МИНСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ

ФЕДЕРАЦИИ

федеральное государственное бюджетное образовательное учреждение

высшего профессионального образования

«АЛТАЙСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Международный институт экономики, менеджмента и информационных систем

Кафедра цифровых технологий и бизнес – аналитики

ОТЧЕТ О ВЫПОЛНЕНИИ ПЕРВОГО ЗАДАНИЯ ПО УЧЕБНОЙ ПРАКТИКЕ

**(практики по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности)**

на кафедре цифровых технологий и бизнес-аналитики МИЭМИС ФГБОУ ВО АлтГУ

Выполнил студент

3 курса, 2009а группы

Д.C. Беляев

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Научный руководитель,

доцент, канд. физ. мат. наук

А. Ю. Юдинцев

Работа защищена

Оценка\_\_\_\_\_\_\_\_\_\_\_

« » \_\_\_\_\_\_\_\_\_\_\_ 2023г.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(подпись)

Барнаул 2023

СОДЕРЖАНИЕ

[1 РАБОТА С ХОСТИНГОМ 3](#_Toc139042815)

[1.1 Создание базы данных 3](#_Toc139042816)

[1.2 Состав и структура базы данных 5](#_Toc139042817)

[2 РАЗРАБОТКА ASP.NET ПРИЛОЖЕНИЯ 12](#_Toc139042818)

[2.1 Создание хранимых процедур 12](#_Toc139042819)

[2.2 Разработка приложения и размещение на somee.com 17](#_Toc139042820)

[3 РАЗРАБОТКА ASMX WEB-СЕРВИСОВ 26](#_Toc139042821)

[3.1 Создание и тестирование Web-сервисов 26](#_Toc139042822)

[3.2 Размещение сервиса на somee.com 32](#_Toc139042823)

[ПРИЛОЖЕНИЯ 34](#_Toc139042824)

## 1 РАБОТА С ХОСТИНГОМ

### 1.1 Создание базы данных

Перед созданием базы данных на хостинге somee.com была сделана резервная копия уже готовой базы данных из курсовой работы в панели управления другого хостинга, а позже загружена на диск (Рис. 1.1).

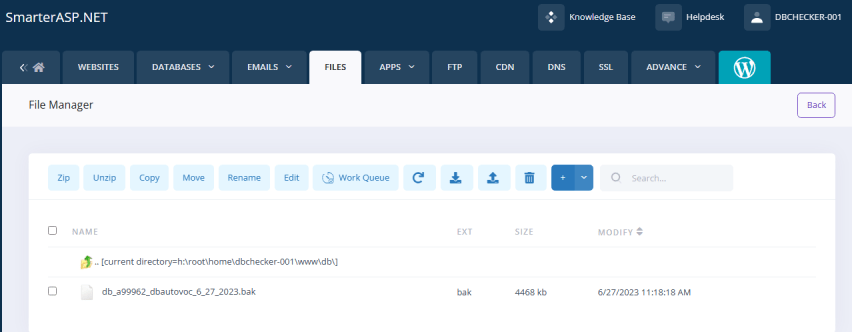


Рис. 1.1 – Созданная резервная копия

Далее создан аккаунт на somee.com, а также создана и настроена база данных, названная тем же именем, что была на другом хостинге (Рис. 1.2).

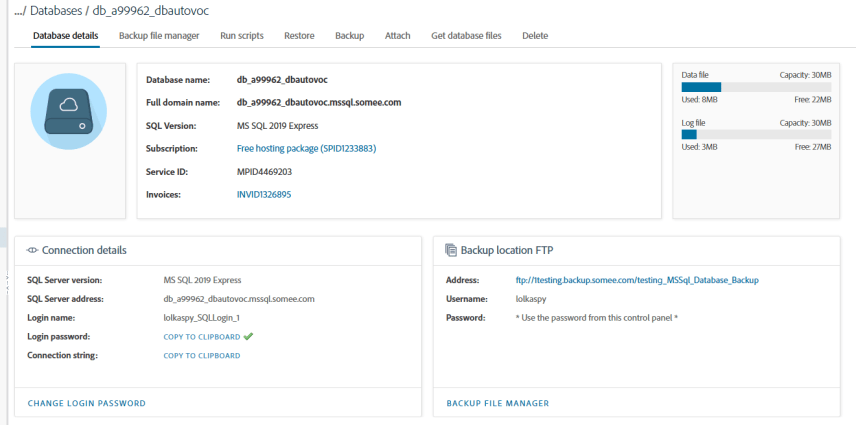


Рис. 1.2 – Созданная база данных на хостинге somee.com

А затем во вкладке «Restore» была восстановлена резервная копия базы данных (Рис. 1.3).

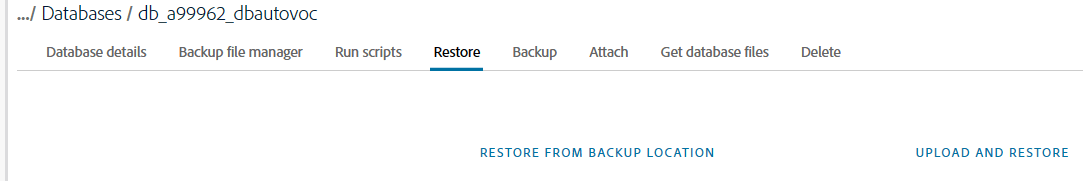


Рис. 1.3 – Восстановление резервной копии базы данных

Реквизиты для подключения к базе данных:

* Сервер: db\_a99962\_dbautovoc.mssql.somee.com
* База: db\_a99962\_dbautovoc
* Логин: lolkaspy\_SQLLogin\_1
* Пароль: q485bx4le2
* Строка подключения:

workstation id=db\_a99962\_dbautovoc.mssql.somee.com;packet size=4096;user id=lolkaspy\_SQLLogin\_1;pwd=q485bx4le2;data source=db\_a99962\_dbautovoc.mssql.somee.com;persist security info=False;initial catalog=db\_a99962\_dbautovoc

После восстановления резервной копии проведена проверка запросов на выборку всех таблиц из базы данных и на выборку из конкретной таблицы «Routes» (Рис. 1.4).

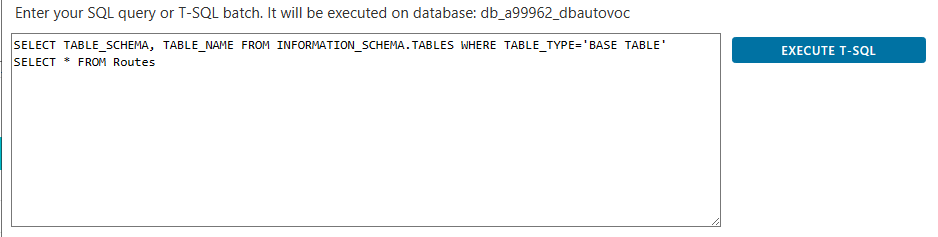


Рис. 1.4 – SQL-запросы на выборку

Результат выполнения на следующей странице (Рис. 1.5).

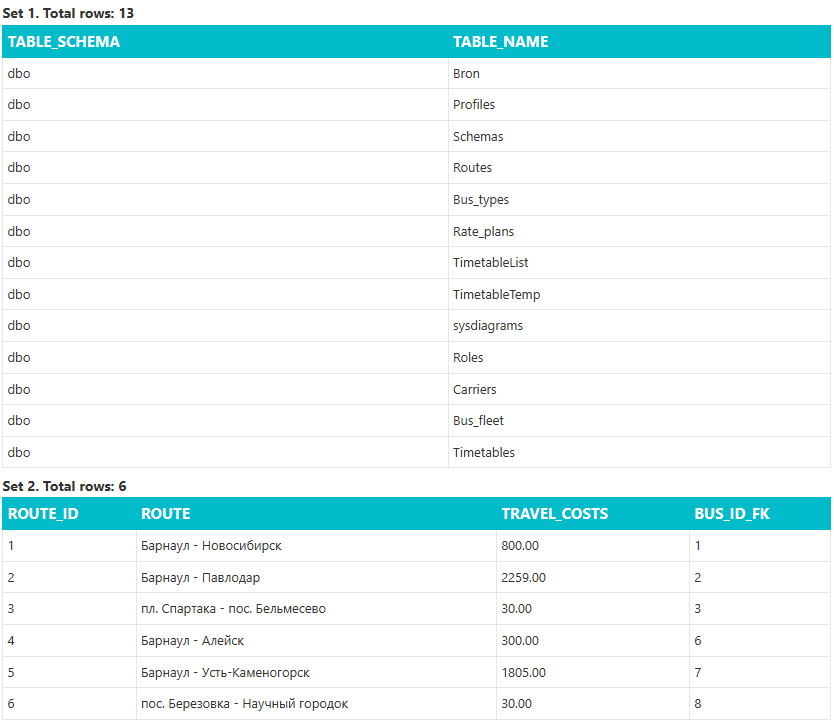


Рис. 1.5 – Результат выполнения запросов на выборку

Тем не менее был сгенерирован скрипт для создания базы данных и заполнения данными, который представлен в Приложении 1.

### 1.2 Состав и структура базы данных

Состав и структура базы данных, связи таблиц базы данных представлены на ER-диаграмме на следующей странице (Рис. 1.6).

В данной структуре есть 4 обособленных компонента:

1. Справочники «Profiles» и «Roles», доступные только администратору системы.
2. Справочники, необходимые для работы по оформлению билетов и расписаний.
3. Справочник «Bron», необходимый для работы по бронированию.
4. Справочник «Rate\_plans», обязательный для оформления проездных билетов.

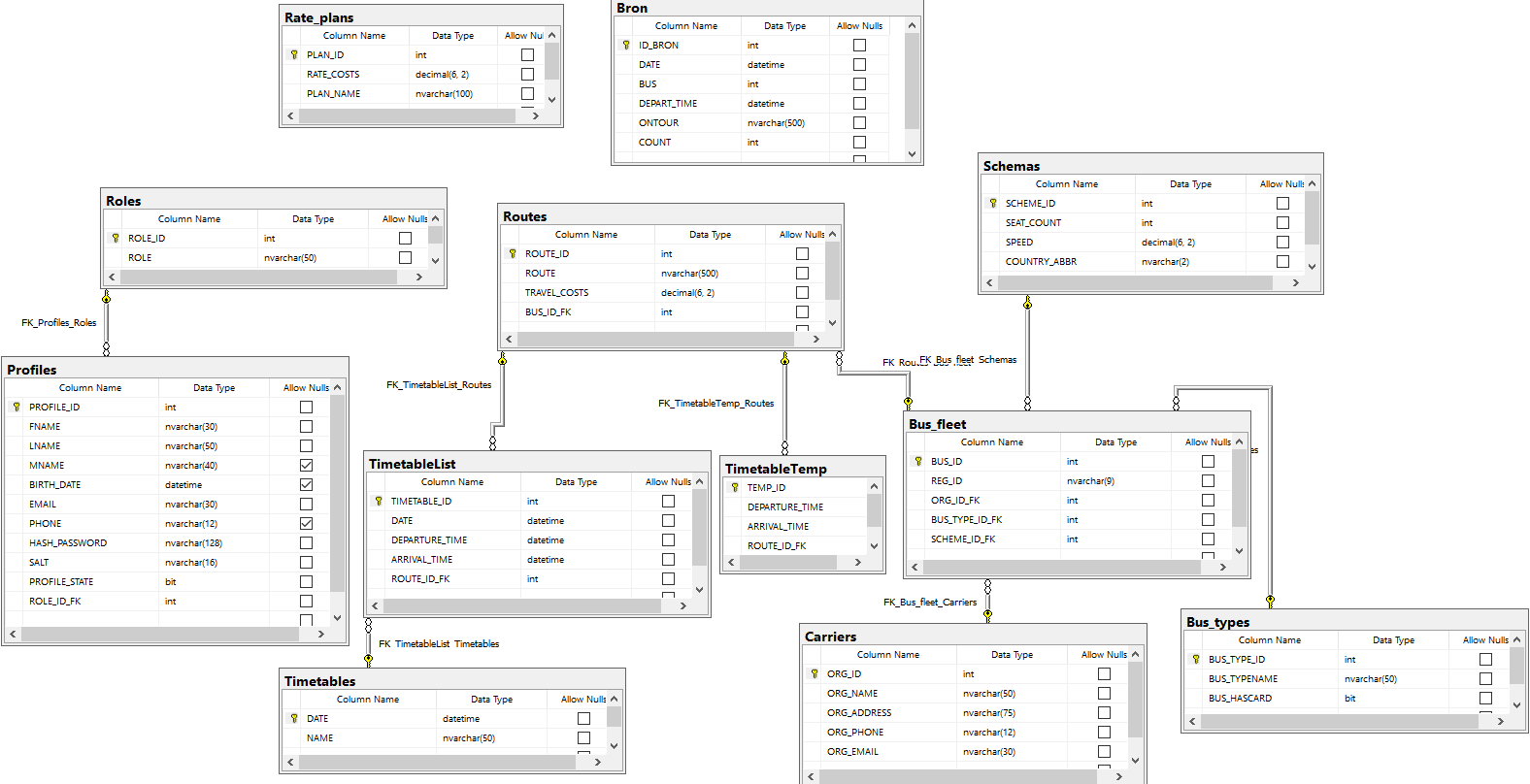


Рис. 1.6 – ER-диаграмма базы данных

Описание всех сущностей и связей будет оформлено в виде таблиц.

Первая сущность «Типы автобуса» представлена в Таблице 1.

Таблица 1 − Описание таблицы «Типы автобуса»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | BUS\_TYPE\_ID | Уникальный ИД типа автобуса | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | BUS\_TYPENAME | Наименование типа автобуса | Текстовый, nvarchar(50) | - | Да |
| Пользователь | BUS\_HASCARD | Возможность приобрести проездной билет | Логический, bit | - | Да |

Вторая сущность «Схемы автобусов» представлена в Таблице 2.

Таблица 2 − Описание таблицы «Схемы автобусов»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | SCHEME\_ID | Уникальный ИД схемы посадочных мест автобуса | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | SEAT\_COUNT | Количество посадочных мест | Числовой,  int | - | Да |
| Пользователь | SPEED | Скорость (км/ч) | Числовой,  decimal(6,2) | - | Да |
| Пользователь | COUNTRY\_ABBR | Код страны-производителя | Текстовый,  nvarchar(2) | - | Да |

Третья сущность «Компании-перевозчики» представлена в Таблице 3.

Таблица 3 − Описание таблицы «Компании-перевозчики»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Заполняется системой | ORG\_ID | Уникальный ИД компании-перевозчика | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | ORG\_NAME | Наименование организации | Текстовый,  nvarchar(50) | - | Да |

Продолжение Таблицы 3

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Пользователь | ORG\_ADDRESS | Юридический адрес | Текстовый,  nvarchar(75) | - | Да |
| Пользователь | ORG\_PHONE | Контактный номер телефона | Текстовый,  nvarchar(12) | - | Да |
| Пользователь | ORG\_EMAIL | Электронная почта | Текстовый,  nvarchar(30) | - | Да |

Четвертая сущность «Парк автобусов» оперирует данными предыдущих трёх сущностей.

Сама сущность представлена в Таблице 4.

Таблица 4 − Описание таблицы «Парк автобусов»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | BUS\_ID | Уникальный ИД автобуса | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | REG\_ID | Регистрационный номер автобуса | Текстовый,  nvarchar(9) | - | Да |
| Пользователь | ORG\_ID\_FK | ИД компании-перевозчика | Числовой,  int | Внешний | Да |
| Пользователь | BUS\_TYPE\_ID\_FK | ИД типа автобуса | Числовой,  int | Внешний | Да |
| Пользователь | SCHEME\_ID\_FK | ИД схемы посадочных мест автобуса | Числовой,  int | Внешний | Да |

Пятая сущность «Маршруты» оперирует данными всех предыдущих сущностей, включая сущность «Парк автобусов». Сущность представлена в Таблице 5.

Таблица 5 − Описание таблицы «Маршруты»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Заполняется системой | ROUTE\_ID | Уникальный ИД маршрута | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | ROUTE | Путь передвижения | Текстовый,  nvarchar(500) | - | Да |

Продолжение Таблицы 5

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Пользователь | TRAVEL\_COSTS | Стоимость поездки | Числовой,  decimal(6,2) | - | Да |
| Пользователь | BUS\_ID\_FK | ИД автобуса | Числовой,  int | Внешний | Да |

Главная сущность в БД «Лист расписания». Оперирует данными всех предыдущих сущностей, включая сущность «Маршруты». Представлена в Таблице 2.6.

Таблица 6 − Описание таблицы «Лист расписания»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Пользователь | TIMETABLE\_ID | Уникальный ИД записи расписания | Счетчик,  int(1,1) | Первичный | Да |
| Пользователь | DATE | Дата | Дата и время, datetime | Внешний | Да |
| Пользователь | DEPARTURE\_TIME | Время отправления | Дата и время, datetime | - | Да |
| Пользователь | ARRIVAL\_TIME | Время прибытия | Дата и время, datetime | - | Да |
| Пользователь | ROUTE\_ID\_FK | ИД маршрута | Числовой,  int | Внешний | Да |

Следующая сущность «Расписания» отвечает за отображение записей расписания в качестве единого расписания. Она описана в таблице 7.

