

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №3

ПО ДИСЦИПЛИНЕ

«ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ ИНФОРМАЦИОННЫХ СИСТЕМ» НА ТЕМУ

«Приложение ASP.NET для работы с базой данных PostgreSQL»

Вариант – 2

Факультет: АВТ Преподаватель: Дубков И.С.

Группа: АВТ-918

Студент: Ванин К.Е.

Цель работы

Создание Web-приложений ASP.NET, взаимодействующих с базой данных PostgreSQL.

Задание

1. Создать базу данных PostgreSQL в соответствии с вариантом задания.
2. Составить программу для создания Web-приложения ASP.NET, данные которого можно будет хранить в базе данных PostgreSQL.
3. С помощью созданного Web-приложения обеспечить заполнение и просмотр содержимого базы данных.
4. Выполнить запросы к базе данных для получения требуемой информации.

**Вариант** 2:

Коллекция картин художественного музея

Ход работы

Была создана база данных Artmuseum, внутри которой были созданы следующие таблицы: Paintings (картины), Authors (авторы).

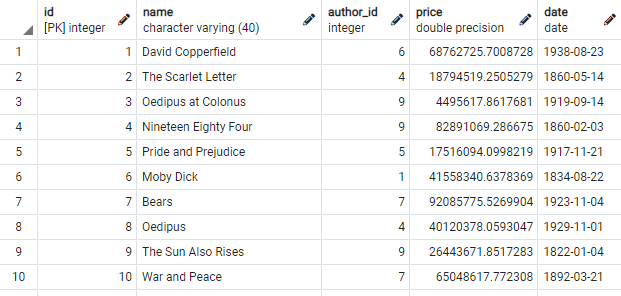
В таблице Paintings существуют колонки:

* id (идентификатор),
* name (название картины),
* date (дата создания картины)
* author\_id (внешний ключ, идентификатор автора),
* price (цена картины);

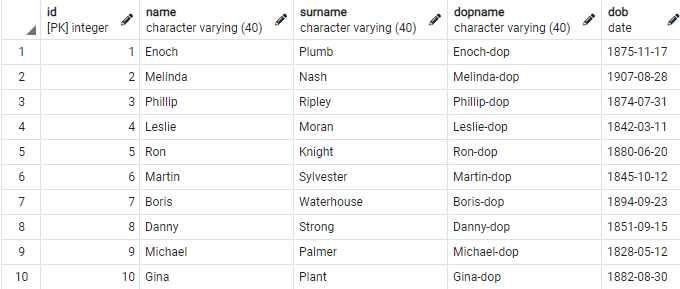
в таблице authors:

* id (идентификатор),
* name (имя),
* surname (фамилия),
* dopname (отчество),
* dob (дата рождения);

Далее заполним таблицы данными.

