**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ**

**УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ ИМ. ПРОФ. М.А. БОНЧ-БРУЕВИЧА»**

**(СПбГУТ)**

**АРХАНГЕЛЬСКИЙ КОЛЛЕДЖ ТЕЛЕКОММУНИКАЦИЙ**

**ИМ. Б.Л. РОЗИНГА (ФИЛИАЛ) СПбГУТ**

**(АКТ (ф) СПбГУТ)**

СОГЛАСОВАНО

Рук. предприятия

\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(Подпись) (И.О. Фамилия)

«31» мая 2025г.

**ТЕХНИЧЕСКИЙ ОТЧЁТ**

**по ПМ.11, ПМ.01**

|  |
| --- |
| МБОУ ОШ №12 |
| Информационные системы и программирование |
| 09.02.07. 25ТО01. 026 ПЗ |
| (Обозначение документа) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИСПП-21 | |  | 31.05.2025 | А.В. Речицкий |
|  | (Группа) | | (Подпись) | (Дата) | (И.О. Фамилия) |
| Рук. практики от предприятия | | |  | 31.05.2025 | И.В. Кресцова |
|  | |  | (Подпись) | (Дата) | (И.О. Фамилия) |

Архангельск 2025

Содержание

[Перечень сокращений и обозначений 3](#_Toc201228119)

[Введение 4](#_Toc201228120)

[1 Охрана труда и техника безопасности при работе на ПК 6](#_Toc201228121)

[1.1 Общие требования безопасности 6](#_Toc201228122)

[2 Выполнение работ по ПМ.11 7](#_Toc201228123)

[2.1 Проектирование базы данных 7](#_Toc201228124)

[2.2 Разработка базы данных и объектов базы данных 10](#_Toc201228125)

[2.3 Администрирование и защита базы данных 13](#_Toc201228126)

[3 Выполнение работ по ПМ.01 15](#_Toc201228127)

[3.1 Постановка задачи 15](#_Toc201228128)

[3.2 Описание алгоритма функционирования системы 16](#_Toc201228129)

[3.3 Разработка мобильного приложения 18](#_Toc201228130)

[3.4 Рефакторинг и оптимизация программного кода 21](#_Toc201228131)

[Заключение 22](#_Toc201228132)

[Список использованных источников 23](#_Toc201228133)

Перечень сокращений и обозначений

В настоящем техническом отчете применяются следующие сокращения и обозначения:

БД – база данных

ВДТ – видео-дисплейный терминал

МБОУ – муниципальное бюджетное общеобразовательное учреждение

ОС – операционная система

ОШ – основная школа

ПК – персональный компьютер

ПО – программное обеспечение

ПЭВМ – персональная электронно-вычислительная машина

СУБД – система управления базами данных

API – программный интерфейс приложения

DDL – язык описания данных

EF – Entity Framework

GUI – графический пользовательский интерфейс

HMAC – Hash-based Message Authentication Code

IDE - интегрированная среда разработки

ORM - объектно-реляционное отображение

PDA - атаки по заранее вычисленным словарям

RAM – оперативная память

SDK – набор инструментов для разработки программного обеспечения

SHA-256 – Secure Hash Algorithm 256-bit

SQL – язык структурированных запросов

UI – пользовательский интерфейс

UML – унифицированный язык моделирования

WPF – Windows Presentation Foundation

Введение

Базой производственной практики является МБОУ ОШ №12. Предприятие занимается обучением школьников, в том числе с использованием информационных технологий.

Целями производственной практики являются:

* получение практического опыта по выполнению работ по ПМ.01 «Разработка программных модулей программного обеспечения для компьютерных систем» и развитие общих и профессиональных компетенций;
* получение практического опыта по выполнению работ по ПМ.11 «Разработка, администрирование и защита баз данных» и развитие общих и профессиональных компетенций.

Задачами производственной практики являются:

* разработка алгоритма поставленной задачи и создание схемы алгоритма с использованием специализированного программного обеспечения;
* описание работы программного модуля с помощью языка моделирование UML с использованием специализированного программного обеспечения;
* разработка кода программного продукта на основе готовой спецификации на уровне модуля;
* разработка на основе готовой спецификации кода программного продукта с графическим интерфейсом пользователя на высокоуровневом языке программирования;
* использование инструментальных средств на этапе отладки программного продукта;
* проведение тестирования программного модуля по определённому сценарию;
* разработка модульных тестов и выполнение автоматизированного тестирования программного модуля;
* рефакторинг программного кода;
* разработка мобильного приложения;
* выделение объектов и атрибутов в соответствии с заданием;
* построение и обоснование концептуальной модели БД;
* проектирование логической и физической схемы БД;
* работа с современными CASE-средствами проектирования БД;
* формирование и настройки схемы БД;
* создание индексов в таблице БД;
* использование средств заполнения БД;
* создание представлений БД;
* создание хранимых процедур и триггеров БД;
* создание запросов к БД;
* установка и настройка SQL-сервера;
* управление пользователями и ролями в конкретной СУБД;
* осуществление контроля доступа к данным и управление привилегиями;
* использование стандартных методов защиты объектов БД;
* выполнение резервного копирования БД;
* восстановление БД из резервной копии.

