

Рисунок 1.1 - Xaml

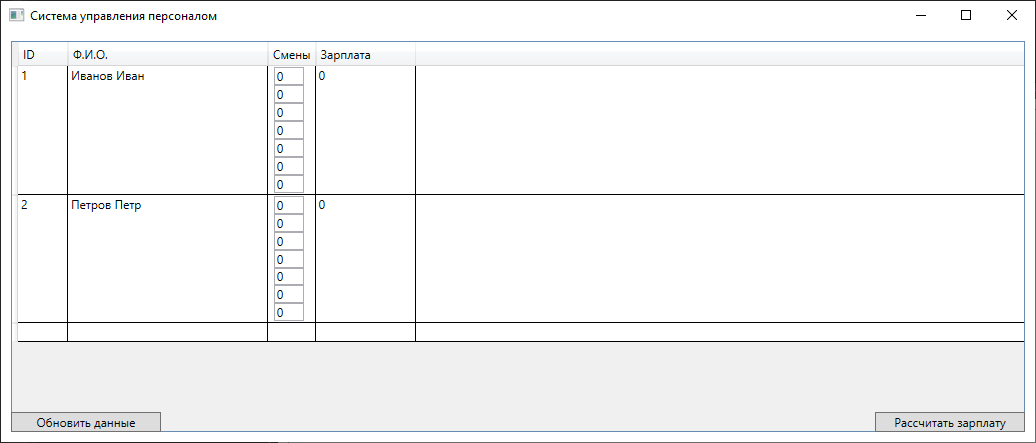


Рисунок 1.2 - Главное окно

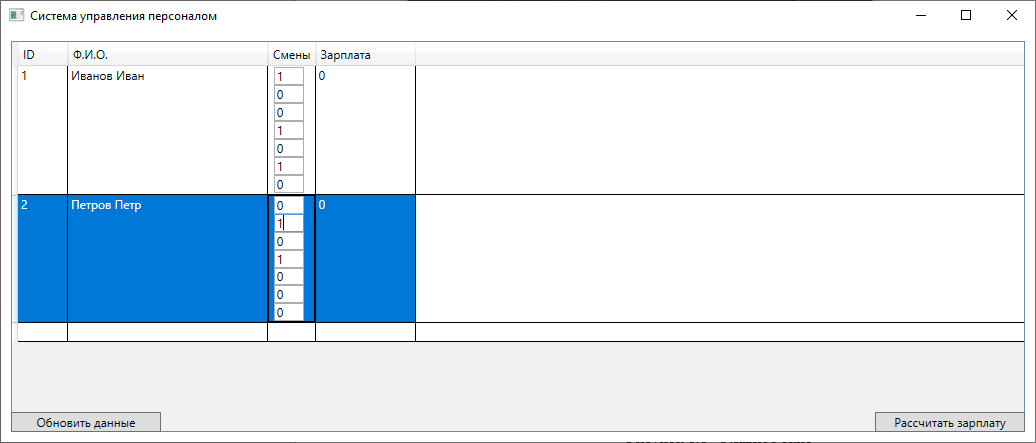


Рисунок 1.3 – Ввод данных

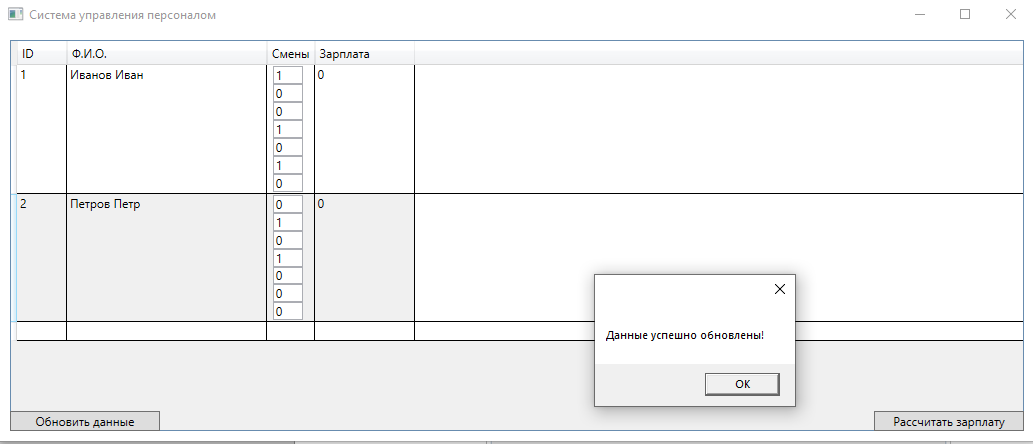


Рисунок 1.4 – Действие кнопки «Обновить данные»

