

### Лабораторная работа №3

В лабораторной работе №3 необходимо было разбить монолитный сервис на несколько микросервисов. Во-первых, нужно было разделить данные по разным базам. Созданы 3 базы: 1)База с таблицей Customers и Deliveries, 2)База с таблицей Goods и 3)база с таблицей Orders.

1)База с таблицей Customers и Deliveries:

Таблица Customers:

The screenshot displays the SQL Server Enterprise Manager interface on the left and a SQL Query window on the right. The Enterprise Manager shows the database structure for '2864\_lab\_2\_1', with the 'dbo.Customers' table selected. The SQL Query window shows a query to select the top 1000 records from the 'Customers' table, including columns for ID, Title, and Phone. Below the query, the 'Results' tab shows the first two rows of data.

```
SQLQuery6.sql - HP\...\jagan (51)
/***** Сценарий для команды SelectTopN
SELECT TOP 1000 [ID]
      , [Title]
      , [Phone]
FROM [2864_lab_2_1].[dbo].[Customers]
```

	ID	Title	Phone
1	2	Петров	123-123-32
2	3	Сидоров	32-33-55

## Таблица Deliveries:

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Обозреватель объектов' (Object Explorer) for the 'HP\SQLEXPRESS (SQL Server 10.0.5538 - HP)' instance. The 'Базы данных' (Databases) folder is expanded, showing '2864\_lab\_2\_1'. Under 'Таблицы' (Tables), the 'dbo.Deliveries' table is selected. The right pane shows the 'SQLQuery7.sql' window with the following query:

```
/****** Сценарий для команды SelectTopNRc
SELECT TOP 1000 [ID]
      , [Title]
      , [Cost]
FROM [2864_lab_2_1].[dbo].[Deliveries]
```

The 'Результаты' (Results) tab is active, displaying the following data:

	ID	Title	Cost
1	1	Курьер	400.00
2	2	Почта	140.00

## 2)База с таблицей Goods:

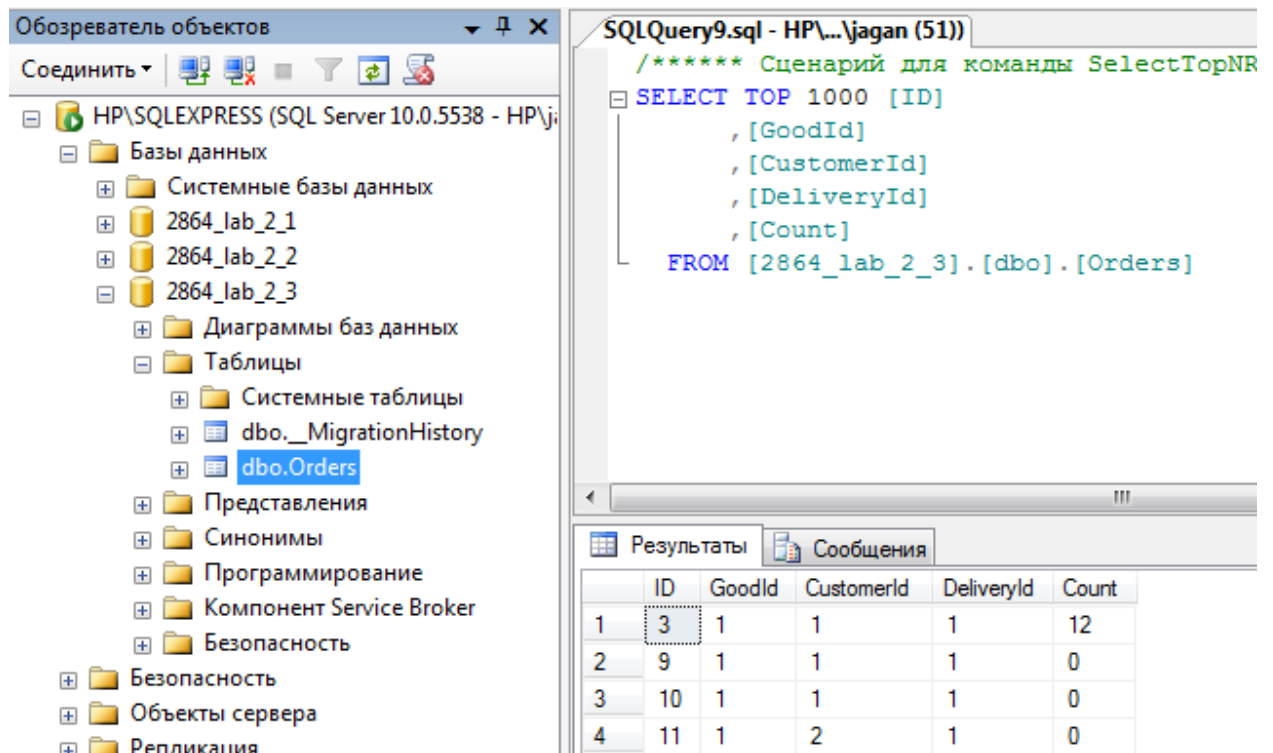
The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the 'Обозреватель объектов' (Object Explorer) for the 'HP\SQLEXPRESS (SQL Server 10.0.5538 - HP)' instance. The 'Базы данных' (Databases) folder is expanded, showing '2864\_lab\_2\_1' and '2864\_lab\_2\_2'. Under 'Таблицы' (Tables), the 'dbo.Goods' table is selected. The right pane shows the 'SQLQuery8.sql' window with the following query:

```
/****** Сценарий для команды SelectTopNRc
SELECT TOP 1000 [ID]
      , [Cost]
      , [Title]
FROM [2864_lab_2_2].[dbo].[Goods]
```