Для практикантов предоставляется рабочее место с персональным компьютером и всем необходимым для работы аппаратным и программным обеспечением:

* процессор: AMD A4-9125 RADEON R3;
* системная плата: LENOVO LNVNB161216;
* видеокарта: встроенная;
* оперативная память: 16ГБ;
* операционная система: Microsoft Windows 10 Pro;
* прикладное ПО: Microsoft Office, Microsoft Edge, Visual Studio 2022, Android Studio, Microsoft SQL Server Management Studio 20, Draw.io.

# Охрана труда и техника безопасности при работе на ПК

## Общие требования безопасности

Действия настоящей инструкции распространяется на всех работников образовательного учреждения, которые при использовании своих должностных обязанностей используют ПЭВМ и ВДТ.

К самостоятельной работе с ВДТ и ПЭВМ допускаются:

* лица не моложе 18 лет, прошедшие медицинский осмотр и не имеющие противопоказаний по состоянию здоровья для работы с ПЭВМ и ВДТ;
* прошедшие, как правило, курс обучения принципам работы с вычислительной техникой и специальное обучение работе с использованием конкретного программного обеспечения;
* прошедшие вводный инструктаж по электробезопасности с присвоением II группы допуска;
* ознакомленные с инструкциями по эксплуатации средств оргтехники (принтеры, сканеры, источники бесперебойного питания и т.п.).

Опасными факторами при работе с ПЭВМ и ВДТ являются:

* физические (низкочастотные электрические и магнитные поля, статическое электричество, лазерное и ультрафиолетовое излучение, повышенная температура, ионизация воздуха, опасное напряжение в электрической сети);
* химические (пыль, вредные химические вещества, выделяемые при работе принтеров);
* психофизиологические (напряжение зрения и внимания, интеллектуальные и эмоциональные нагрузки, длительные статические нагрузки и монотонность труда).

# Выполнение работ по ПМ.11

## Проектирование базы данных

В рамках проекта разработана информационная система для автоматизации ключевых процессов школьного учёта: регистрации учеников и учителей, распределения по классам и предметам, фиксации посещаемости и выставления оценок.

Система обеспечивает регистрацию оценок и учёт посещаемости, а также фиксирует распределение учебной нагрузки преподавателей.



Основная цель – централизованное хранение данных об учебной деятельности, упрощение контроля посещаемости и успеваемости, повышение эффективности взаимодействия между участниками образовательного процесса.

Для хранения данных выбрана реляционная СУБД Microsoft SQL Server, поэтому концептуальная модель (рисунок 1) преобразована в логическую с учётом перехода от ER-модели к реляционной. Модели созданы с помощью средства проектирования Draw.io.



Рисунок 1 – Концептуальная модель

На рисунке 2 показана логическая модель предметной области, полученная путем преобразования концептуальной модели и созданная по принципу 3НФ с помощью средства проектирования Draw.io. При этом учтены следующие связи между сущностями:

* связь между Ученики и Классы – М:1, поэтому формируется два отношений с внешним ключом в отношении Классы;
* связь между Ученики и Посещаемости – М:1, поэтому формируется два отношения с внешним ключом в отношении Посещаемости;
* связь между Ученики и Оценки – М:1 и необязательна для Ученики, поэтому формируется два отношения с внешним ключом в отношении Оценки;
* связь между Оценки и Предметы – М:1, поэтому формируется два отношения с внешним ключом в отношении Оценки;
* связь между Посещаемости и Предметы – М:1, поэтому формируется два отношения с внешним ключом в отношении Посещаемости;
* связь между Посещаемости и Классы – М:1, поэтому формируется два отношения с внешним ключом в отношении Посещаемости;
* связь между Классы и Учителя – М:1, поэтому формируется два отношения с внешним ключом в отношении Классы;
* связь между Классы, Учителя и Предметы– М:М:М, поэтому формируется четыре отношения с внешними ключами в отношении УчительПоПредмету;
* связь между Классы и Предметы М:М, поэтому формируется три отношения с внешними ключами в отношении ПредметыВКлассе.

На рисунке 3 показана физическая модель предметной области, разработанная на основе логической модели для СУБД Microsoft SQL Server Express 2020.

