**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

\_\_\_\_\_\_\_\_\_\_\_ В. Р. Оганян

(Подпись) (И.О. Фамилия)

«31» мая 2025г.

**ТЕХНИЧЕСКИЙ ОТЧЁТ**

**по ПМ.01, ПМ.11**

|  |
| --- |
| ООО «СЕВДОРСТРОЙСЕРВИС» |
| Информационные системы и программирование |
| 09.02.07. 25ТО01. 013 ПЗ |
| (Обозначение документа) |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Студент | ИСПП-21 | |  | 30.05.2025 | М.А. Кореев |
|  | (Группа) | | (Подпись) | (Дата) | (И.О. Фамилия) |
| Рук. практики от предприятия | | |  | 30.05.2025 | Н.А. Зайков |
|  | |  | (Подпись) | (Дата) | (И.О. Фамилия) |

Архангельск 2025

Содержание

[Перечень сокращений и обозначений 3](#_Toc201185421)

[Введение 4](#_Toc201185422)

[1 Охрана труда и техника безопасности при работе на ПК 6](#_Toc201185423)

[2 Выполнение работ по ПМ.11 8](#_Toc201185424)

[2.1 Проектирование базы данных 8](#_Toc201185425)

[2.2 Разработка базы данных и объектов базы данных 10](#_Toc201185426)

[2.3 Администрирование и защита базы данных 13](#_Toc201185427)

[3 Выполнение работ по ПМ.01 16](#_Toc201185428)

[3.1 Проектирование программного обеспечения 16](#_Toc201185429)

[3.2 Разработка программных модулей 17](#_Toc201185430)

[3.3 Отладка и тестирование программных модулей 19](#_Toc201185431)

[3.4 Оптимизация и рефакторинг программного кода 22](#_Toc201185432)

[Заключение 24](#_Toc201185433)

[Список использованных источников 25](#_Toc201185434)

Перечень сокращений и обозначений

В настоящем техническом отчете применяются следующие сокращения и обозначения:

БД – база данных

ПК – персональный компьютер

ПМ – профессиональный модуль

ПО – программное обеспечение

СУБД – система управления базами данных

DDL – язык описания данных

ERD – диаграмма «сущность-связь»

ER-модель – модель «сущность-связь»

FK – внешний ключ

MVVM - Model-View-ViewModel

PK – первичный ключ

SQL – язык структурированных запросов

UI - пользовательский интерфейс

UML – унифицированный язык моделирования

WPF – Windows Presentation Foundation

XAML - eXtensible Application Markup Language

Введение

Базой производственной практики является ООО «Севдорстройсервис». Организация специализируется на строительстве и обслуживании дорожной инфраструктуры в Северо-Западном регионе. Основные направления деятельности включают:

* строительство и ремонт автомобильных дорог, мостов и путепроводов;
* устройство дорожных покрытий различных типов (асфальт, бетон, щебень);
* организация и обеспечение безопасности дорожного движения (установка дорожных знаков, разметка);

Деятельность ООО «Севдорстройсервис» направлена на повышение качества дорожной сети, обеспечение комфортного и безопасного передвижения транспорта, а также на поддержку экономического развития региона.

Целями производственной практики являются:

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

Задачами производственной практики являются:

* построение и обоснование концептуальной модели БД;
* проектирование логической и физической схемы БД;
* работа с современными CASE-средствами проектирования БД;
* создание хранимых процедур и триггеров БД;
* установка и настройка SQL-сервера;
* управление пользователями и ролями в конкретной СУБД;
* осуществление контроля доступа к данным и управления привилегиями;
* использование стандартных методов защиты объектов БД;
* разработка алгоритма поставленной задачи и создание схемы алгоритма с использованием специализированного программного обеспечения;
* описание работы программного модуля с помощью языка моделирования UML с использованием специализированного программного обеспечения;
* разработка кода программного продукта на основе готовой спецификации на уровне модуля;
* проведение тестирования программного модуля по определенному сценарию;
* разработка модульных тестов и выполнение автоматизированного тестирования программного модуля;
* рефакторинг программного кода;
* выявление «узких» мест алгоритма и оптимизация разработанного программного модуля;
* разработка мобильного приложения.