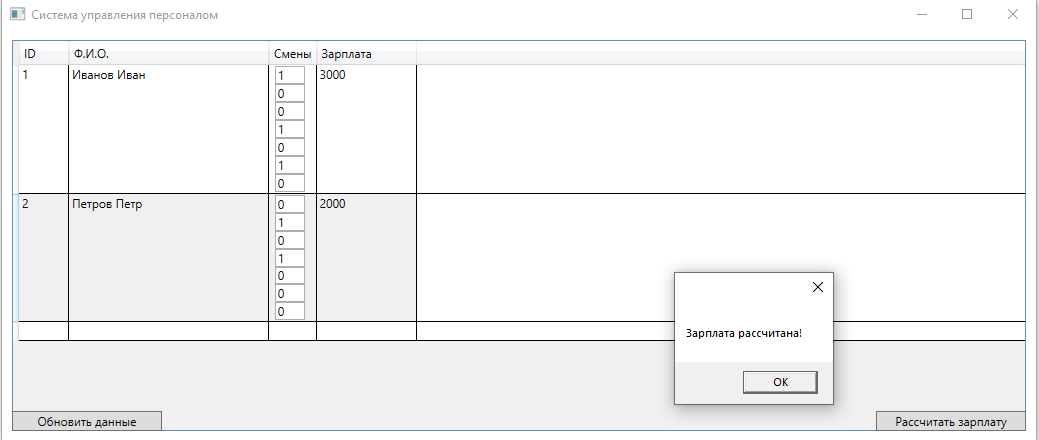


Рисунок 1.5 – Действие кнопки «Расчёт зарплаты»

MainWindow.xaml.cs

|  |
| --- |
| …  public class Shift : INotifyPropertyChanged  {  private int worked;  public int Worked  {  get => worked;  set  {  if (value != 0 && value != 1)  {  MessageBox.Show("0 - Не рабочая смена\n1 - Рабочая смена");  throw new ArgumentException("Значение должно быть 0 или 1");  }  worked = value;  OnPropertyChanged(nameof(Worked));  }  }  public event PropertyChangedEventHandler PropertyChanged;  protected void OnPropertyChanged(string propertyName)  {  PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));  }  }  // Загрузка данных сотрудников  private ObservableCollection<Employee> LoadEmployees()  {  var dbHelper = new DatabaseHelper();  var staffList = dbHelper.LoadStaff();  return new ObservableCollection<Employee>(  staffList.Select(s => new Employee  {  Id = s.Id,  Name = s.FullName,  VacationDays = s.VacationDays.ToString(),  Job = s.CurrentJob,  RatePerShift = s.RatePerShift,  Shifts = GenerateShifts()  }));  }  … |

DataBaseHelper.cs

|  |
| --- |
| …  using Npgsql;  using System;  using System.Collections.Generic;  namespace WpfAppLeon  {  public class DatabaseHelper  {  private const string ConnectionString = "Host=localhost;Port=5432;Database=EmployeeLeon;Username=postgres;Password=sa;";  public List<MainWindow.Staff> LoadStaff()  {  var staffList = new List<MainWindow.Staff>();  try  {  using (var connection = new NpgsqlConnection(ConnectionString))  {  connection.Open();  string query = "SELECT id, fullname, vacationdays, currentjob, ratepershift FROM staff";  using (var command = new NpgsqlCommand(query, connection))  using (var reader = command.ExecuteReader())  {  while (reader.Read())  {  staffList.Add(new MainWindow.Staff  {  Id = reader.GetInt32(0),  FullName = reader.GetString(1),  VacationDays = reader.GetInt32(2),  CurrentJob = reader.GetString(3),  RatePerShift = reader.GetInt32(4)  });  }  }  }  }  catch (Exception ex)  {  throw new Exception($"Ошибка при загрузке сотрудников: {ex.Message}");  }  return staffList;  }  /// <summary>  /// Добавляет нового сотрудника в базу данных.  /// </summary>  public void AddEmployee(string fullName, int vacationDays, string currentJob, int ratePerShift)  {  try  {  using (var connection = new NpgsqlConnection(ConnectionString))  {  connection.Open();  string query = @"  INSERT INTO staff (fullname, vacationdays, currentjob, ratepershift)  VALUES (@fullname, @vacationdays, @currentjob, @ratepershift)";  using (var command = new NpgsqlCommand(query, connection))  {  command.Parameters.AddWithValue("fullname", fullName);  command.Parameters.AddWithValue("vacationdays", vacationDays);  command.Parameters.AddWithValue("currentjob", currentJob);  command.Parameters.AddWithValue("ratepershift", ratePerShift);  command.ExecuteNonQuery();  }  }  }  catch (Exception ex)  {  throw new Exception($"Ошибка при добавлении сотрудника: {ex.Message}");  }  }  }  }… |