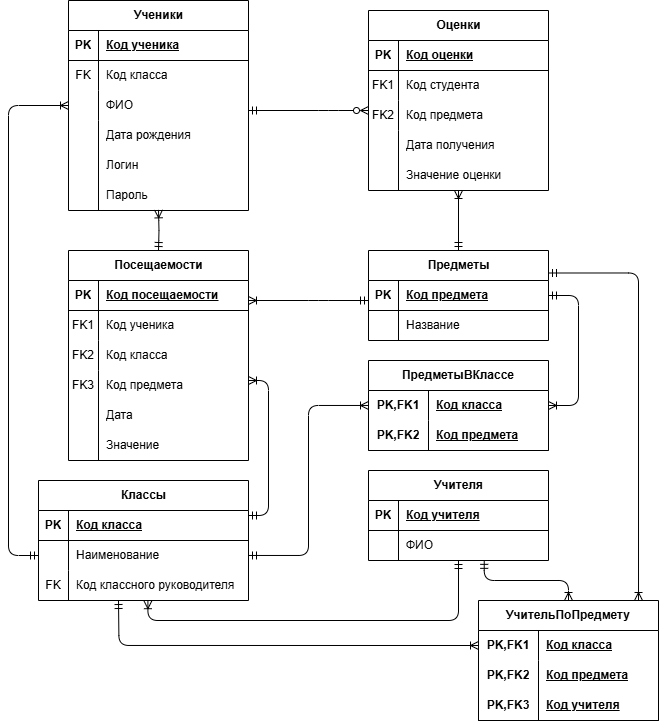


Рисунок 2 – Логическая модель

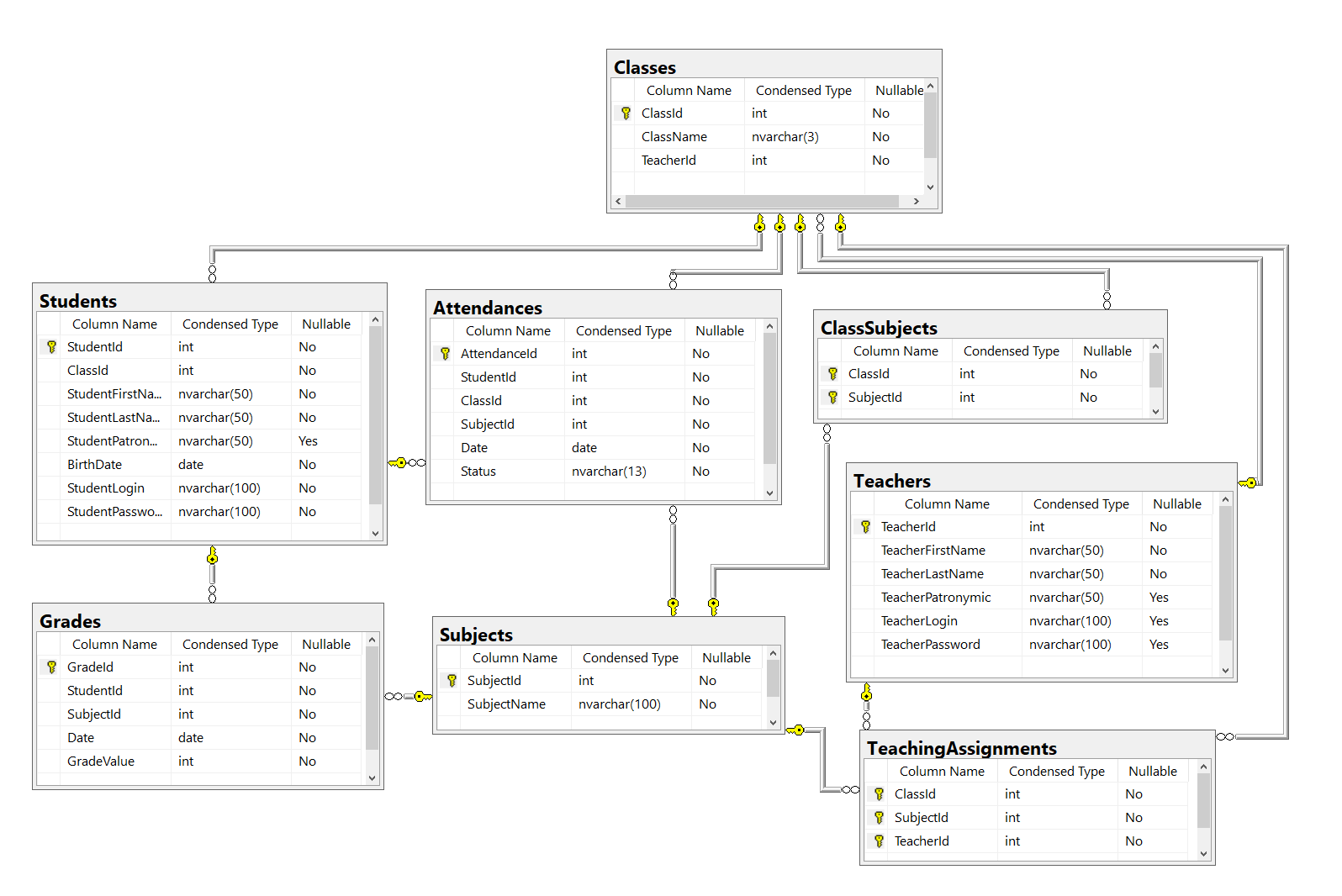


Рисунок 3 – Физическая модель

## Разработка базы данных и объектов базы данных

В таблице 1 в виде словаря данных представлено описание созданных таблиц и ограничений целостности БД.