The 'Результаты' (Results) tab is active, displaying the following data:

	ID	Cost	Title
1	1	333.00	Кнопки
2	2	444.00	Ручка

### 3)База с таблицей Orders:



SQLQuery9.sql - HP\... \jagan (51))

```
/****** Сценарий для команды SelectTopNR
SELECT TOP 1000 [ID]
      , [GoodId]
      , [CustomerId]
      , [DeliveryId]
      , [Count]
FROM [2864_lab_2_3].[dbo].[Orders]
```

	ID	GoodId	CustomerId	DeliveryId	Count
1	3	1	1	1	12
2	9	1	1	1	0
3	10	1	1	1	0
4	11	1	2	1	0

Во-вторых, созданы 3 микросервиса для работы с этими базами. Для реализации микросервисов применена технология WCF сервиса с возможностью самостоятельно хостинга без IIS. За это отвечает приложение RunService.

На примере GoodService опишем принцип работы.

Сделаем сервис

```
[ServiceContract]
[ServiceBehavior(IncludeExceptionDetailInFaults = true)]
public class GoodsService
{
    [OperationContract]
    public string GetById(int id)
    {
        using (GoodsContext context = new GoodsContext())
        {
            return JsonConvert.SerializeObject(context.Goods.Where(x => x.ID ==
id).FirstOrDefault());
        }
    }
    [OperationContract]
    public string GetPage(int page, int size)
    {
        using (GoodsContext context = new GoodsContext())
        {
            return JsonConvert.SerializeObject(context.Goods.OrderBy(x =>
x.ID).Skip((page - 1) * size)
                .Take(size).ToList());
        }
    }
}
```

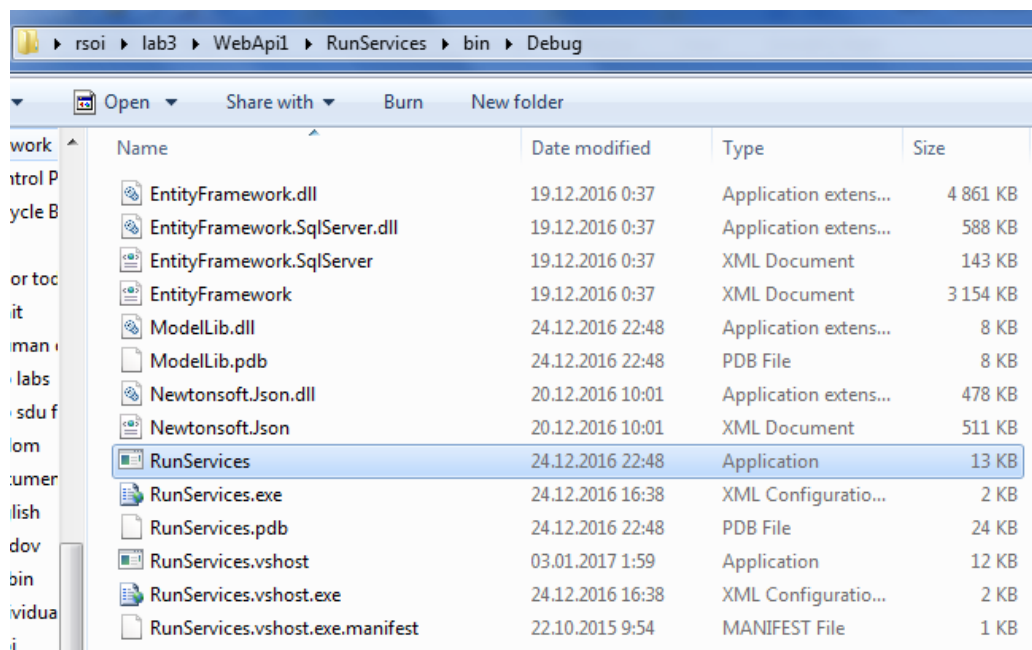
Данный сервис можно запустить командами

```
ServiceHost host = new ServiceHost(type, baseAddress);
ServiceMetadataBehavior smb = new ServiceMetadataBehavior();
smb.HttpGetEnabled = true;
smb.MetadataExporter.PolicyVersion = PolicyVersion.Policy15;
host.Description.Behaviors.Add(smb);
host.Open();
```

