

# **Desgin Document for Weather REST API Service**

## Revision History

Approver Role	Name	Signature
Integration Consultant	Premkumar Natarajan <a href="mailto:premkumar502@gmail.com">premkumar502@gmail.com</a> <a href="http://www.linkedin.com/in/premn">http://www.linkedin.com/in/premn</a>	Version 1.0

## Table of Contents

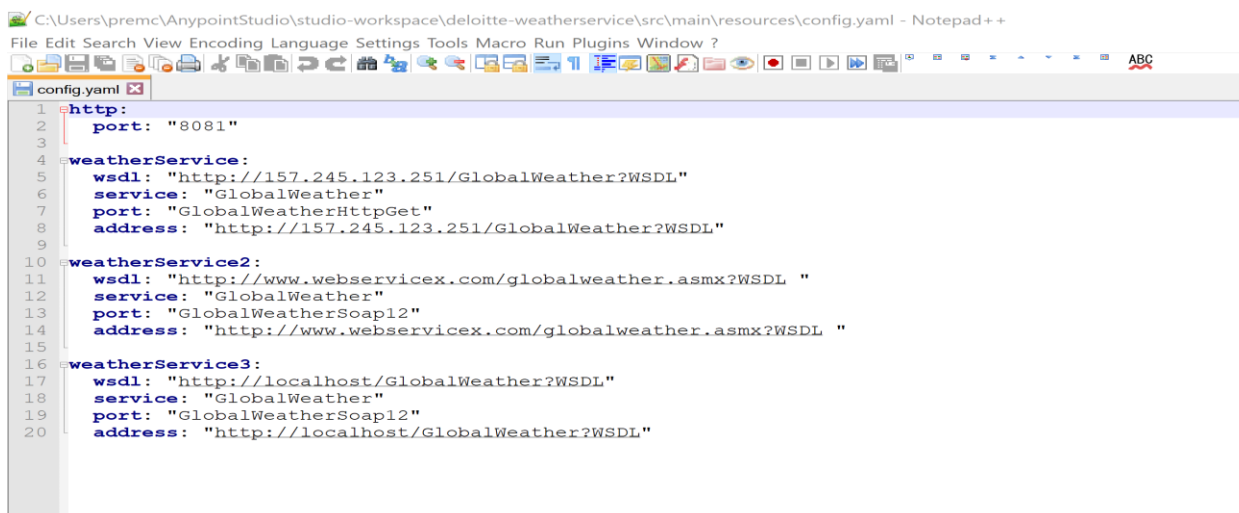
1	INTRODUCTION.....	4
2	ASSUMPTIONS .....	4
3	ENVIRONMENT DETAILS.....	4
4	PREREQUISITES .....	5
5	WEATHER SERVICE REST API OPERATIONS .....	5
6	MULE FLOWS.....	7
6.1	Weather-Service API Main Flow .....	7
6.2	Implementaion Flow .....	7
6.3	External SOAP Webservice .....	8
6.4	Mule Exchange – Public Portal .....	8
7	ERROR HANDLING.....	8
8	UNIT TEST CASE.....	9
8.1	/cities – get .....	9
8.2	/weather – get.....	10
9	SCOPE FOR ENHANCEMENT AND OPEN ISSUES.....	11
9.1	Scope for Enhancement.....	11
9.2	Challenges Faced .....	11
10	APPENDIX.....	11
10.1	Docker Setup to Run NodeJS .....	11
11	REFERENCES.....	11

## 1 INTRODUCTION

Weather REST API exposes the the operations from the external weather SOAP webservice (<http://www.webservicex.com/globalweather.asmx?WSDL>). Weather REST API Service been developed using RAML and Mule 4.

