

КОМИТЕТ ПО ОБРАЗОВАНИЮ ПРАВИТЕЛЬСТВА САНКТ-ПЕТЕРБУРГА

Санкт-Петербургское государственное
бюджетное профессиональное образовательное учреждение
«КОЛЛЕДЖ ЭЛЕКТРОНИКИ И ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ»

«МДК.07.01 Управление и автоматизация баз данных»

ОТЧЁТ

по лабораторной работе №3

«Подготовка и импорт данных»

Работу выполнил студент 325гр.:

Шлычков И. Д.

Преподаватель: Фомин А.В.

Санкт-Петербург 2025

Импорт данных

Для выполнения работы были использованы данные о береговой линии, представленные по ссылке <https://www.naturalearthdata.com/downloads/10m-physical-vectors/> . Рисунок 1



Рисунок 1 – сайт с данными

После скачивания архивы были распакованы в рабочую папку. Данные содержали координаты береговых линий. Рисунок 2

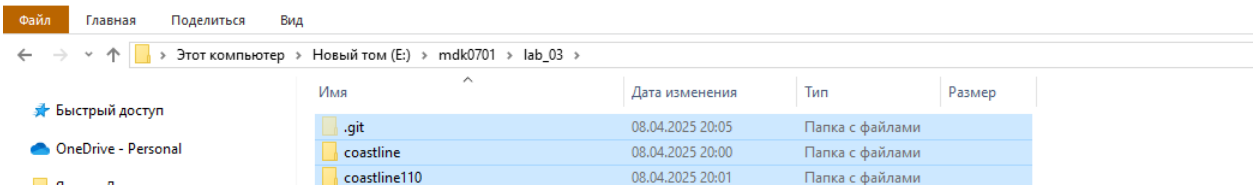


Рисунок 2 – рабочая папка

Для хранения данных ранее была создана база данных (lab02) в СУБД PostgreSQL. Структура базы данных включала координат береговых линий, а также дополнительных таблицы для метаданных. Рисунок 3

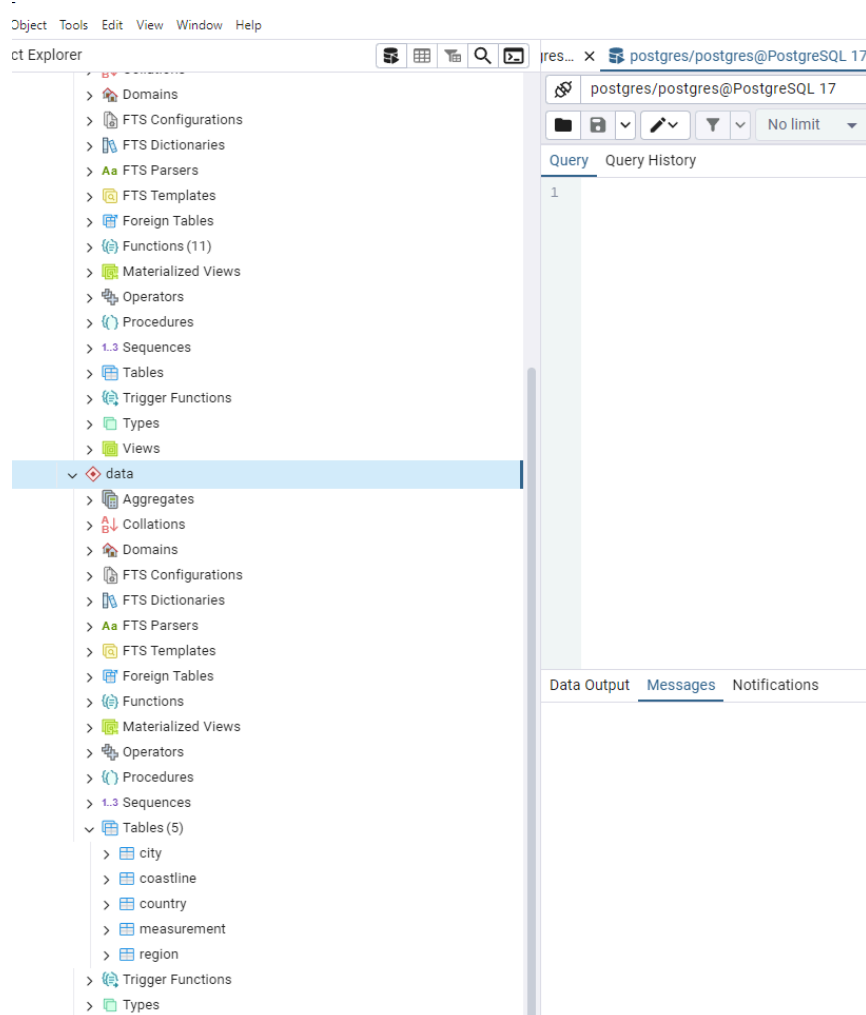


Рисунок 3 – база данных

Создаем проекты

Для импорта данных был разработан скрипт на C#, который подключает проект к базе данных и загружает данные из распакованных файлов. Рисунок 4