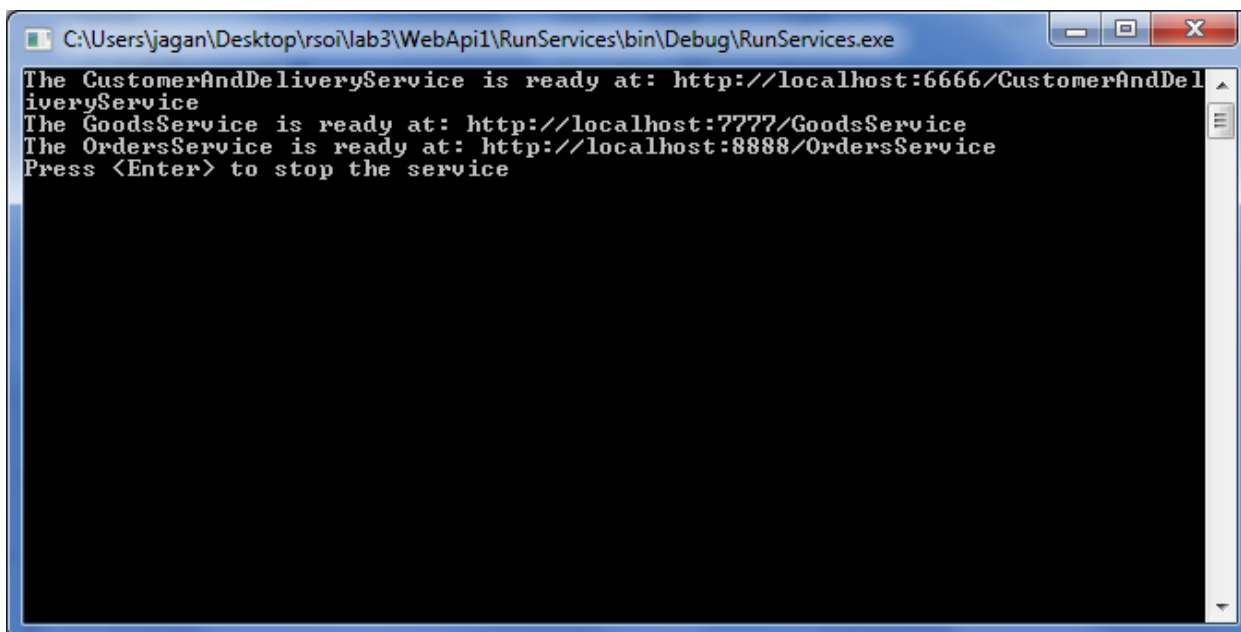
Это позволит обращаться к этому сервису по tcp/ip протоколу.

Консольное приложение содержит 3 микросервиса и после запуска три сервиса становятся активным и доступными для работы (запускать нужно под именем администратора). Внизу показано поэтапно как запускать эти микросервисы:

1-шаг: находим RunServices.exe



2-шаг: запускаем его под именем администратора



```
C:\Users\jagan\Desktop\rsoi\lab3\WebApi1\RunServices\bin\Debug\RunServices.exe
The CustomerAndDeliveryService is ready at: http://localhost:6666/CustomerAndDeliveryService
The GoodsService is ready at: http://localhost:7777/GoodsService
The OrdersService is ready at: http://localhost:8888/OrdersService
Press <Enter> to stop the service
```

Наверху на рисунке показано что после запуска микросервисов в командном строке окна запустились три микросервисов:

- 1)Customer and Delivery сервис на localhost:6666
- 2)Goods сервис на localhost:7777
- 3)Orders сервис на localhost:8888

По аналогии с классом MainControllers.cs сделаем класс MainControllers2.cs который будет содержать API для работы с микросервисами. Внешне набор API методов не изменился (только добавилась двойка например api/orders2).