Таблица 1 – Словарь данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Ключ** | **Поле** | **Тип данных** | **Обязательное** | **Примечание** |
| Students | | | | |
| PK | StudentId | int | + | Автоинкрементный |
| FK | ClassId | Int | + |  |
|  | StudentFirstName | nvarchar(50) | + |  |
|  | StudentLastName | nvarchar(50) | + |  |
|  | StudentPatronymic | nvarchar(50) | - |  |
|  | BirthDate | date | + |  |
|  | StudentLogin | nvarchar(100) | + | Уникальный |
|  | StudentPassword | nvarchar(100) | + |  |

Для создания таблиц БД [6] и ограничений целостности требуется выполнить соответствующие DDL-команды. Код создания таблицы Students представлен листингом 1.

Листинг 1 – SQL-запрос для создания таблицы Students

--Создание таблицы Students с полями и первичным ключом

CREATE TABLE [dbo].[Students](

[StudentId] [int] NOT NULL,

[ClassId] [int] NOT NULL,

[StudentFirstName] [nvarchar](50) NOT NULL,

[StudentLastName] [nvarchar](50) NOT NULL,

[StudentPatronymic] [nvarchar](50) NULL,

[BirthDate] [date] NOT NULL,

[StudentLogin] [nvarchar](100) NOT NULL,

[StudentPassword] [nvarchar](100) NOT NULL,

[StudentPasswordInvis] [nvarchar](100) NOT NULL,

CONSTRAINT [PK\_Students] PRIMARY KEY CLUSTERED ([StudentId] ASC));

Для решения поставленных задач для таблиц БД требуется создать уникальные ограничения в соответствии со словарем данных, используя DDL-команды. Код создания уникальных ограничений для таблиц представлен листингом 2.

Листинг 2 – Код создания уникальных ограничений

--Создание уникального ограничения для таблицы Students

ALTER TABLE Students

ADD CONSTRAINT [UQ\_Students] UNIQUE ([StudentLogin])

GO

Для решения поставленных задач в БД требуется создать:

* скалярную функцию GetStudentAverageGrade, возвращающую среднюю оценку на основе кода ученика и кода предмета. Код создания представлен листингом 3;
* триггер trg\_PreventFutureGrades, выполняющийся после операции INSERT в таблице Grades, и запрещающий выставлять оценки на будущие учебные дни. Код создания процедуры показан листингом 4;
* представление CurrentStudentsWithClassView, возвращающее список ФИО учеников по классам. Код создания представления показан в листинге 5.

Листинг 3 – Код создания скалярной функции GetStudentAverageGrade

--Функция возвращающая средний балл ученика

--по определенному предмету.

CREATE FUNCTION GetStudentAverageGrade

(

@StudentId INT,

@SubjectId INT

)

RETURNS FLOAT

AS

BEGIN

DECLARE @AvgGrade FLOAT;

SELECT @AvgGrade = ROUND(AVG(CAST(GradeValue AS FLOAT)), 2)

FROM Grades

WHERE StudentId = @StudentId AND SubjectId = @SubjectId;

RETURN @AvgGrade;

END;

Листинг 4 – Код создания триггера trg\_PreventFutureGrades

--Триггер, запрещающий добавлять оценки в будущие даты

CREATE TRIGGER trg\_PreventFutureGrades

ON Grades

AFTER INSERT

AS

BEGIN

SET NOCOUNT ON;

IF EXISTS (SELECT 1 FROM inserted WHERE Date > GETDATE())

BEGIN

RAISERROR('Нельзя вставлять оценки в будущие даты.', 16, 1);

ROLLBACK TRANSACTION;

RETURN;

END

END;

Листинг 5 – Код создания представления StudentAverageGradesView

--Представление возвращающее список средней оценки

--учеников по предметам

CREATE VIEW StudentAverageGradesView AS

SELECT

s.StudentId,

s.StudentLastName + ' ' + s.StudentFirstName + ' ' + s.StudentPatronymic AS StudentFullName,

subj.SubjectName,

ROUND(AVG(CAST(g.GradeValue AS FLOAT)), 2) AS AverageGrade

FROM Grades g

JOIN Students s ON g.StudentId = s.StudentId

JOIN Subjects subj ON g.SubjectId = subj.SubjectId

GROUP BY s.StudentId, s.StudentLastName + ' ' + s.StudentFirstName + ' ' + s.StudentPatronymic, subj.SubjectName;

Выполнены базовые операции INSERT INTO для таблицы Subjects в листинге 6. Это необходимо как на этапе начальной инициализации данных, так и для тестирования представлений, процедур и функций.