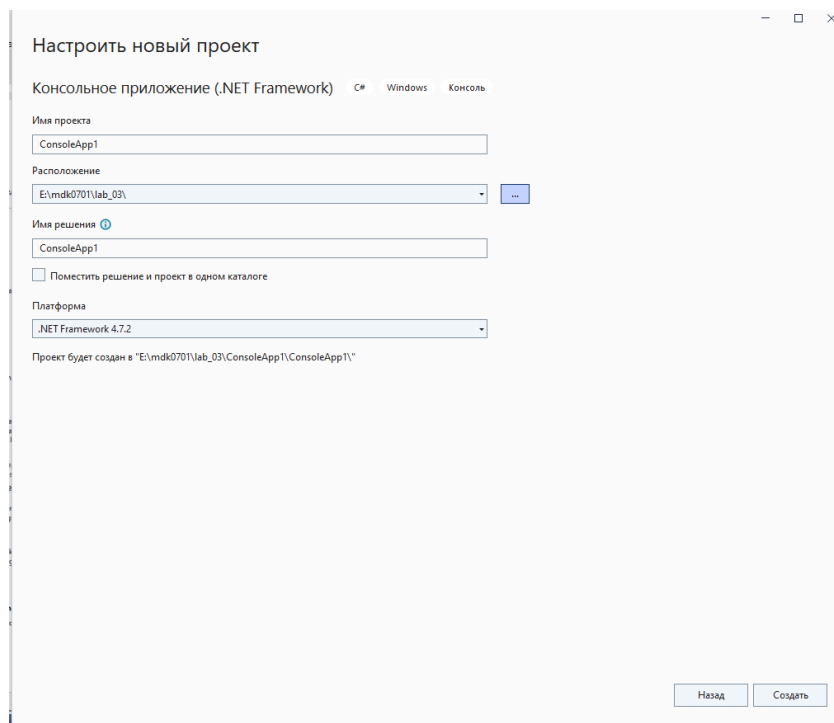
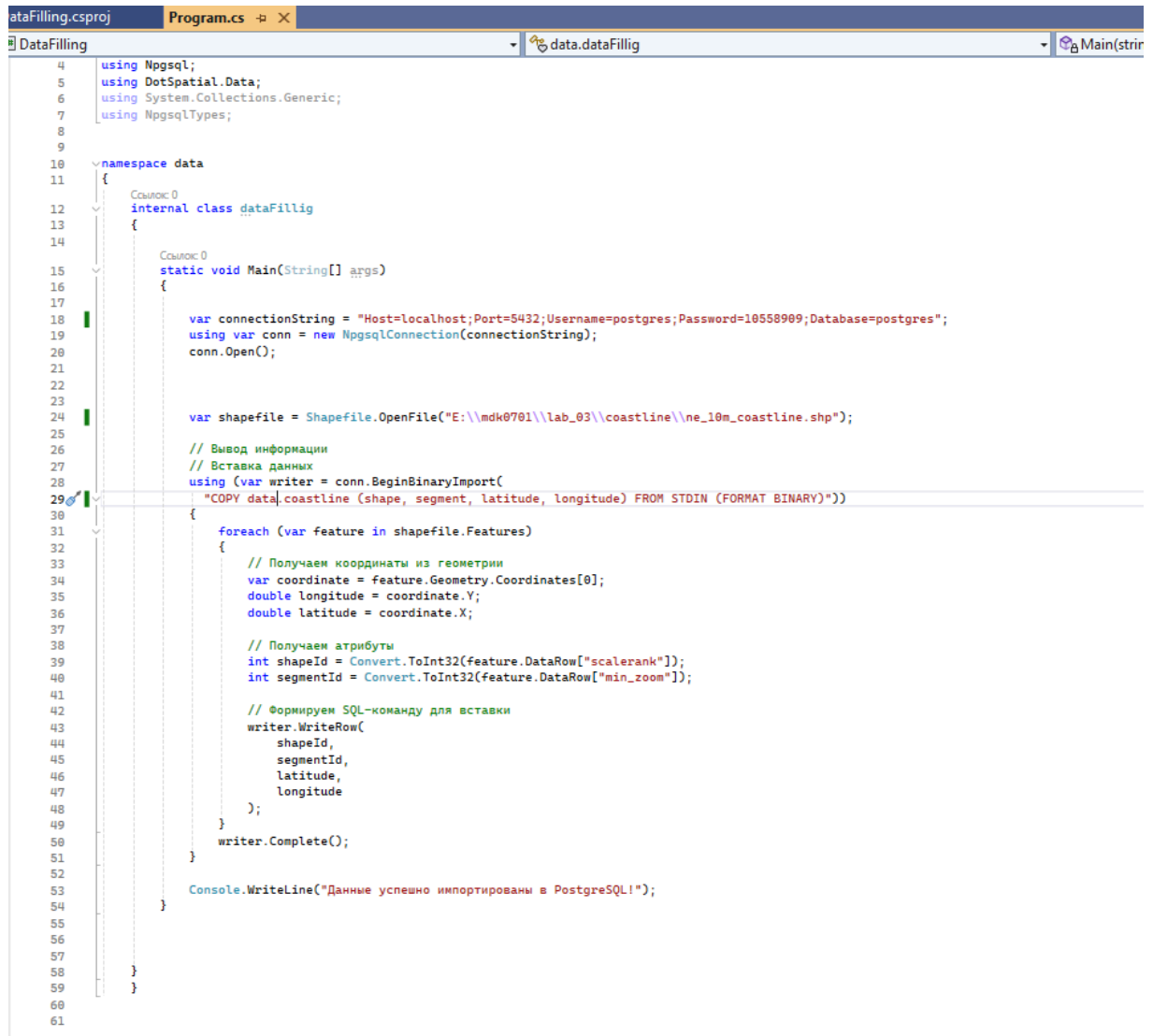