Таблица 7 − Описание таблицы «Расписания»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Пользователь | DATE | Дата расписания | Счётчик, int(1,1) | Первичный | Да |
| Пользователь | NAME | Наименование | Текстовый,  nvarchar(50) | - | Да |

Сущность «Тарифные планы» отвечает за возможность оформления проездных билетов. Она представлена в таблице 8 на следующей странице.

Таблица 8 − Описание таблицы «Тарифные планы»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | RATE\_ID | Уникальный ИД плана | Счётчик, int(1,1) | Первичный | Да |
| Пользователь | RATE\_COSTS | Стоимость | Числовой,  decimal(6,2) | - | Да |
| Пользователь | PLAN\_NAME | Наименование типа плана | Текстовый,  nvarchar(100) | - | Да |

Следующими сущностями являются «Роли» и «Профили». Как раз эти сущности отвечают за предоставление прав пользования системой. Они представлены в таблицах 9. и 10. соответственно.

Таблица 9 − Описание таблицы «Роли»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | ROLE\_ID | Уникальный ИД роли | Счетчик, int(1,1) | Первичный | Да |
| Пользователь | ROLE | Наименование роли | Текстовый,  nvarchar(50) | - | Да |

Таблица 10 − Описание таблицы «Профили»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Заполняется системой | PROFILE\_ID | Уникальный ИД профиля | Счётчик, int(1,1) | Первичный | Да |
| Пользователь | FNAME | Имя | Текстовый,  nvarchar(30) | - | Да |
| Пользователь | LNAME | Фамилия | Текстовый,  nvarchar(50) | - | Да |
| Пользователь | MNAME | Отчество | Текстовый,  nvarchar(40) | - |  |
| Пользователь | BIRTH\_DATE | Дата рождения | Дата и время, datetime | - |  |
| Пользователь | EMAIL | Электронная почта | Текстовый,  nvarchar(30) | - | Да |
| Пользователь | PHONE | Номер телефона | Текстовый,  nvarchar(12) | - |  |

Продолжение Таблицы 10

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| Пользователь | HASH\_PASSWORD | Хэш пароля | Текстовый,  nvarchar(128) | - | Да |
| Пользователь | SALT | Соль | Текстовый,  nvarchar(16) | - | Да |
| Пользователь | PROFILE\_STATE | Статус профиля | Логический, bit | - | Да |
| Пользователь | ROLE\_ID\_FK | ИД роли | Числовой,  int | Внешний | Да |

В таблице 11.описана последняя сущность «Лист бронирования». Именно по ней формируется одноимённый документ в системе.

Таблица 11 − Описание таблицы «Лист бронирования»

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Режим  заполнения | Имя  атрибута | Назначение | Тип данных | Ключ | Обяз. для  заполнения |
| Заполняется системой | ID\_BRON | Уникальный ИД брони | Счётчик, int(1,1) | Первичный | Да |
| Пользователь | DATE | Дата | Дата и время, datetime | - | Да |
| Пользователь | BUS | Номер автобуса | Числовой,  int | - | Да |
| Пользователь | DEPART\_TIME | Время отправления | Дата и время, datetime | - | Да |
| Пользователь | ONTOUR | Путь | Текстовый,  nvarchar(500) | - | Да |
| Пользователь | COUNT | Количество мест | Числовой,  int | - | Да |

## 2 РАЗРАБОТКА ASP.NET ПРИЛОЖЕНИЯ

### 2.1 Создание хранимых процедур

Перед созданием приложения ASP.NET разработано несколько хранимых процедур для вывода расписания автобусов (Рис. 2.1). Первая процедура выводит расписание автобусов на конкретную дату с некоторыми подробностями. Вторая и третья процедуры выводят то же самое, но в XML или JSON форматах.

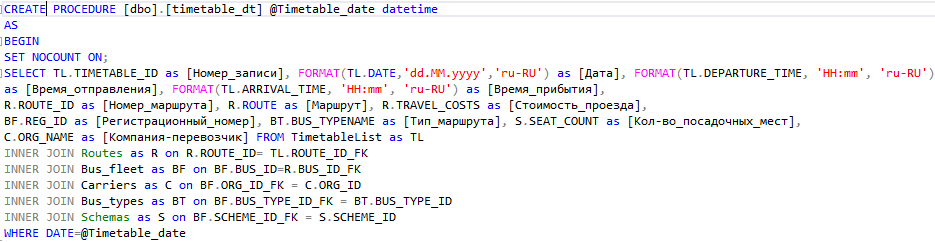


Рис. 2.1 – Создание хранимой процедуры

На Рисунке 2.2 дан результат выполнения первой процедуры в Microsoft SQL Server Management Studio.

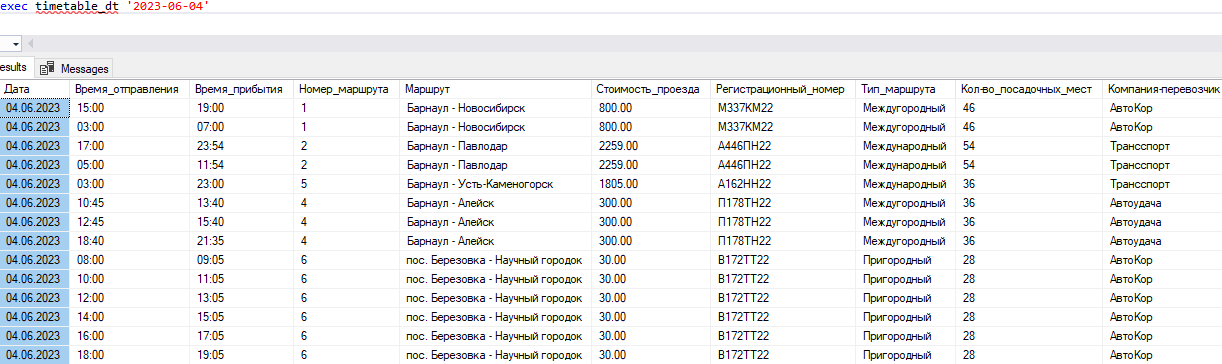


Рис. 2.2 – Выполнение хранимой процедуры

Далее код хранимой процедуры по выводу XML-документа был обновлён так, чтобы корневым элементом было наименование «Расписание», а все внутренние элементы как «Запись». Результат выполнения XML-процедуры показан на следующей странице на Рисунке 2.3. В этой процедуре использован параметр ELEMENTS для придания xml-документу элементной модели. Сама процедура выводит все записи расписания на конкретную дату, при этом дата является атрибутом тега «Расписание». Для процедуры было подготовлено представление, собранное из нескольких справочников (Рис. 2.4).

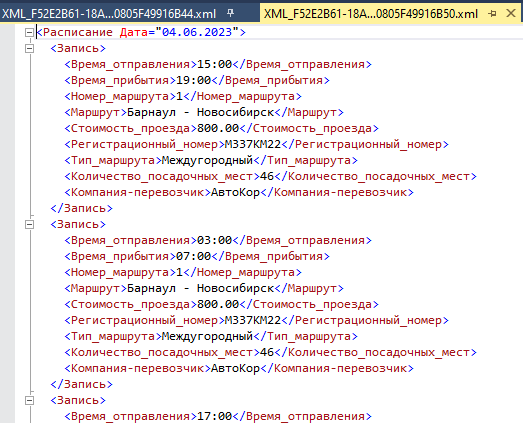


Рис. 2.3 – Результат выполнения обновлённой процедуры

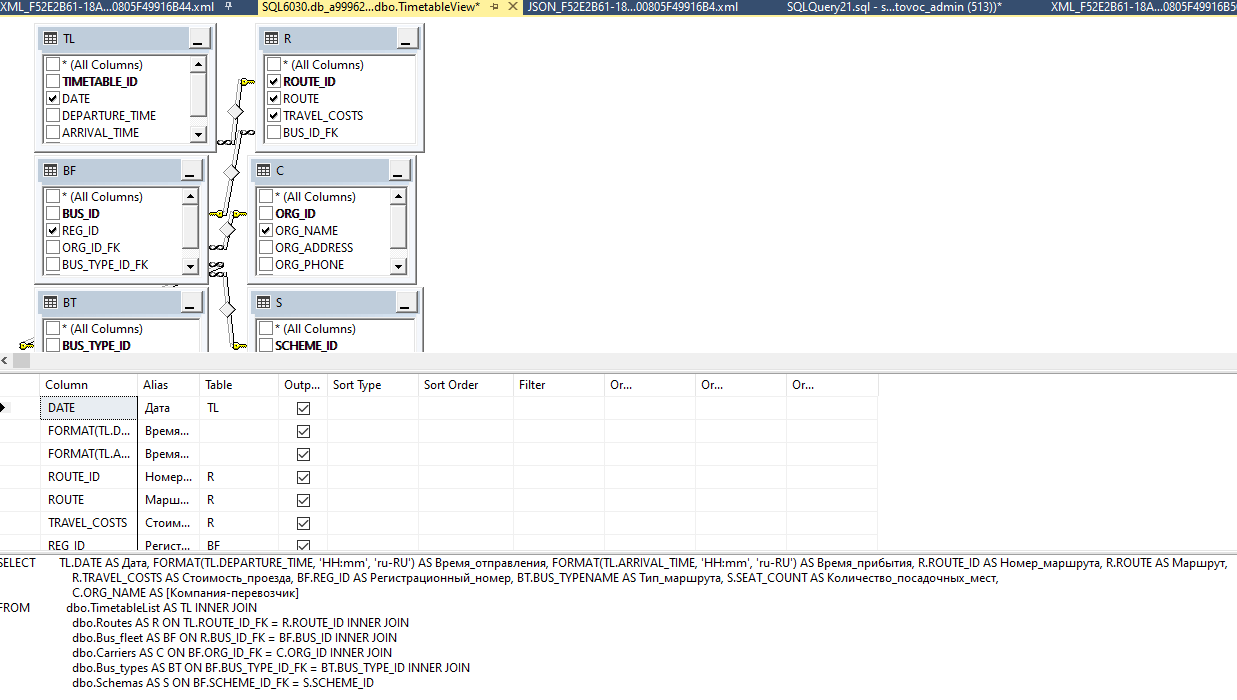


Рис. 2.4 – Конструктор представления

Также были созданы процедуры для вывода всех расписаний (документов), всех организаций (справочника) и отдельной организации по её коду.

Результат одной из процедур «carriers\_byid\_xml» для вывода организации по её коду показан на Рисунке 2.5.

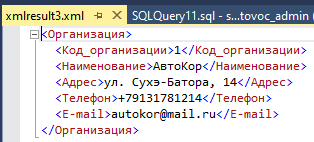


Рис. 2.5 – Вывод организации по её коду в формате xml

На следующем шаге необходимо сделать процедуры добавления, изменения и удаления записей в справочнике и документе.

Процедуры для добавления организаций будут называться new\_carrier\_xml и new\_carrier\_json, а для добавления расписаний – new\_timetable\_xml и new\_timetable\_json.

Процесс создания процедуры new\_carrier\_xml на Рисунке 2.6, а на Рисунке 2.7 результат выполнения следующей команды:

exec new\_carrier\_xml N'<Организация ORG\_NAME="Случайное название" ORG\_ADDRESS="Мало-Тобольская" ORG\_PHONE="89161729099" ORG\_EMAIL="organization@mail.ru"></Организация>'

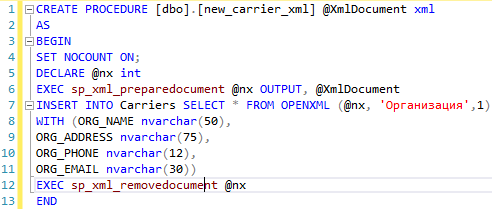


Рис. 2.6 – Создание хранимой процедуры по добавлению записи

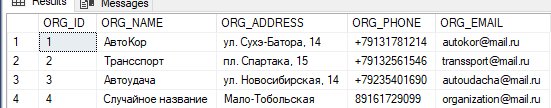


Рис. 2.7 – Выполнение хранимой процедуры по добавлению записи

Далее будут созданы процедуры для обновления данных, их четыре:

update\_carrier\_json, update\_carrier\_xml, update\_timetable\_json, update\_timetable\_xml.

На Рисунке 2.8 представлен процесс создания процедуры update\_carrier\_json, а на Рисунке 2.9 на следующей странице результат выполнения данной команды:

exec update\_carrier\_json N'[{"ORG\_ID":"4", "ORG\_NAME":"Случайное название 2", "ORG\_ADDRESS":"Мало-Тобольская, 15", "ORG\_PHONE":"89161729088", "ORG\_EMAIL":"organization2@mail.ru"}]'

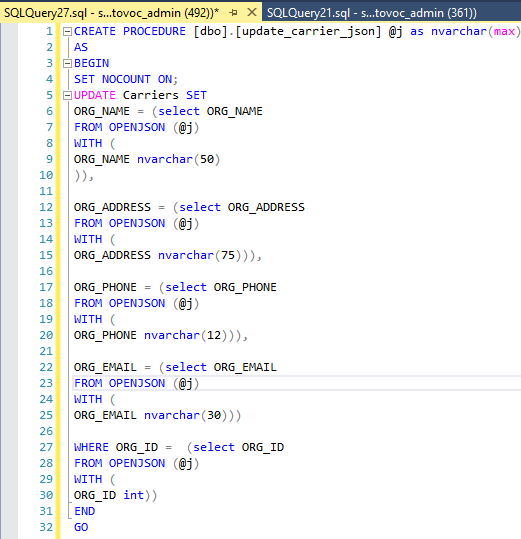


Рис. 2.8 – Создание хранимой процедуры по обновлению записи

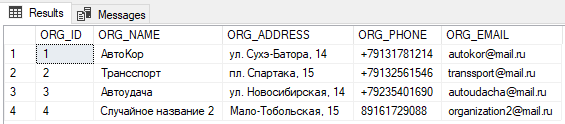


Рис. 2.9 – Выполнение хранимой процедуры по обновлению записи

Последние процедуры нужны для удаления данных, по выбранному коду записи, их также четыре:

delete\_carrier\_json, delete\_carrier\_xml, delete\_ttrecord\_json, delete\_ttrecord\_xml.

На Рисунке 2.10 показан процесс создания процедуры delete\_carrier\_xml, а на Рисунке 2.11 отображён результат выполнения следующей команды:

exec delete\_carrier\_xml N'<Организация ORG\_ID="4"/>'

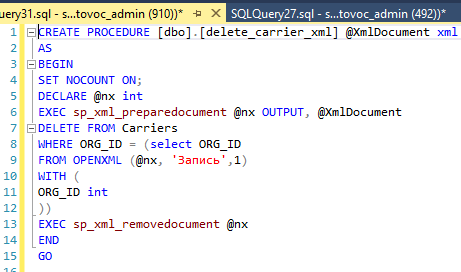


Рис. 2.10 – Создание хранимой процедуры по удалению записи

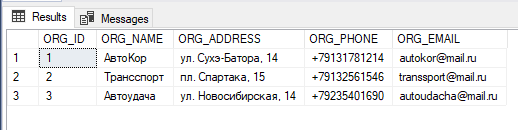


Рис. 2.11 – Выполнение хранимой процедуры по удалению записи

Полный код каждой из процедур и представления размещён в Приложении 2.

### 2.2 Разработка приложения и размещение на somee.com

После разработки необходимых процедур создан новый пустой проект ASP.NET. В файл Web.config добавлена нужная строка соединения (Рис. 2.12).

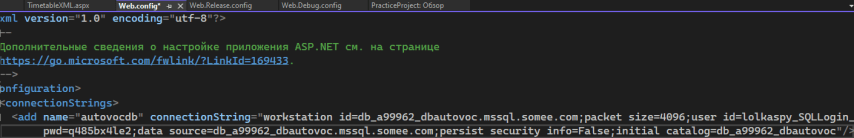


Рис. 2.12 – Строка соединения

Далее были созданы две веб-формы TimetableJSON.aspx и TimetableXML.aspx. Процесс разработки первого веб-приложения представлен на Рисунке 2.13.

Также были созданы следующие веб-формы: TimetablesXML.aspx и TimetablesJSON.aspx для вывода всех расписаний в виде XML и JSON соответственно, CarriersXML.aspx и CarriersJSON.aspx для вывода всех организаций и CarrierByIdXML.aspx и CarrierByIdJSON.aspx для вывода организации по её коду в форматах XML и JSON.

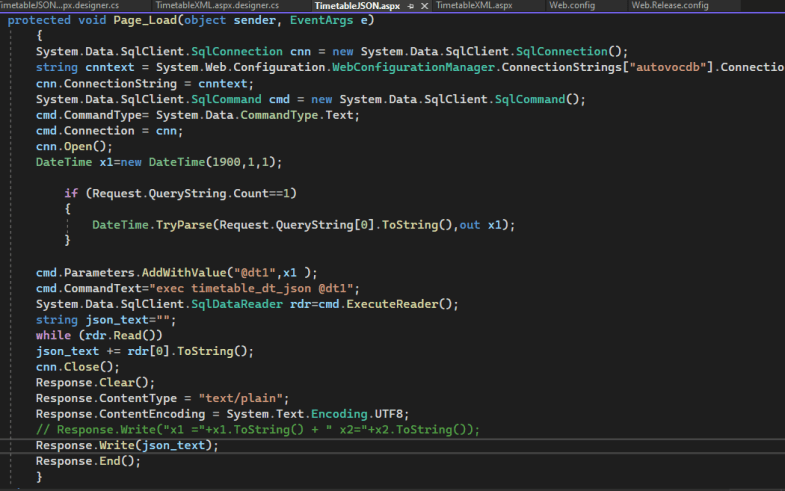


Рис. 2.13 – Разработка приложения TimetableJSON.aspx

После того, как приложения разработаны, их необходимо переместить в «File Manager» веб-сайта, созданном на somee.com. Внутри «File Manager» была создана директория «ASPNETProject». Результат перемещения на Рисунке 2.14.

