# Especificação do Sistema

## Objetivo

Criação de um Web Service com dois métodos que retornem, em formato JSON, a distância entre duas cidades e as cidades mais próximas de uma cidade solicitada.

## Métodos Criados

**1 – Distância entre duas cidades**

Endereço: URLBASE/api/Operation/GetDistance?city01=1&city02=5

Retorna a distância entre duas cidades em quilômetros. Possui os parâmetros city01 e city02, referentes ao código das duas cidades que desejamos calcular a distância.

**2 – Cidades mais próximas**

Endereço: URLBASE /api/Operation/GetClosestCities?city=1 OU

URLBASE /api/Operation/GetClosestCities?city=1&total=3

Deve ser informado o id da cidade base da pesquisa, através do parâmetro “city”. O serviço retorna as cidades mais próximas dessa cidade escolhida.

Existe um segundo parâmetro chamado ”total”, referente ao número de cidades que devem ser retornadas. Esse parâmetro total é opcional e o valor padrão é dois. Isso significa que se ele não for informado na url, o serviço vai retornar as duas cidades mais próximas.

**3 – Lista das Cidades**

Apesar de não ser item obrigatório no teste, a listagem das cidades existentes no cadastro se faz importante para a visualização dos itens e de seus respectivos ids.

Endereço: URLBASE /api/Operation/GetCitiesList

## Arquitetura Utilizada

O teste foi desenvolvido utilizando como IDE o Visual Studio 2012. Os serviços foram codificados em C#, através do framework Web Api.

Foi utilizado o ORM Entity Framework, como uma configuração customizada que, caso o banco de dados não exista, realiza a criação automaticamente e já popula a tabela de cidades. Isso significa que basta adaptar a string de conexão ao seu servidor e rodar o projeto.

O banco de dados pode ser tanto o SQL Server quanto o MySQL, basta comentar uma das strings de conexão e descomentar a relacionada ao SGBD desejado.

## Observações

A latitude e longitude normalmente podem ser identificadas de duas formas: através da notação comum de coordenada, que utiliza graus, minutos e segundos (ex: 23°46'44.3"S) e através de uma notação decimal (ex: -23.954719). O serviço permite que ele seja cadastrado de qualquer uma das duas formas.

## Passos Futuros

Pensando em otimizar a performance, é possível realizar alguma refatoração para melhorar o desempenho, sobretudo no método que retorna as cidades mais próximas. Uma das soluções que podem ser testadas é a criação de uma view ou de uma stored procedure.

## Cálculo da Distância

Os seguintes passos foram realizados no cálculo da distância:

1 – Conversão da latitude e longitude para a notação decimal, através da conversão de minutos e segundos em graus;

2 - Formação da notação com o sinal, considerando "N" e "E" com sinal positivo e números "S" e "W" com sinal negativo;

3 – Conversão em radianos através da fórmula r = d\*(pi/180), onde pi é igual a 3,14159;

4 – Calcular a mudança na latitude e longitude do ponto inicial até o final;

5 – Aplicar os dados obtidos na fórmula [sen²(Δlat/2) + cos(lat1)] x cos(lat2) x sen²(Δlong/2), onde Δlat é a mudança na latitude e Δlong é a mudança na longitude.

6 – Aplicar o resultado da formula anterior na formula 2 x cot(√a/√(1−a)), onde a é igual ao resultado da fórmula anterior.

7 – Multiplicar o valor encontrado pelo raio da Terra (6,371 km).

## Fontes de Consulta

Fórmula:

<http://www.ehow.com.br/calcular-distancia-entre-pontos-latitude-longitude-como_71372/>