Рисунок 4 – создание проекта

Тут представлен фрагмент кода, в котором указаны порт БД, пароль и название самой БД. Также был добавлен путь к папке с данными, чтобы скрипт смог их перенести в нашу схему. Рисунок 5



```
1  using Npgsql;
2  using DotSpatial.Data;
3  using System.Collections.Generic;
4  using NpgsqlTypes;
5
6  namespace data
7  {
8      internal class dataFilling
9      {
10
11          static void Main(String[] args)
12          {
13
14              var connectionString = "Host=localhost;Port=5432;Username=postgres;Password=10558909;Database=postgres";
15              using var conn = new NpgsqlConnection(connectionString);
16              conn.Open();
17
18              var shapefile = Shapefile.OpenFile("E:\\mdk0701\\lab_03\\coastline\\ne_10m_coastline.shp");
19
20              // Вывод информации
21              // Вставка данных
22              using (var writer = conn.BeginBinaryImport(
23                  "COPY data.coastline (shape, segment, latitude, longitude) FROM STDIN (FORMAT BINARY)"))
24              {
25                  foreach (var feature in shapefile.Features)
26                  {
27                      // Получаем координаты из геометрии
28                      var coordinate = feature.Geometry.Coordinates[0];
29                      double longitude = coordinate.Y;
30                      double latitude = coordinate.X;
31
32                      // Получаем атрибуты
33                      int shapeId = Convert.ToInt32(feature.DataRow["scalerank"]);
34                      int segmentId = Convert.ToInt32(feature.DataRow["min_zoom"]);
35
36                      // Формируем SQL-команду для вставки
37                      writer.WriteRow(
38                          shapeId,
39                          segmentId,
40                          latitude,
41                          longitude
42                      );
43                  }
44                  writer.Complete();
45              }
46
47              Console.WriteLine("Данные успешно импортированы в PostgreSQL!");
48          }
49      }
50  }
```

Рисунок 5 – фрагмент кода

Но прежде, чем подключить наш проект к БД, мы должны перейти во вкладку проект. Рисунок 6

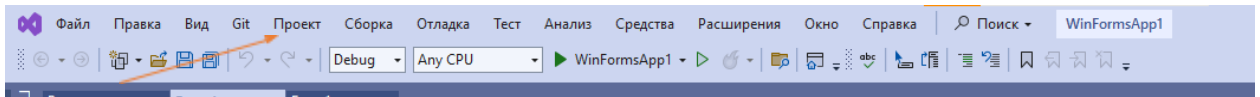


Рисунок 6

И затем скачать дополнительные расширения представленные на Рисунке 7, чтобы всё заработало

