**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](#_Toc194696990)

[Введение 4](#_Toc194696991)

[1 Охрана труда и техника безопасности при работе на ПК 6](#_Toc194696992)

[2 Выполнение работ по ПМ.11 8](#_Toc194696993)

[2.1 Проектирование базы данных 8](#_Toc194696994)

[2.2 Разработка базы данных и объектов базы данных 10](#_Toc194696995)

[2.3 Администрирование и защита базы данных 14](#_Toc194696996)

[3 Выполнение работ по ПМ.01 16](#_Toc194696997)

[3.1 Проектирование программного обеспечения 16](#_Toc194696998)

[3.2 Разработка программных модулей 17](#_Toc194696999)

[3.3 Отладка и тестирование программных модулей 25](#_Toc194697001)

[3.4 Оптимизация и рефакторинг программного кода 26](#_Toc194697002)

[Заключение 27](#_Toc194697003)

[Список использованных источников 28](#_Toc194697004)

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 - User Interface

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" представляет собой отношение "один ко многим" (1:М), где каждая дорога может иметь множество связанных с ней расчетов, что является обязательным для сущности "Расчёт", так как каждый расчет должен быть привязан к конкретной дороге. Для реализации этой связи в таблицу "Расчёт" добавляется внешний ключ IdRoad, который ссылается на первичный ключ IdRoad в таблице "Дорога", обеспечивая тем самым целостность данных и правильное отражение связей между дорогами и их расчетами.,

- Связь между сущностями "Материал" и "Расчёт" также является отношением "один ко многим" (1:М), где один материал может быть использован в нескольких расчетах, что обязательно для сущности "Расчёт", поскольку каждый расчет должен быть связан с конкретным материалом. В таблице "Расчёт" используется внешний ключ IdMaterial, который ссылается на первичный ключ IdMaterial в таблице "Материал", что позволяет поддерживать целостность данных и обеспечивает корректное выполнение расчетов с использованием определенных материалов.

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

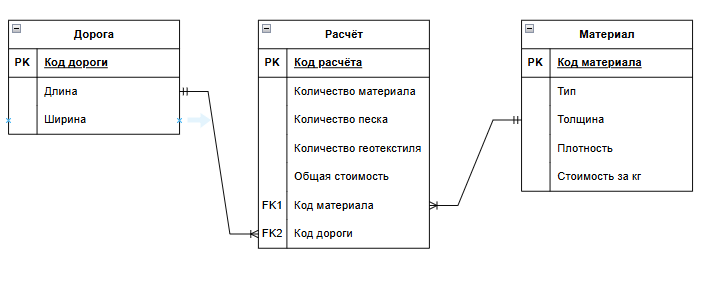


Рисунок 2 – Логическая модель

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

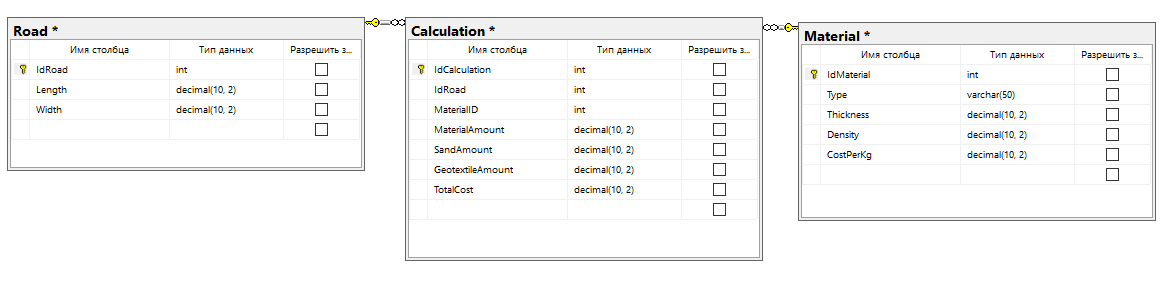


Рисунок 3 – Физическая модель

## 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) | + |  |
|  | Thickness | decimal(10, 2) | + |  |
|  | Density | decimal(10, 2) | + |  |
|  | CostPerKg | decimal(10, 2) | + |  |
| Calculation | | | | |
| PK | IdCalculation | int | + | Автоинкрементный |
| FK | IdRoad | int | + | Автоинкрементный |
| FK | IdMaterial | int | + | Автоинкрементный |
|  | MaterialAmount | decimal(10, 2) | + |  |
|  | SandAmount | decimal(10, 2) | + |  |
|  | GeotextileAmount | decimal(10, 2) | + |  |
|  | TotalCost | decimal(10, 2) | + |  |

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

SELECT

c.IdCalculation,

r.IdRoad,

r.Length,

r.Width,

m.IdMaterial,

m.Type,

m.Thickness,

c.MaterialAmount,

c.SandAmount,

c.GeotextileAmount,

c.TotalCost

FROM

Calculation c

JOIN

Road r ON c.IdRoad = r.IdRoad

JOIN

Material m ON c.IdMaterial = m.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

DECLARE @CostPerKg DECIMAL(10, 2);

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

DECLARE @TotalCost DECIMAL(10, 2);

SET @TotalCost = @CostPerKg \* @MaterialAmount +

10 \* @SandAmount +

2 \* @GeotextileAmount

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

SET NOCOUNT ON;

UPDATE c

SET TotalCost = (

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

10 \* SandAmount +

2 \* GeotextileAmount

)

FROM Calculation c

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

END

## 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 пользователей и ролей

-- 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 создания пользователей и ролей

-- 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.

