СОДЕРЖАНИЕ

Введение 4

1 Технический проект 5

1.1 Анализ предметной области 5

1.2 Постановка задачи 7

1.3 Требования к программному обеспечению и техническим средствам 8

1.4 Проектирование функциональности и структуры программы 9

1.5 Разработка концептуальной модели базы данных 14

1.6 Выбор средств разработки 15

2 Рабочий проект 19

2.1 Проектирование физической модели базы данных 19

2.2 Функциональное взаимодействие модулей программы 21

2.3 Входные и выходные данные 24

2.4 Установка и настройка программы 25

2.5 Работа с программой 26

Заключение 29

Список использованных источников 30

Приложение А – Листинг модулей 31

# ВВЕДЕНИЕ

Темой курсового проекта является разработка и создание приложения для составления расписания уроков для «МБОУ СОШ №3»

Целью данного курсового проекта будет являться написание приложения, позволяющего быстро просмотреть и редактировать расписание уроков.

Написание данного курсового проекта является целесообразным, потому что:

* приложение позволит быстро редактировать расписание уроков;

Для достижения поставленной цели будут рассмотрены и решены следующие задачи:

* разработана структура и функциональность программы;
* произведён выбор средств реализации;
* описаны минимальные требования к программно-техническим средствам;
* описаны входные и выходные данные;
* разработан программный код приложения;
* создано руководство пользователя;
* разработана инсталляция.

Предметом исследования будет применение современных информационных технологий для создания автоматизированных информационных систем.

Объектом исследования будет автоматизированная система расписания занятий высшего учебного заведения.

Практическая значимость проекта заключается в возможности использования приложения в процессе работы для более удобного и быстрого составления расписания.

Основными пользователями данной программы будут преподаватели, заместители директора по учебной работе, школьники.

# 1 ТЕХНИЧЕСКИЙ ПРОЕКТ

## 1.1 Анализ предметной области

Расписание уроков «МБОУ СОШ №3» в республике Адыгея, город Майкоп на 2022-2023 учебный год составлено согласно «Санитарно-эпидемиологическим требованиям к условиям и организации обучения общеобразовательных учреждениях», утверждённое Постановлением государственного санитарного врача Российской Федерации от 29 декабря 2010г №189 г. Москва «Об утверждении СанПиН 2.4.2.2821-10».

Расписание составлено в соответствии с Учебным планом образовательного учреждения на 2022-2023 учебный год. Также учитывались сведения о распределении учебной нагрузки преподавателей, сведения о количестве классов на начало учебного года и о численности обучающихся.

Рационально составленное расписание занятий способствует эффективности учебно-воспитательного процесса, снижению и ликвидации перегрузок учащихся, повышению работоспособности обучающихся, преподавателей, оптимальному использованию кабинетов и других учебных помещений.

Расписание устанавливает распорядок занятий в течение дня, недели, всего учебного года, определяет характер и продолжительность работы учащихся над выполнением домашних заданий и труд учителей по подготовке к урокам. При формировании расписания уроков на 2020 – 2021 учебный год учитывались данные о дневном и недельном циклах изменения работоспособности учащихся.

Образовательное учреждение занимается в одну смену. Таким образом, расписание уроков составлено для учащихся первой смены.

Расписание 1-4 классов составлено в соответствии с учебным планом по 5-дневной рабочей неделе, с максимальной нагрузкой учащимся, не превышающей допустимую.

Ежедневное количество уроков в 1-х классах не более 4-х, один раз в неделю 5 уроков, за счет урока физической культуры. Для обучающихся 2-4 классов – не более 5 уроков.

Обучение в 1-м классе осуществляется с соблюдением следующих дополнительных требований:

* учебные занятия проводятся по 5-дневной неделе и только в первую смену;
* использование «ступенчатого» режима обучения в первом полугодии (в сентябре –октябре по 3 урока в день по 35 минут каждый; со второй четверти – по 4 урока по 35минут каждый; январь - май – по 4 урока по 45 минут каждый); Остальное время заполняется целевыми прогулками, экскурсиями, физкультурными занятиями, развивающими играми.

Расписание 5-8 классов составлено в соответствии с учебным планом по 5-дневной рабочей неделе, с максимальной нагрузкой учащимся, не превышающей допустимую.

Ежедневное количество уроков для обучающихся 5-6 классов – не более 6 уроков, 7-8 классов — не более 7 уроков.

Учащиеся 9-11 классов обучаются по 6-дневной рабочей неделе. В основное расписание включены все уроки по учебному плану и элективные курсы. Ежедневное количество уроков для обучающихся 9-11 классов — не более 7 уроков, в субботу — не более 5уроков. В целях оптимизации работы элективных курсов в 11 классе на 8 урок вынесены элективы по истории и биологии, чтобы их могли посещать обучающиеся 11 «А» и 11 «Б» классов.

## 1.2 Постановка задачи

Необходимо разработать программу для составления расписания уроков для «МБОУ СОШ №3», а также создать документационное сопровождение программы, включающее в себя технический и рабочие проекты.