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

* процессор: Intel Core i3-10100F;
* системная плата: GIGABYTE H510M S2H V2;
* видеокарта: Palit GeForce GTX 1050 Ti 4 ГБ;
* оперативная память – 16ГБ;
* операционная система: Microsoft Windows 10;
* прикладное ПО: пакет Microsoft Office, Google Chrome, Visual Studio

2022, Microsoft SQL Server Management Studio 21, draw.io.

# Охрана труда и техника безопасности при работе на ПК

К работе на персональном компьютере допускаются лица, прошедшие обучение безопасным методам труда, вводный инструктаж, первичный инструктаж на рабочем месте.

При эксплуатации персонального компьютера на работника могут оказывать действие следующие опасные и вредные производственные факторы:

* повышенный уровень электромагнитных излучений;
* повышенный уровень статического электричества;
* пониженная ионизация воздуха;
* статические физические перегрузки;
* перенапряжение зрительных анализаторов.

Работник обязан:

* выполнять только ту работу, которая определена его должностной инструкцией;
* содержать в чистоте рабочее место;
* соблюдать режим труда и отдыха;
* соблюдать меры пожарной безопасности.

Рабочая мебель для пользователей компьютерной техникой должна отвечать следующим требованиям:

* высота рабочей поверхности стола должна регулироваться в пределах 800 мм;
* рабочий стол должен иметь пространство для ног высотой не менее 600 мм, глубиной на уровне колен не менее 450 мм и на уровне вытянутых ног не менее 650 мм;
* рабочий стул должен быть подъемно-поворотным и регулируемым по высоте и углам наклона сиденья и спинки, а также – расстоянию спинки от переднего края сиденья;
* рабочее место должно быть оборудовано подставкой для ног, имеющей ширину не менее 300 мм, глубину не менее 400 мм, регулировку по высоте в пределах до 150 мм и по углу наклона опорной поверхности подставки до 20 градусов; поверхность подставки должна быть рифленой и иметь по переднему краю бортик высотой 10 мм;
* рабочее место с персональным компьютером должно быть оснащено легко перемещаемым пюпитром для документов.

Для нормализации аэроионного фактора помещений с компьютерами необходимо использовать устройства автоматического регулирования ионного режима воздушной среды.

# Выполнение работ по ПМ.11

## Проектирование базы данных

Для разработки базы данных, предназначенной для эффективного хранения, поиска и защиты данных о дорожных материалах, система должна обеспечивать учет материалов, их типов, принадлежность к дорогам и разграничение доступа.

Каждый материал имеет свой уникальный номер, тип, толщину, плотность и стоимость за килограмм. Список дорог содержит их длину и ширину. На рисунке 1 показана концептуальная модель предметной области в виде ERD, созданная с помощью средства проектирования Draw.io.



Рисунок 1 – Концептуальная модель

СУБД Microsoft SQL Server, выбранная для хранения данных, является реляционной, поэтому требуется преобразовать разработанную концептуальную модель в логическую с учетом правил преобразования ER модели в реляционную модель данных:

* связь между сущностями Дорога и Расчёт 1:М, и является обязательной для сущности Расчёт, поэтому формируется два отношения с внешним ключом в отношении Расчет;
* связь между сущностями Материал и Расчёт 1:М, и является обязательной для сущности Расчёт, поэтому формируется два отношения с внешним ключом в отношении Материал

На рисунке 2 показана логическая модель предметной области, полученная путем преобразования концептуальной модели и созданная с помощью средства проектирования Draw.io.

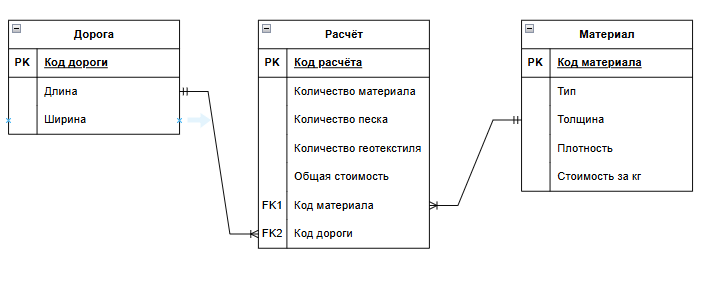


Рисунок 2 – Логическая модель

Согласно разработанной схеме БД, все значения в атрибутах являются атомарными, все значения функционально зависят от первичного ключа и отсутствуют транзитивные зависимости, что соответствует требованиям третьей нормальной формы [2].

