Introdução

Este documento descreve a aplicação desenvolvida para o teste técnico enviado. As seguintes tecnologias foram utilizadas:

- Visual Studio 2013
- SQL Server 2014
- SoupUI

A tecnologia utilizada na implementação do projeto foi o WCF Library para construção dos endpoints, esta template foi escolhida devido a possibilidade de que a mesma possa ser hospedada de diferentes formas, como por exemplo: Console Aplication, Windows Service e IIS/WAS .

A tecnologia WCF foi escolhida considerando que não foi definida a forma de hospedagem do serviço, logo foram implementadas três possibilidades, são elas hospedagem no IIS, serviços windows e como uma aplicação desktop. Essa hospedagem deve ser realizada para que o serviço possa ser executado no servidor que centraliza o jogos.

Para gerenciamento da massa de dados foi criado um banco de dado. O banco escolhido foi o SQL Server 2014.

Descrição da solução

A solução , WcfServiceLibraryGame, é composta por 4 projetos.

- ConsoleHosting:

  Implementa a hospedagem do serviço com o console application.

- WcfServiceHosting:

  Implementa a hospedagem do serviço para publicação no IIS.

- WcfServiceLibraryGame

  Implementa os endpoints para gravação na base de dados. No capítulo abaixo encontra-se toda a descrição desta implementação

- WindowsServiceHostingGames

  Implementa a hospedagem do serviço como windows service.

Descrição da implementação

Na solução do Visual Studio o projeto que contém a implementação dos endpoints é o **WcfServiceLibraryGame**. Neste projeto apresenta a interface (IServiceGames) que descreve os dois endpois requisitados no teste. A classe ServiceGames implementa os métodos e realiza a invocações de métodos para gravação de dados.

A classe DBGamesPlayers realiza as operações de gravação e busca de dados, no banco de dados criado.

Para criação do serviço foi utilizada a arquitetura REST e o formato das mensagens escolhido foi JSON. Esta arquitetura foi escolhida, pois foi trabalhado somente o protocolo HTTP e o formato JSON foi escolhido devido ao fato que as mensagens neste formato são menores e mais eficiente, ou seja sua carga de dados é menor assim facilitando a transição de informações.

Para gerenciamento da massa de dados foi escolhido o banco de dados SQL Server. O banco foi nominado dbGames. As tabelas criadas são players, game, scoreGamePlayers. A tabela scoreGamePlayers possui relacionamento com as duas tabelas criadas. Para gravação de registro foi criada uma stored procedure, onde é gravado registros enviados pelos servidores. Para busca do top 100 jogadores outra stored procedure foi criada.

Passos para Executar a Aplicação Implementada

1. Criação da base de dados:
   Para a criação da base de dados é necessário executar o script enviado (createDataBaseGame.sql) no sql management studio. Este arquivo esta no diretório Scripts do projeto enviado.
2. Configuração da string de conexão para a base de dados:
   Junto aos binários do projeto existe um arquivo denominado parametro.xml, neste arquivo há um campo chamado stringdeConexao. Neste campo insira a sua string de conexão.
3. Execução da aplicação:

   Para executar a aplicação você pode fazer 3 tipos de hospedagem:

   1. Desktop (Console Application)

      Para hospedar neste modo é necessário executar em **modo administrador** o arquivo consoleHost.exe. e realizar todos os teste desejados. Para realização de teste foi utilizada a ferramenta SoupUI, no próximo tópico será demonstrado como foi realizado os teste. Utilizando esta forma de hospedagem o endereço será http://localhost:8081/GameScore

   2. Desktop (Windows Service)

      Desta forma será criado um serviço no windows para hospedar a aplicação. É necessário instalar o serviço WindowsServiceHostingGames.exe , utilizando o UtilInstall.exe do framework .NET