Материал должен быть предоставлен в файлах, поддерживающих текст.

Данная программа должна будет содержать все необходимые сведения для составления расписания, иметь удобный и понятный пользователю графический интерфейс.

Программа должна обеспечить выполнение следующих функций:

* авторизация;
* переход между страницами;
* возможность просмотра расписания;
* возможность редактирования расписания;
* хранение информации в БД;
* возможность выбора класса, для редактирования расписания;
* возможность выбора определённого дня для изменения расписания;
* вывод информации и изображений в текстовые и графические поля.

Программный продукт должен включать в себя две части:

* ознакомительную часть, которая должна содержать все необходимые сведения о расписании школы.
* Контролирующую часть, которая должна содержать редактирование расписания. На каждый день должно быть расписание для всех школьных классов.

Входными данными в приложении будет логин и пароль пользователя, информация о преподавателях, предметах, классах и днях, которые будут хранится в базе данных.

Выходными данными в приложении будет информация с расписанием, вывод количества, времени и названия уроков, для каждого класса.

Приложение будет состоять из страниц, на которых будет располагаться вся необходимая пользователю информация.

Способ решения поставленных задач зависит от выбора средств разработки.

Тестирование программы будет проводится путем выбора пунктов меню и проверки корректности получаемого отклика программы.

Блок тестирования будет проверяться путем запуска теста и эмуляцией различных вариантов пользовательских действий.

## 1.3 Требования к программному обеспечению и техническим средствам

Для корректной работы разрабатываемой программы необходимо, чтобы компьютер удовлетворял следующим минимальным требованиям:

* процессор Intel® Core™ i3-4130;
* объем свободной оперативной памяти не менее 2 Гб;
* видеоадаптер Intel HD Graphics 405;
* 100 Мб свободного места на жестком диске;
* стандартная клавиатура с 101/109 клавишами;
* манипулятор типа «мышь».

Для нормального функционирования системы на компьютере должна быть установлена операционная система Windows 10, СУБД Microsoft SQL Server.

1.4 Проектирование функциональности и структуры программы

Для описания основных функций проектируемой программы воспользуемся методологией SADT с использованием диаграмм IDEF0.

Сначала строим самую общую, контекстную диаграмму (рисунок 1). На ней описываем процесс контроля с помощью программного приложения одним функциональным блоком, определяем входные и выходные данные, управляющие воздействия и механизмы.

Входные данные:

* Логин (I1);
* Пароль (I2);
* Исходная БД (I3);
* Данные о классах (I4);
* Данные о расписании (I5);
* Запросы на формирование нового расписания (I6).
* Выходные данные:
* Сообщение об ошибке (O1);
* Изменённое БД (O2);
* Изменённое расписание (O3).
* Управление:
* Правила работы с программой. (C1);
* Механизмом будет являться создаваемое приложение (M1).

Субъект: приложение для составления расписания уроков для «МБОУ СОШ №3».

Точка зрения: процесс обучения и контроля знаний рассматривается с точки зрения корректного визуального отображения текстовой и графической информации.

Цель: рассмотрение функций системы, выполнение которых позволит корректно отобразить текстовую и графическую информацию.

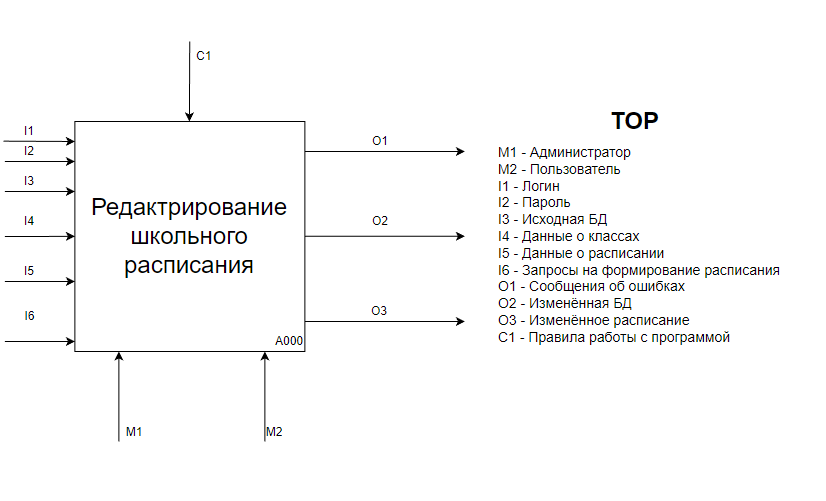


Рисунок 1 – Контекстная IDEF0-диаграмма

Далее производим декомпозицию функционального блока A000 (рисунок 2), выделяя подфункции «Выполнить аутентификацию в системе» – блок A100, и «Выполнить администрирование системы» – блок A200.