**

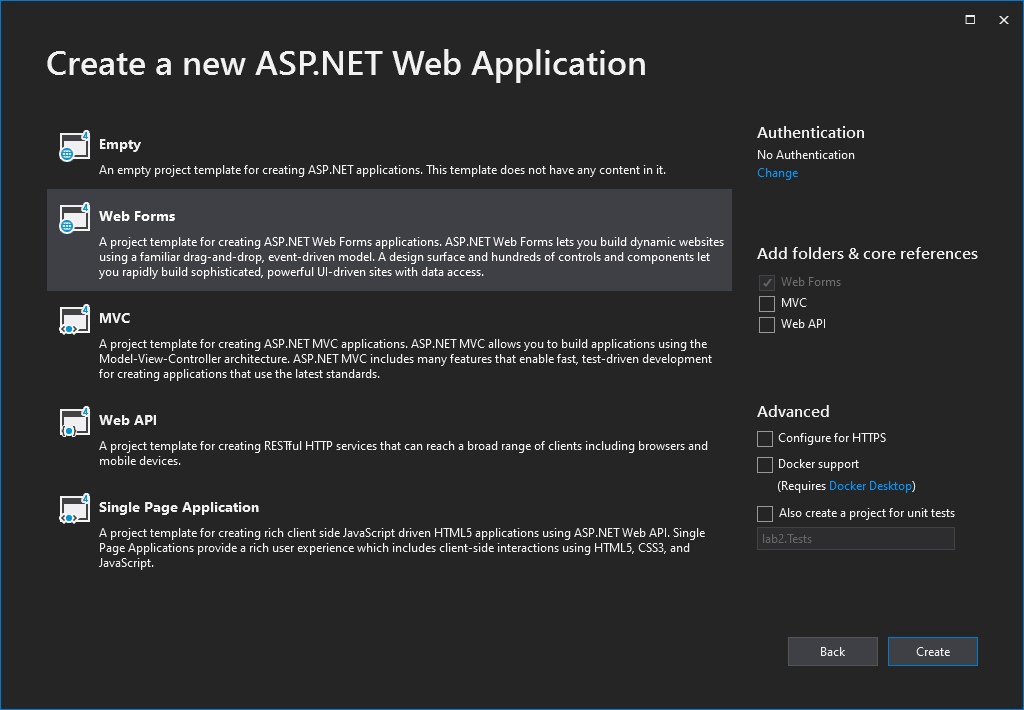
*Рис. 1 – Содержание таблицы Paintings*

**

*Рис. 2 – Содержание таблицы Authors*

SQL-код для создания базы данных и таблиц приведен в приложении A.

Для выполнения лабораторной работы был выбран ASP.NET Web Forms для системы Windows.



*Рис. 3 – Создание проекта*

Согласно заданию, был создан пустой проект, в который была добавлена мастер-страница с именем по умолчанию (Site1.Master). Созданная мастер-страница представляет собой «каркас» сайта – она содержат HTML-разметку с шапкой и подвалом сайта (также туда входит все, что не относится к контентной части).

После этого была создана веб-форма. Веб-форма позволяет просматривать список картин, а также добавлять новые картины путем добавления данных в таблицу Paintings.

На главной странице приводится таблица, которая создается динамически, после выборки данных из БД с помощью ADO.NET компонентов: SqlConnection, SqlDataAdapter, которые описаны ниже.

Подключение к базе данных:

connectionString = String.Format("Server = {0}; Database = {1}; Port = {2}; Username = {3}; Password = {4}", "localhost", "postgres", "5432", "postgres", "12345");

conn = new NpgsqlConnection(connectionString);

conn.Open();

С использованием строки запроса и объекта класса NpgsqlConnection создадим объект класса NpgsqlCommand:

var command1 = new NpgsqlCommand("select id, name, surname from authors;", conn);

С использованием объекта класса NpgsqlCommand, создадим объект класса NpgsqlDataReader:

NpgsqlDataReader reader1 = command1.ExecuteReader();

Далее в цикле добавляем строки и столбцы к таблице.

while (reader.Read()){

for (int i = 0; i < numcols; i++){

textContent = reader[i].ToString();

cell.Text = textContent;

row.Cells.Add(cell);

}

table.Rows.Add(row);

}

После окончания чтения, закрываем соединение.

На странице пользователю предлагается отправить форму с данными, которые будут записаны в таблицу Paintings. При нажатии на кнопку «Отправить» вызывается обработчик события Button1\_Click.

protected void Button1\_Click(object sender, EventArgs e)

{

connectionString = String.Format("Server = {0}; Database = {1}; Port = {2}; Username = {3}; Password = {4}",

"localhost", "postgres", "5432", "postgres", "12345");

conn = new NpgsqlConnection(connectionString);

conn.Open();

var commandString = String.Format("insert into paintings (name, author\_id, price, date) " +

"values('{0}', {1}, {2}, '{3}');",

name.Text, DropDownList1.SelectedValue, price.Text, date.Text);

var command1 = new NpgsqlCommand(commandString, conn);

NpgsqlCommand sqlCmd = new NpgsqlCommand(commandString, conn);

try

{

sqlCmd.ExecuteNonQuery();

}

catch (Exception ex)

{

Label1.Text = "Ошибка добавления записи!";