## 2 ASSUMPTIONS

- Provided external webservice URL <http://www.webservicex.com/globalweather.asmx?WSDL> is not available. So, Mule REST API, references the web service provided by Deloitte which been build using NodeJS/Docker.
- I have used my DigitalOcean Cloud portal to run the NodeJS service in docker provided by Deloitte and refereed my cloud ip for all development. <http://157.245.123.251/GlobalWeather?WSDL>
- If deloitte team needs to change the external SOAP service URL, then update config.yaml file in the mule application code as shown below:
  - **Line No 4:** Update `weatherService:` to `weatherService2`
  - **Line No 10:** Update `weatherService2:` to `weatherService`



```
1 http:
2   port: "8081"
3
4 weatherService:
5   wsdl: "http://157.245.123.251/GlobalWeather?WSDL"
6   service: "GlobalWeather"
7   port: "GlobalWeatherHttpGet"
8   address: "http://157.245.123.251/GlobalWeather?WSDL"
9
10 weatherService2:
11   wsdl: "http://www.webservicex.com/globalweather.asmx?WSDL "
12   service: "GlobalWeather"
13   port: "GlobalWeatherSoap12"
14   address: "http://www.webservicex.com/globalweather.asmx?WSDL "
15
16 weatherService3:
17   wsdl: "http://localhost/GlobalWeather?WSDL"
18   service: "GlobalWeather"
19   port: "GlobalWeatherSoap12"
20   address: "http://localhost/GlobalWeather?WSDL"
```

- Provided NodeJS webservice response for getCities and getWeather operations are not formatted xml and it contains CDATA. So, the REST API also displays exactly as the response from the external webservice.

## 3 ENVIRONMENT DETAILS

- |                       |   |   |
|-----------------------|---|---|
| • Mule Server Version | : | 4.2.0   |
| • Anypoint Studio     | : | 7.3.4   |
| • Mule Modules        | : | APIKit (1.1.9), HTTP (1.5.3), Sockets (1.1.5), Web Service Consumer (1.3.1) |
| • RAML Version        | : | 1.0   |

- Cloud to host webservice : Digital Ocean
- Cloud Weather service URL : <http://157.245.123.251/GlobalWeather?WSDL>
- Mule Exchange Public Portal : <https://anypoint.mulesoft.com/exchange/portals/gmail-394/62e8b8e3-8c39-403f-814c-31b913ca0f5d/weather-service-api/>
- SOAP UI Tool : 5.5
- GIT : <https://github.com/premkumarmlp/deloitte-weatherwesbservice>

## 4 PREREQUISITES

1. External SOAP WebService should be available. Code been developed using <http://157.245.123.251/GlobalWeather?WSDL>
2. Access to Mule Platform
3. SOAP UI tool to test the REST and SOAP service.

## 5 WEATHER SERVICE REST API OPERATIONS

Weather REST API provides the below service and respective operation are listed below.

### 1. Cities Service: **/api/cities/**

#### 1.1. GET

*Description:*

Get the all the cities for the given country in the query parameters. If the query parmeter is not given, API retrieves all the cities from country Australia.

*Query Parameter:*

Parameter	Type	Optional	Example and URI
countryName	String	Yes	countryName=Australia URI: <a href="http://localhost:8081/api/cities?countryName=Australia">http://localhost:8081/api/cities?countryName=Australia</a>