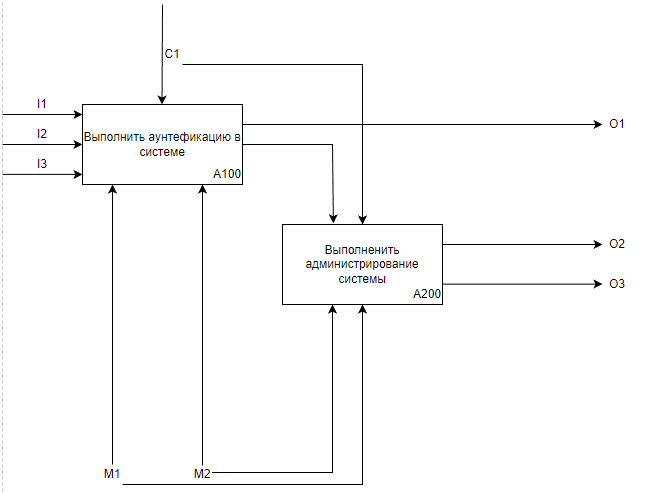


Рисунок 2 – IDEF0-диаграмма основных процессов обучающей и контролирующей программы

На вход функции «Выполнить аутентификацию в системе» будет подаваться информация о логине (I1) и его пароль (I2) из базы данных (I3), после чего будет происходить проверка введённых данных с базой данных, после чего появится либо вывод об ошибке (O1), либо вы будете допущены к следующему блоку.

Для блока «Выполнить администрирование системы» входными данными будут материалы хранящимися в базе данных (I3) о расписании (I5) Выходными данными будет изменённая база данных (O2) и изменённое расписание (O3).

Обеспечивать выполнение функций «Выполнить аутентификацию в системе» и «Выполнить администрирование системы» будет разрабатываемое приложение.

Теперь на отдельных диаграммах произведем декомпозицию функциональных блоков «Выполнить администрирование системы» и «Выполнить администрирование системы».

На диаграмме A100 (рисунок 3) представлены основные процессы, позволяющие корректно выполнить аутентификацию в систему:

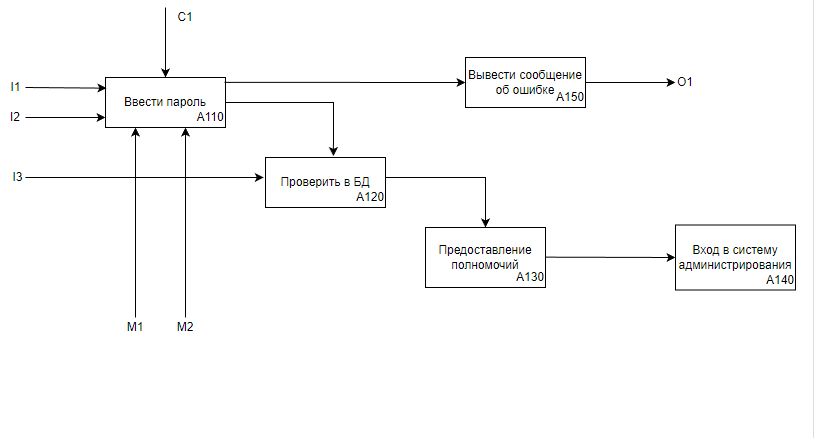


Рисунок 3 – IDEF0-диаграмма основных процессов аутентификации

На диаграмме A200 (рисунок 4) представлены основные процессы, администрирования системы:

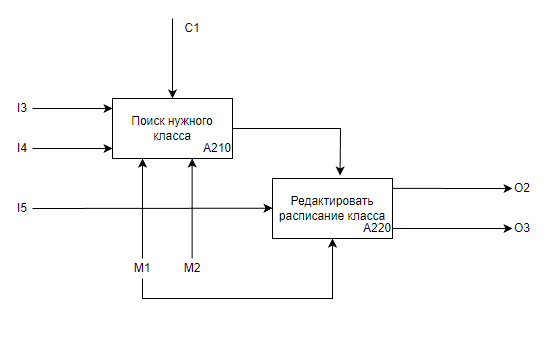


Рисунок 4 – IDEF0-диаграмма основных процессов администрирования системы

На диаграмме A210 (рисунок 5) представлены основные функции, осуществляемые проектируемым приложением, которые позволят корректно провести тестирование:

* «Отобразить блоки для ввода информации» (A211);
* «Проверка правильности ввода данных» (A212);
* «Сохранение изменений в базу данных» (A213).

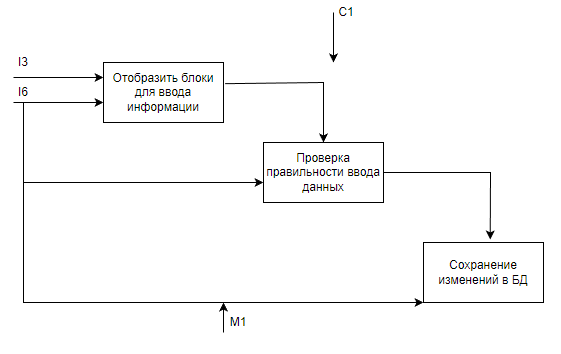


Рисунок 5 - IDEF0-диаграмма основных процессов добавления/редактирования информации о участниках

Исходная база данных (I3) и данные о расписании (I6) будут входными данными для первого блока A210. Заполненные поля будут выходным данным для блока A212 и входным для блока A213.