На рисунке 3 показана физическая модель предметной области, разработанная на основе логической модели для СУБД Microsoft SQL Server Express 2021.

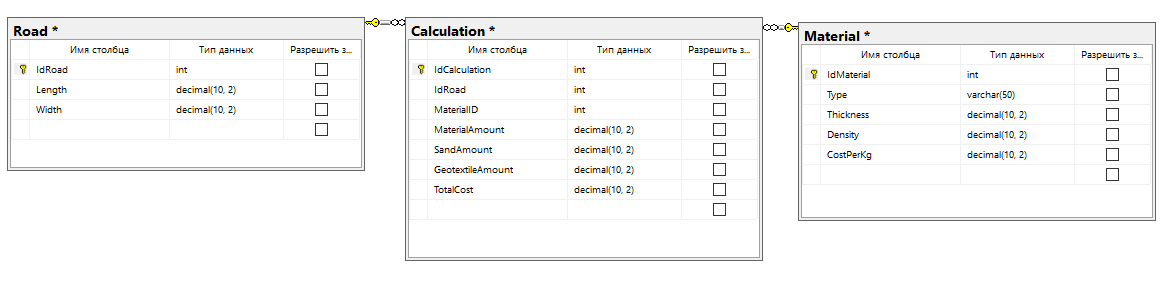


Рисунок 3 – Физическая модель

## 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** |
| Road | | | | |
| PK | IdRoad | int | + | Автоинкрементный |
|  | Length | decimal(10, 2) | + |  |
|  | Width | decimal(10, 2) | + |  |
| Material | | | | |
| PK | IdMaterial | int | + | Автоинкрементный |
|  | Type | varchar(50) | + | Значение по умолчанию: ‘ShPS’ |
|  | Thickness | decimal(10, 2) | + |  |
|  | Density | decimal(10, 2) | + |  |
|  | CostPerKg | decimal(10, 2) | + |  |
| Calculation | | | | |
| PK | IdCalculation | int | + | Автоинкрементный |
| FK | IdRoad | int | + |  |
| FK | IdMaterial | int | + |  |
|  | MaterialAmount | decimal(10, 2) | + |  |
|  | SandAmount | decimal(10, 2) | + |  |
|  | GeotextileAmount | decimal(10, 2) | + |  |
|  | TotalCost | decimal(10, 2) | + | Значение по умолчанию: 0.00 |

Для создания таблиц БД и ограничений целостности требуется выполнить соответствующие DDL команды. Код создания таблиц Road, Material, Calculation представлен листингом 1.

Листинг 1 – SQL-запрос для создания таблиц

CREATE TABLE Road (

IdRoad INT PRIMARY KEY IDENTITY(1,1),

Length DECIMAL(10, 2) NOT NULL,

Width DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Material (

IdMaterial INT PRIMARY KEY IDENTITY(1,1),

Type VARCHAR(50) NOT NULL,

Thickness DECIMAL(10, 2) NOT NULL,

Density DECIMAL(10, 2) NOT NULL,

CostPerKg DECIMAL(10, 2) NOT NULL

);

CREATE TABLE Calculation (

IdCalculation INT PRIMARY KEY IDENTITY(1,1),

IdRoad INT NOT NULL,

IdMaterial INT NOT NULL,

MaterialAmount DECIMAL(10, 2) NOT NULL,

SandAmount DECIMAL(10, 2) NOT NULL,

GeotextileAmount DECIMAL(10, 2) NOT NULL,

TotalCost DECIMAL(10, 2) NOT NULL,

FOREIGN KEY (IdRoad) REFERENCES Road(IdRoad),

FOREIGN KEY (IdMaterial) REFERENCES Material(IdMaterial)

);

Для решения поставленных задач в БД требуется создать представление vCalculationDetails, упрощающее доступ к данным о расчетах, включая информацию о дорогах и материалах. Код создания представления показан в листинге 2.