### 2. Weather Service: **/api/weather/**

#### 2.1. GET

*Description:*

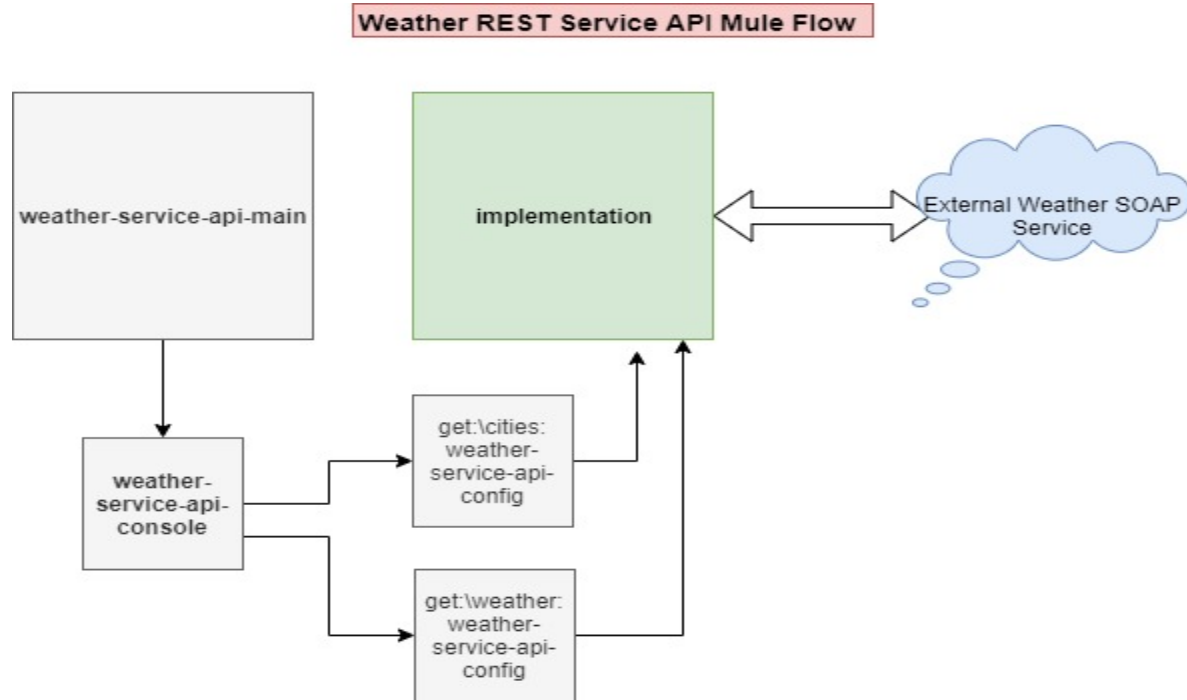
Get the weather data for the given citiy in the country which will be passed in the query parameters. If the query parmeter is not given, API retrieves weather data for the city Melbourne from country Australia.

*Query Parameter:*

Parameter	Type	Optional	Example and URI

countryName	String	Yes	countryName=Australia http://localhost:8081/api/weather?countryName=Australia&cityName=Melbourne
cityName	String	Yes	cityName=Melbourne http://localhost:8081/api/weather?countryName=Australia&cityName=Melbourne

## 6 MULE FLOWS



Overall technical architecture illustrates the mule flow used for developing this REST service.

### 6.1 Weather-Service API Main Flow

Its an interface for the implementation of the weather service REST API and the it provides the console for the testing the service. Its been built using APIKit plugin in anypoint Studio.

As defined in the RAML file (weather-service-api.raml), based on the url APIRoute in the main flow routes the request to the respective operation in the implementation flow.

Interface flow is build based on the RAML definition (weather-service-api.raml).

### 6.2 Implementaion Flow

Implementation flow been developed in the anypoint studio and its been invoked by the interace weather-service-api-main (API gateway) for the services. Implementation message flow comprises 2 flows for the operations **getCitiesByCountry** and **getWeather** from the external SOAP service.

#### **getCities:**

Its used to consume the external webservice using the queryparameter (countryName) and displays the list of cities from the country.

#### **getWeather:**

Its used to consume the external webservice using the queryparameter (countryName, cityName) and displays the weather data of cities from the country.

### 6.3 External SOAP Webservice

Deloitte provide the below external soap webservice to retrieve the citted and weather data of the city.  
<http://www.webservices.com/globalweather.asmx?WSDL>.

Due to service wsdl unavailability, used the NodeJS mock implementation of webservice for weather service. And this service been hosted in my digital ocean cloud portal (<http://157.245.123.251/GlobalWeather?WSDL>). If the code to be modified for the anyother webservice wsdl , please refer the section **Assumptions** to update the wsdl.

### 6.4 Mule Exchange – Public Portal

I have published this REST service in my exchange portal and external users can mock test this service.

<https://anypoint.mulesoft.com/exchange/portals/gmail-394/62e8b8e3-8c39-403f-814c-31b913ca0f5d/weather-service-api/>

## 7 ERROR HANDLING

Weather-Service API Main interface handles below error:

- APIKIT:BAD\_REQUEST
- APIKIT:METHOD\_NOT\_ALLOWED
- APIKIT:NOT\_ACCEPTABLE
- APIKIT:UNSUPPORTED\_MEDIA\_TYPE
- APIKIT:NOT\_IMPLEMENTED

Implementation Flow handles below error:

- getCities
  - 400 : "Error retrieving cities from the Weather Service API."
- getWeather
  - 400 : "Error retrieving weather data from the Weather Service API."