Выходными потоками для блока A213 будут:

* Изменённая БД(O2);
* Изменённое расписание (O3).

Результатом выполнения блока «Выполнить администрирование системы» будет отображение новой информации расписания и сохранения её в базе данных.

С целью удостоверения предъявляемых к программе требований, приведенных в разделе 1.2, разрабатываемое приложение будет состоять из следующих окон:

* загрузки – будет запускаться перед отображением главного окна;
* авторизация – будет содержать блоки для ввода информации и кнопку входа в приложение;
* главная – окно с выбором класса для просмотра/редактирования расписания;
* просмотр расписания – окно, где можно посмотреть расписание;
* редактирование расписания – окно, где можно сменить расписание;
* справки – будет выводить справочный материал для пользователя.

Переход к блоку «окно с просмотром расписания» будет происходить при выборе пункта «Сменить расписание» в окне с расписанием.

Материалы теста будут хранится в БД. Проектируемая структура приложения представлена на рисунке 6.

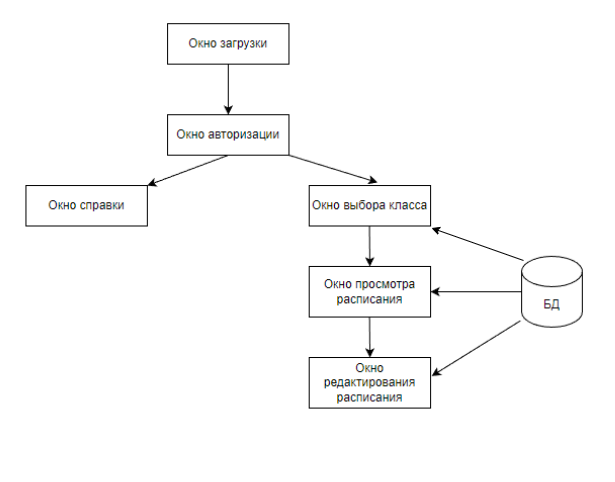


Рисунок 6 – Схема проектируемой структуры приложения

Тестирование программы планируется проводить путём выбора каждого пункта меню и проверки корректности получаемого отклика программы. Блок тестирования будет проверяться путем запуска тестов и эмуляцией различных вариантов пользовательских действий.

## 1.5 Разработка концептуальной модели базы данных

Для хранения информации о пользователях и расписании будет разработана БД. Она будет состоять из таблиц: «Пользователи», «Роли», «Классы» и «Расписание», которые включают в себя следующие поля:

Пользователи:

* Id – служебное поле для обеспечения уникальности записей;
* RoleId – внешний ключ к таблице «Роли»;
* Login – логин;
* Password – пароль;
* FullName – Ф.И.О администратора.

Роли:

* Id – служебное поле для обеспечения уникальности записей;
* RoleName – наименование роли.

Классы:

* Id – служебное поле для обеспечения уникальности записей;
* ClassName – название класса;
* Smena – смена.

Расписание:

* Id – служебное поле для обеспечения уникальности записей;
* ClassId – внешний ключ класса;
* DayWeekId – день недели;
* NumTask – номер урока;
* TasksName – название урока;
* Office – кабинет;
* TeacherName – имя преподавателя.

Концептуальная модель БД представлена на рисунке 7.

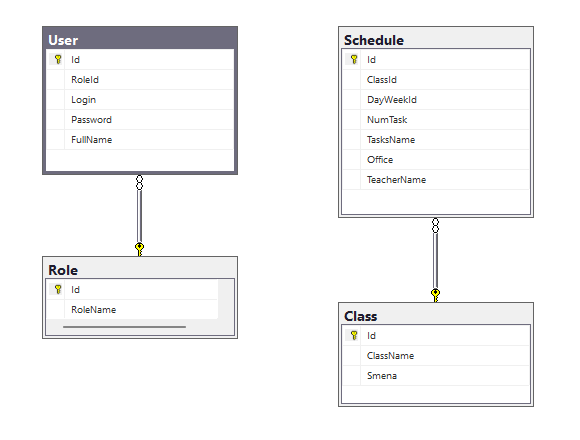


Рисунок 7 – Концептуальная модель БД

## 1.6 Выбор средств разработки

Из множества различных языков программирования для реализации курсового проекта был выбран C# и интегрированная среда разработки Microsoft Visual Studio 2019.

C# является объектно-ориентированным и в этом плане много перенял у Java и С++. Он поддерживает полиморфизм, наследование, перегрузку операторов, статическую типизацию. Объектно-ориентированный подход позволяет решить задачи по построению крупных, но в тоже время гибких, масштабируемых и расширяемых приложений. C# продолжает активно развиваться, и с каждой новой версией появляется все больше интересной функциональности, как, например, лямбда-выражения, динамическое связывание, асинхронные методы и так далее.