Листинг 2 – Код создания представления vCalculationDetails

CREATE VIEW [dbo].[vCalculationDetails] AS

-- Создаем представление для объединения данных из таблиц Calculation, Road и Material

Calculation, Road и Material

SELECT

c.IdCalculation, -- Идентификатор расчета

r.IdRoad, -- Идентификатор дороги

r.Length, -- Длина дороги

r.Width, -- Ширина дороги

m.IdMaterial, -- Идентификатор материала

m.Type, -- Тип материала

m.Thickness, -- Толщина материала

c.MaterialAmount, -- Количество материала

c.SandAmount, -- Количество песка

c.GeotextileAmount, -- Количество геотекстиля

c.TotalCost -- Общая стоимость

FROM

Calculation c

JOIN

Road r ON c.IdRoad = r.IdRoad -- Соединяем таблицу Calculation с таблицей Road по IdRoad

JOIN

Material m ON c.IdMaterial = m.IdMaterial; -- Соединяем таблицу Calculation с таблицей Material по IdMaterial;

Для решения поставленных задач в БД требуется создать хранимую процедуру AddCalculation, позволяющая добавить новый расчет в таблицу Calculation (код создания представлен листингом 3).

Листинг 3 – Код создания хранимой процедуры AddCalculation

CREATE PROCEDURE [dbo].[AddCalculation]

@IdRoad INT,

@IdMaterial INT,

@MaterialAmount DECIMAL(10, 2),

@SandAmount DECIMAL(10, 2),

@GeotextileAmount DECIMAL(10, 2)

AS

BEGIN

-- Объявляем переменную для хранения стоимости материала за килограмм

DECLARE @CostPerKg DECIMAL(10, 2);

-- Получаем стоимость материала за килограмм из таблицы Material по IdMaterial

SELECT @CostPerKg = CostPerKg FROM Material WHERE IdMaterial = @IdMaterial;

-- Объявляем переменную для хранения общей стоимости

DECLARE @TotalCost DECIMAL(10, 2);

-- Рассчитываем общую стоимость на основе количества материалов и их стоимости

SET @TotalCost = @CostPerKg \* @MaterialAmount +

10 \* @SandAmount +

2 \* @GeotextileAmount;

-- Вставляем запись в таблицу Calculation с рассчитанными значениями

INSERT INTO Calculation (IdRoad, IdMaterial, MaterialAmount, SandAmount, GeotextileAmount, TotalCost)

VALUES (@IdRoad, @IdMaterial, @MaterialAmount, @SandAmount, @GeotextileAmount, @TotalCost);

END;

Для решения поставленных задач в БД требуется создать триггер trg\_UpdateTotalCost, выполняющийся при изменении данных в таблице TotalCost. Код создания триггера представлен листингом 4.

Листинг 4 – Код создания триггера trg\_UpdateTotalCost

CREATE TRIGGER [dbo].[trg\_UpdateTotalCost]

ON Calculation

AFTER INSERT, UPDATE

AS

BEGIN

-- Отключаем отправку сообщений о количестве обработанных строк клиенту

SET NOCOUNT ON;

-- Обновляем поле TotalCost в таблице Calculation на основе данных из вставленной или обновленной записи

UPDATE c

SET TotalCost = (

-- Получаем стоимость за килограмм материала из таблицы Material

(SELECT CostPerKg FROM Material WHERE IdMaterial = c.IdMaterial) \* MaterialAmount +

-- Добавляем стоимость песка (10 за единицу)

10 \* SandAmount +

-- Добавляем стоимость геотекстиля (2 за единицу)

2 \* GeotextileAmount

)

-- Обновляем записи в таблице Calculation, которые соответствуют вставленным или обновленным записям

FROM Calculation c

INNER JOIN inserted i ON c.IdCalculation = i.IdCalculation;

END

## 2.3 Администрирование и защита базы данных

Для защиты информации в базе данных необходимо создать учетную запись, пользователя и назначить встроенные роли, выполнив набор SQL-команд, представленный листингом 5.