}

conn.Close();

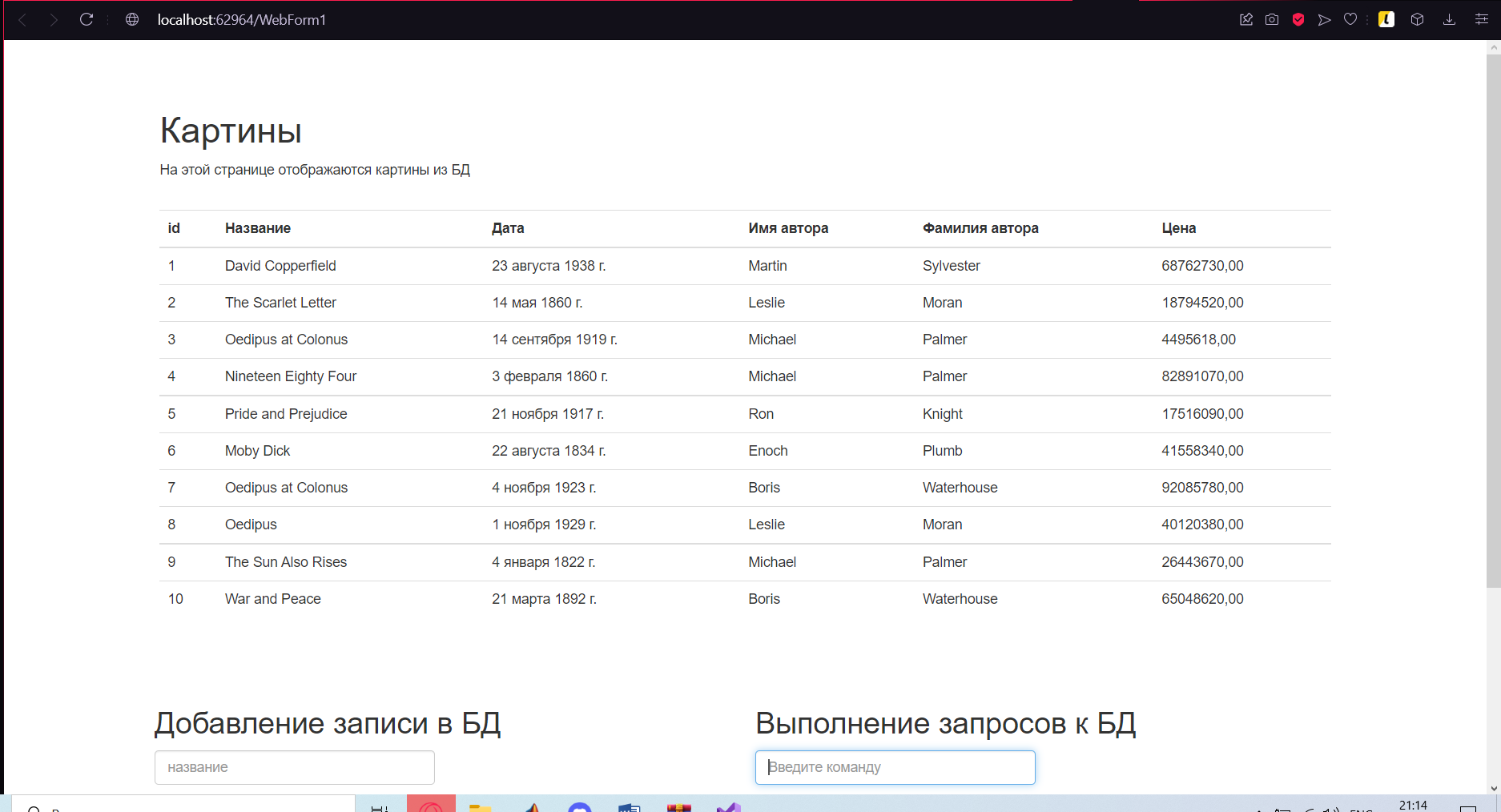
Page.Response.Redirect(Page.Request.Url.ToString(), true);