Тест: Простановка пользователем смен и просчет зарплаты

Цель: Проверка что пользователь сможет установить смены рабочим и подсчитать им зарплату.

Ход выполнения:

1. Пользователь запускает программу.

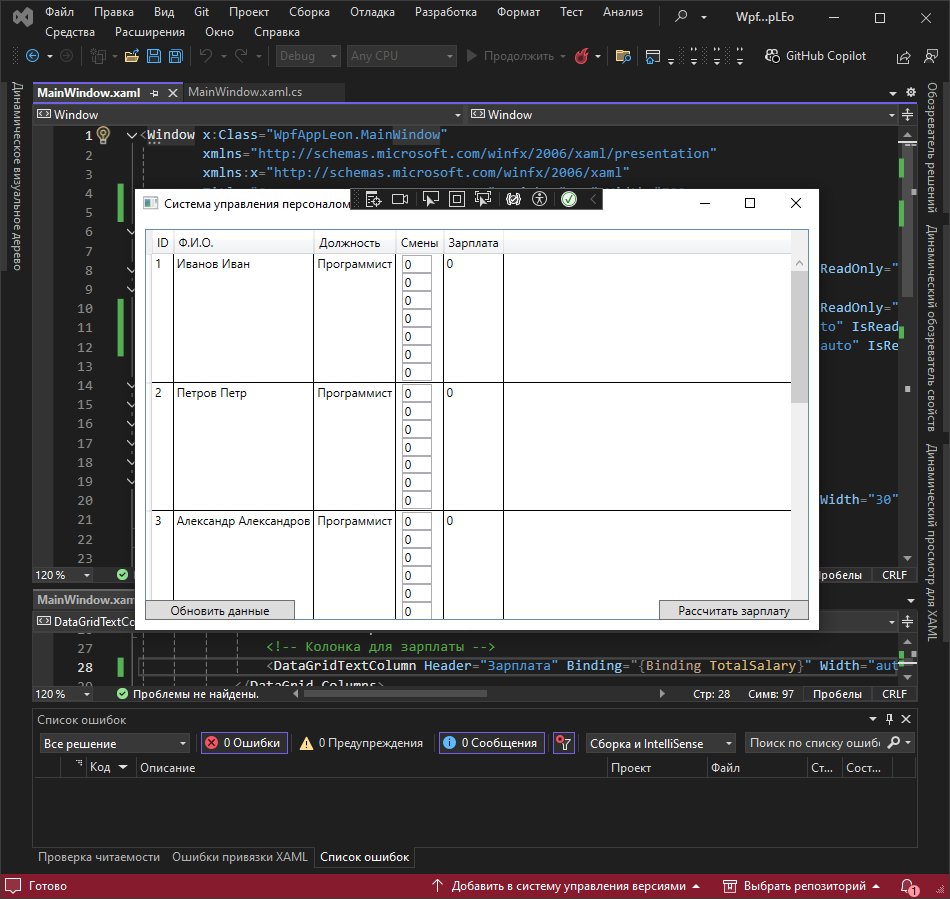


Рисунок 1.6 – Открытия приложения пользователем

1. Выбирает для работников нужное число смен меняя значения.