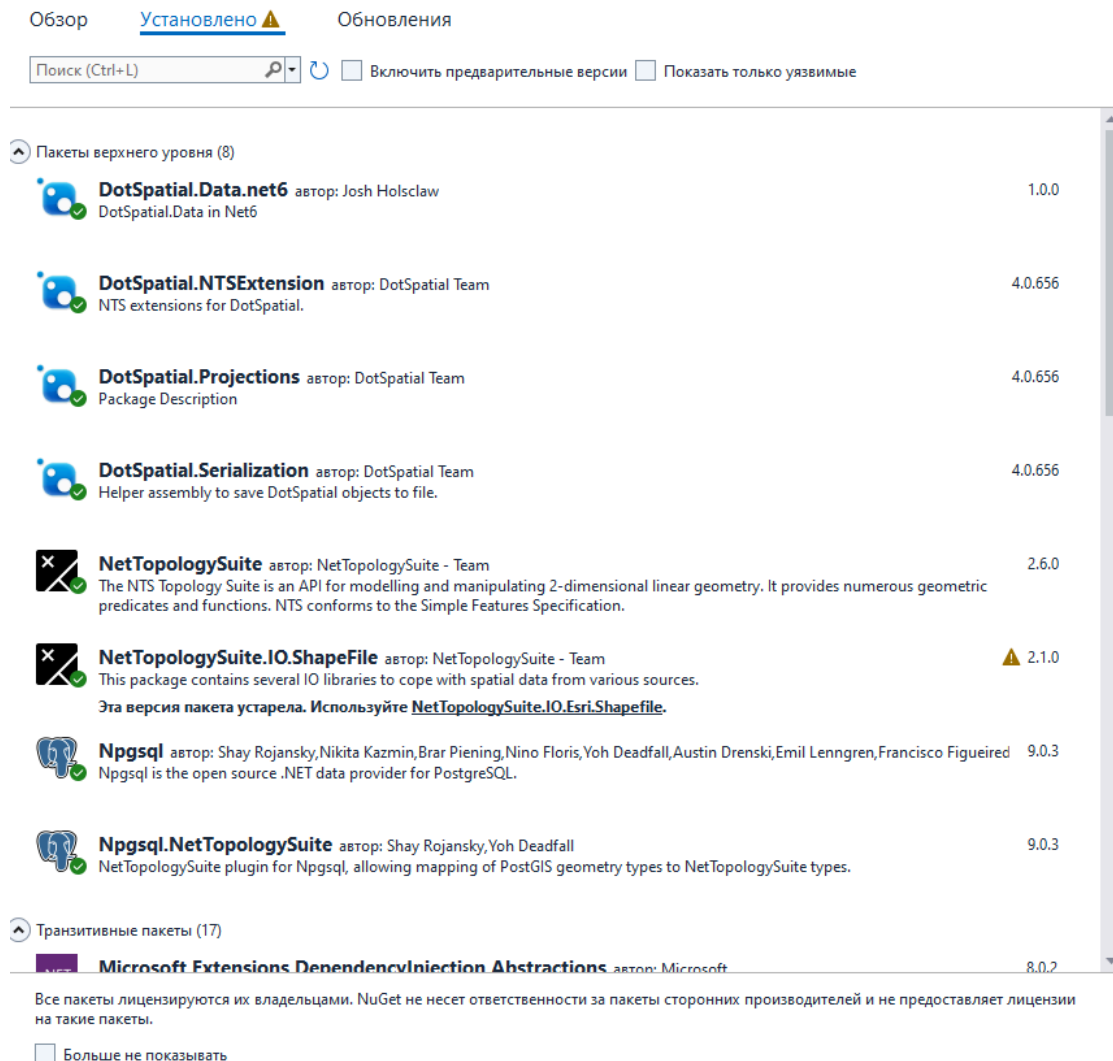


Рисунок 7 – дополнительные расширения

После чего мы запускаем нашу программу. И убеждаемся что всё работает.
Рисунок 8

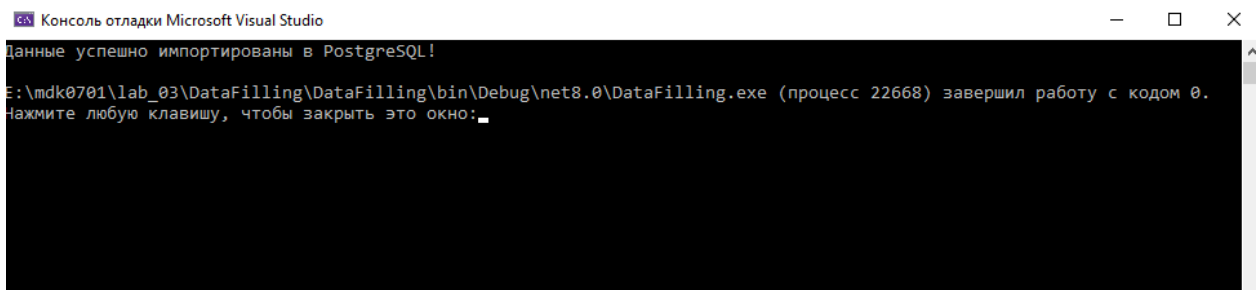


Рисунок 8

Выводим через select, чтобы посмотреть результат. Рисунок 9