}

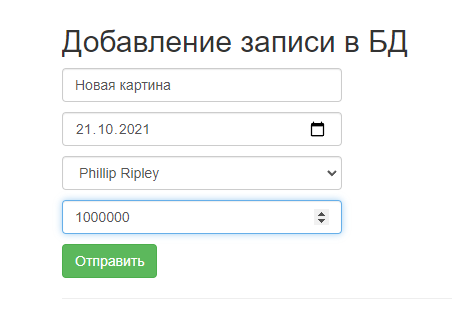
В этом обработчике открывается соединение с БД, выполняется команда добавления записи в таблицу. В случае ошибки, пользователю выводится сообщение об ошибке и закрывает соединение. В случае успешного выполнения команды сначала закрывается соединение, после чего делаем редирект для того, чтобы предотвратить повторную отправку формы.

Также добавил форму, в которую можно ввести команду для выборки данных из базы данных. Добавил для нее обработчик, в котором открывается соединение с БД, выполняется команда выборки из таблицы. В случае ошибки, пользователю выводится сообщение о некорректной команде и закрывается соединение. В случае успешного выполнения команды сначала выводим полученные данные, после чего закрывается соединение.

Скриншоты



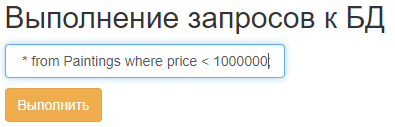
*Рис. 2 – Главная страница*



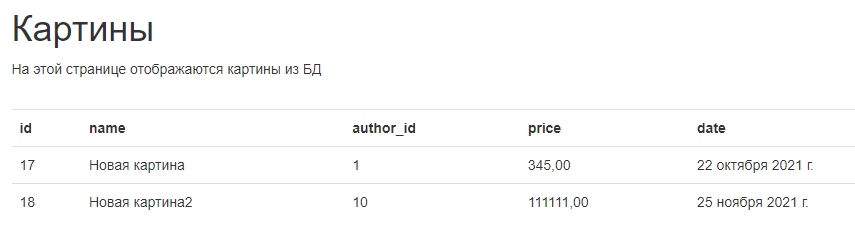
*Рис. 3 – Форма добавления записи в таблицу*

**

*Рис. 4 – Результат добавления записи в таблицу*



*Рис. 5 – Форма выполнения запроса для получения информации*

**

*Рис. 6 – Пример выполнения запроса для получения информации о картинах с ценой меньше 1000000*

Ответы на вопросы

1. **Какие компоненты требуется включить в программный код ASP.NET приложения, чтобы обеспечить доступ к данным по технологии ADO.NET?**

В первую очередь нужно подключить необходимый драйвер (для разных СУБД он отличается). После этого можно подключаться к базе данных с помощью компонента SqlConnection, делать запросы с помощью компонентов SqlCommand, SqlParameter, SqlDataAdapter, SqlDataReader. После осуществления выборки данными удобно оперировать с помощью классов Data\*: DataSet, DataTable, DataView и т.д.

1. **Какие элементы управления ASP.NET приложения можно использовать для отображения результатов запросов к базе данных?**

Для визуализации выборки можно использовать asp:GridView. GridView – гибкий табличный элемент управления, предназначенный для демонстрации данных в виде двумерной сетки, или экранной таблицы, состоящей из строк и столбцов. Он включает в себя широкий диапазон встроенных средств, включая выделение, разбиение на страницы и редактирование. К тому же он может быть расширен с помощью шаблонов. Огромным преимуществом GridView перед DataGrid (ASP.NET 1.x) является его поддержка сценариев без кода. Используя GridView, можно без написания кода решать множество распространенных задач, таких как перемещение по страницам и выделение. В DataGrid для получения тех же средств нужно было кодировать обработку событий.