   3. IIS

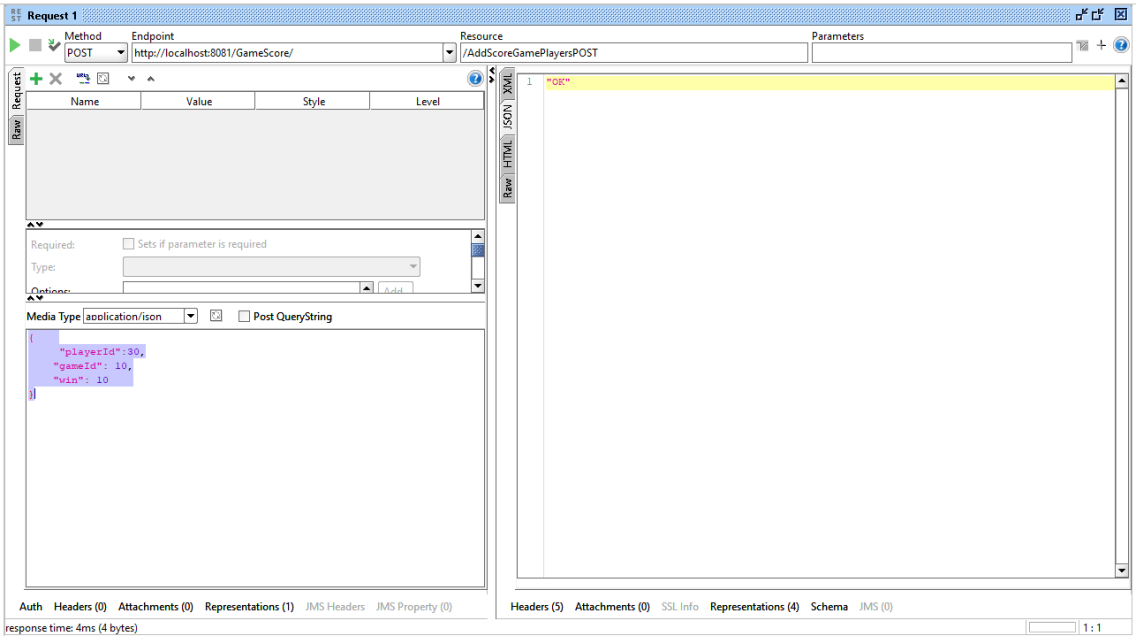
Há possibilidade de hospedagem no IIS. Caso deseje utilizar este método hospedagem deve-se o utilizar o projeto WcfServiceHosting para fazer o deploy no IIS e criar o serviço arquivo .svc

Testes Realizados

Foi utilizado a ferramenta SoupUI para realizar testes de chamadas dos métodos , abaixo segue os teste:

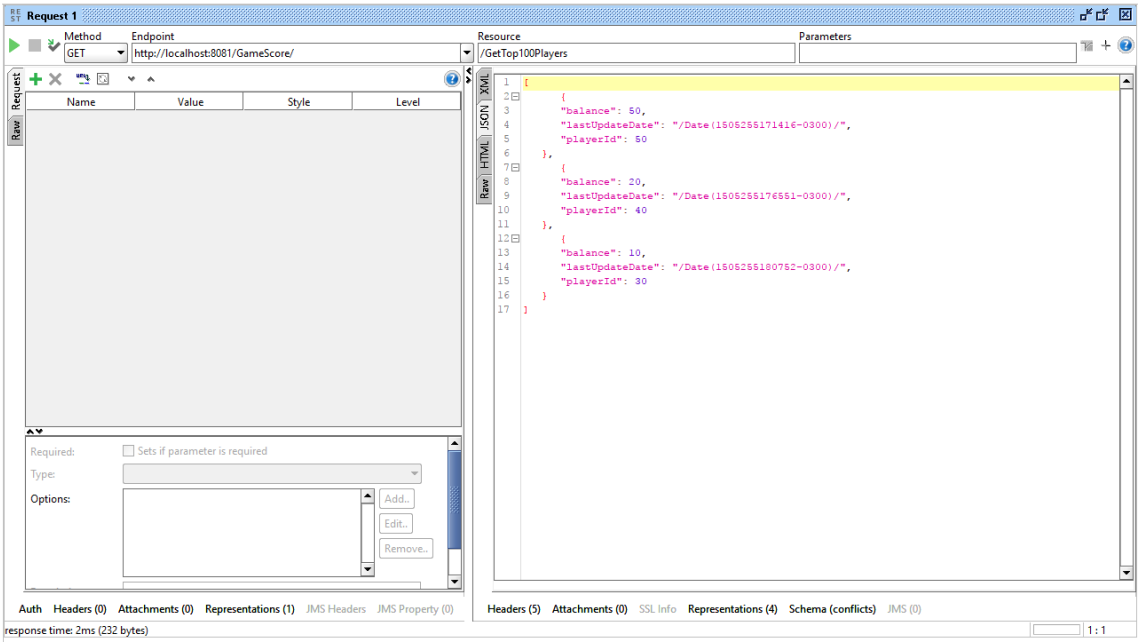
- **Teste com o endpoit /AddScoreGamePlayersPOST**

Na figura abaixo esta representado o teste de POST para gravação do score dos jogadores na base de dados



- Teste com o endpoit /GetTop100Players

Na figura abaixo esta representado o teste de GET onde é retornado os top 100 jogadores com maior score.


Request 1
Method
GET
Endpoint
http://localhost:8081/GameScore/
Resource
/GetTop100Players
Parameters
Request
Raw
Name
Value
Style
Level
Required:
Sets if parameter is required
Type:
Options:
Add..
Edit..
Remove..
Auth
Headers (0)
Attachments (0)
Representations (1)
JMS Headers
JMS Property (0)
XML
JSON
HTML
Raw
[
{
"balance": 50,
"lastUpdateDate": "/Date(1505255171416-0300)/",
"playerId": 50
},
{
"balance": 20,
"lastUpdateDate": "/Date(1505255176551-0300)/",
"playerId": 40
},
{
"balance": 10,
"lastUpdateDate": "/Date(1505255180752-0300)/",
"playerId": 30
}
]
Headers (5)
Attachments (0)
SSL Info
Representations (4)
Schema (conflicts)
JMS (0)
response time: 2ms (232 bytes)
1 : 1