Схема алгоритма для выполнения поставленных задач представлена на рисунке 4.

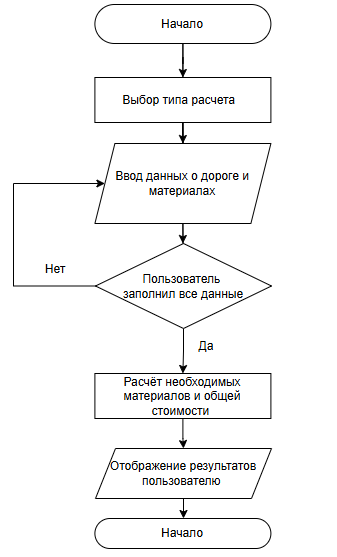


Рисунок 4 - Схема алгоритма

Действия, доступные пользователю приложения, отображены на диаграмме прецедентов, представленной на рисунке 5.

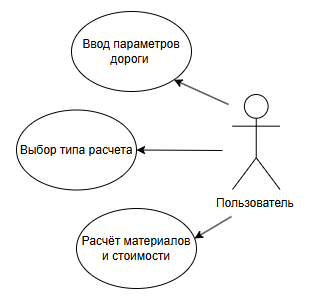


Рисунок 5 - Диаграмма прецедентов

## 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 (Model-View-ViewModel);
* библиотека для реактивного программирования: CommunityToolkit.

Класс RoadConstructionViewModel является ViewModel в паттерне MVVM и отвечает за логику расчета материалов для строительства дорог. Он использует свойства и команды для взаимодействия с пользовательским интерфейсом, код класса представлен листингом 8.

Листинг 8 – Код класса RoadConstructionViewModel

public partial class RoadConstructionViewModel : ObservableObject

{

private const double GeotextileThickness = 0.002;

private const double SandDensity = 1500;

private const double MaterialDensity = 2400;

[ObservableProperty]

private double length;

[ObservableProperty]

private double width;

[ObservableProperty]

private RoadMaterialType selectedMaterialType;

[ObservableProperty]

private double sandThicknessCm;

[ObservableProperty]

private double surfaceMaterialThicknessCm;

[ObservableProperty]

private double requiredMaterialAmount;

[ObservableProperty]

private double requiredSandAmount;

[ObservableProperty]

private double requiredGeotextileAmount;

[ObservableProperty]

private double totalCost;

[ObservableProperty]

private double materialCostPerKg = 0.5;

[ObservableProperty]

private double sandCostPerCubicMeter = 10;

[ObservableProperty]

private double geotextileCostPerSquareMeter = 2;

[ObservableProperty]

private string selectedCalculationType;

[ObservableProperty]

private bool showMaterialAmounts;

[ObservableProperty]

private bool showCost;

[ObservableProperty]

private string errorMessage;

public ObservableCollection<string> CalculationTypes { get; }

public ObservableCollection<RoadMaterialType> MaterialTypes { get; }

public string FormattedTotalCost => ConvertCost(TotalCost);

public RoadConstructionViewModel()

{

CalculationTypes = new ObservableCollection<string>

{

"Количество материалов",

"Примерная стоимость",

"Всё вместе"

};

MaterialTypes = new ObservableCollection<RoadMaterialType>

{

RoadMaterialType.ShMA\_22,

RoadMaterialType.ShMA\_16,

RoadMaterialType.ShMA\_11,

RoadMaterialType.ShPS

};

SelectedCalculationType = CalculationTypes[0];

SelectedMaterialType = MaterialTypes[0];

}

[RelayCommand]

private void Calculate()

{

if (!ValidateInputs())

{

return;

}

CalculateMaterials();

switch (SelectedCalculationType)

{

case "Количество материалов":

ShowMaterialAmounts = true;

ShowCost = false;

break;

case "Примерная стоимость":

ShowMaterialAmounts = false;

CalculateCost();

ShowCost = true;

break;

case "Всё вместе":

CalculateCost();

ShowMaterialAmounts = true;

ShowCost = true;

break;

}

}

private bool ValidateInputs()

{

if (Length <= 0)

{

ErrorMessage = "Длина дороги должна быть больше нуля.";

return false;

}

if (Width <= 0)

{

ErrorMessage = "Ширина дороги должна быть больше нуля.";

return false;