Преимущества C# по сравнению с аналогичными программными продуктами:

* поддержка подавляющего большинства продуктов Microsoft;
* типы данных имеют фиксированный размер что повышает мобильность языка и упрощает программирование;
* автоматическая очистка оперативной память;
* большое количество специальных конструкций, разработанных для понимания и написания кода.

Microsoft Visual Studio – полнофункциональная интегрированная среда разработки (IDE) с поддержкой популярных языков программирования, среди которых С, C++, VB.NET, C#, F#, JavaScript, Python. Функциональность Visual Studio охватывает все этапы разработки программного обеспечения, предоставляя современные инструменты для написания кода, проектирования графических интерфейсов, сборки, отладки и тестирования приложений. Возможности Visual Studio могут быть дополнены путем подключения необходимых расширений. Visual Studio поддерживает подсветку синтаксиса, вставку фрагментов кода, отображение структуры и связанных функций. Существенно ускорить работу помогает технология IntelliSense – автозавершение кода по мере ввода.

Встроенный отладчик Visual Studio используется для поиска и исправления ошибок в исходном коде, в том числе на низком аппаратном уровне. Инструменты диагностики позволяют оценить качество кода с точки зрения производительности и использования памяти.

Для создания инсталляции проекты был выбран Smart InstallMaker. Это условно-бесплатная утилита с закрытым исходным кодом, которая представляет пользователям мощный и простой в использовании инструмент для создания высокоэффективных инсталляторов в 32-битных операционных системах Microsoft Windows.

Smart InstallMaker обладает богатыми функциональными возможностями для создания надёжных и красивых инсталляционных пакетов. Утилита предоставляет мощный алгоритм Cabinet для сжатия и создания компактных инсталляторов с маленьким размером, поддерживает создание многоязычного инсталлятора (до 25 языков), а также предоставляет средства для управления внешним видом программы-установки. Среди прочих возможностей Smart InstallMaker, можно выделить создание деинсталлятора, ярлыков в указанных директориях системы, регистрацию новых шрифтов или расширений в операционной системе, поддержку ActiveX и переменных, проверку установленного .NET Framework, разделение инсталлятора на установочные диски, детальную настройку информации об установочном файле, системных требований (например, запретить установку программы в Windows 95), а также его внешнего вида.

Недостатки:

* отсутствие кроссплатформенности;
* закрытый исходный код;
* отсутствует поддержка 64-разрядных операционных систем.

Microsoft SQL Server – система управления реляционными БД, разработанная корпорацией Microsoft.

# 2 РАБОЧИЙ ПРОЕКТ

## 2.1 Проектирование физической модели базы данных

Физическая модель данных описывает то, как данные хранятся в компьютере, представляя информацию о структуре записей, их упорядоченности и существующих путях доступа. Также описываются типы, идентификаторы и разрядность полей.

На основе логической модели были создана физическая модель БД, которая хранится в компьютере в файле SchoolDb.mdf (таблица 1).

Таблица 1 – Физическая модель таблицы «Пользователи»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Подпись поля | Тип данных | Длина | Ключ |
| Id | Уникальный номер пользователя | int | - | Первичный |
| Login | Логин | nvarchar | 255 | - |
| Password | Пароль | char | 55 | - |
| FullName | Ф.И,О | nvarchar | 255 | - |

Таблица 2 – Физическая модель таблицы «Роли»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Подпись поля | Тип данных | Длина | Ключ |
| Id | Уникальный номер | int | - | Первичный |
| RoleName | Наименование роли | nvarchar | 255 | - |

Таблица 3 – Физическая модель таблицы «Классы»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Подпись поля | Тип данных | Длина | Ключ |
| Id | Уникальный номер класса | int | - | Первичный |
| ClassName | Название класса | nvarchar | 255 | - |
| Smena | Смена | nvarchar | 255 | - |

Таблица 4 – Физическая модель таблицы «Расписание»

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя поля | Подпись поля | Тип данных | Длина | Ключ |
| Id | Уникальный номер | int | - | Первичный |
| ClassId | Внешний ключ класса | int | - | Внешний |
| DayWeekId | День недели | nvarchar | 255 | - |
| NumTask | Номер урока | int | - | - |
| TasksName | Название урока | nvarchar | 255 | - |
| Office | Кабинет | int | - | - |
| TeacherName | Ф.И.О преподавателя | nvarchar | 255 | - |

## 2.2 Функциональное взаимодействие модулей программы

На основе разработанного технического проекта было написано приложение по теме «Расписание школы». Для создания программы был использован язык программирования C# и ИСР Visual studio 2019. В программе используются следующие языковые модули:

* System − содержит фундаментальные и базовые классы, определяющие часто используемые типы значений и ссылочных данных, события и обработчики событий, интерфейсы, атрибуты и исключения обработки;
* System.Windows − содержит типы, используемые в приложениях
* System.IO − содержит типы, поддерживающие ввод и вывод, включая возможности чтения и записи данных в потоках как синхронно, так и асинхронно, реализации пользовательского ведения журнала и обработки входящих и исходящих потоков данных в последовательных портах;
* System.Data − содержит типы для доступа к данным из различных источников и для управления этими данными. Пространство имен верхнего уровня и несколько дочерних пространств имен образуют архитектуру ADO.NET и поставщиков данных ADO.NET.
* System.Threading− содержат типы, обеспечивающие возможности многопотокового программирования. Дочернее пространство имен предоставляет типы, которые упрощают задачу написания параллельного и асинхронного кода;
* System.Collections.Generic – работа с обобщёнными коллекциями (List<string>).

Также были разработаны следующие пользовательские модули, взаимодействие которых представлено на рисунке 8:

* SplashScreen.xaml.cs – Экран загрузки;
* Autorization.xaml.cs – Экран авторизации;
* MainWindow.xaml.cs – Главное меню выбора класса;
* SheldureWindow.xaml.cs – Экран с расписанием класса;
* SupportWindow.xaml.cs – Экран со справкой;
* EditSheldureWindow.xaml.cs – Экран с редактированием расписания;
* BaseViewModel.cs – Расширенная реализация, для упрощения работы со свойством;
* AutorizationWindowViewModel.cs – Расширенная реализация авторизации;
* RelayCommand.cs – Взаимодействие окон (xaml) с классами ViewModel;
* CurrentUser.cs – Хранение текущего пользователя;
* EditWindowViewModel – Расширенная реализация изменения расписания.

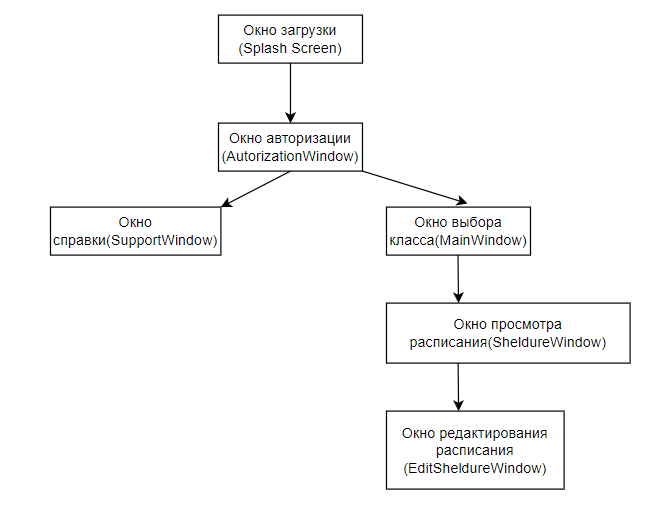


Рисунок 8 – Схема взаимодействия основных модулей программы

В модуле AutorizationWindow.xaml.cs описаны и используются следующие процедуры:

* private void ExiteBtn\_Click(object sender, RoutedEventArgs e) – Выход из программы;
* private void PasswordBox\_PasswordChanged(object sender, RoutedEventArgs e) – Передача информации из TextBox в PasswordBox
* private void CheckBox\_Checked(object sender, RoutedEventArgs e) – Передача информации из PasswordBox в TextBox. Смена видимости у PasswordBox в режим невидимки и TextBox в режим видимости.
* private void CheckBox\_Unchecked(object sender, RoutedEventArgs e) - Смена видимости у TextBox в режим невидимки и PasswordBox в режим видимости.

В модуле MainWindow.xaml.cs описаны и используются следующие процедуры:

* private void ExiteBtn\_Click(object sender, RoutedEventArgs e) – Выход из программы;
* private void AddBtn\_Click(object sender, RoutedEventArgs e) – Переход в окно с расписанием.

В модуле SheldureWindow.xaml.cs описаны и используются следующие процедуры:

* private void ExiteBtn\_Click(object sender, RoutedEventArgs e) – Выход из программы;
* private void EditBtn\_Click(object sender, RoutedEventArgs e) – Переход в окно с редактированием расписания.

В модуле EditSheldureWindow.xaml.cs описаны и используются следующие процедуры:

* private void ExiteBtn\_Click(object sender, RoutedEventArgs e) – Выход из программы;
* private void ReturnBtn\_Click(object sender, RoutedEventArgs e) – Переход в главное окно.

## 2.3 Входные и выходные данные

Входными данными для модуля AutorizationWindow.xaml.cs являются данные из базы данных, которая загружается из класса MainWindowViewModel.cs. Ввод данных происходит в LoginTextBox и PasswordTextBox (PasswordBox). Выходными данными будет отображение информации об ошибке MessageBox или авторизация и переход к следующему модулю.

Входными данными для модуля MainWindow.xaml.cs являются данные из базы данных, которая загружается из класса MainWindowViewModel.cs. Информация выводиться в ComboBox.