Листинг 6 – Код для добавления предмета в таблицу Subjects

INSERT INTO Subjects (SubjectName)

VALUES ('Математика'), ('Русский язык'), ('Литература'), ('Физика'), ('Химия'), ('Биология'), ('История'), ('Обществознание'), ('География'), ('Английский язык'), ('Информатика'), ('Физкультура'), ('ОБЖ'), ('Технология'), ('Искусство');

## Администрирование и защита базы данных

Для защиты информации в БД требуется создать отдельные имена для входа и пользователей для учителей и учеников.

У учителя должна быть возможность чтения и записи данных, ученик должен иметь доступ только к чтению.

Для создания имени для входа, пользователи и выдачи пользователю роли БД для ограничения доступа требуется выполнить набор SQL-команд, представленный листингом 7 и 8.

Листинг 7 – Код создания пользователя и имени для входа ученика

CREATE LOGIN [Student] WITH PASSWORD = 'SomeStrongPassword123!'

GO --Создание имени для входа с паролем

CREATE USER [Student] FOR LOGIN [Student]

GO --Создание пользователя для имени для входа

EXEC sp\_addrolemember 'db\_datareader', 'Student'

GO --Выдача прав на чтение данных пользователю

Листинг 8 – Код создания пользователя и имени для входа учителя

CREATE LOGIN [Teacher] WITH PASSWORD = 'SomeStrongPassword123!'

GO --Создание имени для входа с паролем

CREATE USER [Teacher] FOR LOGIN [Teacher]

GO --Создание пользователя для имени для входа

EXEC sp\_addrolemember 'db\_datareader', 'Teacher'

GO --Выдача прав на чтение данных пользователю

EXEC sp\_addrolemember 'db\_datawriter', 'Teacher'

GO --Выдача прав на запись данных пользователю

Во избежание потери данных необходимо периодически выполнять полное резервное копирование БД. Резервные копии позволяют восстановить данные после сбоя и других непредвиденных проблем.

Для выполнения резервного копирования БД требуется выполнить SQL-скрипт, представленный листингом 9.

Листинг 9 – Код для создания резервной копии БД School12

--Выполнение резервного копирования

BACKUP DATABASE [School12] TO

DISK = N'D:\Practika\School12.bak' --Путь

WITH NOFORMAT, NOINIT,

NAME = N'School12-Резервное копирование', -- Имя резервной копии

SKIP, NOREWIND, NOUNLOAD, STATS = 10;

Для восстановления данных из созданной резервной копии требуется выполнить SQL-скрипт, представленный листингом 10.

Листинг 10 – Код для восстановления из резервной копии БД

RESTORE DATABASE School12 FROM

DISK = N'D:\Practika\School12.bak' --Путь

WITH FILE = 2, NOUNLOAD, STATS = 5;

# Выполнение работ по ПМ.01

МБОУ ОШ №12 осуществляет реализацию общеобразовательных программ среднего общего образования. В рамках учебного процесса учителя обязаны вести учёт успеваемости и посещаемости учащихся.

На данный момент подобная работа зачастую ведётся вручную либо в сторонних сервисах, не всегда подходящих под конкретные внутренние нужды учебного заведения. В связи с этим была поставлена задача разработки электронного дневника, обеспечивающего:

* ведение цифрового журнала оценок;
* контроль посещаемости;
* формирование отчётов по успеваемости учащихся;
* визуализацию данных об успеваемости.

Программное обеспечение предназначено для использования учителями учебного заведения и реализовано в виде настольного клиентского приложения на платформе Windows.

## Постановка задачи

Необходимо разработать однопользовательское приложение с графическим интерфейсом, обеспечивающее учителю возможность:

* входа в систему с использованием логина и пароля;
* отображения списка классов, учеников и предметов;
* внесения и редактирования оценок;
* расчёта средней оценки по предмету;
* формирования отчётов по отдельному ученику или всему классу;
* визуализации данных в виде графиков;
* сохранения отчётов в файл.

## Описание алгоритма функционирования системы

Система реализована как автономное клиентское приложение, взаимодействующее с локальной реляционной базой данных. Приложение построено по принципу ввода, обработки и вывода информации.

Действия пользователя:

1. запуск приложения;
2. ввод логина и пароля;
3. при успешной авторизации – переход на главное окно;
4. выбор класса и предмета;
5. просмотр или редактирование оценок;
6. формирование отчёта и его экспорт.

Действия, доступные пользователю приложения, отображены на диаграмме прецедентов, представленной на рисунке 4.