}

if (SandThicknessCm <= 0)

{

ErrorMessage = "Толщина песчаной подложки должна быть больше нуля.";

return false;

}

if (SurfaceMaterialThicknessCm <= 0)

{

ErrorMessage = "Толщина материала должна быть больше нуля.";

return false;

}

ErrorMessage = string.Empty;

return true;

}

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);

}

private void CalculateCost()

{

TotalCost = (RequiredMaterialAmount \* MaterialCostPerKg \* 1000) +

(RequiredSandAmount \* SandCostPerCubicMeter \* 1000) +

(RequiredGeotextileAmount \* GeotextileCostPerSquareMeter);

OnPropertyChanged(nameof(FormattedTotalCost));

}

private double GetMaterialCoefficient(RoadMaterialType materialType)

{

return materialType switch

{

RoadMaterialType.ShMA\_22 => 1.22,

RoadMaterialType.ShMA\_16 => 1.16,

RoadMaterialType.ShMA\_11 => 1.11,

RoadMaterialType.ShPS => 1.0,

\_ => 1.0

};

}

private string ConvertCost(double cost)

{

if (cost >= 1000000)

{

return $"{cost / 1000000:F2} млн руб";

}

else if (cost >= 1000)

{

return $"{cost / 1000:F2} тыс руб";

}

else

{

return $"{cost:F2} руб";

}

}

}

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

Вид главного окна программы представлен на рисунке 5.

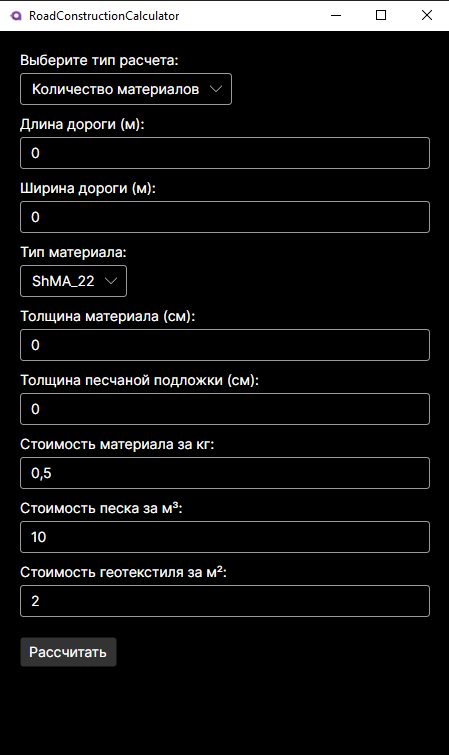


Рисунок 5 – Вид главной формы

## Отладка и тестирование программных модулей

Для отладки приложения используется комплекс средств Visual Studio

2022, который позволяет произвести диагностику и локализацию ошибок, внести исправления в программу с целью их устранения.

Стандартные средства отладчика Visual Studio 2022:

* для установки точки останова необходимо нажать клавишу F9, когда курсор находится на необходимой строке;
* для включения и отключения точек останова необходимо использовать комбинацию Ctrl+F9;
* для запуска отладки с остановками на точках останова необходимо нажать F5;
* для пошагового прохода без захода и с заходом в методы необходимо использовать F10 и F11 соответственно.

Для проверки работы приложения необходимо провести тестирование.

Для формирования теста необходимо указать выполняемое действие, ожидаемый результат и полученный результат. В таблице 2 приведен набор тестов разработанного приложения.

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 |

## Оптимизация и рефакторинг программного кода

Для улучшения читаемости и поддержки кода, константы были вынесены в начало класса., представленный листингом 9.

Листинг 9 – 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 9.

Листинг 9 – 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 оптимизация скорости кода не требуется.

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.

Список использованных источников

1. Microsoft. Документация по .NET и C# / Microsoft Docs. – URL: https://docs.microsoft.com/en-us/dotnet/ (дата обращения: 15.05.2025). – Режим доступа: свободный. – Текст: электронный.
2. Avalonia UI. Официальная документация по Avalonia UI / Avalonia UI Documentation. – URL: https://docs.avaloniaui.net/ (дата обращения: 15.05.2025). – Режим доступа: свободный. – Текст: электронный.
3. Microsoft. Документация по Microsoft SQL Server / Microsoft SQL Server Documentation. – URL: https://docs.microsoft.com/en-us/sql/sql-server/ (дата обращения: 15.05.2025). – Режим доступа: свободный. – Текст: электронный.
4. Microsoft. Документация по Entity Framework / Entity Framework Documentation. – URL: https://docs.microsoft.com/en-us/ef/ (дата обращения: 15.05.2025). – Режим доступа: свободный. – Текст: электронный.
5. Stack Overflow. Вопросы и ответы по программированию / Stack Overflow. – URL: https://stackoverflow.com/ (дата обращения: 15.05.2025). – Режим доступа: свободный. – Текст: электронный.