Листинг 5 – Код создания пользователей и назначения ролей

-- Создание учетной записи на уровне SQL сервера с именем пользователя и паролем

CREATE LOGIN road\_user WITH PASSWORD = 'StrongPassword!';

-- Создание пользователя в текущей базе данных и связывание его с созданной учетной записью

CREATE USER road\_user FOR LOGIN road\_user;

-- Предоставление пользователю road\_user стандартной роли для чтения данных

EXEC sp\_addrolemember 'db\_datareader', 'road\_user';

-- Предоставление пользователю road\_user стандартной роли для записи данных

EXEC sp\_addrolemember 'db\_datawriter', 'road\_user';

-- Создание новой роли в базе данных с именем road\_admin

CREATE ROLE road\_admin;

-- Назначение прав на выполнение операций с таблицей Calculation для роли road\_admin

GRANT SELECT, INSERT, UPDATE, DELETE ON Calculation TO road\_admin;

Чтобы избежать потери данных, необходимо периодически выполнять полное резервное копирование базы данных. Резервные копии позволяют восстановить данные после сбоев и других непредвиденных ситуаций. Для выполнения резервного копирования базы данных необходимо выполнить SQL-скрипт, представленный листингом 6.

Листинг 6 – Код создания полной резервной копии базы данных

-- Создание полной резервной копии базы данных RoadConstructionCalculator

BACKUP DATABASE RoadConstructionCalculator

TO DISK = 'C:\Backup\RoadConstructionCalculator.bak'

WITH FORMAT, -- Перезапись существующего резервного файла

INIT, -- Инициализация нового резервного набора данных

NAME = 'Full Backup of RoadConstructionCalculator', -- Название резервной копии

SKIP, -- Пропуск проверки на существование файла

STATS = 5; -- Отображение статуса выполнения каждые 5%

Для восстановления данных из созданной резервной копии требуется выполнить SQL-скрипт, представленный листингом 7.

Листинг 7 – Код восстановления базы данных

-- Восстановление базы данных RoadConstructionCalculator из резервной копии

RESTORE DATABASE RoadConstructionCalculator

FROM DISK = 'C:\Backup\RoadConstructionCalculator.bak'

WITH REPLACE, -- Замена существующей базы данных

RECOVERY, -- Завершение восстановления и открытие базы данных для использования

STATS = 5; -- Отображение статуса выполнения каждые 5%

# Выполнение работ по ПМ.01

## Проектирование программного обеспечения

Программное обеспечение, предназначено для расчета материалов, необходимых для строительства дорог. В связи с этим поставлена задача по разработке системы для автоматизации расчетов и управления данными о дорогах и материалах.

ПО должно обеспечивать выполнение следующих задач:

* ввод и хранение данных о дорогах, включая длину и ширину;
* ввод и хранение данных о материалах, включая тип, толщину, плотность и стоимость;
* проведение расчетов необходимых материалов для строительства дорог;
* оценка общей стоимости строительства на основе введенных данных.

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

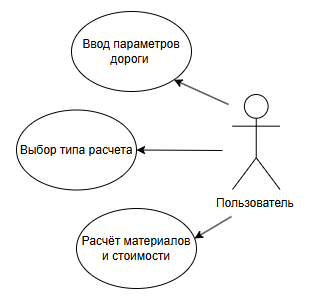


Рисунок 4 - Диаграмма прецедентов

На рисунке 5 изображена схема, которая выполнена в соответствии с ГОСТ 19.701 ЕСПД [1].

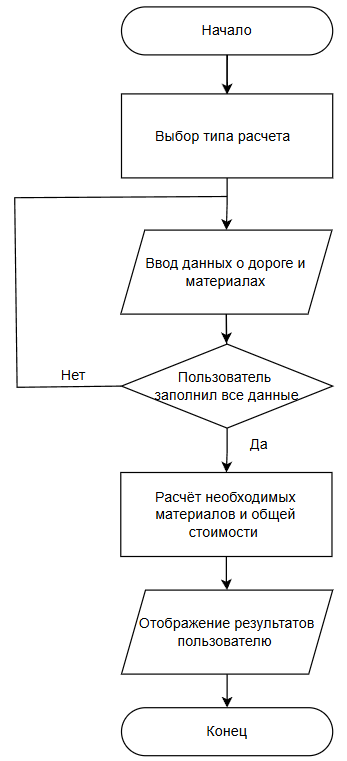


Рисунок 5 - Схема алгоритма

## Разработка программных модулей

Для разработки программного обеспечения используются следующие технологии:

* язык программирования: C# с использованием .NET 8.0;
* фреймворк UI: Avalonia UI – кроссплатформенный фреймворк для создания настольных и мобильных приложений;
* паттерн проектирования: MVVM;
* библиотека для реактивного программирования: CommunityToolkit.