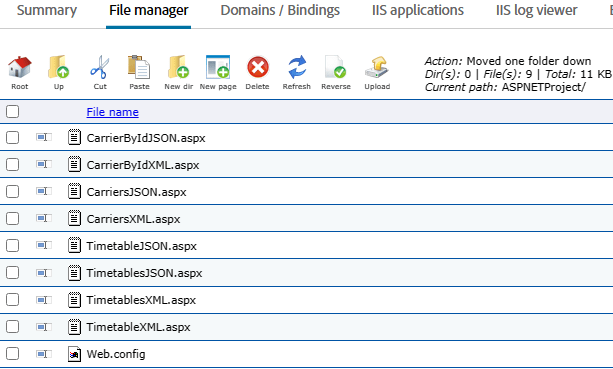


Рис. 2.14 – Каталог с приложениями на somee.com

Были введены следующие адреса в адресную строку бразуера:

1. [webservicesaspnet.somee.com/ASPNETProject/TimetableXML.aspx?dt1=2023-06-04](http://webservicesaspnet.somee.com/ASPNETProject/TimetableXML.aspx?dt1=2023-06-04)
2. [webservicesaspnet.somee.com/ASPNETProject/TimetableJSON.aspx?dt1=2023-06-04](http://webservicesaspnet.somee.com/ASPNETProject/TimetableJSON.aspx?dt1=2023-06-04)

Результат выполнения приложения по выводу расписания на конкретную дату XML на Рисунке 2.15, а по выводу JSON – на Рисунке 2.16 на следующей странице.

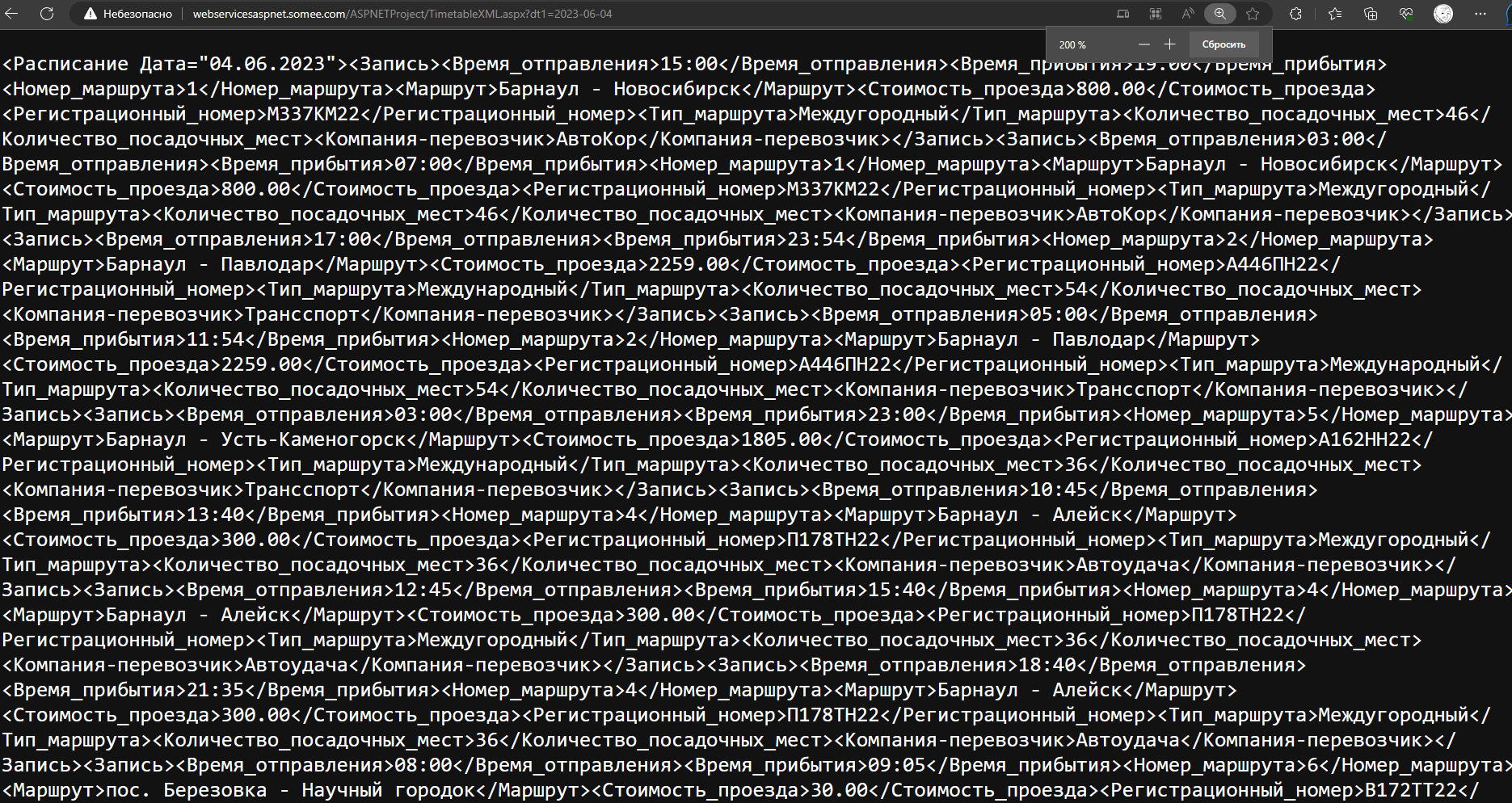


Рис. 2.15 – Работа приложения по выводу XML на сервере

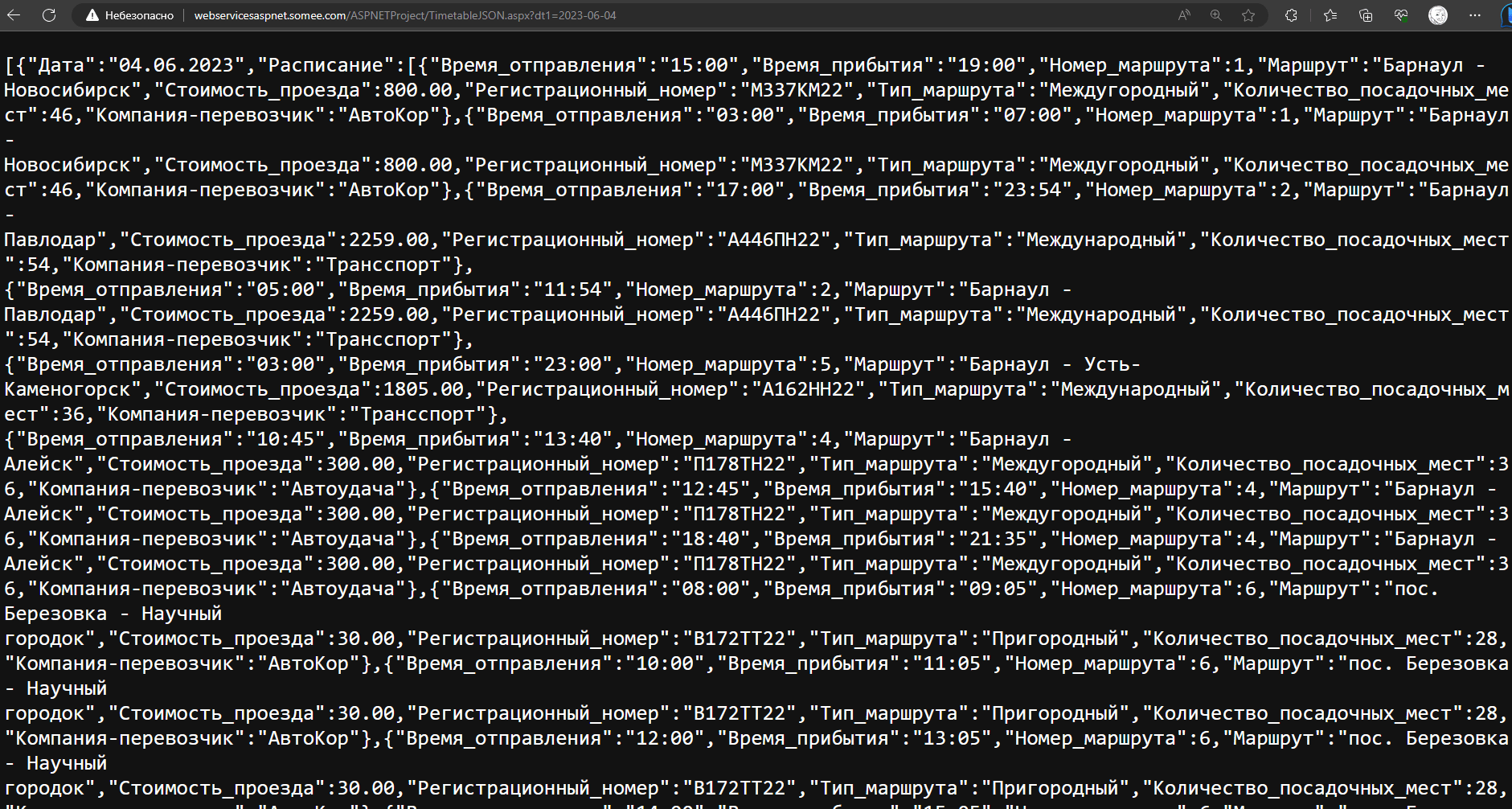


Рис. 2.16 – Работа приложения по выводу JSON на сервере

Следующие два приложения выводят все расписания в виде XML или JSON. Они доступны по следующим адресам:

1. [webservicesaspnet.somee.com/ASPNETProject/TimetablesXML.aspx](http://webservicesaspnet.somee.com/ASPNETProject/TimetablesXML.aspx)
2. [webservicesaspnet.somee.com/ASPNETProject/TimetablesJSON.aspx](http://webservicesaspnet.somee.com/ASPNETProject/TimetablesJSON.aspx)

Результат работы первого приложения показан на Рисунке 2.17, а результат работы второго приложения на Рисунке 2.18 на следующей странице.

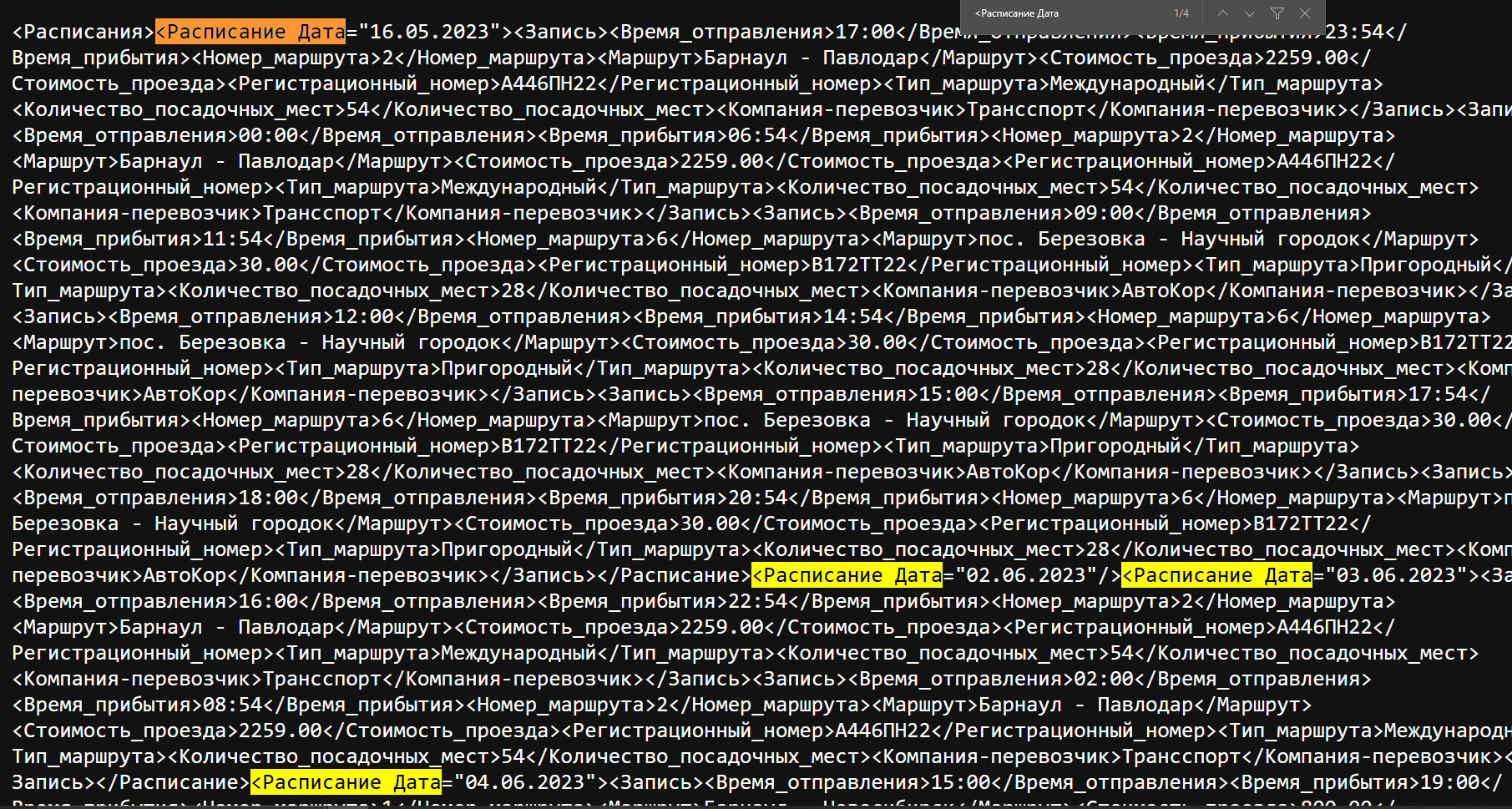


Рис. 2.17 – Вывод всех расписаний в виде XML

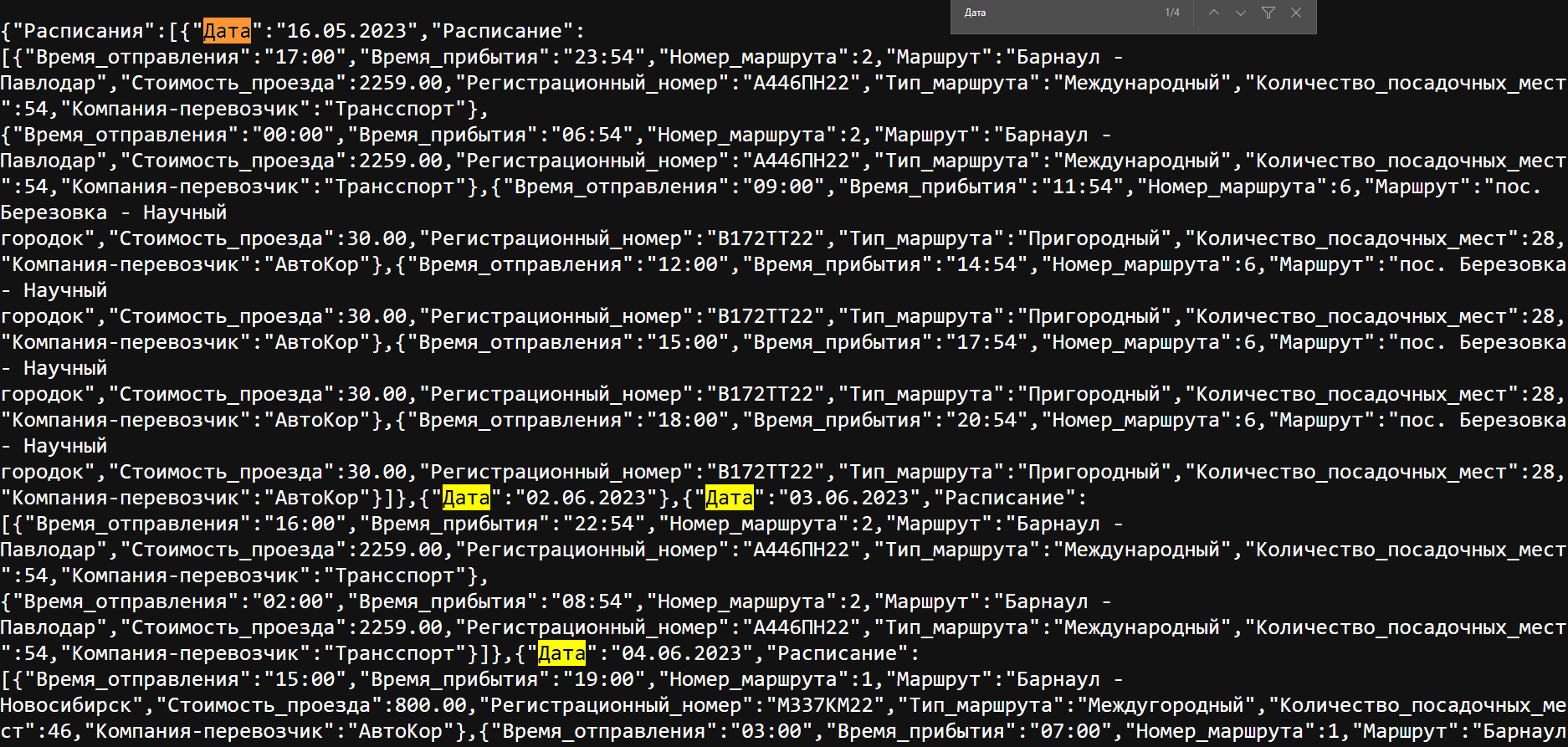


Рис. 2.18 – Вывод всех расписаний в виде JSON

Следующие два приложения выводят список организаций в формате XML или JSON. Они расположены на следующих адресах:

1. [webservicesaspnet.somee.com/ASPNETProject/CarriersXML.aspx](http://webservicesaspnet.somee.com/ASPNETProject/CarriersXML.aspx)
2. [webservicesaspnet.somee.com/ASPNETProject/CarriersJSON.aspx](http://webservicesaspnet.somee.com/ASPNETProject/CarriersJSON.aspx)

Результат работы первого приложения на Рисунке 2.19, а результат второго – на Рисунке 2.20.