1. **Как осуществляется привязка данных к элементам управления?**

В ASP.NET большинство элементов управления (включая TextBox, LinkButton, Image и многие другие) поддерживают привязку данных с одним значением (single-value). Такая привязка позволяет связать свойство элемента управления с источником данных, но элемент управления может отображать единственное значение. Для использования привязки одного значения создаются выражения привязки данных. Многие веб-элементы управления поддерживают привязку с многократными значениями (repeated-value), это значит, что они могут отображать наборы элементов (например, ListBox и GridView). Чтобы наполнить элемент управления данными, необходим метод элемента управления DataBind().

1. **В чем преимущество использования технологии ADO.NET в ASP.NET приложениях?**

Масштабируемость: при использовании DataSet работа происходит с отсоединенными наборами данных. Это означает, что вы используете соединение с источником данных очень короткое время. Во многих системах количество подключений к базам данных является самым узким местом в плане масштабируемости. И для этих систем ADO.NET является очень хорошим решением, резко повышающим их масштабируемость. Отключенный набор данных может использоваться несколькими частями программы (или пользователями) одновременно.

Независимость от источника данных: в ADO.NET DataSet действительно независим от источника данных, и изменение провайдера, с помощью которого заполняется DataSet, не влияет на функциональность DataSet.

Способность к взаимодействию: ADO.NET использует XML как стандартный формат передачи данных, программа, которой необходимо получить данные из компонента ADO.NET, не обязана сама быть компонентом ADO.NET. В общем случае она вообще может не быть Windows-программой. Единственное требование – эта программа должна понимать XML.

1. **Какими средствами технологии ADO.NET можно обеспечить прямой доступ к данным?**

Компоненты SqlCommand, SqlParameter, SqlDataAdapter, SqlDataReader предоставляют прямой доступ к считываемым данным.

Выводы

В ходе выполнения лабораторной работы было создано Web-приложение ASP.NET, взаимодействующих с базой данных PostgreSQL с помощью компонентов ADO.NET.

Приложение А

Листинг SQL

1. **CREATE** **TABLE** **IF** **NOT** **EXISTS** authors
2. (
3. id      **INTEGER** **PRIMARY** **KEY**,
4. **NAME**    varchar(40),
5. surname varchar(40),
6. dopname varchar(40),
7. dob     **DATE**
8. );
10. **CREATE** **TABLE** **IF** **NOT** **EXISTS** paintings
11. (
12. id        **INTEGER** **PRIMARY** **KEY**,
13. **NAME**      varchar(40),
14. author\_id **INTEGER**,
15. price     **FLOAT**,
16. **DATE**      **DATE**,
17. **CONSTRAINT** paintings **FOREIGN** **KEY** ("author\_id") **REFERENCES** authors ("id")
18. );
20. **INSERT** **INTO** authors VALUES('1','Enoch','Plumb','Enoch-dop','1875-11-17');
21. **INSERT** **INTO** authors VALUES('2','Melinda','Nash','Melinda-dop','1907-08-28');
22. **INSERT** **INTO** authors VALUES('3','Phillip','Ripley','Phillip-dop','1874-07-31');
23. **INSERT** **INTO** authors VALUES('4','Leslie','Moran','Leslie-dop','1842-03-11');
24. **INSERT** **INTO** authors VALUES('5','Ron','Knight','Ron-dop','1880-06-20');
25. **INSERT** **INTO** authors VALUES('6','Martin','Sylvester','Martin-dop','1845-10-12');;
26. **INSERT** **INTO** authors VALUES('7','Boris','Waterhouse','Boris-dop','1894-09-23');
27. **INSERT** **INTO** authors VALUES('8','Danny','Strong','Danny-dop','1851-09-15');;
28. **INSERT** **INTO** authors VALUES('9','Michael','Palmer','Michael-dop','1828-05-12');
29. **INSERT** **INTO** authors VALUES('10','Gina','Plant','Gina-dop','1882-08-30');
31. **INSERT** **INTO** paintings VALUES('1','David Copperfield',6,68762725.7008728,'1938-08-23');
32. **INSERT** **INTO** paintings VALUES('2','The Scarlet Letter',4,18794519.2505279,'1860-05-14');
33. **INSERT** **INTO** paintings VALUES('3',' Oedipus at Colonus',9,4495617.8617681,'1919-09-14');
34. **INSERT** **INTO** paintings VALUES('4','Nineteen Eighty Four',9,82891069.286675,'1860-02-03');
35. **INSERT** **INTO** paintings VALUES('5','Pride and Prejudice',5,17516094.0998219,'1917-11-21');
36. **INSERT** **INTO** paintings VALUES('6','Moby Dick',1,41558340.6378369,'1834-08-22');
37. **INSERT** **INTO** paintings VALUES('7',' Oedipus at Colonus',7,92085775.5269904,'1923-11-04');
38. **INSERT** **INTO** paintings VALUES('8','Oedipus',4,40120378.0593047,'1929-11-01');
39. **INSERT** **INTO** paintings VALUES('9','The Sun Also Rises',9,26443671.8517283,'1822-01-04');
40. **INSERT** **INTO** paintings VALUES('10','War and Peace',7,65048617.772308,'1892-03-21');