Входными данными для модуля SheldureWindow.xaml.cs являются данные из базы данных, которая загружается из класса SheldureWindowViewModel.cs. Информация выводиться в ItemSource.

Входными данными для модуля EditSheldureWindow.xaml.cs являются данные ведёнными пользователем в TextBox и ComboBox. При клике на Button вся ведённая информация сохраняется в базу данных.

## 2.4 Установка и настройка программы

Для установки программы необходимо запустить файл инсталляции Edit Sheldure.exe. В результате в выбранном пользователем каталоге будет создана папка «Sheldure», в которой будут размещены исполняемых файл программы (Edit Sheldure.exe), файл системы помощи (help.chm), и файлы с материалами (БД) Также будет создан ярлык для запуска приложения на рабочем столе. Ярлык для деинсталляции программы будет размещен в папке программы.

Для создания инсталляции была использована программа Smart Install Maker. В ходе инсталляции пользователю показываются следующие окна:

* выбор папки установки;
* создание дополнительных ярлыков – позволяет создать ярлык на рабочем столе или отказаться от создания ярлыка;
* ход выполнения процесса установки;
* завершение установки – позволяет выбрать файлы, которые нужно запустить после установки.

После запуска программы инсталляции нужно следовать указаниям в появляющихся окнах.

## 2.5 Работа с программой

Приложение было разработано на языке C# на платформе .NET Framework 4.8. Для создания графического интерфейса использовалась технология WPF. На рисунке 9 представлено окно загрузки программы.

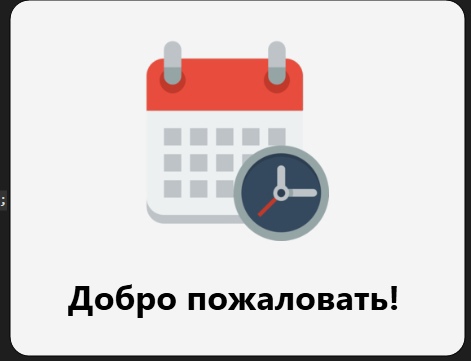


Рисунок 9 – Окно загрузки

После завершения загрузки, откроется окно с авторизацией в приложении (Рисунок 10)

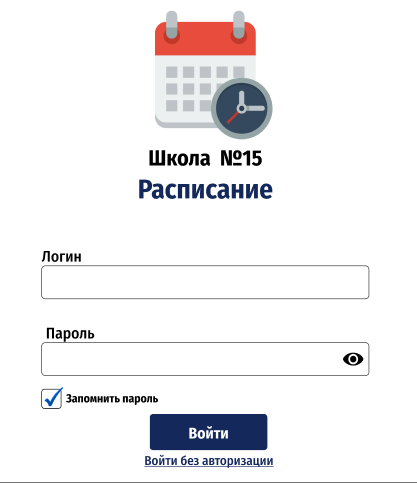


Рисунок 10 – Окно авторизации

При неправильном вводе логина или пароля, выскакивает предупреждение, что какие-то данные были введены неправильно. (Рисунок 11)

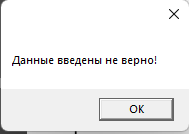


Рисунок 11 – Диалоговое окно с предупреждением

После авторизации появиться окно главного меню. В этом окне необходимо выбрать нужный класс, литеру и форму обучения, для поиска нужного расписания. (Рисунок 12)

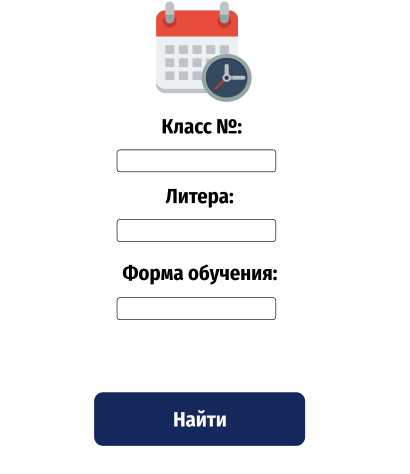


Рисунок 12 – Главное меню

При нажатии на кнопку найти, отобразиться расписание выбранного класса (Рисунок 14). Здесь отображено время урока, ФИО преподавателя и день недели.

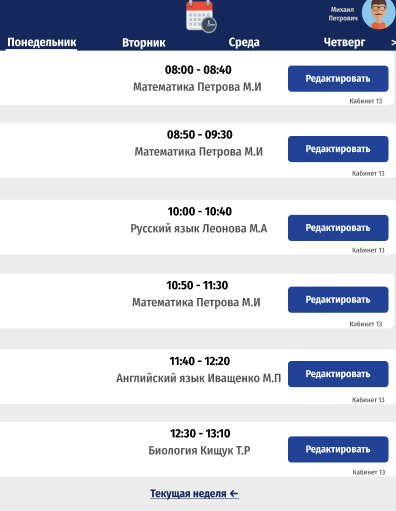


Рисунок 13 – Окно с расписанием класса

При нажатии на кнопку редактировать откроется ещё одно окно, где можно изменить выбранный урок. (Рисунок 14)



Рисунок 14 – Окно редактирование урока

# ЗАКЛЮЧЕНИЕ

В результате выполнения данного курсового проекта было написано приложение для составления расписания уроков для «МБОУ СОШ №3» с помощью которой можно легко и быстро просмотреть и изменить расписание

Основными достоинствами программы являются:

* программа имеет простую установку, настройку и эксплуатацию;
* наличие удобного графического интерфейса;
* данный программный продукт будет удобен для преподавателей и школьников;
* Быстрое изменение расписания.