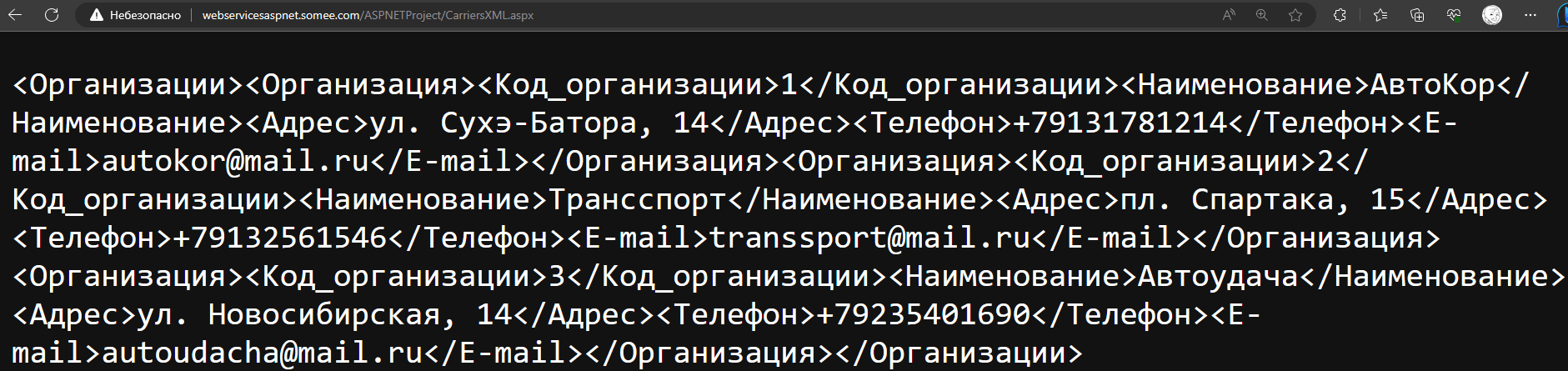


Рис. 2.19 – Вывод всех организаций в виде XML

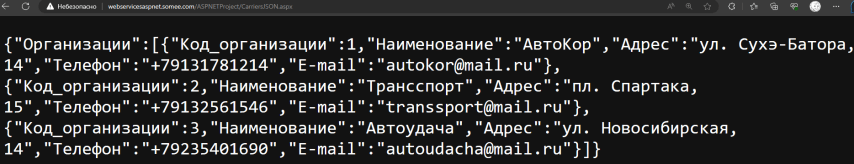


Рис. 2.20 – Вывод всех организаций в виде JSON

Последние два приложения выводят данные об организации по её коду и доступны на следующих адресах:

1. [webservicesaspnet.somee.com/ASPNETProject/CarrierByIdXML.aspx?id=2](http://webservicesaspnet.somee.com/ASPNETProject/CarrierByIdXML.aspx?id=2)
2. [webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx?id=3](http://webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx?id=3)

В первом приложении использован код 2, результат работы на Рисунке 2.21, а во втором приложении использован код 3, результат на Рисунке 2.22.

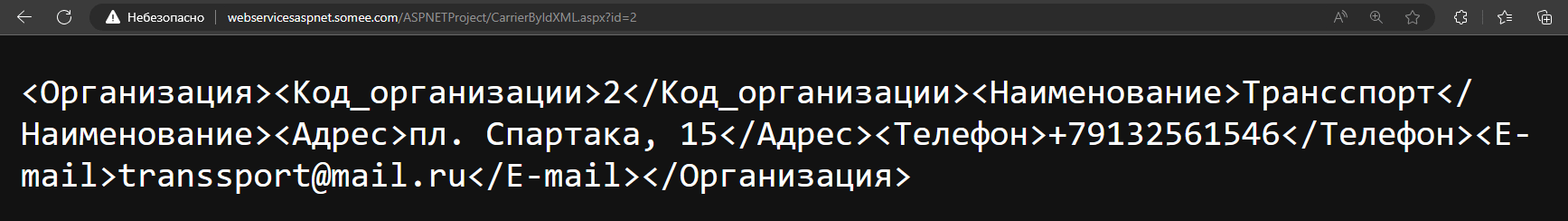


Рис. 2.21 – Работа приложения по выводу JSON на сервере

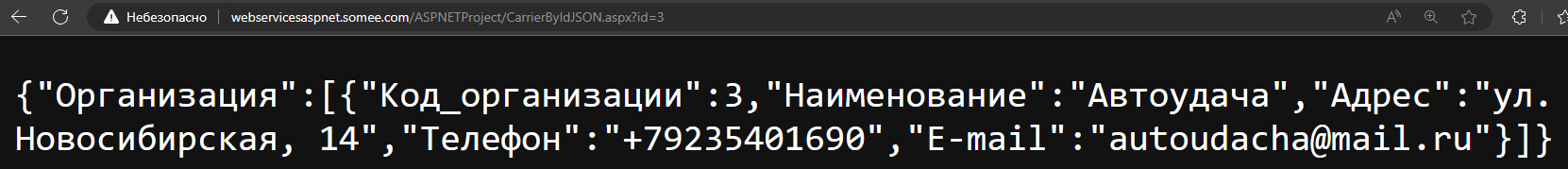


Рис. 2.22 – Работа приложения по выводу JSON на сервере

Также для примера были введены следующие адреса:

1. [webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx](http://webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx)
2. [webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx?id=4](http://webservicesaspnet.somee.com/ASPNETProject/CarrierByIdJSON.aspx?id=4)

В случае, если параметр отсутствует, выводится факт отсутствия данных и совет при работе с приложением (Рис. 2.23), а если записи, удовлетворяющей условию, нет, то приложение выдаёт только сообщение о том, что данные отсутствуют (Рис. 2.24).

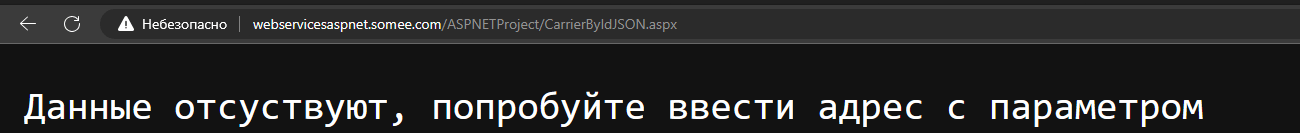


Рис. 2.23 – Работа приложения по выводу JSON на сервере

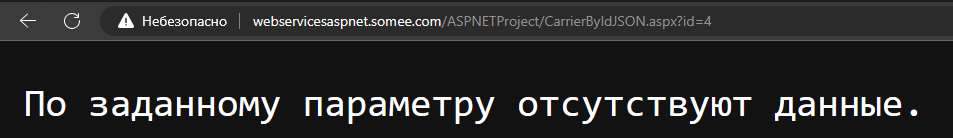


Рис. 2.24 – Работа приложения по выводу JSON на сервере

Таким образом были созданы приложения для вывода документа, а именно расписания автобусов на конкретную дату в форматах XML и JSON; вывода всех расписаний автобусов в форматах XML и JSON; вывода всех элементов справочника и вывода конкретного элемента по заданному параметру в форматах XML и JSON.

Добавление, изменение и удаление записей также будет реализовано в виде ASP.NET приложений, которые принимают параметры.

Количество приложений равно 12, столько же, сколько и процедур, созданных на втором шаге.

Полный список наименований приложений такой:

* DeleteCarrierXML.aspx
* DeleteCarrierJSON.aspx
* DeleteTTRecordXML.aspx
* DeleteTTRecordJSON.aspx
* AddTTRecordXML.aspx
* AddTTRecordJSON.aspx
* AddCarrierXML.aspx
* AddCarrierJSON.aspx
* UpdateTTRecordXML.aspx
* UpdateTTRecordJSON.aspx
* UpdateCarrierXML.aspx
* UpdateCarrierJSON.aspx

Также для каждого приложения создана отдельная html-страница, где передаётся параметр в виде строки Base64. Пример страницы для DeleteCarrierXML.aspx представлен на Рисунке 2.25.

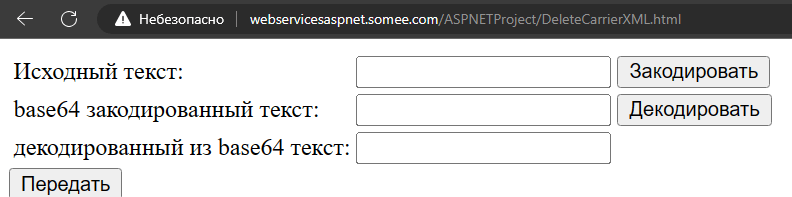


Рис. 2.25 – Пример страницы по передаче параметров

Далее будут рассмотрены только три приложения, так как код не сильно видоизменён в каждом отдельном случае.

Удалим запись с помощью приложения DeleteCarrierXML.aspx, используя соответствующую html-страницу. Введём следующий параметр в первое поле и декодируем (Рис. 2.26, стр. 24):

<Организация ORG\_ID="10" />

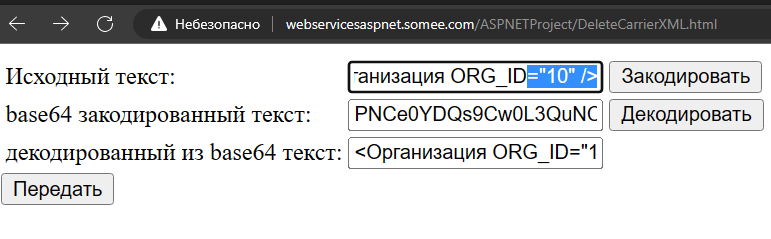


Рис. 2.26 – Ввод параметра для удаления записи

Результат передачи параметра изображён на Рисунке 2.27.

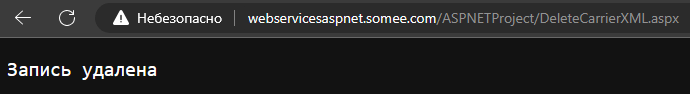


Рис. 2.27 – Результат, выдаваемый при удалении записи

Добавим новую запись с помощью приложения AddTTRecordJSON.aspx. Введём следующий параметр в первое поле и закодируем, а затем передадим в приложение (Рис. 2.28):

[{"DEPARTURE\_TIME":"16:15","ARRIVAL\_TIME":"20:15","ROUTE\_ID\_FK":"3"}]

Результат выполнения на Рисунке 2.29, а на Рисунке 2.30 на следующей странице выборка из таблицы, в которую была помещена запись (с кодом 65).

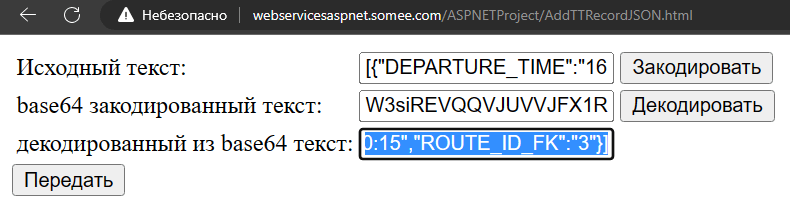


Рис. 2.28 – Ввод параметра для добавления записи

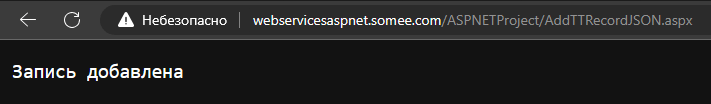


Рис. 2.29 – Результат, выдаваемый при добавлении записи

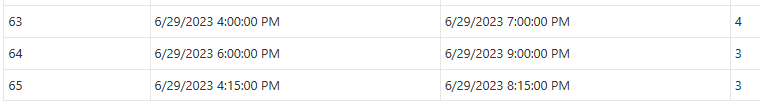


Рис. 2.30 – Проверка таблицы с записями расписаний после добавления

Последнее приложение на очереди – UpdateCarrierJSON. Введём следующий параметр в первое поле (Рис. 2.31):

[{"ORG\_ID":"6", "ORG\_NAME":"Неслучайное название", "ORG\_ADDRESS":"Московская, 27а", "ORG\_PHONE":"89261239281", "ORG\_EMAIL":"organization6@gmail.com"}]

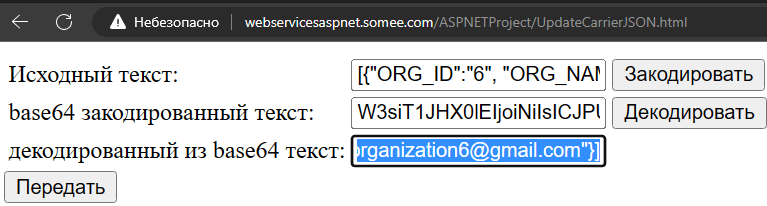


Рис. 2.31 – Ввод параметра для обновления записи

Результат после передачи параметра можно наблюдать на Рисунке 2.32, а изменения в записи с кодом 6 на Рисунке 2.33.

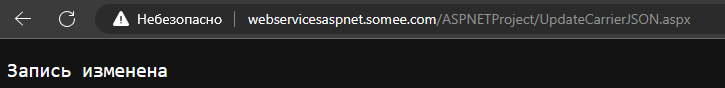


Рис. 2.32 – Результат, выдаваемый при обновлении записи

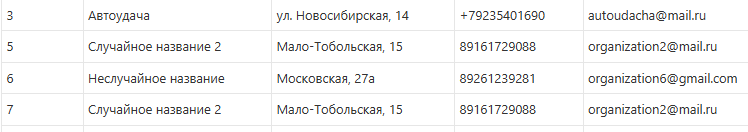


Рис. 2.33 – Проверка таблицы с записями организаций после обновления

Полные коды программ и пример одной из HTML-страниц размещены в Приложении 3.

## 3 РАЗРАБОТКА ASMX WEB-СЕРВИСОВ

### 3.1 Создание и тестирование Web-сервисов

На последнем шаге необходимо создать ASMX Web-сервис, поддерживающий несколько методов для получения данных, а также редактирования данных с помощью XML и JSON.

В этот раз опять используется пустой шаблон ASP.NET приложения. Здесь необходимо добавить новую веб-службу (ASMX). Результат создания веб-службы на Рисунке 3.1.

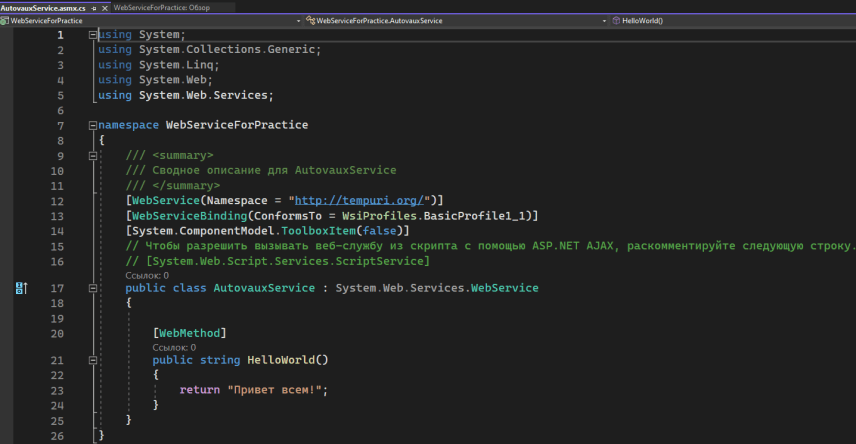


Рис 3.1 – Результат создания веб-службы

По умолчанию здесь есть один метод «HelloWorld», который выводит сообщение «Привет всем!».

Перед тем, разработать свои методы, необходимо добавить строку подключения в файл Web.config, а также добавить тег webServices для того, чтобы сервис мог работать по методам POST и GET. Результат представлен на Рисунке 3.2 на следующей странице.