Метод CalculateMaterials является ViewModel в паттерне MVVM и выполняет расчет необходимых материалов для строительства дорог на основе введенных пользователем данных [4]. Он использует параметры, такие как длина и ширина дороги, а также толщина материалов, для вычисления объема необходимых материалов, код класса представлен листингом 8.

Листинг 8 – Метод CalculateMaterials

private void CalculateMaterials()

{

// Рассчитываем площадь дороги и объемы материалов

double roadArea = Length \* Width;

double surfaceMaterialVolume = roadArea \* surfaceMaterialThicknessM;

double sandVolume = roadArea \* sandThicknessM;

// Конвертируем толщину поверхностного материала из сантиметров в метры

double surfaceMaterialThicknessM = SurfaceMaterialThicknessCm / 100;

// Конвертируем толщину песчаной подложки из сантиметров в метры

double sandThicknessM = SandThicknessCm / 100;

// Получаем коэффициент материала

double materialCoefficient = GetMaterialCoefficient(SelectedMaterialType);

// Рассчитываем необходимое количество материалов

RequiredMaterialAmount = surfaceMaterialVolume \* MaterialDensity \* materialCoefficient / 1000;

RequiredSandAmount = sandVolume \* SandDensity / 1000;

RequiredGeotextileAmount = Length \* (Width + 2 \* GeotextileThickness);

}

UI реализован в XAML (MainView.axaml) с использованием привязок к ViewModel.

Вид главного окна программы представлен на рисунке 6.

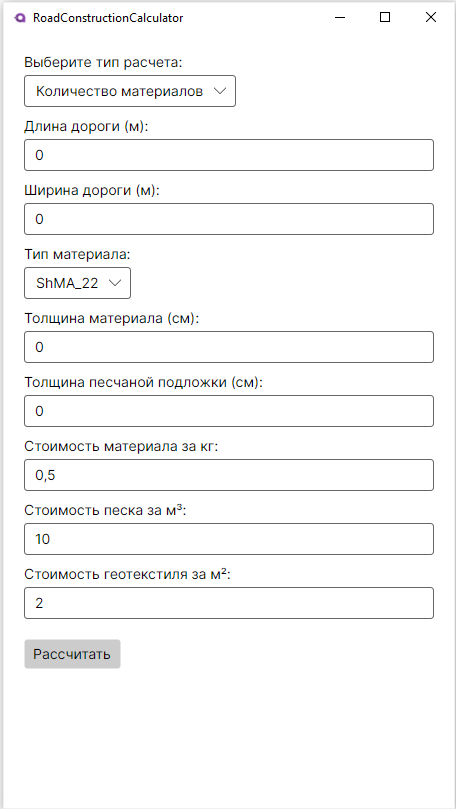


Рисунок 6 – RoadConstructionCalculator. Вид главной формы

## Отладка и тестирование программных модулей

В процессе разработки приложения были использованы различные средства отладки, доступные в среде разработки Visual Studio 2022. В частности, применялись точки останова и контрольные значения, что позволило эффективно выявлять и устранять ошибки в коде. На рисунке 7 представлены точки останова, установленные в ключевых местах кода для контроля выполнения программы.

Кроме того, в ходе тестирования было проведено тестирование методом "черного ящика" для проверки функциональности приложения. Результаты этого тестирования представлены в таблице 2, где описаны различные сценарии использования приложения и их результаты.