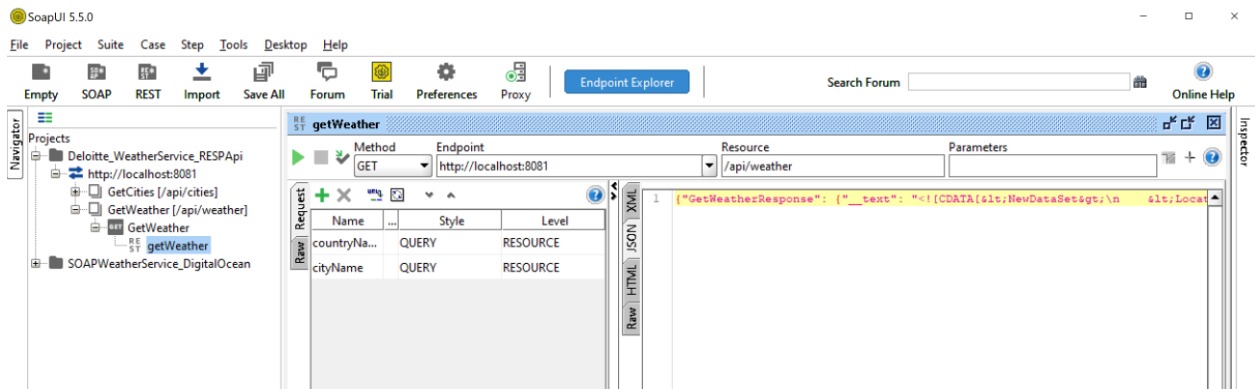


## 8 UNIT TEST CASE

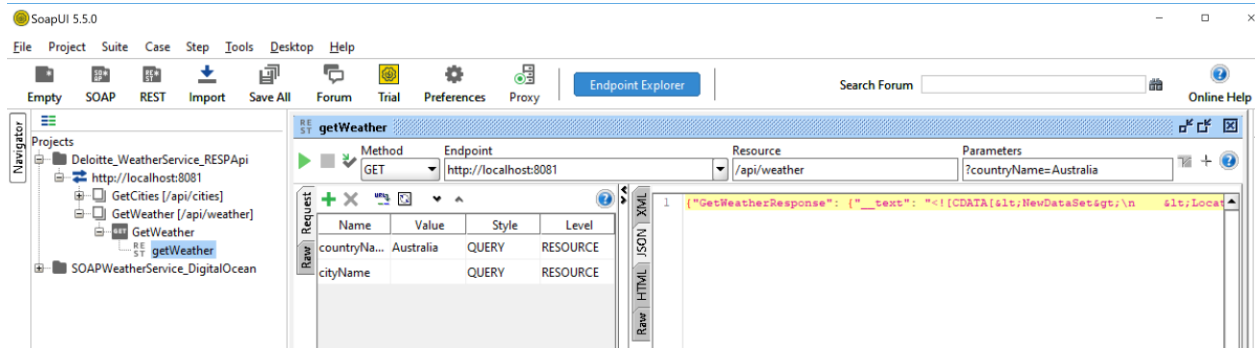
SOAP UI tool beed used to test the REST Service developed using Mule 4 and SOAP UI projects been uploaded in the GIT (deliverables).