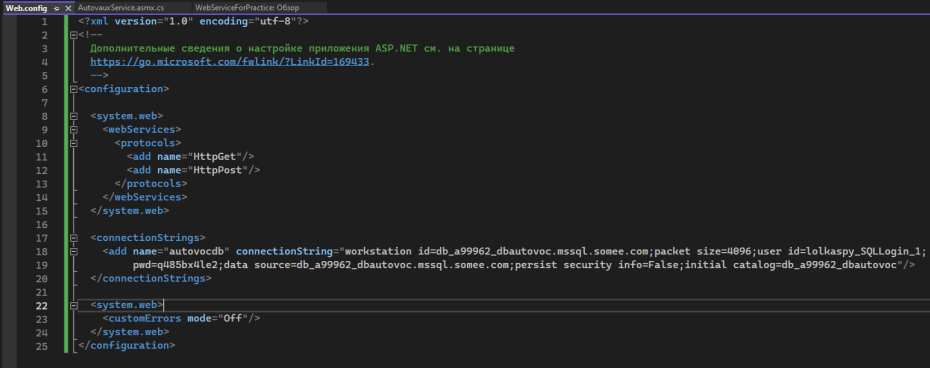


Рис 3.2 – Изменение конфигурационного файла

Для вывода информации в виде JSON необходимо к проекту подключить службу NewtonSoft.Json. Это удалось сделать благодаря добавлению nuget-пакета к проекту.

В целом создано 8 методов для вывода информации в виде XML или JSON, список методов для вывода информации представлен на Рисунке 3.3.

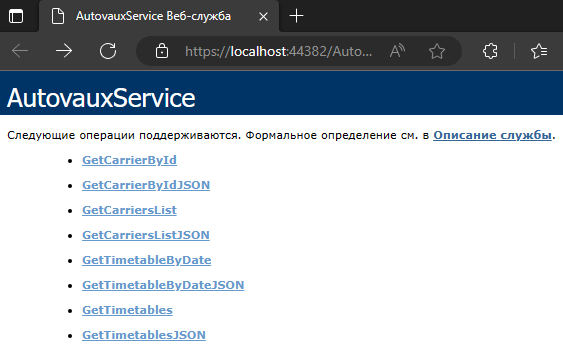


Рис 3.3 – Список методов по выводу информации

Примеры работы метода GetTimetablesJSON представлены на Рисунке 3.4 и Рисунке 3.5 на следующей странице.

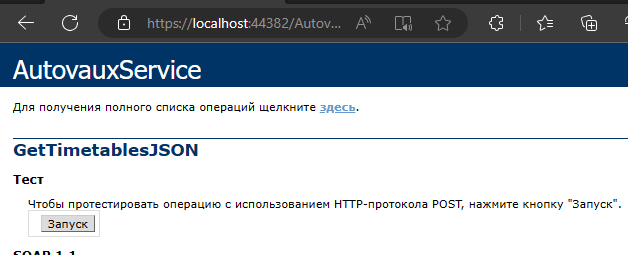


Рис 3.4 – Процесс запуска метода GetTimetablesJSON

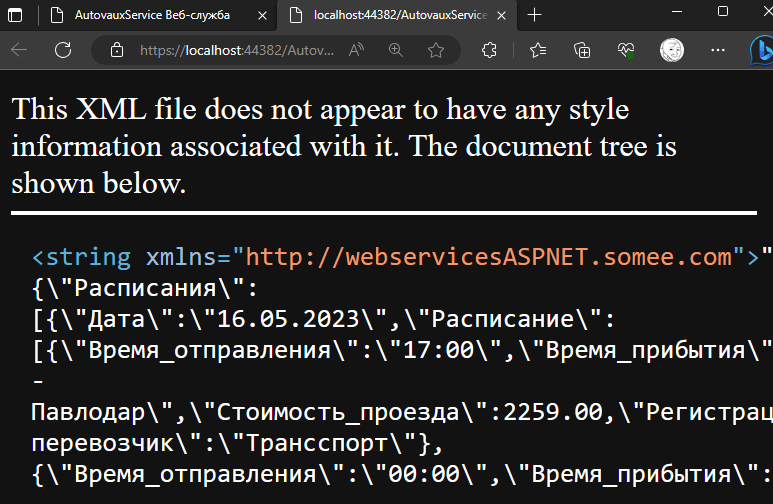


Рис 3.5 – Результат работы метода GetTimetables.JSON

Также создано несколько методов для редактирования справочников и документов. Полный список методов изображён на Рисунке 3.6 на следующей странице.

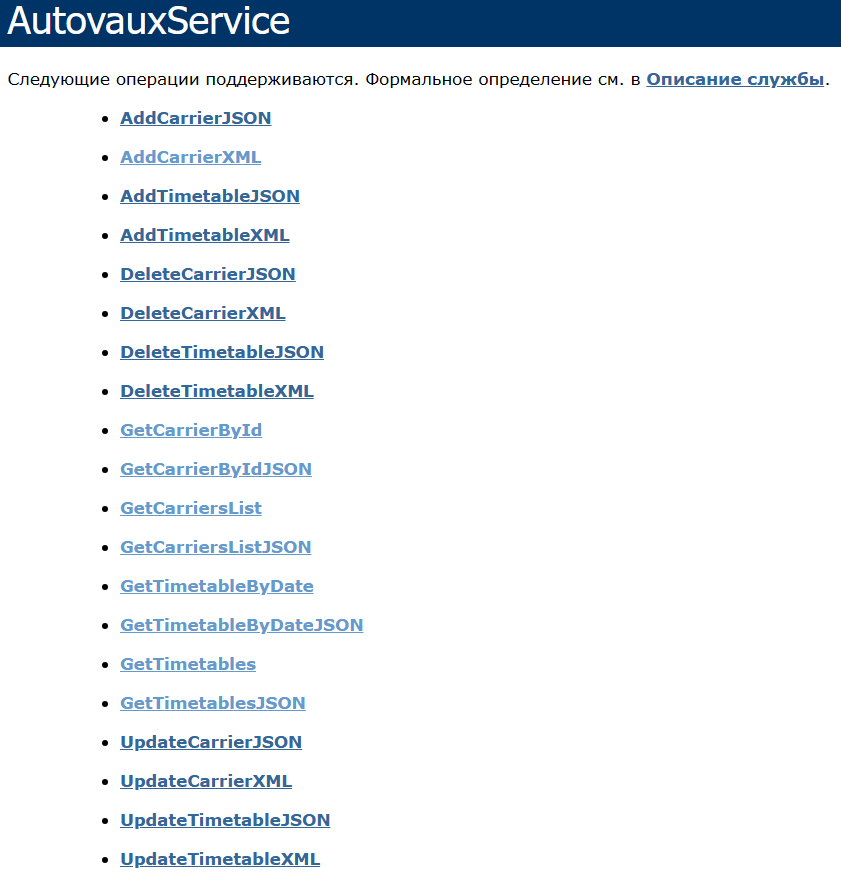


Рис 3.6 – Список методов веб-сервиса

В каждый из методов по редактированию справочника или документа в качестве параметра принимается Base-64 строка некоторого xml или json текста. Для её формирования можно воспользоваться ранее разработанной HTML-страницей.

Строка для кодирования будет такой:

<Организация ORG\_NAME="Случайное название ff" ORG\_ADDRESS="Мало-Тобольский, 27" ORG\_PHONE="89161729011" ORG\_EMAIL="arganization@mail.ru"></Организация>

После формирования строки, её можно передать в качестве параметра, результаты на Рисунке 3.7 и на Рисунке 3.8.

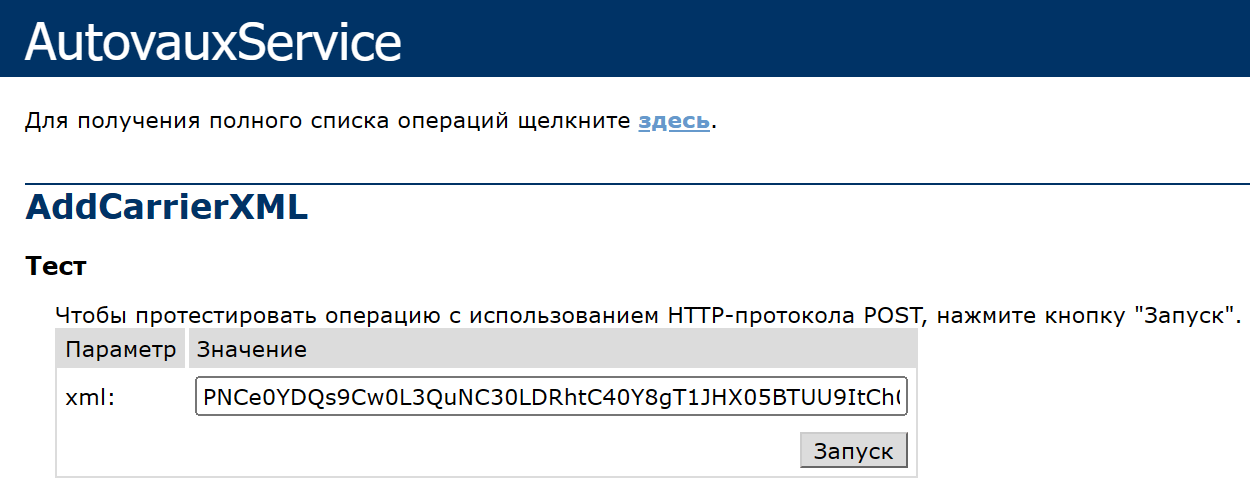


Рис. 3.7 – Процесс запуска метода AddCarrierXML

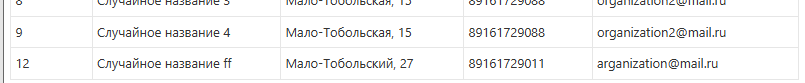


Рис. 3.8 – Результат в таблице после выполнения метода AddCarrierXML

Также можно продемонстрировать работу метода с помощью GET-запроса в командной строке. Для примера передадим параметр в метод UpdateCarrierJSON.

Используем для этого такой параметр:

[{"ORG\_ID":"7", "ORG\_NAME":"название для рисунка", "ORG\_ADDRESS":"Мало-Тобольская, 15", "ORG\_PHONE":"89161729088", "ORG\_EMAIL":"organization10@mail.ru"}]

Результат показан на Рисунке 3.9 и на Рисунке 3.10 на следующей странице.

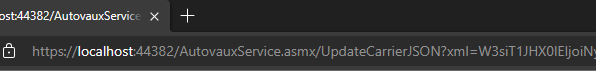


Рис. 3.9 – Процесс запуска метода в командной строке с GET-запросом

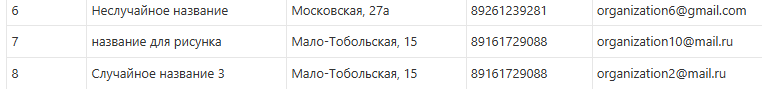


Рис. 3.10 – Результат в таблице после выполнения метода UpdateCarrierJSON

По факту методы выполняют то же самое, что и приложения из предыдущего шага практического задания. Однако в случае методов по редактированию никакого результата на странице не отображается.

Поэтому далее будет разработана универсальная клиентская часть в виде html-страницы.

Здесь есть возможность ввести адрес любого сервиса, закодировать в base64 некоторые параметры (для редактирования данных) и получить некоторый ответ.

Пример вызова метода из веб-сервиса изображён на Рисунке 3.11 на следующей странице.

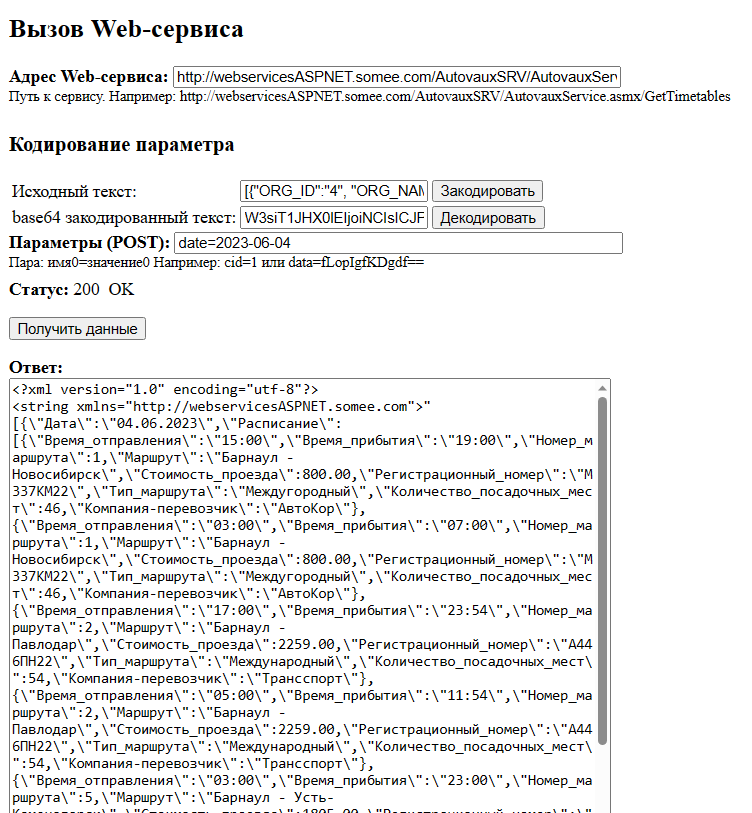


Рис. 3.11 – Результат вызова одного из методов сервиса

Полный код веб-сервиса и код клиентской части размещён в Приложении 4.

### 3.2 Размещение сервиса на somee.com

Перед размещением сервиса на хостинге необходимо, чтобы новый каталог работал как приложение, поэтому следует превратить его в приложение во вкладке IIS applications. Процесс конвертирования на Рисунке 3.12.

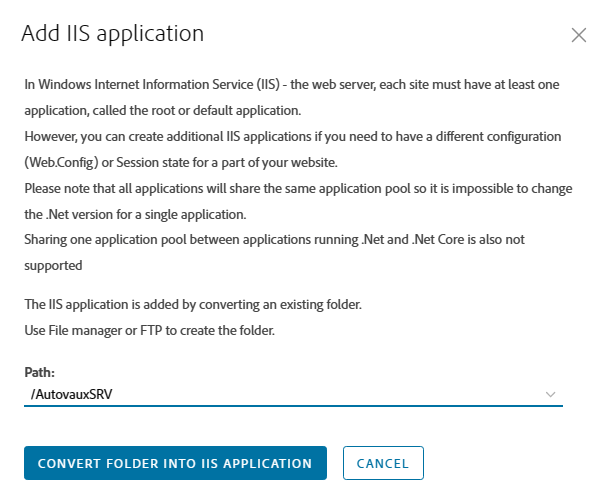


Рис. 3.12 – Конвертирование директории в приложение

Далее помещаем весь проект, созданный в Visual Studio на хостинг в директорию «AutovauxSRV».

Результат перемещения можно наблюдать на Рисунке 3.13.