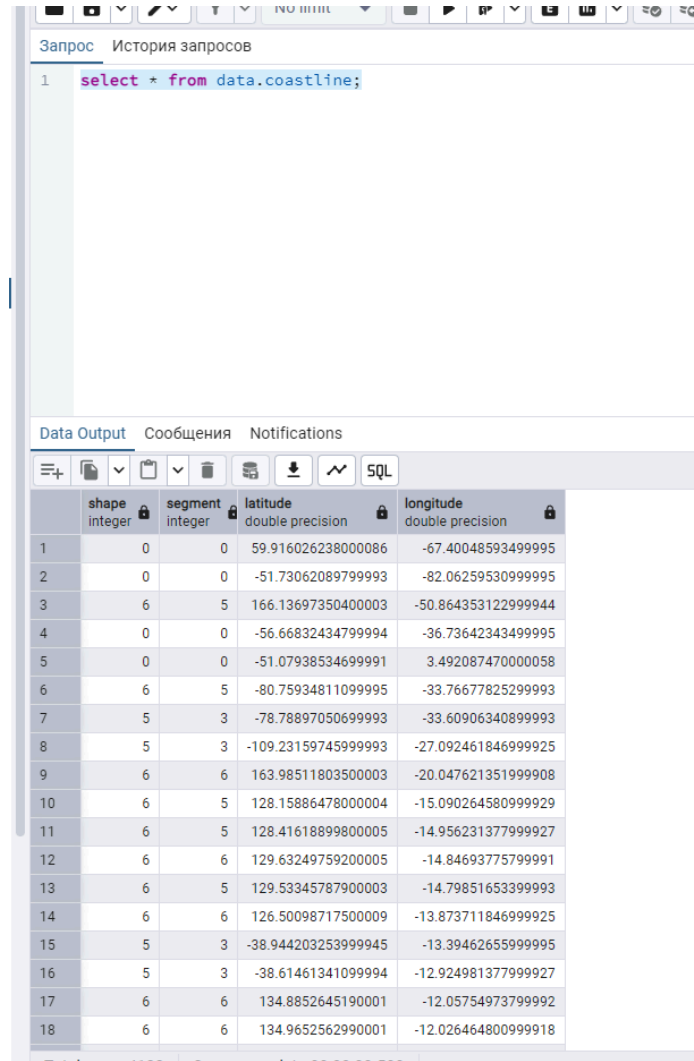


Рисунок 9

Для визуализации данных был создан второй проект на C#. Приложение подключалось к БД, извлекало координаты береговых линий и отображало их на карте мира. Рисунок 10-11