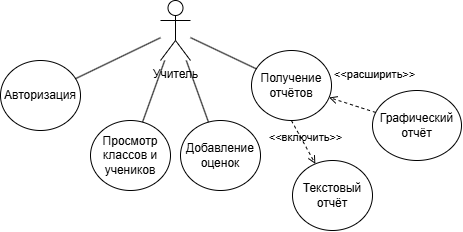


Рисунок 4 – Диаграмма прецедентов

В качестве языка разработки выбран C#, как современный и безопасный язык программирования, обеспечивающий удобную работу с данными и широкую поддержку GUI через WPF [5].

Используемые технологии:

* WPF – для реализации пользовательского интерфейса;
* Entity Framework Core [3] – для доступа к базе данных;
* LiveChartsCore [2] – для визуальных данных в виде графиков;
* SQL Server – в качестве реляционной БД;
* PBKDF2 – для хеширования паролей.

Среда разработки: Microsoft Visual Studio 2022.

Для решения задач приложения, в первую очередь, необходимо разработать графический интерфейс пользователя.

Используя средства WPF, требуется спроектировать интерфейс авторизации преподавателя в соответствии с рисунком 5.

Изображение выглядит как текст, снимок экрана, программное обеспечение, Шрифт

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 5 – TeacherCab. Вид окна авторизации

В качестве основной формы приложения выступает главное окно, предоставляющее учителю доступ к ключевому функционалу: выбору класса, предмета, расчёту средней успеваемости и генерации отчётов. Интерфейс главного окна представлен на рисунке 6.

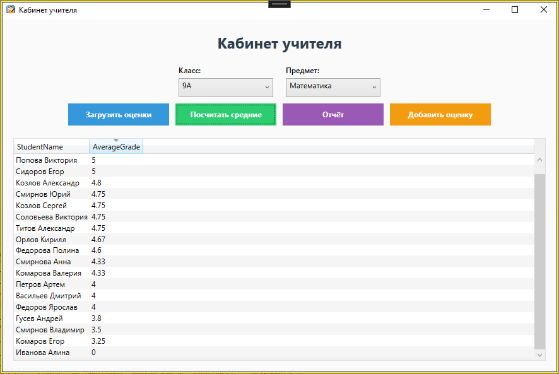


Рисунок 6 – TeacherCab. Вид главного окна

## Разработка мобильного приложения

Мобильная версия приложения TeacherCab будет разработана для обеспечения удобного и быстрого доступа учащихся к своей успеваемости.

Пользователи смогут авторизоваться в системе и просматривать сводную таблицу с оценками по предметам в режиме онлайн.

Приложение ориентировано на платформу Android и реализовано на языке Kotlin с использованием среды разработки Android Studio.

В процессе разработки применются современные технологии создания мобильных интерфейсов, что обеспечивает высокую производительность и стабильность.

В приложении реализовано два экрана:

* LoginScreen (рисунок 9). Окно авторизации ученика. Содержит поля для ввода логина и пароля, чекбокс «Запомнить меня» и кнопку входа. При авторизации данные передаются в виде параметров в следующий экран.
* GradesScreen (рисунок 10). Главный экран. После успешной авторизации отображается таблица с оценками.

Изображение выглядит как текст, снимок экрана, Шрифт, дизайн

Содержимое, созданное искусственным интеллектом, может быть неверным.