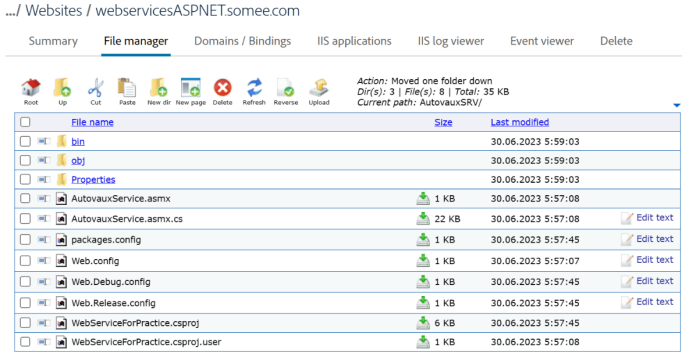


Рис. 3.13 – Перемещение проекта на хостинг

## ПРИЛОЖЕНИЯ

ПРИЛОЖЕНИЕ 1

Скрипт создания и заполнения базы данных

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Bron](

[ID\_BRON] [int] IDENTITY(1,1) NOT NULL,

[DATE] [datetime] NOT NULL,

[BUS] [int] NOT NULL,

[DEPART\_TIME] [datetime] NOT NULL,

[ONTOUR] [nvarchar](500) NOT NULL,

[COUNT] [int] NOT NULL,

CONSTRAINT [PK\_Bron] PRIMARY KEY CLUSTERED

(

[ID\_BRON] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Bus\_fleet](

[BUS\_ID] [int] IDENTITY(1,1) NOT NULL,

[REG\_ID] [nvarchar](9) NOT NULL,

[ORG\_ID\_FK] [int] NOT NULL,

[BUS\_TYPE\_ID\_FK] [int] NOT NULL,

[SCHEME\_ID\_FK] [int] NOT NULL,

CONSTRAINT [PK\_Bus\_fleet] PRIMARY KEY CLUSTERED

(

[BUS\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Bus\_types](

[BUS\_TYPE\_ID] [int] IDENTITY(1,1) NOT NULL,

[BUS\_TYPENAME] [nvarchar](50) NOT NULL,

[BUS\_HASCARD] [bit] NOT NULL,

CONSTRAINT [PK\_Bus\_types] PRIMARY KEY CLUSTERED

(

[BUS\_TYPE\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Carriers](

[ORG\_ID] [int] IDENTITY(1,1) NOT NULL,

[ORG\_NAME] [nvarchar](50) NOT NULL,

[ORG\_ADDRESS] [nvarchar](75) NOT NULL,

[ORG\_PHONE] [nvarchar](12) NOT NULL,

[ORG\_EMAIL] [nvarchar](30) NOT NULL,

CONSTRAINT [PK\_Carriers] PRIMARY KEY CLUSTERED

(

[ORG\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Profiles](

[PROFILE\_ID] [int] IDENTITY(1,1) NOT NULL,

[FNAME] [nvarchar](30) NOT NULL,

[LNAME] [nvarchar](50) NOT NULL,

[MNAME] [nvarchar](40) NULL,

[BIRTH\_DATE] [datetime] NULL,

[EMAIL] [nvarchar](30) NOT NULL,

[PHONE] [nvarchar](12) NULL,

[HASH\_PASSWORD] [nvarchar](128) NOT NULL,

[SALT] [nvarchar](16) NOT NULL,

[PROFILE\_STATE] [bit] NOT NULL,

[ROLE\_ID\_FK] [int] NOT NULL,

CONSTRAINT [PK\_Profiles] PRIMARY KEY CLUSTERED

(

[PROFILE\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Rate\_plans](

[PLAN\_ID] [int] IDENTITY(1,1) NOT NULL,

[RATE\_COSTS] [decimal](6, 2) NOT NULL,

[PLAN\_NAME] [nvarchar](100) NOT NULL,

CONSTRAINT [PK\_Rate\_plans] PRIMARY KEY CLUSTERED

(

[PLAN\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Roles](

[ROLE\_ID] [int] IDENTITY(1,1) NOT NULL,

[ROLE] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_ROLES] PRIMARY KEY CLUSTERED

(

[ROLE\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Routes](

[ROUTE\_ID] [int] IDENTITY(1,1) NOT NULL,

[ROUTE] [nvarchar](500) NOT NULL,

[TRAVEL\_COSTS] [decimal](6, 2) NOT NULL,

[BUS\_ID\_FK] [int] NOT NULL,

CONSTRAINT [PK\_Routes] PRIMARY KEY CLUSTERED

(

[ROUTE\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Schemas](

[SCHEME\_ID] [int] IDENTITY(1,1) NOT NULL,

[SEAT\_COUNT] [int] NOT NULL,

[SPEED] [decimal](6, 2) NOT NULL,

[COUNTRY\_ABBR] [nvarchar](2) NOT NULL,

CONSTRAINT [PK\_Schemas] PRIMARY KEY CLUSTERED

(

[SCHEME\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[TimetableList](

[TIMETABLE\_ID] [int] IDENTITY(1,1) NOT NULL,

[DATE] [datetime] NOT NULL,

[DEPARTURE\_TIME] [datetime] NOT NULL,

[ARRIVAL\_TIME] [datetime] NOT NULL,

[ROUTE\_ID\_FK] [int] NOT NULL,

CONSTRAINT [PK\_Timetable] PRIMARY KEY CLUSTERED

(

[TIMETABLE\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[Timetables](

[DATE] [datetime] NOT NULL,

[NAME] [nvarchar](50) NOT NULL,

CONSTRAINT [PK\_Timetables\_1] PRIMARY KEY CLUSTERED

(

[DATE] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

CREATE TABLE [dbo].[TimetableTemp](

[TEMP\_ID] [int] IDENTITY(1,1) NOT NULL,

[DEPARTURE\_TIME] [datetime] NOT NULL,

[ARRIVAL\_TIME] [datetime] NOT NULL,

[ROUTE\_ID\_FK] [int] NOT NULL,

CONSTRAINT [PK\_TimetableTemp] PRIMARY KEY CLUSTERED

(

[TEMP\_ID] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

SET IDENTITY\_INSERT [dbo].[Bron] ON

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (1, CAST(N'2023-05-31T00:00:00.000' AS DateTime), 13, CAST(N'2023-05-31T05:02:00.000' AS DateTime), N'До заповедника', 28)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (2, CAST(N'2023-05-25T00:00:00.000' AS DateTime), 12, CAST(N'2023-05-25T08:02:00.000' AS DateTime), N'До заповедника', 54)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (3, CAST(N'2023-05-26T00:00:00.000' AS DateTime), 13, CAST(N'2023-05-26T02:04:00.000' AS DateTime), N'До школы', 28)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (4, CAST(N'2023-05-27T00:00:00.000' AS DateTime), 13, CAST(N'2023-05-27T08:00:00.000' AS DateTime), N'Тальменка', 28)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (6, CAST(N'2023-06-05T00:00:00.000' AS DateTime), 12, CAST(N'2023-06-05T13:00:00.000' AS DateTime), N'Детский утренник на улице Матросова, 42', 27)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (7, CAST(N'2023-06-04T00:00:00.000' AS DateTime), 13, CAST(N'2023-06-04T15:00:00.000' AS DateTime), N'К школе отвезти инвентарь, учебники, а также рабочих', 28)

INSERT [dbo].[Bron] ([ID\_BRON], [DATE], [BUS], [DEPART\_TIME], [ONTOUR], [COUNT]) VALUES (8, CAST(N'2023-06-04T00:00:00.000' AS DateTime), 13, CAST(N'2023-06-04T15:00:00.000' AS DateTime), N'К школе отвезти инвентарь, учебники, а также рабочих', 28)

SET IDENTITY\_INSERT [dbo].[Bron] OFF

GO

SET IDENTITY\_INSERT [dbo].[Bus\_fleet] ON

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (1, N'М337КМ22', 1, 4, 1)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (2, N'А446ПН22', 2, 5, 2)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (3, N'Р221КТ22', 3, 6, 3)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (6, N'П178ТН22', 3, 4, 4)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (7, N'А162НН22', 2, 5, 4)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (8, N'В172ТТ22', 1, 6, 3)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (10, N'А200ММ22', 3, 4, 3)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (11, N'А567БВ22\_', 1, 4, 4)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (12, N'П177ВВ22\_', 2, 8, 2)

INSERT [dbo].[Bus\_fleet] ([BUS\_ID], [REG\_ID], [ORG\_ID\_FK], [BUS\_TYPE\_ID\_FK], [SCHEME\_ID\_FK]) VALUES (13, N'В282КК22', 3, 8, 3)

SET IDENTITY\_INSERT [dbo].[Bus\_fleet] OFF

GO

SET IDENTITY\_INSERT [dbo].[Bus\_types] ON

INSERT [dbo].[Bus\_types] ([BUS\_TYPE\_ID], [BUS\_TYPENAME], [BUS\_HASCARD]) VALUES (4, N'Междугородный', 0)

INSERT [dbo].[Bus\_types] ([BUS\_TYPE\_ID], [BUS\_TYPENAME], [BUS\_HASCARD]) VALUES (5, N'Международный', 0)

INSERT [dbo].[Bus\_types] ([BUS\_TYPE\_ID], [BUS\_TYPENAME], [BUS\_HASCARD]) VALUES (6, N'Пригородный', 1)

INSERT [dbo].[Bus\_types] ([BUS\_TYPE\_ID], [BUS\_TYPENAME], [BUS\_HASCARD]) VALUES (7, N'Городской', 1)

INSERT [dbo].[Bus\_types] ([BUS\_TYPE\_ID], [BUS\_TYPENAME], [BUS\_HASCARD]) VALUES (8, N'Бронируемый', 0)

SET IDENTITY\_INSERT [dbo].[Bus\_types] OFF

GO

SET IDENTITY\_INSERT [dbo].[Carriers] ON

INSERT [dbo].[Carriers] ([ORG\_ID], [ORG\_NAME], [ORG\_ADDRESS], [ORG\_PHONE], [ORG\_EMAIL]) VALUES (1, N'АвтоКор', N'ул. Сухэ-Батора, 14', N'+79131781214', N'autokor@mail.ru')

INSERT [dbo].[Carriers] ([ORG\_ID], [ORG\_NAME], [ORG\_ADDRESS], [ORG\_PHONE], [ORG\_EMAIL]) VALUES (2, N'Трансспорт', N'пл. Спартака, 15', N'+79132561546', N'transsport@mail.ru')

INSERT [dbo].[Carriers] ([ORG\_ID], [ORG\_NAME], [ORG\_ADDRESS], [ORG\_PHONE], [ORG\_EMAIL]) VALUES (3, N'Автоудача', N'ул. Новосибирская, 14', N'+79235401690', N'autoudacha@mail.ru')

SET IDENTITY\_INSERT [dbo].[Carriers] OFF

GO

SET IDENTITY\_INSERT [dbo].[Profiles] ON

INSERT [dbo].[Profiles] ([PROFILE\_ID], [FNAME], [LNAME], [MNAME], [BIRTH\_DATE], [EMAIL], [PHONE], [HASH\_PASSWORD], [SALT], [PROFILE\_STATE], [ROLE\_ID\_FK]) VALUES (1, N'Дмитрий', N'Беляев', N'Сергеевич', CAST(N'1999-12-27T00:00:00.000' AS DateTime), N'spydetector@mail.ru', N'89831735033', N'b568fff3720615fc0dc4b94564ac8e9092b9042ee82c541a17bf759346e3f8815dccba7cc7e6e686c06f46779ba4a026f71ae666b25deab9f99fdba9217b5ce4', N'joXtgVrJkmE3VQV', 1, 1)

INSERT [dbo].[Profiles] ([PROFILE\_ID], [FNAME], [LNAME], [MNAME], [BIRTH\_DATE], [EMAIL], [PHONE], [HASH\_PASSWORD], [SALT], [PROFILE\_STATE], [ROLE\_ID\_FK]) VALUES (2, N'Андрей', N'Шишкин', N'Петрович', CAST(N'1987-06-12T00:00:00.000' AS DateTime), N'vasya@mail.ru', N'+79813567547', N'8f10f0a15f372363c746136e3a1246b8da8048015bcb740ce0603e09aa6e7c39411f19fbdb27310183c7c7afb433b00fbb3c7e7456cfe5b77d99f7d6f92c4538', N'YKt1FyFyhHhb7vb', 1, 2)

INSERT [dbo].[Profiles] ([PROFILE\_ID], [FNAME], [LNAME], [MNAME], [BIRTH\_DATE], [EMAIL], [PHONE], [HASH\_PASSWORD], [SALT], [PROFILE\_STATE], [ROLE\_ID\_FK]) VALUES (3, N'Максим', N'Петров', N'Валерьевич', CAST(N'1998-05-04T00:00:00.000' AS DateTime), N'petya@mail.ru', N'+79141261688', N'011db9c536e1c6232dfd16014e2a88b24204ef3d3f33731fb95f299b3f3c9b43802993542006467086b432edf8117f41c75f7dd0b4186a3c2f6e533ec04bdff4', N'T6G5PjDXafcKhJd', 1, 1)

SET IDENTITY\_INSERT [dbo].[Profiles] OFF

GO

SET IDENTITY\_INSERT [dbo].[Rate\_plans] ON

INSERT [dbo].[Rate\_plans] ([PLAN\_ID], [RATE\_COSTS], [PLAN\_NAME]) VALUES (1, CAST(600.00 AS Decimal(6, 2)), N'Школьный')

INSERT [dbo].[Rate\_plans] ([PLAN\_ID], [RATE\_COSTS], [PLAN\_NAME]) VALUES (2, CAST(700.00 AS Decimal(6, 2)), N'Студенческий')

INSERT [dbo].[Rate\_plans] ([PLAN\_ID], [RATE\_COSTS], [PLAN\_NAME]) VALUES (3, CAST(900.00 AS Decimal(6, 2)), N'Пенсионный')

INSERT [dbo].[Rate\_plans] ([PLAN\_ID], [RATE\_COSTS], [PLAN\_NAME]) VALUES (4, CAST(1200.00 AS Decimal(6, 2)), N'Единый')

SET IDENTITY\_INSERT [dbo].[Rate\_plans] OFF

GO

SET IDENTITY\_INSERT [dbo].[Roles] ON

INSERT [dbo].[Roles] ([ROLE\_ID], [ROLE]) VALUES (1, N'Администратор')

INSERT [dbo].[Roles] ([ROLE\_ID], [ROLE]) VALUES (2, N'Диспетчер')

SET IDENTITY\_INSERT [dbo].[Roles] OFF

GO

SET IDENTITY\_INSERT [dbo].[Routes] ON

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (1, N'Барнаул - Новосибирск', CAST(800.00 AS Decimal(6, 2)), 1)

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (2, N'Барнаул - Павлодар', CAST(2259.00 AS Decimal(6, 2)), 2)

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (3, N'пл. Спартака - пос. Бельмесево', CAST(30.00 AS Decimal(6, 2)), 3)

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (4, N'Барнаул - Алейск', CAST(300.00 AS Decimal(6, 2)), 6)

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (5, N'Барнаул - Усть-Каменогорск', CAST(1805.00 AS Decimal(6, 2)), 7)

INSERT [dbo].[Routes] ([ROUTE\_ID], [ROUTE], [TRAVEL\_COSTS], [BUS\_ID\_FK]) VALUES (6, N'пос. Березовка - Научный городок', CAST(30.00 AS Decimal(6, 2)), 8)

SET IDENTITY\_INSERT [dbo].[Routes] OFF

GO

SET IDENTITY\_INSERT [dbo].[Schemas] ON

INSERT [dbo].[Schemas] ([SCHEME\_ID], [SEAT\_COUNT], [SPEED], [COUNTRY\_ABBR]) VALUES (1, 46, CAST(80.00 AS Decimal(6, 2)), N'KR')

INSERT [dbo].[Schemas] ([SCHEME\_ID], [SEAT\_COUNT], [SPEED], [COUNTRY\_ABBR]) VALUES (2, 54, CAST(90.00 AS Decimal(6, 2)), N'KR')

INSERT [dbo].[Schemas] ([SCHEME\_ID], [SEAT\_COUNT], [SPEED], [COUNTRY\_ABBR]) VALUES (3, 28, CAST(90.00 AS Decimal(6, 2)), N'CN')

INSERT [dbo].[Schemas] ([SCHEME\_ID], [SEAT\_COUNT], [SPEED], [COUNTRY\_ABBR]) VALUES (4, 36, CAST(60.00 AS Decimal(6, 2)), N'RU')

INSERT [dbo].[Schemas] ([SCHEME\_ID], [SEAT\_COUNT], [SPEED], [COUNTRY\_ABBR]) VALUES (10, 30, CAST(67.00 AS Decimal(6, 2)), N'KZ')

SET IDENTITY\_INSERT [dbo].[Schemas] OFF

GO

SET IDENTITY\_INSERT [dbo].[TimetableList] ON

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (14, CAST(N'2023-06-03T00:00:00.000' AS DateTime), CAST(N'2023-06-03T16:00:00.000' AS DateTime), CAST(N'2023-06-03T22:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (15, CAST(N'2023-06-03T00:00:00.000' AS DateTime), CAST(N'2023-06-04T02:00:00.000' AS DateTime), CAST(N'2023-06-04T08:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (16, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T15:00:00.000' AS DateTime), CAST(N'2023-06-04T19:00:00.000' AS DateTime), 1)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (17, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-05T03:00:00.000' AS DateTime), CAST(N'2023-06-05T07:00:00.000' AS DateTime), 1)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (18, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T17:00:00.000' AS DateTime), CAST(N'2023-06-04T23:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (19, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-05T05:00:00.000' AS DateTime), CAST(N'2023-06-05T11:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (20, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T03:00:00.000' AS DateTime), CAST(N'2023-06-04T23:00:00.000' AS DateTime), 5)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (21, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T10:45:00.000' AS DateTime), CAST(N'2023-06-04T13:40:00.000' AS DateTime), 4)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (22, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T12:45:00.000' AS DateTime), CAST(N'2023-06-04T15:40:00.000' AS DateTime), 4)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (23, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T18:40:00.000' AS DateTime), CAST(N'2023-06-04T21:35:00.000' AS DateTime), 4)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (24, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T08:00:00.000' AS DateTime), CAST(N'2023-06-04T09:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (25, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T10:00:00.000' AS DateTime), CAST(N'2023-06-04T11:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (26, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T12:00:00.000' AS DateTime), CAST(N'2023-06-04T13:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (27, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T14:00:00.000' AS DateTime), CAST(N'2023-06-04T15:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (28, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T16:00:00.000' AS DateTime), CAST(N'2023-06-04T17:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (29, CAST(N'2023-06-04T00:00:00.000' AS DateTime), CAST(N'2023-06-04T18:00:00.000' AS DateTime), CAST(N'2023-06-04T19:05:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (30, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-30T17:00:00.000' AS DateTime), CAST(N'2023-05-30T23:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (31, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-31T00:00:00.000' AS DateTime), CAST(N'2023-05-31T06:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (32, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-30T09:00:00.000' AS DateTime), CAST(N'2023-05-30T11:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (33, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-30T12:00:00.000' AS DateTime), CAST(N'2023-05-30T14:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (34, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-30T15:00:00.000' AS DateTime), CAST(N'2023-05-30T17:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableList] ([TIMETABLE\_ID], [DATE], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (35, CAST(N'2023-05-16T00:00:00.000' AS DateTime), CAST(N'2023-05-30T18:00:00.000' AS DateTime), CAST(N'2023-05-30T20:54:00.000' AS DateTime), 6)