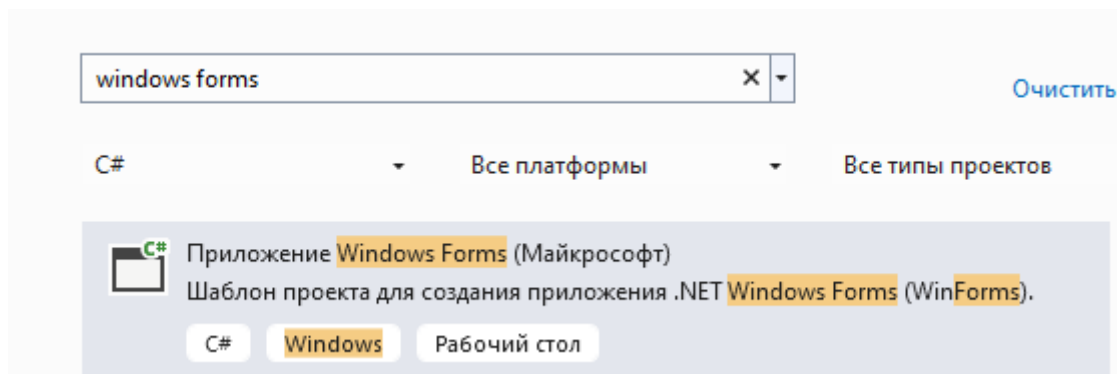


Рисунок 10 -создание проекта