Приложение Б

WebForms1.aspx

<%@ Page Language="C#" MasterPageFile="/Site.Master" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs" Inherits="lab3.WebForm1" %>

<asp:Content ID="MainContent" ContentPlaceHolderID="MainContent" runat="server">

<h1>Картины</h1>

<p>На этой странице отображаются картины из БД</p>

<br/>

<asp:Table ID="table" runat="server" CssClass="table" HorizontalAlign="Center"></asp:Table>

<br/><br/>

<form id="form1" runat="server">

<div class="row">

<div class="column">

<h2>Добавление записи в БД</h2>

<asp:Label ID="Label1" runat="server"></asp:Label>

<p><asp:TextBox ID="name" runat="server" type="text" CssClass="form-control" placeholder="название" ></asp:TextBox></p>

<p><asp:TextBox ID="date" runat="server" type="date" CssClass="form-control" placeholder="дата"></asp:TextBox></p>

<p><asp:DropDownList ID="DropDownList1" runat="server" CssClass="form-control" placeholder="автор" ></asp:DropDownList></p>

<p><asp:TextBox ID="price" runat="server" type="number" CssClass="form-control" placeholder="цена" ></asp:TextBox></p>

<asp:Button ID="Button1" runat="server" Text="Отправить" CssClass="btn btn-success" OnClick="Button1\_Click" />

</div>

<div class="column">

<h2>Выполнение запросов к БД</h2>

<asp:Label ID="Label2" runat="server"></asp:Label>

<p><asp:TextBox ID="query" runat="server" type="text" CssClass="form-control" placeholder="Введите команду" ></asp:TextBox></p>

<asp:Button ID="Button2" runat="server" Text="Выполнить" CssClass="btn btn-warning" OnClick="Button2\_Click" />

</div>

</div>

</form>

</asp:Content>

WebForms1.aspx.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Diagnostics;

using System.Data;

using Npgsql;

using System.Text;

namespace lab3

{

public partial class WebForm1 : System.Web.UI.Page

{

private string connectionString;

private NpgsqlConnection conn;

protected void Page\_Load(object sender, EventArgs e)

{

connectionString = String.Format("Server = {0}; Database = {1}; Port = {2}; Username = {3}; Password = {4}",

"localhost", "postgres", "5432", "postgres", "12345");

conn = new NpgsqlConnection(connectionString);

conn.Open();

var command1 = new NpgsqlCommand("select id, name, surname from authors;", conn);

NpgsqlDataReader reader1 = null;

reader1 = command1.ExecuteReader();

while (reader1.Read())

{

ListItem lst = new ListItem(reader1[1].ToString() + " " + reader1[2].ToString(), reader1[0].ToString());

DropDownList1.Items.Insert(DropDownList1.Items.Count, lst);

}

reader1.Close();

var command = new NpgsqlCommand("select paintings.id, paintings.name, " +

"paintings.date, authors.name, authors.surname, paintings.price from paintings, " +

"authors WHERE paintings.author\_id = authors.id;", conn);