SET IDENTITY\_INSERT [dbo].[TimetableList] OFF

GO

INSERT [dbo].[Timetables] ([DATE], [NAME]) VALUES (CAST(N'2023-05-16T00:00:00.000' AS DateTime), N'Расписание от 16-05-2023')

INSERT [dbo].[Timetables] ([DATE], [NAME]) VALUES (CAST(N'2023-06-02T00:00:00.000' AS DateTime), N'Расписание от 02.06.2023 0:00:00')

INSERT [dbo].[Timetables] ([DATE], [NAME]) VALUES (CAST(N'2023-06-03T00:00:00.000' AS DateTime), N'Расписание от 03.06.2023 0:00:00')

INSERT [dbo].[Timetables] ([DATE], [NAME]) VALUES (CAST(N'2023-06-04T00:00:00.000' AS DateTime), N'Расписание от 04.06.2023 0:00:00')

GO

SET IDENTITY\_INSERT [dbo].[TimetableTemp] ON

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (55, CAST(N'2023-05-10T09:00:00.000' AS DateTime), CAST(N'2023-05-10T11:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (56, CAST(N'2023-05-10T12:00:00.000' AS DateTime), CAST(N'2023-05-10T14:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (57, CAST(N'2023-05-10T15:00:00.000' AS DateTime), CAST(N'2023-05-10T21:54:00.000' AS DateTime), 2)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (58, CAST(N'2023-05-10T18:00:00.000' AS DateTime), CAST(N'2023-05-10T20:54:00.000' AS DateTime), 6)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (59, CAST(N'2023-05-10T08:20:00.000' AS DateTime), CAST(N'2023-05-10T10:00:00.000' AS DateTime), 3)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (60, CAST(N'2023-05-10T10:40:00.000' AS DateTime), CAST(N'2023-05-10T12:20:00.000' AS DateTime), 3)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (61, CAST(N'2023-05-10T13:00:00.000' AS DateTime), CAST(N'2023-05-10T14:40:00.000' AS DateTime), 3)

INSERT [dbo].[TimetableTemp] ([TEMP\_ID], [DEPARTURE\_TIME], [ARRIVAL\_TIME], [ROUTE\_ID\_FK]) VALUES (62, CAST(N'2023-05-16T03:00:00.000' AS DateTime), CAST(N'2023-05-16T07:02:00.000' AS DateTime), 1)

SET IDENTITY\_INSERT [dbo].[TimetableTemp] OFF

GO

ALTER TABLE [dbo].[Bus\_fleet] WITH CHECK ADD CONSTRAINT [FK\_Bus\_fleet\_Bus\_types] FOREIGN KEY([BUS\_TYPE\_ID\_FK])

REFERENCES [dbo].[Bus\_types] ([BUS\_TYPE\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Bus\_fleet] CHECK CONSTRAINT [FK\_Bus\_fleet\_Bus\_types]

GO

ALTER TABLE [dbo].[Bus\_fleet] WITH CHECK ADD CONSTRAINT [FK\_Bus\_fleet\_Carriers] FOREIGN KEY([ORG\_ID\_FK])

REFERENCES [dbo].[Carriers] ([ORG\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Bus\_fleet] CHECK CONSTRAINT [FK\_Bus\_fleet\_Carriers]

GO

ALTER TABLE [dbo].[Bus\_fleet] WITH CHECK ADD CONSTRAINT [FK\_Bus\_fleet\_Schemas] FOREIGN KEY([SCHEME\_ID\_FK])

REFERENCES [dbo].[Schemas] ([SCHEME\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Bus\_fleet] CHECK CONSTRAINT [FK\_Bus\_fleet\_Schemas]

GO

ALTER TABLE [dbo].[Profiles] WITH CHECK ADD CONSTRAINT [FK\_Profiles\_Roles] FOREIGN KEY([ROLE\_ID\_FK])

REFERENCES [dbo].[Roles] ([ROLE\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Profiles] CHECK CONSTRAINT [FK\_Profiles\_Roles]

GO

ALTER TABLE [dbo].[Routes] WITH CHECK ADD CONSTRAINT [FK\_Routes\_Bus\_fleet] FOREIGN KEY([BUS\_ID\_FK])

REFERENCES [dbo].[Bus\_fleet] ([BUS\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[Routes] CHECK CONSTRAINT [FK\_Routes\_Bus\_fleet]

GO

ALTER TABLE [dbo].[TimetableList] WITH CHECK ADD CONSTRAINT [FK\_TimetableList\_Routes] FOREIGN KEY([ROUTE\_ID\_FK])

REFERENCES [dbo].[Routes] ([ROUTE\_ID])

ON DELETE CASCADE

GO

ALTER TABLE [dbo].[TimetableList] CHECK CONSTRAINT [FK\_TimetableList\_Routes]

GO

ALTER TABLE [dbo].[TimetableList] WITH CHECK ADD CONSTRAINT [FK\_TimetableList\_Timetables] FOREIGN KEY([DATE])

REFERENCES [dbo].[Timetables] ([DATE])

GO

ALTER TABLE [dbo].[TimetableList] CHECK CONSTRAINT [FK\_TimetableList\_Timetables]

GO

ALTER TABLE [dbo].[TimetableTemp] WITH CHECK ADD CONSTRAINT [FK\_TimetableTemp\_Routes] FOREIGN KEY([ROUTE\_ID\_FK])

REFERENCES [dbo].[Routes] ([ROUTE\_ID])

GO

ALTER TABLE [dbo].[TimetableTemp] CHECK CONSTRAINT [FK\_TimetableTemp\_Routes]

GO

ПРИЛОЖЕНИЕ 2

Скрипты создания хранимых процедур и представлений

Представление TimetableView

CREATE VIEW [dbo].[TimetableView]

AS

SELECT TL.DATE AS Дата, FORMAT(TL.DEPARTURE\_TIME, 'HH:mm', 'ru-RU') AS Время\_отправления, FORMAT(TL.ARRIVAL\_TIME, 'HH:mm', 'ru-RU') AS Время\_прибытия, R.ROUTE\_ID AS Номер\_маршрута, R.ROUTE AS Маршрут,

R.TRAVEL\_COSTS AS Стоимость\_проезда, BF.REG\_ID AS Регистрационный\_номер, BT.BUS\_TYPENAME AS Тип\_маршрута, S.SEAT\_COUNT AS Количество\_посадочных\_мест,

C.ORG\_NAME AS [Компания-перевозчик] FROM dbo.TimetableList AS TL INNER JOIN

dbo.Routes AS R ON TL.ROUTE\_ID\_FK = R.ROUTE\_ID INNER JOIN

dbo.Bus\_fleet AS BF ON R.BUS\_ID\_FK = BF.BUS\_ID INNER JOIN

dbo.Carriers AS C ON BF.ORG\_ID\_FK = C.ORG\_ID INNER JOIN

dbo.Bus\_types AS BT ON BF.BUS\_TYPE\_ID\_FK = BT.BUS\_TYPE\_ID INNER JOIN

dbo.Schemas AS S ON BF.SCHEME\_ID\_FK = S.SCHEME\_ID

GO

Хранимая процедура timetable\_dt

CREATE PROCEDURE [dbo].[timetable\_dt] @Timetable\_date datetime

AS

BEGIN

SET NOCOUNT ON;

SELECT [Время\_отправления], [Время\_прибытия], [Номер\_маршрута], [Маршрут], [Стоимость\_проезда], [Регистрационный\_номер], [Тип\_маршрута], [Количество\_посадочных\_мест], [Компания-перевозчик] FROM TimetableView as TV

WHERE TV.Дата = @Timetable\_date

END

GO

Хранимая процедура timetable\_dt\_xml

CREATE PROCEDURE [dbo].[timetable\_dt\_xml] @Timetable\_date datetime

AS

BEGIN

SET NOCOUNT ON;

SELECT FORMAT(T.DATE,'dd.MM.yyyy','ru-RU') as [Дата],

(SELECT [Время\_отправления], [Время\_прибытия], [Номер\_маршрута], [Маршрут], [Стоимость\_проезда], [Регистрационный\_номер], [Тип\_маршрута], [Количество\_посадочных\_мест], [Компания-перевозчик] FROM TimetableView as TV

WHERE TV.Дата = @Timetable\_date

FOR XML RAW('Запись'), type, ELEMENTS)

FROM Timetables as T

WHERE T.DATE = @Timetable\_date

FOR XML RAW('Расписание')

END

GO

Хранимая процедура timetable\_dt\_json

CREATE PROCEDURE [dbo].[timetable\_dt\_json] @Timetable\_date datetime

AS

BEGIN

SET NOCOUNT ON;

select FORMAT(T.DATE,'dd.MM.yyyy','ru-RU') as [Дата],

(SELECT [Время\_отправления], [Время\_прибытия], [Номер\_маршрута], [Маршрут], [Стоимость\_проезда], [Регистрационный\_номер], [Тип\_маршрута], [Количество\_посадочных\_мест], [Компания-перевозчик] FROM TimetableView as TV

WHERE TV.Дата = @Timetable\_date

FOR JSON AUTO) [Расписание]

FROM Timetables as T

WHERE T.DATE = @Timetable\_date

FOR JSON AUTO

END

GO

Хранимая процедура timetables\_json

CREATE PROCEDURE [dbo].[timetables\_json]

AS

BEGIN

SET NOCOUNT ON;

select FORMAT(T.DATE,'dd.MM.yyyy','ru-RU') as [Дата],

(SELECT [Время\_отправления], [Время\_прибытия], [Номер\_маршрута], [Маршрут], [Стоимость\_проезда], [Регистрационный\_номер], [Тип\_маршрута], [Количество\_посадочных\_мест], [Компания-перевозчик] FROM TimetableView as TV

WHERE [Дата]=T.DATE

FOR JSON AUTO) [Расписание]

FROM Timetables as T

FOR JSON PATH, ROOT('Расписания')

END

GO

Хранимая процедура timetables\_xml

CREATE PROCEDURE [dbo].[timetables\_xml]

AS

BEGIN

SET NOCOUNT ON;

select FORMAT(T.DATE,'dd.MM.yyyy','ru-RU') as [Дата],

(SELECT [Время\_отправления], [Время\_прибытия], [Номер\_маршрута], [Маршрут], [Стоимость\_проезда], [Регистрационный\_номер], [Тип\_маршрута], [Количество\_посадочных\_мест], [Компания-перевозчик] FROM TimetableView as TV

WHERE [Дата]=T.DATE

FOR XML RAW('Запись'), type, ELEMENTS)

FROM Timetables as T

FOR XML RAW('Расписание'), ROOT('Расписания')

END

GO

Хранимая процедура carriers\_query

CREATE PROCEDURE [dbo].[carriers\_query]

AS

BEGIN

SET NOCOUNT ON;

SELECT ORG\_ID as [Код\_организации], ORG\_NAME as [Наименование], ORG\_ADDRESS as [Адрес], ORG\_PHONE as [Телефон], ORG\_EMAIL as [E-mail] FROM Carriers as C

END

GO

Хранимая процедура carriers\_query\_json

CREATE PROCEDURE [dbo].[carriers\_query\_json]

AS

BEGIN

SET NOCOUNT ON;

SELECT ORG\_ID as [Код\_организации], ORG\_NAME as [Наименование], ORG\_ADDRESS as [Адрес], ORG\_PHONE as [Телефон], ORG\_EMAIL as [E-mail] FROM Carriers as C

FOR JSON PATH, ROOT('Организации')

END

GO

Хранимая процедура carriers\_query\_xml

CREATE PROCEDURE [dbo].[carriers\_query\_xml]

AS

BEGIN

SET NOCOUNT ON;

SELECT ORG\_ID as [Код\_организации], ORG\_NAME as [Наименование], ORG\_ADDRESS as [Адрес], ORG\_PHONE as [Телефон], ORG\_EMAIL as [E-mail] FROM Carriers as C

FOR XML RAW('Организация'), ROOT('Организации'), ELEMENTS

END

GO

Хранимая процедура carriers\_byid\_json

CREATE PROCEDURE [dbo].[carriers\_byid\_json] @org\_id int

AS

BEGIN

SET NOCOUNT ON;

SELECT ORG\_ID as [Код\_организации], ORG\_NAME as [Наименование], ORG\_ADDRESS as [Адрес], ORG\_PHONE as [Телефон], ORG\_EMAIL as [E-mail] FROM Carriers as C

WHERE ORG\_ID = @org\_id

FOR JSON PATH, ROOT('Организация')

END

GO

Хранимая процедура carriers\_byid\_xml

CREATE PROCEDURE [dbo].[carriers\_byid\_xml] @org\_id int

AS

BEGIN

SET NOCOUNT ON;

SELECT ORG\_ID as [Код\_организации], ORG\_NAME as [Наименование], ORG\_ADDRESS as [Адрес], ORG\_PHONE as [Телефон], ORG\_EMAIL as [E-mail] FROM Carriers as C

WHERE ORG\_ID = @org\_id

FOR XML RAW('Организация'), type, ELEMENTS

END

GO

Хранимая процедура delete\_carrier\_json

CREATE PROCEDURE [dbo].[delete\_carrier\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

DELETE FROM Carriers

WHERE ORG\_ID = (select ORG\_ID

FROM OPENJSON (@j)

WITH (

ORG\_ID int

))

END

GO

Хранимая процедура delete\_carrier\_xml

CREATE PROCEDURE [dbo].[delete\_carrier\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

DELETE FROM Carriers

WHERE ORG\_ID = (select ORG\_ID

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_ID int

))

EXEC sp\_xml\_removedocument @nx

END

GO

Хранимая процедура delete\_ttrecord\_json

CREATE PROCEDURE [dbo].[delete\_ttrecord\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

DELETE FROM TimetableTemp

WHERE TEMP\_ID = (select TEMP\_ID

FROM OPENJSON (@j)

WITH (

TEMP\_ID int

))

END

GO

Хранимая процедура delete\_ttrecord\_xml

CREATE PROCEDURE [dbo].[delete\_ttrecord\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

DELETE FROM TimetableTemp

WHERE TEMP\_ID = (select TEMP\_ID

FROM OPENXML (@nx, 'Запись',1)

WITH (

TEMP\_ID int

))

EXEC sp\_xml\_removedocument @nx

END

GO

Хранимая процедура new\_carrier\_json

CREATE PROCEDURE [dbo].[new\_carrier\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO Carriers SELECT \* FROM OPENJSON (@j)

WITH (ORG\_NAME nvarchar(50),

ORG\_ADDRESS nvarchar(75),

ORG\_PHONE nvarchar(12),

ORG\_EMAIL nvarchar(30))

END

GO

Хранимая процедура new\_carrier\_xml

CREATE PROCEDURE [dbo].[new\_carrier\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

INSERT INTO Carriers SELECT \* FROM OPENXML (@nx, 'Организация',1)

WITH (ORG\_NAME nvarchar(50),

ORG\_ADDRESS nvarchar(75),

ORG\_PHONE nvarchar(12),

ORG\_EMAIL nvarchar(30))

EXEC sp\_xml\_removedocument @nx

END

GO

Хранимая процедура new\_timetable\_json

CREATE PROCEDURE [dbo].[new\_timetable\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO TimetableTemp SELECT CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), DEPARTURE\_TIME) as datetime) as DEPARTURE\_TIME,

CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), ARRIVAL\_TIME) as datetime) as ARRIVAL\_TIME,

ROUTE\_ID\_FK

FROM OPENJSON (@j)

WITH (DEPARTURE\_TIME datetime,

ARRIVAL\_TIME datetime,

ROUTE\_ID\_FK int)

END

GO

Хранимая процедура new\_timetable\_xml

CREATE PROCEDURE [dbo].[new\_timetable\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

INSERT INTO TimetableTemp SELECT CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), DEPARTURE\_TIME) as datetime) as DEPARTURE\_TIME,

CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), ARRIVAL\_TIME) as datetime) as ARRIVAL\_TIME,

ROUTE\_ID\_FK

FROM OPENXML (@nx, 'Запись',1)

WITH (DEPARTURE\_TIME datetime,

ARRIVAL\_TIME datetime,

ROUTE\_ID\_FK int)

EXEC sp\_xml\_removedocument @nx

END

GO

Хранимая процедура update\_carrier\_xml

CREATE PROCEDURE [dbo].[update\_carrier\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

UPDATE Carriers SET

ORG\_NAME = (select ORG\_NAME

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_NAME nvarchar(50))),

ORG\_ADDRESS = (select ORG\_ADDRESS

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_ADDRESS nvarchar(75))),

ORG\_PHONE = (select ORG\_PHONE

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_PHONE nvarchar(12))),

ORG\_EMAIL = (select ORG\_EMAIL

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_EMAIL nvarchar(30)))

WHERE ORG\_ID = (select ORG\_ID

FROM OPENXML (@nx, 'Организация',1)

WITH (

ORG\_ID int))

EXEC sp\_xml\_removedocument @nx

END

GO

Хранимая процедура update\_carrier\_json

CREATE PROCEDURE [dbo].[update\_carrier\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

UPDATE Carriers SET

ORG\_NAME = (select ORG\_NAME

FROM OPENJSON (@j)

WITH (

ORG\_NAME nvarchar(50)

)),

ORG\_ADDRESS = (select ORG\_ADDRESS

FROM OPENJSON (@j)