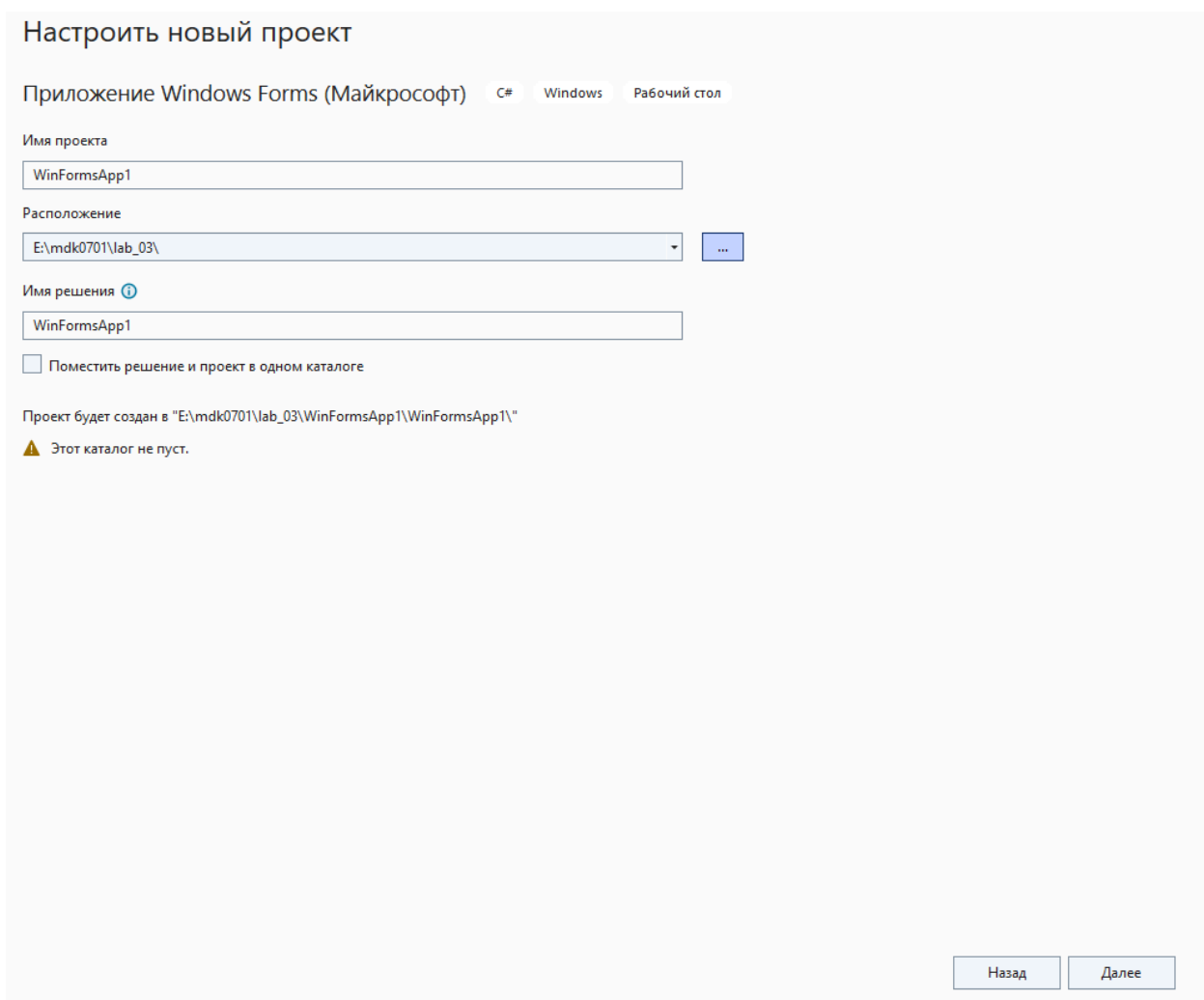
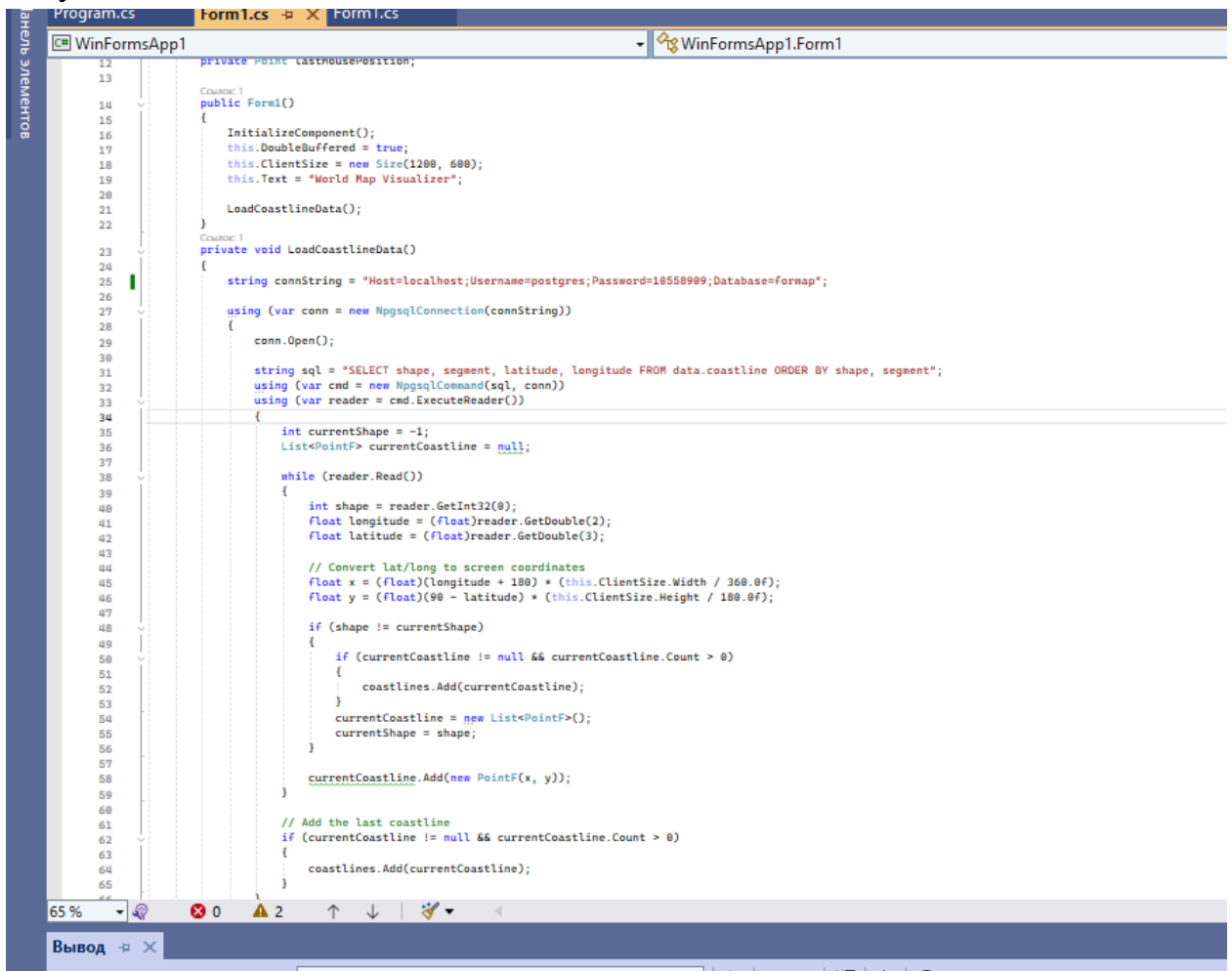