Рисунок 9 – Наша школа. Вид окна авторизации

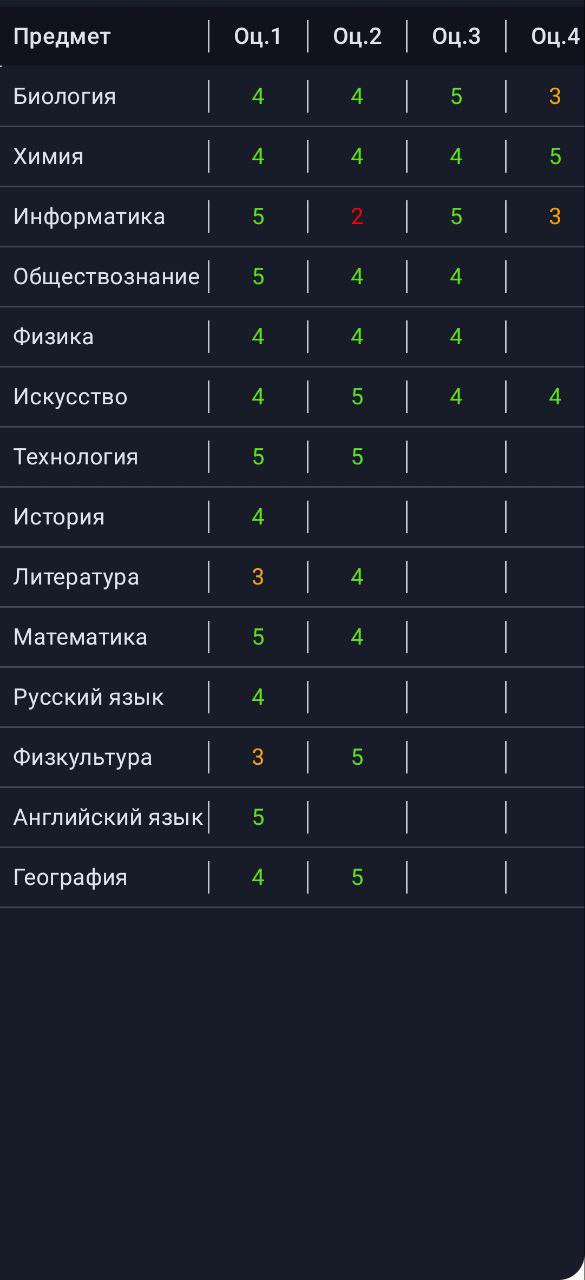


Рисунок 10 – Наша школа. Вид окна электронного журнала

Вёрстка выполнена на Jetpack Compose, таблица реализована через LazyColumn, с горизонтальной прокруткой (horizontalScroll) и адаптацией под количество оценок.

Для обеспечения стабильной и надёжной работы приложения проведено тестирование ключевых компонентов с целью выявления логических, функциональных и пользовательских ошибок.

В процессе тестирования применялось ручное тестирование методом «черного ящика».

Краткое описание действий, ожидаемых и фактических результатов представлено в таблице 2.

Таблица 2 – Набор тестовых приложений

|  |  |  |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Полученный результат** |
| Открыть окно авторизации. Ввести корректный логин и пароль учителя. Нажать кнопку «Войти». | Переход к окну «Кабинет учителя». Закрытие окна авторизации. | Соответствует ожидаемому |
| Ввести корректный логин, но неверный пароль. Нажать «Войти». | Вывод окна с сообщением: «Неверный пароль». | Соответствует ожидаемому |
| Нажать кнопку «Добавить оценку». Выбрать класс, предмет, ученика и оценку. Нажать «Сохранить». | Вывод окна с сообщением: «Оценка успешно добавлена!» | Соответствует ожидаемому |
| Перейти к окну формирования отчёта. Выбрать класс и ученика. Нажать «Сформировать отчёт». | Отображение отчёта в текстовом поле. Отображение графика успеваемости. | Соответствует ожидаемому |
| unit-test. Рассчёт средней оценки ученика (листинг 11). | Метод расчёта средней оценки возвращает корректное значение. | Соответствует ожидаемому |
| UI-test. Успешная авторизация с использованием корректных данных (листинг 12). | После ввода корректных логина и пароля открывается окно "Кабинет учителя". | Соответствует ожидаемому |

В ходе тестирования разработан unit-test с использованием фреймворка xUnit [4], библиотеки FlaUI 4.0.0 [1], а также встроенной базы InMemory для эмуляции SQL Server при тестировании.

Листинг 11 – Код unit-теста GetStudentAverageGrade\_Test()

//Тест на правильность вычисления среднего балла.

[Fact]

public void GetStudentAverageGrade\_Test()

{

var options = new DbContextOptionsBuilder<SchoolContext>()

.UseInMemoryDatabase("AverageTest")

.Options;

var studentId = 1;

var subjectId = 1;

using (var context = new SchoolContext(options))

{

// Добавляем ученика и предмет

context.Students.Add(new Student { StudentId = studentId, StudentFirstName = "Тест", StudentLastName = "Тестов", ClassId = 1, StudentLogin = "TESTOV\_T", StudentPassword = "123", StudentPasswordInvis = "123" });

context.Subjects.Add(new Subject { SubjectId = subjectId, SubjectName = "Информатика" });

context.Grades.AddRange(

new Grade { StudentId = studentId, SubjectId = subjectId, GradeValue = 5, Date = DateOnly.FromDateTime(DateTime.Today) },

new Grade { StudentId = studentId, SubjectId = subjectId, GradeValue = 3, Date = DateOnly.FromDateTime(DateTime.Today) }

);

context.SaveChanges();

}

using (var context = new SchoolContext(options))

{

var average = context.Grades

.Where(g => g.StudentId == studentId && g.SubjectId == subjectId)

.Average(g => g.GradeValue);

Assert.Equal(4.0, average);

}

}

Листинг 12 – Код FlaUI-теста для входа в систему

[Fact]

public void Login\_ShouldOpenMainWindow()

{

using var app = Application.Launch(@"ПУТЬ/ДО/ПРОЕКТА");

using var automation = new UIA3Automation();

var loginWindow = app.GetMainWindow(automation);

var loginBox = loginWindow.FindFirstDescendant(cf => cf.ByAutomationId("LoginTextBox"))?.AsTextBox();

var passBox = loginWindow.FindFirstDescendant(cf => cf.ByAutomationId("PasswordBox"))?.AsTextBox();

var loginButton = loginWindow.FindFirstDescendant(cf => cf.ByAutomationId("LoginButton"))?.AsButton();