Использование данной программы избавит от необходимости:

* печатать расписание;
* тратить время на поиск расписания;

В ходе выполнения курсового проекта были достигнуты поставленные цели и решены следующие задачи:

* исследованы все основные элементы при составлении расписания для школы;
* спроектированы структура приложения и БД;
* разработаны справочные и тестовые материалы;
* определены минимальные требования к техническим и программным средствам;
* описаны входные и выходные данные;
* выбраны средства реализации приложения;
* разработана БД;
* разработан программный код приложения;
* создано руководство пользователя;
* создана инсталляция.

К недостаткам программы можно отнести небольшой функционал.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Уроки C# - https://metanit.com/sharp/tutorial/
2. Шилдт Г. C#: полное руководство.М.:Вильямс, 2017. 730 с.
3. Работа с Microsoft Office - https://office-apps.net/microsoft-word/
4. Справочный материал по C# -<https://www.cyberforum.ru/>
5. СеппаД. MicrosoftADO.NET 2015.М.:Вильямс, 2019. 785 с.
6. Оппель Э. Дж. SQL: Полное руководство. М.:Вильямс, 2015. 960 с.
7. Албахари Д. C# 6.0. Справочник. Полное описание языка. М.:Вильямс, 2018. 1040 с.
8. Скит Д. С# Для профессионалов. Тонкости программирования. М.: Вильямс, 2019. 608 с.
9. Документация Microsoft. URL: https://docs.microsoft.com/ru-ru/documentation (дата обращения: 11.11.2020).
10. Вольная борьба в городе Майкоп. Сайт с информацией о вольной борьбе. URL: https://volnayaborba.adg.sportsng.ru/ (дата обращения: 06.11.2022).
11. Официальный сайт HTML HelpWorkShop. URL: https:// www.microsoft.com (дата обращения: 10.12.2020).
12. Официальный сайт Smart InstallMaker. URL: <http://ru.sminstall.com> (дата обращения: 06.12.2022).

Приложение А

Листинг модуля BaseViewModel.cs

public abstract class BaseViewModel : INotifyPropertyChanged

{

public event PropertyChangedEventHandler PropertyChanged;

// Стандартная реализация

protected virtual void OnPropertyChanged([CallerMemberName] string propertyName = null)

{

PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));

}

// Расширеная реализация, для упрощения работы со свойствами.

protected virtual bool SetPropertyChanged<T>(ref T source, T value, [CallerMemberName] string propertyName = null)

{

if (EqualityComparer<T>.Default.Equals(source, value))

return false;

source = value;

OnPropertyChanged(propertyName);

return true;

}

}

}

Листинг модуля RelayCommand.cs

public class RelayCommand : ICommand

{

private Action<object> execute;

private Func<object, bool> canExecute;

public event EventHandler CanExecuteChanged

{

add { CommandManager.RequerySuggested += value; }

remove { CommandManager.RequerySuggested -= value; }

}

public RelayCommand(Action<object> execute, Func<object, bool> canExecute = null)

{

this.execute = execute;

this.canExecute = canExecute;

}

public bool CanExecute(object parameter)

{

return this.canExecute == null || this.canExecute(parameter);

}

public void Execute(object parameter)

{

this.execute(parameter);

}

}

}

Листинг модуля MainWindowViewModel.cs

public class MainWindowViewModel : BaseViewModel

{

private string \_password;

private string \_login;

private RelayCommand \_loginCommand;

List<User> Users = new List<User>();

public string Password

{

get => \_password;

set => SetPropertyChanged(ref \_password, value, nameof(\_password));

}

public string Login

{

get => \_login;

set => SetPropertyChanged(ref \_login, value, nameof(\_login));

}

public RelayCommand LoginCommand

{

get { return \_loginCommand ?? (\_loginCommand = new RelayCommand(obj => { enterbtn(); })); }

}

public void enterbtn()

{

using (var context = new WrestlingDbEntities())

{

var user = context.User.FirstOrDefault(e => e.Login == Login && e.Password == Password);

if (user != null)

{

CurrentUser.Name = user.FullName;

View.WindowMain contentWindow = new View.WindowMain();

contentWindow.Show();

Application.Current.MainWindow.Hide();

return;

}

else

{

MessageBox.Show("Данные введенны неверно. Повторите попытку.");

}

}

}

}

}

Листинг модуля CurrentUser.cs

public class CurrentUser

{

public static string Name { get; set; }

}

}