Рисунок 11 -создание проекта

Тут представлен фрагмент кода, в котором указаны порт БД, пароль и название самой БД. С помощью этого сценария приложение выводит карту мира с помощью данных, которые были занесены прошлым скриптом.

Рисунок 12



```
12 private POINT lasthousePosition;
13
14 COORDINATE:1
15 public Form1()
16 {
17     InitializeComponent();
18     this.DoubleBuffered = true;
19     this.ClientSize = new Size(1200, 600);
20     this.Text = "World Map Visualizer";
21
22     LoadCoastlineData();
23 }
24 COORDINATE:1
25 private void LoadCoastlineData()
26 {
27     string connString = "Host=localhost;Username=postgres;Password=10550909;Database=ormap";
28
29     using (var conn = new NpgsqlConnection(connString))
30     {
31         conn.Open();
32
33         string sql = "SELECT shape, segment, latitude, longitude FROM data.coastline ORDER BY shape, segment";
34         using (var cmd = new NpgsqlCommand(sql, conn))
35         using (var reader = cmd.ExecuteReader())
36         {
37             int currentShape = -1;
38             List<PointF> currentCoastline = null;
39
40             while (reader.Read())
41             {
42                 int shape = reader.GetInt32(0);
43                 float longitude = (float)reader.GetDouble(2);
44                 float latitude = (float)reader.GetDouble(3);
45
46                 // Convert lat/long to screen coordinates
47                 float x = (float)(longitude + 180) * (this.ClientSize.Width / 360.0f);
48                 float y = (float)(90 - latitude) * (this.ClientSize.Height / 180.0f);
49
50                 if (shape != currentShape)
51                 {
52                     if (currentCoastline != null && currentCoastline.Count > 0)
53                     {
54                         coastlines.Add(currentCoastline);
55                     }
56                     currentCoastline = new List<PointF>();
57                     currentShape = shape;
58                 }
59                 currentCoastline.Add(new PointF(x, y));
60             }
61
62             // Add the last coastline
63             if (currentCoastline != null && currentCoastline.Count > 0)
64             {
65                 coastlines.Add(currentCoastline);
66             }
67         }
68     }
69 }
```

Рисунок 12 -фрагмент кода

Здесь представлен результат работы приложения. Рисунок 13

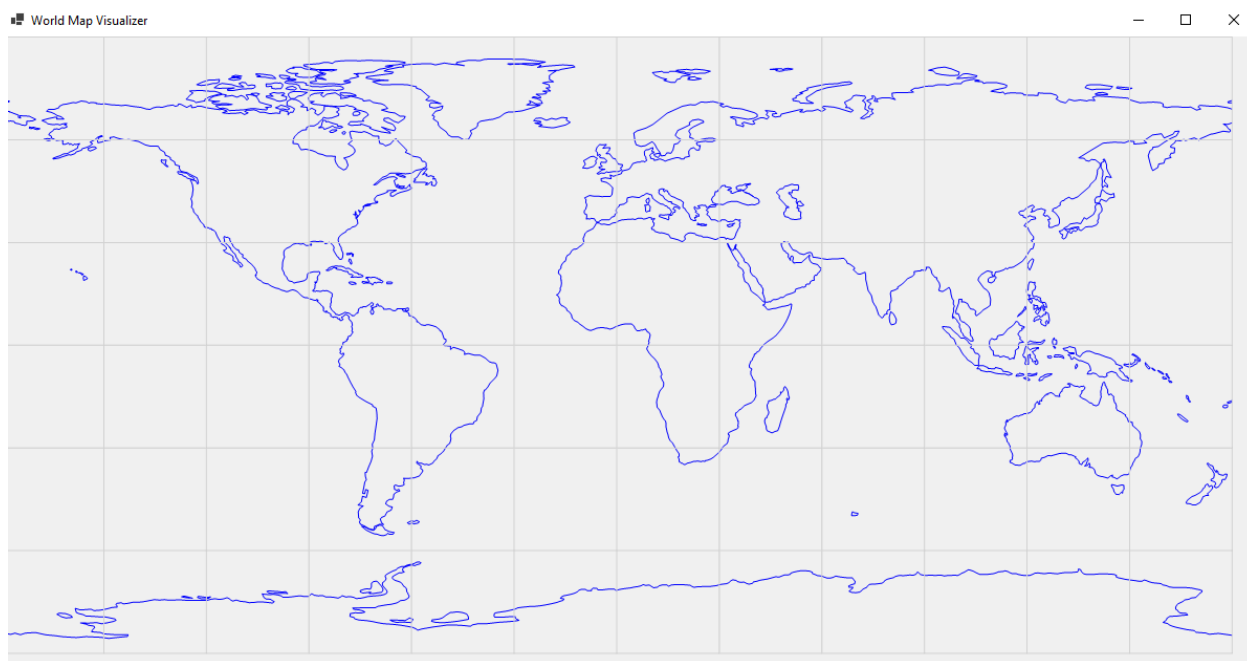


Рисунок 13 -результат

Заключение

В ходе работы была успешно спроектирована база данных для хранения данных береговой линии, выполнены операции импорта данных и разработано приложение для их визуализации. Работа позволила закрепить навыки работы с базами данных, и с программой как Visual Studio.