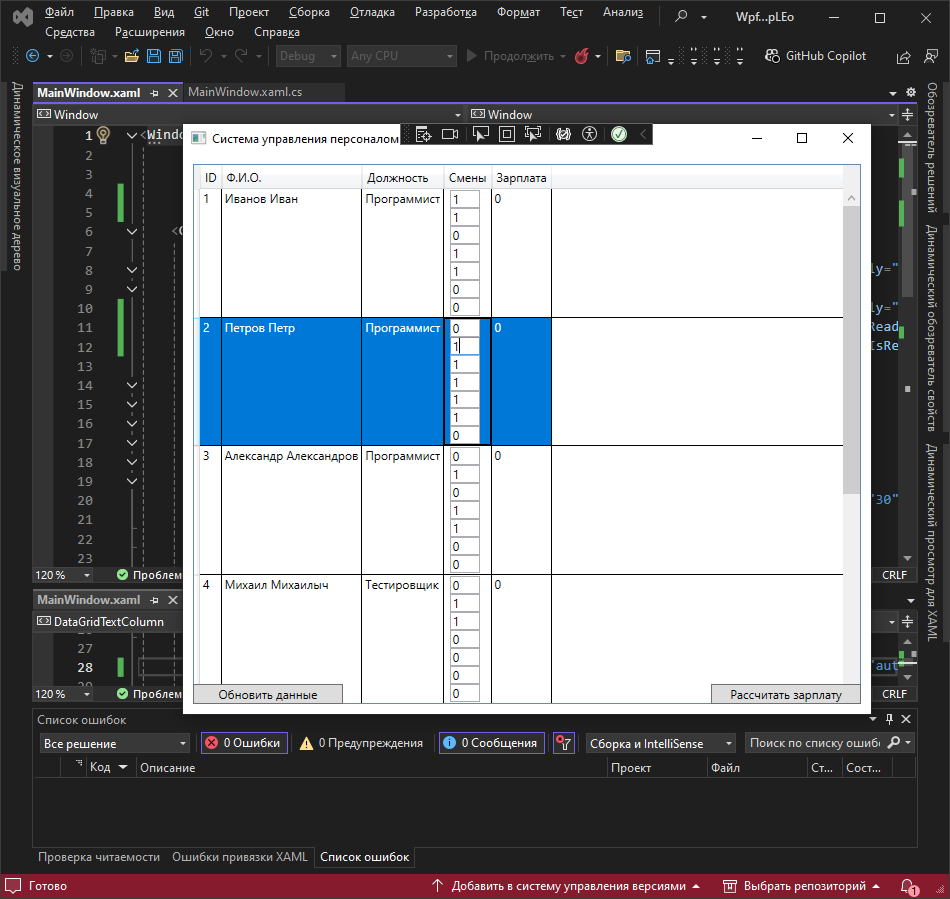


Рисунок 1.7 – Изменение пользователем смен

1. Обновляет данные в таблице путем нажатия кнопки «Обновить данные»