Сервисы подключаются через диалог AddServiceReference. По wsdl генерируется клиентские классы, через которые удобно обращаться к методам сервисов.

Процедуры авторизации те же как в лабораторной работе-2, только страница для инициализации другая – contract и которая содержит всю логику тестирования.

Давайте покажем авторизацию программу поэтапно:

**1-шаг:** запускаем программу и видим ссылку «Home» и «Тест лаб. 3»

## Home Page.

 Modify this template to jump-start your ASP.NET MVC application.

To learn more about ASP.NET MVC visit <http://asp.net/mvc>. The page features [videos](#), [tutorials](#), and [samples](#) to help you get the most from ASP.NET MVC. If you have any questions about ASP.NET MVC visit [our forums](#).

### We suggest the following:

#### 1 Getting Started

## 2-шаг: На нем нажимаем на ссылку «Тест лаб. 3» без авторизаций

Запрос: `http://localhost:60116/api/customers2?page=1&size=5`

Ответ: Недостаточно прав для выполнения операции

Запрос: `http://localhost:60116/api/orders2?page=1&size=5`

Ответ: Недостаточно прав для выполнения операции

Запрос: `http://localhost:60116/api/orders2`

Ответ: Недостаточно прав для выполнения операции

Запрос: `http://localhost:60116/api/orders2?orderId=0&goodId=1&customerId=2&deliveryId=1`

Ответ: Недостаточно прав для выполнения операции

Запрос: `http://localhost:60116/api/orders2/0`

Ответ: Недостаточно прав для выполнения операции

Запрос: `http://localhost:60116/api/orders2?orderId=0`

Ответ: Недостаточно прав для выполнения операции

[Авторизовать приложение по OAuth 2](#)

И мы видим что без авторизаций у нас не достаточно прав для выполнения операции, но на самом конце мы видим еще и ссылку «Авторизовать приложение по OAuth 2».

**3-шаг:** если нажать на ссылку «Авторизовать приложение по OAuth 2» то получаем:

your logo here

Register Log in

Home Тест лаб. 3

### Log in.

#### Use a local account to log in.

User name

Password

☐ Remember me?

Log in

[Register](#) if you don't have an account.

#### Use another service to log in.

There are no external authentication services configured. See [this article](#) for details on setting up this ASP.NET application to support logging in via external services.

**4-шаг:** можно регистрироваться или если ранее регистрировались то можно и вести логин и пароль.

your logo here

Register Log in

Home Тест лаб. 3

### Log in.

#### Use a local account to log in.

User name

Password

☒ Remember me?

Log in

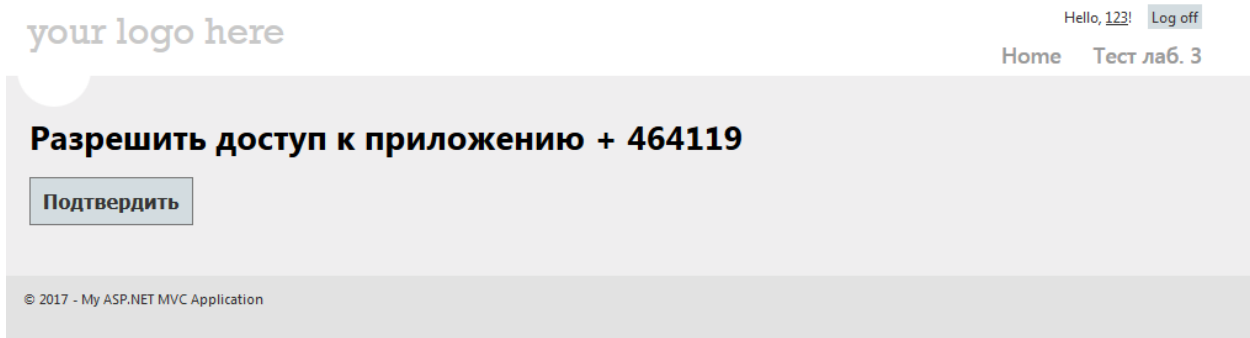
[Register](#) if you don't have an account.