WITH (

ORG\_ADDRESS nvarchar(75))),

ORG\_PHONE = (select ORG\_PHONE

FROM OPENJSON (@j)

WITH (

ORG\_PHONE nvarchar(12))),

ORG\_EMAIL = (select ORG\_EMAIL

FROM OPENJSON (@j)

WITH (

ORG\_EMAIL nvarchar(30)))

WHERE ORG\_ID = (select ORG\_ID

FROM OPENJSON (@j)

WITH (

ORG\_ID int))

END

GO

Хранимая процедура update\_timetable\_json

CREATE PROCEDURE [dbo].[update\_timetable\_json] @j as nvarchar(max)

AS

BEGIN

SET NOCOUNT ON;

UPDATE TimetableTemp SET

DEPARTURE\_TIME = (select CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), DEPARTURE\_TIME) as datetime) as DEPARTURE\_TIME

FROM OPENJSON (@j)

WITH (

DEPARTURE\_TIME datetime)

),

ARRIVAL\_TIME = (select CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), ARRIVAL\_TIME) as datetime) as ARRIVAL\_TIME

FROM OPENJSON (@j)

WITH (

ARRIVAL\_TIME datetime)),

ROUTE\_ID\_FK = (select ROUTE\_ID\_FK

FROM OPENJSON (@j)

WITH (

ROUTE\_ID\_FK int))

WHERE TEMP\_ID = (select TEMP\_ID

FROM OPENJSON (@j)

WITH (

TEMP\_ID int))

END

GO

Хранимая процедура update\_timetable\_xml

CREATE PROCEDURE [dbo].[update\_timetable\_xml] @XmlDocument xml

AS

BEGIN

SET NOCOUNT ON;

DECLARE @nx int

EXEC sp\_xml\_preparedocument @nx OUTPUT, @XmlDocument

UPDATE TimetableTemp SET

DEPARTURE\_TIME = (select CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), DEPARTURE\_TIME) as datetime) as DEPARTURE\_TIME

FROM OPENXML (@nx, 'Запись',1)

WITH (

DEPARTURE\_TIME datetime)

),

ARRIVAL\_TIME = (select CAST(DATEADD(day, DATEDIFF(day, 0, GETDATE()), ARRIVAL\_TIME) as datetime) as ARRIVAL\_TIME

FROM OPENXML (@nx, 'Запись',1)

WITH (

ARRIVAL\_TIME datetime)),

ROUTE\_ID\_FK = (select ROUTE\_ID\_FK

FROM OPENXML (@nx, 'Запись',1)

WITH (

ROUTE\_ID\_FK int))

WHERE TEMP\_ID = (select TEMP\_ID

FROM OPENXML (@nx, 'Запись',1)

WITH (

TEMP\_ID int))

EXEC sp\_xml\_removedocument @nx

END

GO

ПРИЛОЖЕНИЕ 3

Коды ASP.NET приложения

Web.config

<?xml version="1.0" encoding="utf-8"?>

<configuration>

<connectionStrings>

<add name="autovocdb" connectionString="workstation id=db\_a99962\_dbautovoc.mssql.somee.com;packet size=4096;user id=lolkaspy\_SQLLogin\_1;

pwd=q485bx4le2;data source=db\_a99962\_dbautovoc.mssql.somee.com;persist security info=False;initial catalog=db\_a99962\_dbautovoc"/>

</connectionStrings>

<system.web>

<compilation debug="true" targetFramework="4.8" />

<httpRuntime targetFramework="4.8" />

<customErrors mode="Off"/>

</system.web>

</configuration>

TimetableXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

DateTime x1=new DateTime(1900,1,1);

if (Request.QueryString.Count==1)

{

DateTime.TryParse(Request.QueryString[0].ToString(),out x1);

cmd.Parameters.AddWithValue("@dt1",x1 );

cmd.CommandText="exec timetable\_dt\_xml @dt1";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string xml\_text="";

while (rdr.Read())

xml\_text += rdr[0].ToString();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

if (xml\_text != "")

{

Response.Write(xml\_text);

}

else

{

Response.Write("По заданному параметру отсутствуют данные.");

}

Response.End();

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

TimetableJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

DateTime x1=new DateTime(1900,1,1);

if (Request.QueryString.Count==1)

{

DateTime.TryParse(Request.QueryString[0].ToString(),out x1);

cmd.Parameters.AddWithValue("@dt1",x1 );

cmd.CommandText="exec timetable\_dt\_json @dt1";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string json\_text="";

while (rdr.Read())

json\_text += rdr[0].ToString();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

if (json\_text != "")

{

Response.Write(json\_text);

}

else

{

Response.Write("По заданному параметру отсутствуют данные.");

}

Response.End();

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

TimetablesXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText="exec timetables\_xml";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string xml\_text="";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(xml\_text);

Response.End();

}

</script>

TimetablesJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText="exec timetables\_json";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string json\_text="";

while (rdr.Read())

json\_text += rdr[0].ToString();

cnn.Close();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(json\_text);

Response.End();

}

</script>

CarriersXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText="exec carriers\_query\_xml";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string xml\_text="";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(xml\_text);

Response.End();

}

</script>

CarriersJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText="exec carriers\_query\_json";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string json\_text="";

while (rdr.Read())

json\_text += rdr[0].ToString();

cnn.Close();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(json\_text);

Response.End();

}

</script>

CarriersByIdXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

int x1=0;

if (Request.QueryString.Count==1)

{

int.TryParse(Request.QueryString[0].ToString(),out x1);

cmd.Parameters.AddWithValue("@id",x1 );

cmd.CommandText="exec carriers\_byid\_xml @id";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string xml\_text="";

while (rdr.Read())

xml\_text += rdr[0].ToString();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

if (xml\_text != "")

{

Response.Write(xml\_text);

}

else

{

Response.Write("По заданному параметру отсутствуют данные.");

}

Response.End();

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

CarriersByIdJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

int x1=0;

if (Request.QueryString.Count==1)

{

int.TryParse(Request.QueryString[0].ToString(),out x1);

cmd.Parameters.AddWithValue("@id",x1 );

cmd.CommandText="exec carriers\_byid\_json @id";

System.Data.SqlClient.SqlDataReader rdr=cmd.ExecuteReader();

string json\_text="";

while (rdr.Read())

json\_text += rdr[0].ToString();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

if (json\_text != "")

{

Response.Write(json\_text);

}

else

{

Response.Write("По заданному параметру отсутствуют данные.");

}

Response.End();

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

AddCarrierJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec new\_carrier\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись добавлена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при добавлении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

AddCarrierXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec new\_carrier\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись добавлена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

//byte[] byteMessage = System.Convert.FromBase64String(r);

//string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при добавлении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n");

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

AddTTRecordJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec new\_timetable\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись добавлена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при добавлении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

AddTTRecordXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec new\_timetable\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись добавлена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при добавлении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

DeleteCarrierJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec delete\_carrier\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись удалена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при удалении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

DeleteCarrierXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec delete\_carrier\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись удалена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при удалении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

DeleteTTRecordJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec delete\_ttrecord\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись удалена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при удалении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

DeleteTTRecordXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec delete\_ttrecord\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись удалена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при удалении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

UpdateCarrierJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec update\_carrier\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись изменена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при изменении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

UpdateCarrierXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec update\_carrier\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись изменена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при изменении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

UpdateTTRecordJSON.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec update\_timetable\_json @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись изменена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при изменении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

UpdateTTRecordXML.aspx

<%@ Page Language="C#" %>

<script runat="server">

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

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType= System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

if (Request.Form.Count==1)

{

try

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText="exec update\_timetable\_xml @data";

cmd.ExecuteNonQuery();

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write("Запись изменена");

}

catch(Exception ex)

{

string r = Request.Form[0].ToString();

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(r+"\nПроизошла ошибка при изменении записи. Возможен неверный формат параметра. Попробуйте изменить параметр так, чтобы запрос выполнился\n" + ex.Message +"\n" + strMessage);

Response.End();

}

}

else

{

string text="Данные отсуствуют, попробуйте ввести адрес с параметром";

Response.Clear();

Response.ContentType = "text/plain";

Response.ContentEncoding = System.Text.Encoding.UTF8;

Response.Write(text);

Response.End();

}

cnn.Close();

}

</script>

ПРИЛОЖЕНИЕ 4

Код Web.config

<?xml version="1.0"?>

<!--

Дополнительные сведения о настройке приложения ASP.NET см. на странице

https://go.microsoft.com/fwlink/?LinkId=169433.

-->

<configuration>

<system.web>

<webServices>

<protocols>

<add name="HttpGet"/>

<add name="HttpPost"/>

</protocols>

</webServices>

<compilation debug="true"/>

</system.web>

<connectionStrings>

<add name="autovocdb" connectionString="workstation id=db\_a99962\_dbautovoc.mssql.somee.com;packet size=4096;user id=lolkaspy\_SQLLogin\_1; pwd=q485bx4le2;data source=db\_a99962\_dbautovoc.mssql.somee.com;persist security info=False;initial catalog=db\_a99962\_dbautovoc"/>

</connectionStrings>

<system.web>

<customErrors mode="Off"/>

</system.web>

</configuration>

Код веб-сервиса AutovauxService.asmx

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Text;

using System.Web;

using System.Web.Script.Services;

using System.Web.Services;

using Newtonsoft.Json;

namespace WebServiceForPractice

{

/// <summary>

/// Сводное описание для AutovauxService

/// </summary>

[WebService(Namespace = "http://webservicesASPNET.somee.com")]

[WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1\_1)]

[System.ComponentModel.ToolboxItem(false)]

// Чтобы разрешить вызывать веб-службу из скрипта с помощью ASP.NET AJAX, раскомментируйте следующую строку.

[System.Web.Script.Services.ScriptService]

public class AutovauxService : System.Web.Services.WebService

{

[WebMethod]

//Одноуровневый справочник

public string GetCarriersList()

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText = "exec carriers\_query\_xml";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return xml\_text;

}

[WebMethod]

//Элемент справочника по коду

public string GetCarrierById(int cid)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.Parameters.AddWithValue("@id", cid);

cmd.CommandText = "exec carriers\_byid\_xml @id";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return xml\_text;

}

[WebMethod]

//Документы "Расписания" на некоторую дату

public string GetTimetables()

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText = "exec timetables\_xml";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return xml\_text;

}

[WebMethod]

//Документ "Расписание" на некоторую дату

public string GetTimetableByDate(string date)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

DateTime x1 = new DateTime(1900, 1, 1);

DateTime.TryParse(date, out x1);

cmd.Parameters.AddWithValue("@dt1", x1);

cmd.CommandText = "exec timetable\_dt\_xml @dt1";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return xml\_text;

}

[ScriptMethod(ResponseFormat=ResponseFormat.Json)]

[WebMethod]

//Одноуровневый справочник

public string GetCarriersListJSON()

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText = "exec carriers\_query\_json";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return JsonConvert.SerializeObject(xml\_text);

}

[ScriptMethod(ResponseFormat = ResponseFormat.Json)]

[WebMethod]

//Элемент справочника по коду

public string GetCarrierByIdJSON(int cid)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.Parameters.AddWithValue("@id", cid);

cmd.CommandText = "exec carriers\_byid\_json @id";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return JsonConvert.SerializeObject(xml\_text);

}

[ScriptMethod(ResponseFormat = ResponseFormat.Json)]

[WebMethod]

//Документы "Расписания" на некоторую дату

public string GetTimetablesJSON()

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

cmd.CommandText = "exec timetables\_json";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return JsonConvert.SerializeObject(xml\_text);

}

[ScriptMethod(ResponseFormat = ResponseFormat.Json)]

[WebMethod]

//Документ "Расписание" на некоторую дату

public string GetTimetableByDateJSON(string date)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

DateTime x1 = new DateTime(1900, 1, 1);

DateTime.TryParse(date, out x1);

cmd.Parameters.AddWithValue("@dt1", x1);

cmd.CommandText = "exec timetable\_dt\_json @dt1";

System.Data.SqlClient.SqlDataReader rdr = cmd.ExecuteReader();

string xml\_text = "";

while (rdr.Read())

xml\_text += rdr[0].ToString();

cnn.Close();

return JsonConvert.SerializeObject(xml\_text);

}

[WebMethod]

//Добавление организации в справочник xml

public void AddCarrierXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec new\_carrier\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Добавление организации в справочник json

public void AddCarrierJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec new\_carrier\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Добавление расписания xml

public void AddTimetableXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec new\_timetable\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Добавление расписания json

public void AddTimetableJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec new\_timetable\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Удаление организации из справочника xml

public void DeleteCarrierXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec delete\_carrier\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Удаление организации из справочника json

public void DeleteCarrierJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec delete\_carrier\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Удаление расписания xml

public void DeleteTimetableXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec delete\_ttrecord\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Удаление расписания json

public void DeleteTimetableJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec delete\_ttrecord\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Обновление организации в справочнике xml

public void UpdateCarrierXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec update\_carrier\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Обновление организации в справочнике json

public void UpdateCarrierJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec update\_carrier\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Обновление записи расписания xml

public void UpdateTimetableXML(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec update\_timetable\_xml @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

[WebMethod]

//Обновление записи расписания json

public void UpdateTimetableJSON(string xml)

{

System.Data.SqlClient.SqlConnection cnn = new System.Data.SqlClient.SqlConnection();

string cnntext = System.Web.Configuration.WebConfigurationManager.ConnectionStrings["autovocdb"].ConnectionString;

cnn.ConnectionString = cnntext;

System.Data.SqlClient.SqlCommand cmd = new System.Data.SqlClient.SqlCommand();

cmd.CommandType = System.Data.CommandType.Text;

cmd.Connection = cnn;

cnn.Open();

try

{

string r = xml;

byte[] byteMessage = System.Convert.FromBase64String(r);

string strMessage = Encoding.GetEncoding("utf-8").GetString(byteMessage);

cmd.Parameters.AddWithValue("@data", strMessage);

cmd.CommandText = "exec update\_timetable\_json @data";

cmd.ExecuteNonQuery();

}

catch (Exception ex)

{

}

cnn.Close();

}

}

}

Клиентская часть WebServices.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<script src="https://cdn.jsdelivr.net/npm/js-base64@3.7.5/base64.min.js"></script>

<script>

function utf8\_to\_b64(str) {

console.log(Base64.encode(str));

return Base64.encode(str);

}

function ff()

{

var x1 = document.getElementById("p1");

var t1 = document.getElementById("t1");

x1.value= utf8\_to\_b64(t1.value);

}

</script>

<script>

function loadDoc() {

let xhr = new XMLHttpRequest();

let URL = document.getElementById('WS\_address').value;

document.getElementById('A1').innerHTML = "Открываю URL:";

xhr.timeout = 5000;

xhr.open("POST", URL, true);

document.getElementById('A1').innerHTML = "Посылаю запрос";

xhr.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");

let param = document.getElementById('WS\_param').value;

xhr.send(param);

xhr.onload = function() {

if (xhr.status != 200) {

// анализируем HTTP-статус ответа, если статус не 200, то произошла ошибка

alert('Ошибка '+ xhr.status+' '+ xhr.statusText);

} else { // запрос выполнился

document.getElementById('A1').innerHTML='Готово, получили '+xhr.response.length+'байт'; // response -- это ответ сервера

document.getElementById('A1').innerHTML = xhr.status;

document.getElementById('A2').innerHTML = xhr.statusText;

var s = xhr.response;

document.getElementById('A3').value = s;

}

};

xhr.onprogress = function(event) {

if (event.lengthComputable) {

document.getElementById('A1').innerHTML='Получено '+event.loaded+' из'+event.total+' байт';

} else {

document.getElementById('A1').innerHTML='Получено '+event.loaded+' байт'; // если в ответе нет заголовка Content-Length

}

};

xhr.onerror = function(e) {

alert("Во время загрузки документа произошла ошибка: " + e.target.status);

};

};

</script>

</head>

<body>

<h2>Вызов Web-сервиса</h2>

<p><b>Адрес Web-сервиса:</b> <input type="text" id="WS\_address"

style="width:400px"/><br/>

<sup>Путь к сервису. Например:

http://webservicesASPNET.somee.com/AutovauxSRV/AutovauxService.asmx/GetTimetables</sup><br/>

<h3>Кодирование параметра</h3>

<table>

<tr><td> Исходный текст:</td> <td><input type="text" id="t1"/></td><td> <input type="button"

value="Закодировать" onclick="ff()"/></td></tr>

<tr><td> base64 закодированный текст: </td> <td><input type="text" name="p1"

id="p1"></input></td><td>

<input type="button" value="Декодировать" onclick="ff1()"/></td></tr>

</table>

<b>Параметры (POST):</b> <input type="text" id="WS\_param" style="width:400px"/><br/>

<sup>Пара: имя0=значение0 Например: cid=1 или data=fLopIgfKDgdf==</sup><br/>

<b>Статус:</b> <span id="A1"></span>&nbsp; <span id="A2"></span></p>

<button onclick="loadDoc()">Получить данные</button>

<p><b>Ответ:</b><br/> <textarea rows="40" cols="72" name="text" id="A3"></textarea></p>

</body>

</html>