Assert.NotNull(loginBox);

Assert.NotNull(passBox);

Assert.NotNull(loginButton);

loginBox.Enter("LOGIN");

passBox.Enter("PASSWORD");

loginButton.Invoke();

var result = Retry.WhileFalse(() =>

{

var windows = app.GetAllTopLevelWindows(automation);

return windows.Any(w => w.Title.Contains("Кабинет учителя"));

}, timeout: TimeSpan.FromSeconds(10));

Assert.True(result.Success);

app.Close();

}

## Рефакторинг и оптимизация программного кода

Во время разработки приложения «TeacherCab» проведено несколько этапов рефакторинга:

* избавление от повторяющихся запросов. Логика загрузки классов и предметов ранее дублировалась в нескольких окнах. В ходе рефакторинга были созданы отдельные методы LoadClasses() и LoadSubject(), которые теперь вызываются централизованно в соответствующих окнах.
* унификация проверки данных. Ранее каждый метод самостоятельно проверял поля на null. Впоследствии эти проверки вынесены в вспомогательные методы;
* улучшение наименования. Переменные и методы были переименованы согласно их назначению, например “button\_Click” стало “SaveGradeButton\_Click”, что упрощает поддержку кода.

Листинг 13– унификация проверки данных.

private bool ValidateLoginInput(string login, string password)

{

return !string.IsNullOrWhiteSpace(login) && !string.IsNullOrWhiteSpace(password); }

Некоторые операции взаимодействия с БД были переведены на асинхронные версии:

* загрузка данных в ComboBox;
* выполнение длительных вычислений перенесены в Task.Run.

Код представлен в листинге 14.

Листинг 14 – Оптимизация программного кода.

// Асинхронно загружает всех студентов из базы данных в список.

var students = await \_context.Students.ToListAsync();

// Считает средние значения в отдельном потоке, не блокируя

// основной.

var averages = await Task.Run(() => CalculateAverages());.

Заключение

В ходе производственной практики была успешно реализована многоплатформенная информационная система «TeacherCab», предназначенная для автоматизации учёта успеваемости и посещаемости учащихся.

Цель практики – получение практического опыта по выполнению профессиональных модулей ПМ.01 «Разработка программных модулей программного обеспечения» и ПМ.11 «Разработка, администрирование и защита баз данных», а также развитие профессиональных и общих компетенций – достигнута.

Также выполнены следующие задачи:

* разработка алгоритма решения и создание схемы с использованием специализированного ПО;
* описание работы модуля с помощью языка UML;
* реализация программного модуля и графического интерфейса по готовой спецификации на высокоуровневом языке программирования;
* разработка мобильного приложения;
* использование инструментальных средств для отладки и проведения модульного и автоматизированного тестирования по заданному сценарию;
* рефакторинг и оптимизация программного кода;
* проектирование базы данных: построение концептуальной, логической и физической моделей, формирование структуры, создание индексов, представлений, хранимых процедур и триггеров;
* настройка СУБД: установка SQL-сервера, управление пользователями, ролями и привилегиями, реализация защиты данных, выполнение резервного копирования и восстановления БД;
* создание и выполнение SQL-запросов, наполнение БД данными.
* таким образом, все поставленные задачи выполнены, а система готова к развёртыванию и практическому применению в учебном процессе.

Список использованных источников

1. FlaUI : [сайт] / FlaUI Community. – Берлин, 2025. – URL: <https://github.com/FlaUI/FlaUI/wiki> (дата обращения: 23.05.2025). – Текст : электронный.
2. LiveCharts : [сайт] / разработчик Beto Rodriguez. – Мехико, 2025. – URL: <https://v0.lvcharts.com/> (дата обращения: 20.05.2025). – Текст : электронный.
3. Entity Framework Core. Текст : электронный // Microsoft Learn : [сайт] – 2024. URL: <https://learn.microsoft.com/en-us/ef/core/> (дата обращения: 01.05.2025).
4. Начало работы с модульным тестированием. – Текст : электронный // Microsoft Learn : [сайт] – 2025. – URL: <https://learn.microsoft.com/en-us/dotnet/core/testing/unit-testing-with-dotnet-test> (дата обращения: 07.05.2025).
5. Документация по Windows Presentation Foundation. Текст : электронный // Microsoft Learn : [сайт] – 2025. – URL: <https://learn.microsoft.com/en-us/dotnet/desktop/wpf/?view=netdesktop-8.0> (дата обращения: 07.05.2025).
6. Шустова, Л. И. Базы данных: учебник / Л. И. Шустова, О. В. Тараканов. – Москва : ИНФРА-М, 2024. – 304 с. – Текст : электронный. – URL: <https://znanium.ru/catalog/product/2149043> (дата обращения: 19.05.2025). – Режим доступа: по подписке.