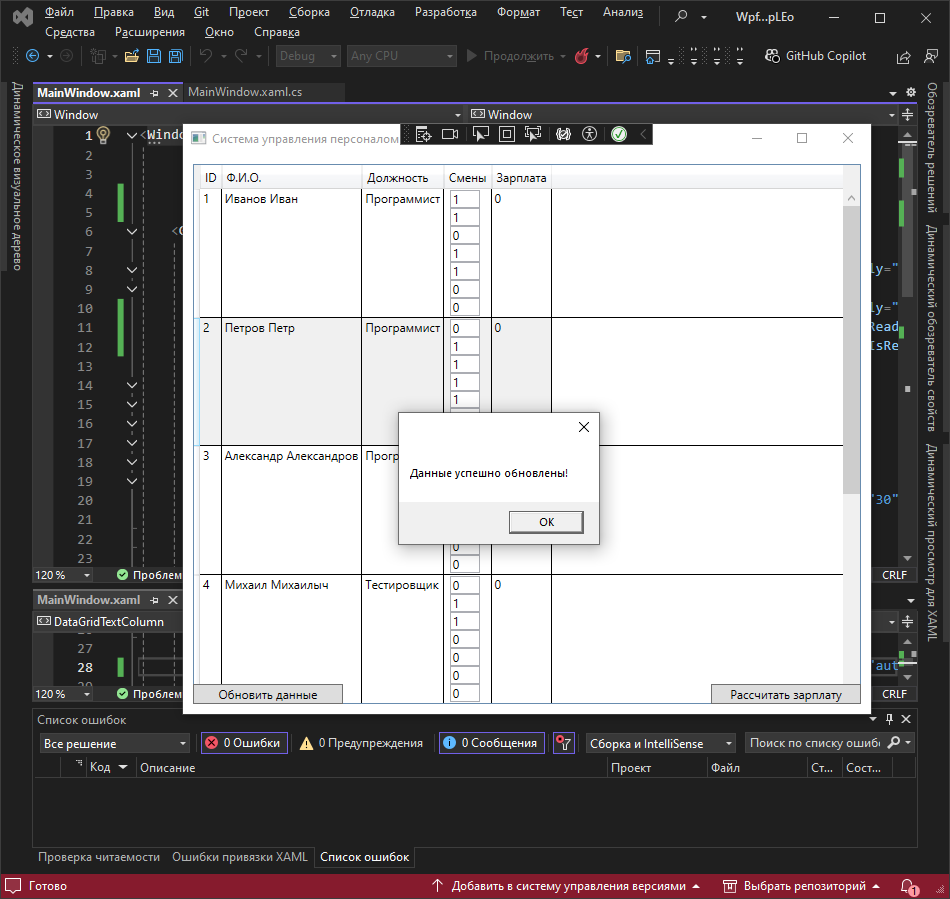


Рисунок 1.8 – Действие кнопки «Обновить данные»

1. Пользователь нажимает кнопку «Рассчитать зарплату»

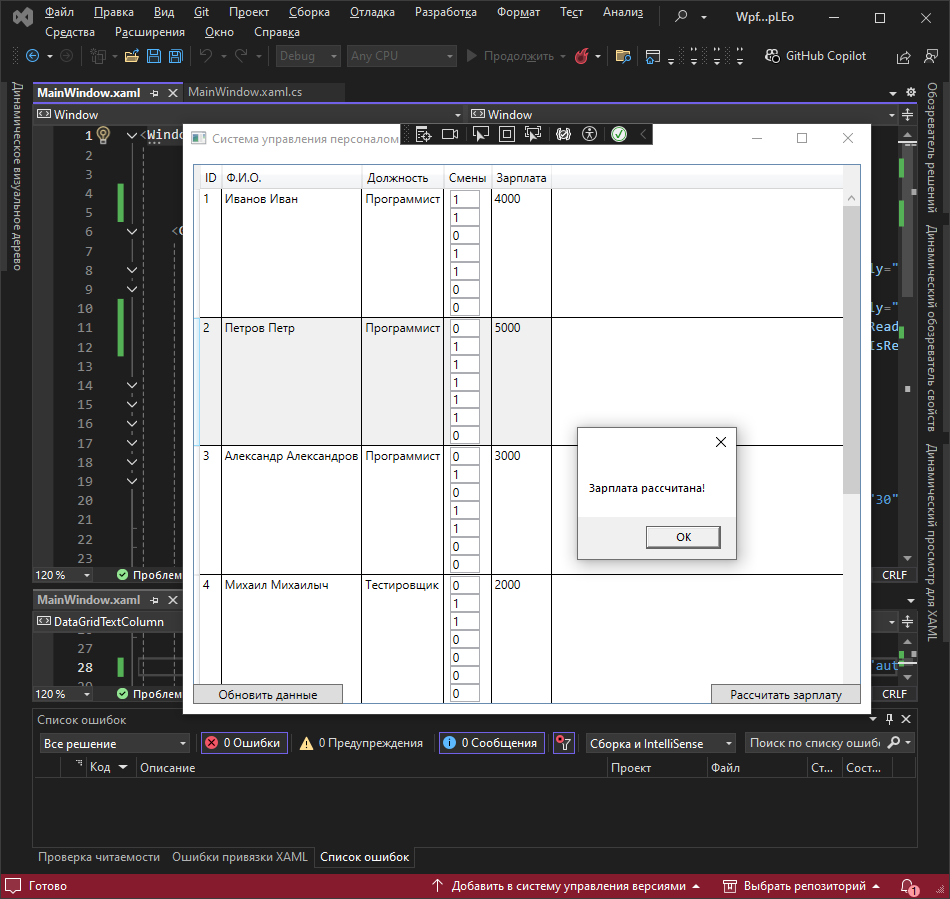


Рисунок 1.9 – Действие кнопки «Обновить данные»

Ожидаемый результат: Пользователь расставил смены и смог подсчитать зарплату.

Итог: Пользователь успешно изменил данные и посчитал зарплату работников