#### Use another service to log in.

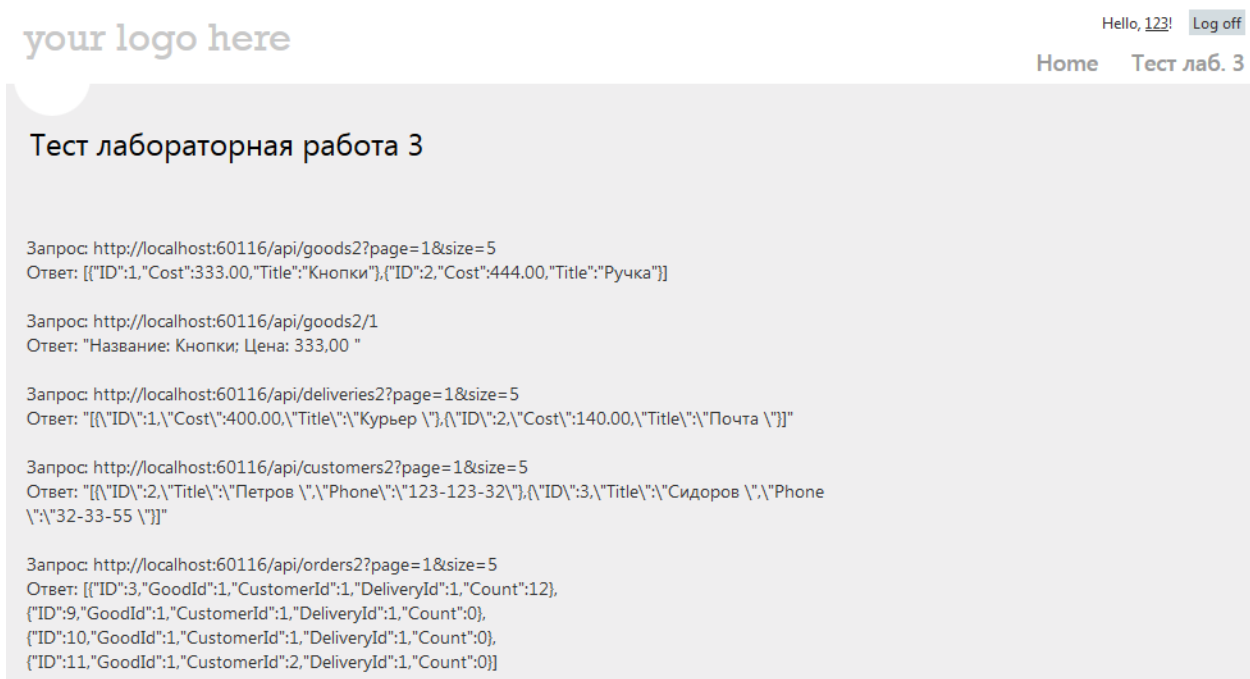
There are no external authentication services configured. See [this article](#) for details on setting up this ASP.NET application to support logging in via external services.

© 2017 - My ASP.NET MVC Application

**5-шаг:** после того как ввели логин и пароль, нажимаем на log in:



**6-шаг:** нажимаем на подтверждение досупа:



Вот так мы запускаем тест для лабораторной работы-3 с авторизацией

Если посмотреть внимательно на содержимое, то видно отличие по данным, т.к. данные для лабораторной работы №2 и №3 берутся из разных источников.



## Соответствие требованиям

1. Данные каждого сервиса можно хранить как в SQL, так и в NoSQL базе. Для упрощения допускается хранить данные на одной базе, но в разных схемах. При этом каждый сервис должен взаимодействовать только со своей схемой, получение данных, не относящихся к текущему сервису строго запрещено.

3 базы MS SQL и каждый микросервис общается со своей базой.

2. Нельзя использовать готовые библиотеки для авторизации по OAuth2.0.

Сделано в ручную

3. Для токена нужно реализовать время жизни (expires) и обновление токена (через refresh token).

Сделано HomeController/AuthMethod

4. Должен быть хотя бы один запрос, требующий агрегированной информации от двух и более сервисов.

Orders2Controller метод Get(id)

5. Все взаимодействие между сервисами выполнить в парадигме RESTful.

Да, есть.

6. Предусмотреть работу системы в случае отказа одного из компонентов системы.

В случае отказа микросервиса данные из других сервисов будут читаться.

7. При получении списков данных предусмотреть пагинацию.

Есть

8. Сделать подробное логгирование выполняемых действий на каждом сервисе.

Есть. В конфиге Nlog.config в каждом проекте указан файл в который пишутся логи. Сейчас он везде один – C:\log.txt.

В него пишут все микрофреймворки выполняя методы.

9. Подготовить шаблоны запросов или маленький скрипт для демонстрации работы.

Есть тестовая веб страница