NpgsqlDataReader reader = null;

reader = command.ExecuteReader();

int numcols = reader.FieldCount;

string[] headerPhrases = new string[] {

"id",

"Название",

"Дата",

"Имя автора",

"Фамилия автора",

"Цена"

};

TableHeaderRow header = new TableHeaderRow();

for (int k = 0; k < headerPhrases.Length; k++)

{

TableHeaderCell cell = new TableHeaderCell();

cell.Text = headerPhrases[k];

header.Cells.Add(cell);

}

table.Rows.Add(header);

while (reader.Read())

{

TableRow row = new TableRow();

for (int i = 0; i < numcols; i++)

{

TableCell cell = new TableCell();

string textContent;

if (headerPhrases[i] == "Дата")

{

DateTime date = DateTime.Parse(reader[i].ToString());

textContent = date.ToString("D");

}

else if(headerPhrases[i] == "Цена")

{

float price = float.Parse(reader[i].ToString());

textContent = price.ToString("F");

}

else

{

textContent = reader[i].ToString();

}

cell.Text = textContent;

row.Cells.Add(cell);

}

table.Rows.Add(row);

}

reader.Close();

conn.Close();

}

protected void Button1\_Click(object sender, EventArgs e)

{

Label1.Text = "";

if (name.Text != "" && date.Text != "" && price.Text != "")

{

connectionString = String.Format("Server = {0}; Database = {1}; Port = {2}; Username = {3}; Password = {4}",

"localhost", "postgres", "5432", "postgres", "12345");

conn = new NpgsqlConnection(connectionString);

conn.Open();

var commandString = String.Format("insert into paintings (name, author\_id, price, date) " +

"values('{0}', {1}, {2}, '{3}');",

name.Text, DropDownList1.SelectedValue, price.Text, date.Text);

var command1 = new NpgsqlCommand(commandString, conn);

NpgsqlCommand sqlCmd = new NpgsqlCommand(commandString, conn);

try

{

sqlCmd.ExecuteNonQuery();

}

catch (Exception ex)

{

Label1.Text = "Ошибка добавления записи!";

}

conn.Close();

}

else

Label1.Text = "Заполните форму!";

if (Label1.Text == "")

Page.Response.Redirect(Page.Request.Url.ToString(), true);

}

protected void Button2\_Click(object sender, EventArgs e)

{

if (query.Text != "") {

Label2.Text = "";

try

{

connectionString = String.Format("Server = {0}; Database = {1}; Port = {2}; Username = {3}; Password = {4}",

"localhost", "postgres", "5432", "postgres", "12345");

conn = new NpgsqlConnection(connectionString);

conn.Open();

NpgsqlCommand command;

command = new NpgsqlCommand(query.Text, conn);

NpgsqlDataReader reader = null;

reader = command.ExecuteReader();

int numcols = reader.FieldCount;

table.Rows.Clear();

TableHeaderRow header = new TableHeaderRow();

for (int i = 0; i < numcols; i++)

{

TableHeaderCell cell = new TableHeaderCell();

cell.Text = reader.GetName(i);

header.Cells.Add(cell);

}

table.Rows.Add(header);

while (reader.Read())

{

TableRow row = new TableRow();

for (int i = 0; i < numcols; i++)

{

TableCell cell = new TableCell();

string textContent;

if (header.Cells[i].Text == "date")

{

DateTime date = DateTime.Parse(reader[i].ToString());

textContent = date.ToString("D");

}

else if (header.Cells[i].Text == "price")

{

float price = float.Parse(reader[i].ToString());

textContent = price.ToString("F");

}

else

{

textContent = reader[i].ToString();

}

cell.Text = textContent;

row.Cells.Add(cell);

}

table.Rows.Add(row);

}

reader.Close();

conn.Close();

}

catch (Exception ex)

{

Label2.Text = "Некорректная команда!";

}

}

else

Label2.Text = "Введите команду!";

//Page.Response.Redirect(Page.Request.Url.ToString(), true);

}

}

}