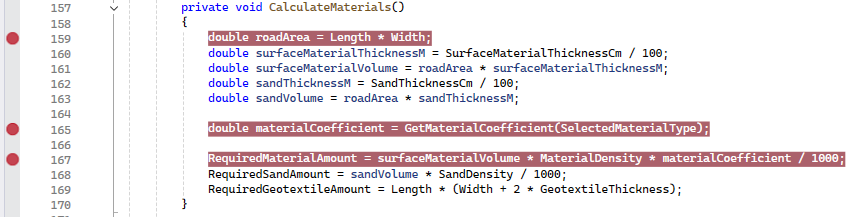


Рисунок 7 – VisualStudio 2022.Фрагмент кода с точками остановки

Таблица 2 – Набор тестов приложения

|  |  |  |
| --- | --- | --- |
| **Действие** | **Ожидаемый результат** | **Полученный результат** |
| Открыть приложение, ввести корректные параметры дороги и материалов, нажать "Рассчитать" | Отображаются рассчитанные значения материалов и общая стоимость | Соответствует ожидаемому |
| Открыть приложение, оставить поля пустыми, нажать "Рассчитать" | Приложение не выполняет расчет, отображает сообщение об ошибке о незаполненных полях | Соответствует ожидаемому |
| Открыть приложение, ввести корректные параметры дороги и материалов, выбрать "Количество материалов", нажать "Рассчитать" | Отображаются рассчитанные значения материалов | Соответствует ожидаемому |

Для проверки корректности выполнения расчетов при вводе валидных данных было проведено автоматизированное тестирование метода расчета материалов. Этот тест представлен в листинге 9 и демонстрирует работу с корректными входными параметрами.

Листинг 9 – Код автоматизированного теста для метода расчета материалов

public class RoadConstructionViewModelTests

{

[Fact]

Public void CalculateMaterials\_WithValidInputs\_CalculatesCorrectly()

{

// Подготовка: создаем экземпляр ViewModel и задаем начальные значения

var viewModel = new RoadConstructionViewModel

{

Length = 100, // Длина дороги в метрах

Width = 5, // Ширина дороги в метрах

SurfaceMaterialThicknessCm = 10, // Толщина материала в сантиметрах

SandThicknessCm = 5, // Толщина песчаной подложки в сантиметрах

SelectedMaterialType = RoadMaterialType.ShMA\_22 // Тип материала

};

// Действие: вызываем метод расчета материалов

viewModel.CalculateMaterials();

// Проверка: убеждаемся, что рассчитанные значения не равны нулю и соответствуют ожидаемым значениям

Assert.NotEqual(0, viewModel.RequiredMaterialAmount);

Assert.NotEqual(0, viewModel.RequiredSandAmount);

Assert.NotEqual(0, viewModel.RequiredGeotextileAmount);

// Примерные ожидаемые значения, которые могут быть уточнены на основе точных расчетов

Assert.Equal(61, Math.Round(viewModel.RequiredMaterialAmount, 0));

Assert.Equal(75, Math.Round(viewModel.RequiredSandAmount, 0));

Assert.Equal(502, Math.Round(viewModel.RequiredGeotextileAmount, 0));

}

};

## Оптимизация и рефакторинг программного кода

Для улучшения читаемости и поддержки кода, константы были вынесены в начало класса., представленный листингом 10.

Листинг 10 – Организация констант

private const double GeotextileThickness = 0.002;

private const double SandDensity = 1500;

private const double MaterialDensity = 2400;

Для улучшения читаемости и компактности кода использованы switch выражения, код которого представлен листингом 11.

Листинг 11 – Использование switch выражения

private string ConvertCost(double cost)

{

return cost switch

{

var \_ when cost >= 1000000 => $"{cost / 1000000:F2} млн руб",

var \_ when cost >= 1000 => $"{cost / 1000:F2} тыс руб",

\_ => $"{cost:F2} руб"

};

}

В ходе первичной отладки приложения проведено тестирование скорости выполнения программного кода средствами Microsoft Visual Studio. Тестирование показало, что потребуется оптимизация скорости кода [3].

Для создания экземпляра клиента в БД используется метод, представленный листингом 12.