### 8.1 /cities – get

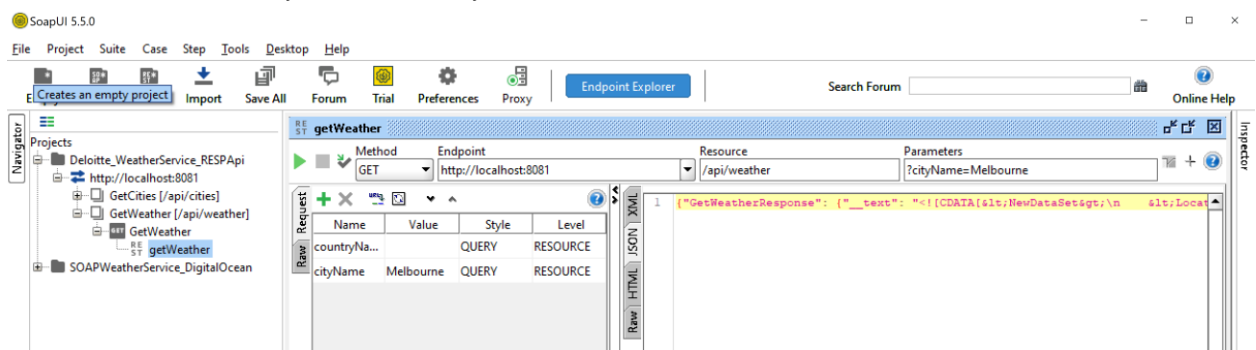
1. Test Case 1: Without Query Parameter:



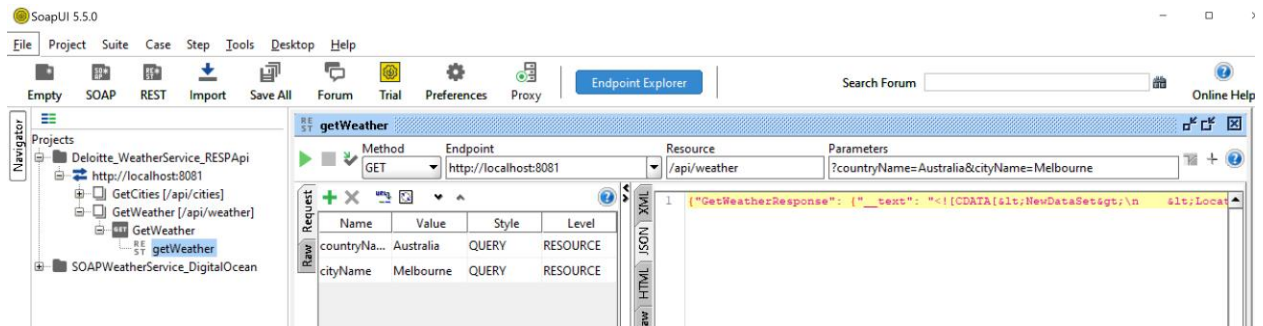
2. Test Case 2: With Query Parameter: countryName



