**Mulesoft Technical Design Document**

**Project Name: Weather System API**

| Date | Author | Version | Change Reference |
| --- | --- | --- | --- |
| 26/01/2020 | Divya N | 1.0 | Initial draft |

Table of Contents

[Approval 3](#_Toc25141311)

[1. Introduction 4](#_Toc25141313)

[2. Mulesoft Design 4](#_Toc25141316)

[3. MUNIT TESTS](#_Toc25141318) 8

[4. LOGGING](#_Toc25141319) 8

[5. Error Handling](#_Toc25141319) 8

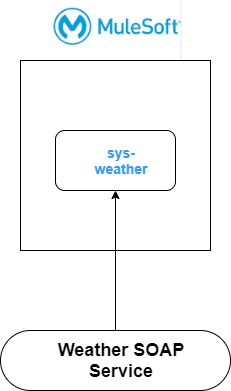
# Approval:

|  |  |  |
| --- | --- | --- |
|  | **Name** | **Date** |
| **Created By:** | Divya N | Creation Date: 26th January 2020 |
| **Updated By** |  |  |
| **Reviewed By:** |  |  |
| **Approved By:** |  |  |

# Introduction:

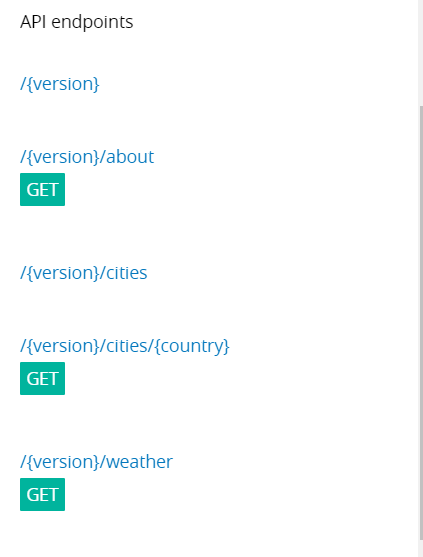
This document addresses the design of a Mulesoft application to extract a list of cities, to get weather data from **Weather SOAP Service** andtransform the data into JSON format.

# MuleSoft Design:



**Below are the endpoints exposed to get the data:**

* Get all city names for a country by passing a “country” name in the URI parameter
* Get weather details for a city/country by passing a “city” or “country” name as query parameters.



/{version}/about

**GET**

To do the health check on the API

/{version}/cities/{country}

**GET**

Get city names for a country

/{version}/weather

**GET**

Get weather details for a city/country by passing city/country name as query parameter.

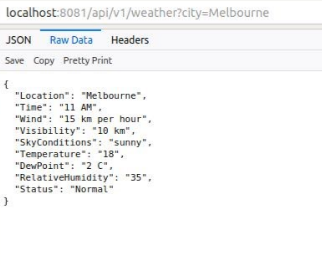
**How to invoke the endpoints:**

* Run the Application in Anypoint Studio.
* Open Postman or simply go to a browser and invoke the below urls (Sample output is given for each endpoint)

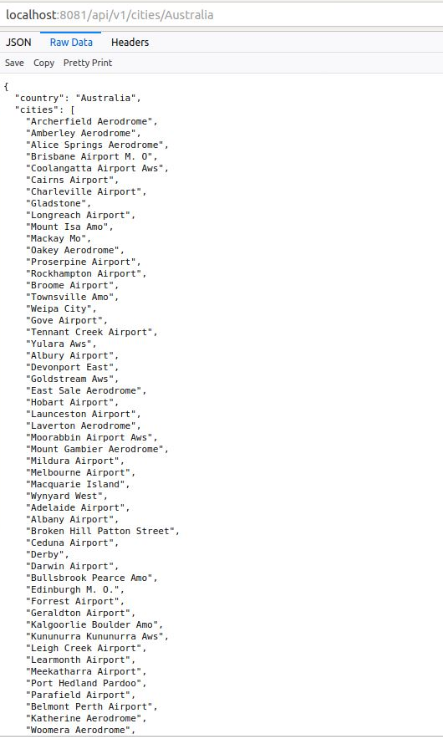
<http://localhost:8081/api/v1/about>



<http://localhost:8081/api/v1/weather?city=Melbourne>



<http://localhost:8081/api/v1/cities/Australia>



**Public Exchange url:** The API is published in Exchange. Please refer to the below Public Exchange link:

<https://anypoint.mulesoft.com/exchange/portals/qwerty-98/>

# MUnit Tests :

MUnit Tests are written to cover the possible success and error scenarios with code coverage.

# Logging :

Standard logging in JSON format is followed along with a correlation ID to track transaction.

# Error Handling :

If any error occurs in the application, global error handler handles the error and gives a standard error response with error details and correlation ID.