Листинг 12 – Код метода CalculateMaterials

private void CalculateMaterials()

{

double roadArea = Length \* Width;

double surfaceMaterialThicknessM = SurfaceMaterialThicknessCm / 100;

double surfaceMaterialVolume = roadArea \* surfaceMaterialThicknessM;

double sandThicknessM = SandThicknessCm / 100;

double sandVolume = roadArea \* sandThicknessM;

double materialCoefficient = GetMaterialCoefficient(SelectedMaterialType);

RequiredMaterialAmount = surfaceMaterialVolume \* MaterialDensity \* materialCoefficient / 1000;

RequiredSandAmount = sandVolume \* SandDensity / 1000;

RequiredGeotextileAmount = Length \* (Width + 2 \* GeotextileThickness);

}

Для оптимизации данного метода использованы улучшенные алгоритмы расчета, код которого представлен листингом 13.

Листинг 13 – Оптимизированный код метода AddClient

private void CalculateMaterials()

{

double roadArea = Length \* Width;

double surfaceMaterialThicknessM = SurfaceMaterialThicknessCm \* 0.01;

double sandThicknessM = SandThicknessCm \* 0.01;

double materialCoefficient = GetMaterialCoefficient(SelectedMaterialType);

RequiredMaterialAmount = roadArea \* surfaceMaterialThicknessM \* MaterialDensity \* materialCoefficient \* 0.001;

RequiredSandAmount = roadArea \* sandThicknessM \* SandDensity \* 0.001;

RequiredGeotextileAmount = Length \* (Width + 2 \* GeotextileThickness);

}

Заключение

Прохождение практики является неотъемлемой частью процесса подготовки будущего специалиста. Практика дает возможность применить полученные теоретические знания в реальных условиях профессиональной деятельности, формирует трудовую дисциплинированность и ответственность.

В ходе производственной практики были изучены характеристика, организационная структура и направления деятельности ООО «Севдорстройсервис».

На базе производственной практики было предоставлено рабочее место с персональным компьютером и всем необходимым для работы аппаратным и программным обеспечением.

Цели производственной практики достигнуты:

* получен практический опыт по выполнению работ по ПМ.11 «Разработка, администрирование и защита баз данных» и развиты общие и профессиональные компетенции;
* получен практический опыт по выполнению работ по ПМ.01 «Разработка модулей программного обеспечения для компьютерных систем» и развиты общие и профессиональные компетенции.

Для достижения целей производственной практики выполнены все поставленные задачи.

В ходе прохождения производственной практики были успешно выполнены все поставленные задачи. Эта практика стала ценным опытом, который способствовал развитию моих профессиональных навыков и самостоятельности. За время практики я не только закрепил теоретические знания, но и узнал много нового и интересного о своей будущей профессии. Полученные знания и навыки помогли мне лучше понять свои профессиональные интересы и укрепить уверенность в выборе карьерного пути.

Список использованных источников

1. ГОСТ 19.701–90 ЕСПД. СХЕМЫ АЛГОРИТМОВ, ПРОГРАММ, ДАННЫХ И СИСТЕМ: дата введения 1992-01-01. – URL: <https://docs.cntd.ru/document/9041994> (дата обращения: 23.04.2025). – Текст: электронный.
2. Дадян, Э. Г. Данные: хранение и обработка : учебник / Э. Г. Дадян. – Москва : ИНФРА-М, 2020. – 205 с. – URL: <https://thelib.net/3347467-dannye-hranenie-i-obrabotka.html> (дата обращения: 29.04.2025). – Текст : электронный.
3. Мартин Роберт. Чистый код: создание, анализ и рефакторинг. Библиотека программиста. – Санкт-Петербург : Питер, 2018. – 464 с. – URL: https://ibooks.ru/bookshelf/377265/reading (дата обращения: 07.05.2025) – Режим доступа: открытый. – Текст : электронный.
4. Павловская, Т. А. C#. Программирование на языке высокого уровня: Учебник для вузов / Т. А. Павловская. – Санкт-Петербург : Питер, 2021. – 432 с. – URL: <https://ibooks.ru/bookshelf/377952/reading> (дата обращения: 05.05.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный
5. Фленов, М. Е. Библия C# / М. Е. Фленов. – 4-е изд., перераб. и доп. – Санкт-Петербург : БХВ-Петербург, 2019. – 512 с. – URL: <https://ibooks.ru/bookshelf/366634/reading> (дата обращения: 27.05.2025). – Режим доступа: для зарегистрир. пользователей. – Текст : электронный.