3. Test Case 3: With Query Parameter: cityName

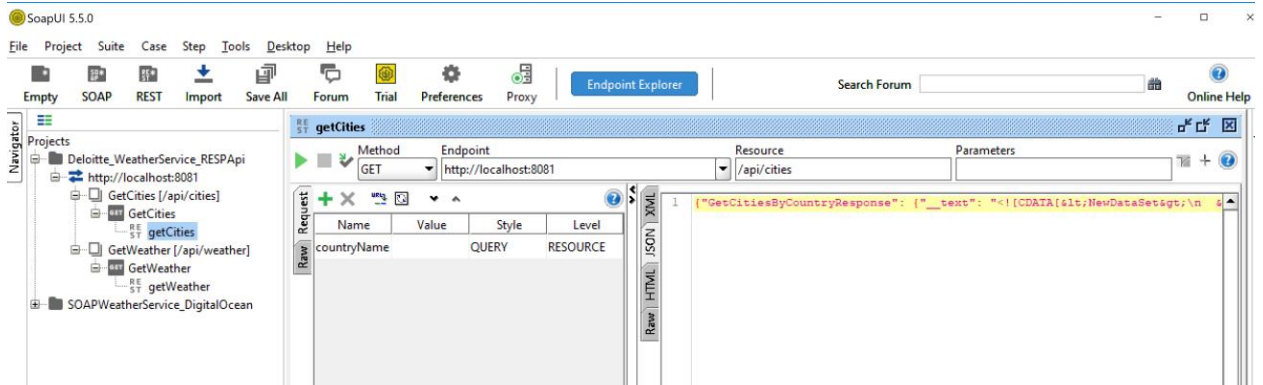


4. Test Case 4: With Query Parameter: countryName and cityName

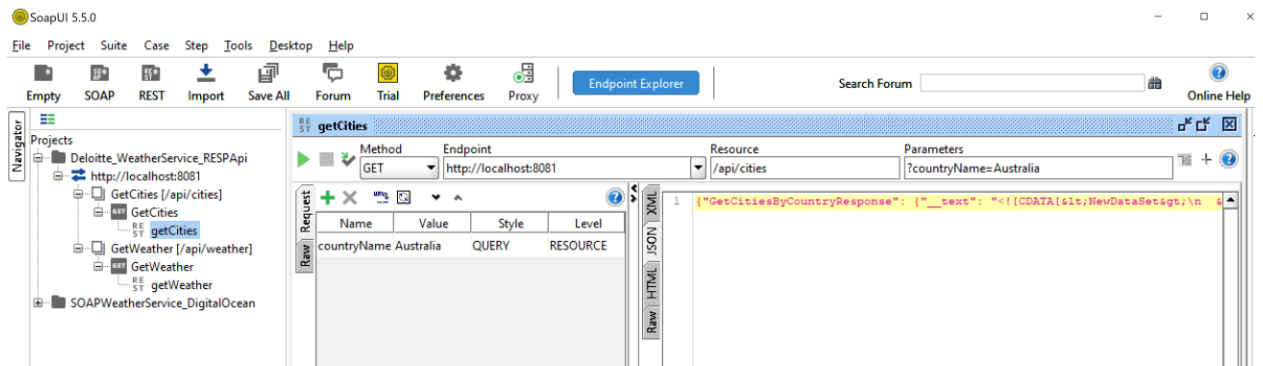


## 8.2 /weather – get

### 1. Test Case 1: Without Query Parameter:



### 2. Test Case 2: With Query Parameter: countryName



Also I have proved the SOAP UI project in the git to test my SOAP service running in the digital ocean docker.

## 9 SCOPE FOR ENHANCEMENT AND OPEN ISSUES

### 9.1 Scope for Enhancement

Webservice can be enhanced further using the below features:

1. External SOAP Webservice can be enhanced to streamline the data in as per the standard schema.
2. Implement the operation POST, PUT and DELETE for the cities and weather service REST API.
3. Schema validation for the JSON data can be applied in the REST validator.
4. Can create docker images of this application and upload in the docker hub.

### 9.2 Challenges Faced

1. Response from the SOAP webservice is not standard xml and it contains CDATA. So had to be tricky to implement the datatype and example.
2. Facing error in mule due to wsdl metadata:  

```
ERROR 2019-10-02 13:11:58,882 [[MuleRuntime].io.07: [deloitte-  
weatherservice].getWeather.BLOCKING @96d2c72] [event: 635df0c0-e4c2-11e9-84c7-  
aeb6d0920871] org.mule.wsdl.parser.operation.WsdlOperationTypeParser$Companion:  
Error building operation [GetCitiesByCountry] typeTrying to resolve metadata  
for a nameless part, probably the provided WSDL is invalid.  
java.lang.RuntimeException: Trying to resolve metadata for a nameless part,  
probably the provided WSDL is invalidError.
```
3. Port binding issue (8081) and resolved after identifying the vpn port blocks.

## 10 APPENDIX

### 10.1 Docker Setup to Run NodeJS

I have uploaded the webservice code provided by deloitte in my github. From the repo, I have built the docker images and executed in the digital ocean droplet.

1. Login to the cloud (UNIX) machine with your account.
2. Run the below commands
  - o git clone <https://github.com/premkumarmpl/deloitte-weatherwebservice.git>
  - o cd deloitte-weatherwebservice/weatherExerciseDockerFile/
  - o docker build -t deloittews .
  - o docker images
  - o docker run -d -it -p 80:8080 --name=ds deloittews
  - o docker ps

## 11 REFERENCES

1. <https://help.mulesoft.com/s/forum>
2. <https://docs.docker.com/engine/reference/commandline>
3. <https://www.digitalocean.com/>

--