**ICT375 Assignment 2 report**

**Name: Ow Jing Wei**

**Student No.: 34053405**

# Introduction

It is a weather aggregator website where user selects the type of data (windspeed and/or solar radiation) and result they want to display based on the year selected. User options will be sent and communicate with the server, and this is implemented using Node.js. The web browser serves as the client and HTTP protocol of Node.js core module will be used for all communication between client and server.

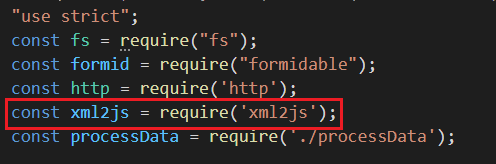
There is only 1 page for this website and user is required to select options of data they want to display, year, range of months and how the data will be displayed.

# Full description of XML and JSON

## XML

For parsing of XML data, I have decided to use xml2js node npm module to parse XML data to JavaScript object after the retrieval of the data from URL provided. Below shows some of the screenshots of code on how this technology is implemented in my solution.

**The xml2js npm module was installed and imported to the project:**



**On line 83, the parser was instantiated: Text

Description automatically generated**

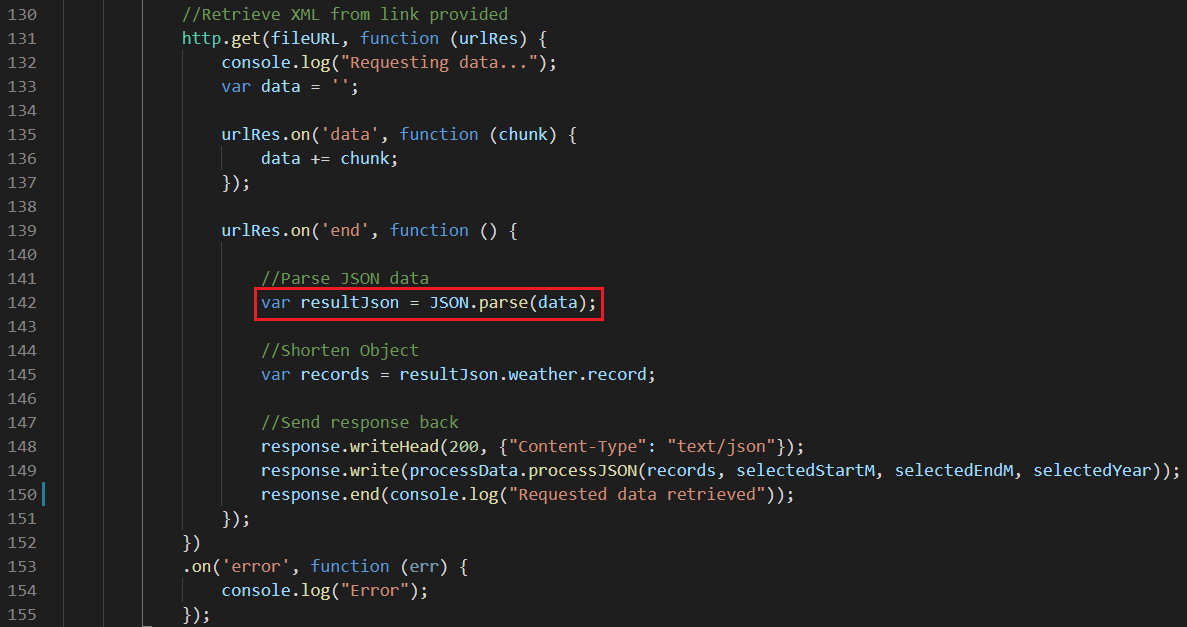
**On line 91, the parser was used to parse the XML data to JavaScript object with error handling: Text

Description automatically generated**

## JSON

For parsing of JSON data, no external node npm was used. JSON.parse() method was implemented to parse the string from JSON data retrieved from the URL provided. Below shows some of the screenshots of code on how this technology is implemented in my solution.

**On line 142, the JSON.parse() method was used to parse the JSON data**



# Overall design (Flow chart)

## Request website

When requesting website, user will enter the URL of the website, such as ceto.murdoch.edu.au:40019, and the browser will send request to 2 request handlers “/css/styles.css” and “/js/client.js ” on the server for the CSS and JavaScript of the website and send it back to the client to render the design and functionality of the website.

## Request XML

**Sending GET request (Client-side)**

After the form is checked and selected with the correct information of the user’s choice, the form will be sent using XMLHttpRequest to the server requesting “/reqXml” request handler.

**Receiving the data from client (Server-side)**

After receiving the data from the client, the server will route to trigger “/reqXml” request handler and it will run the process.

**“/reqXML” request handler**

## Request JSON

**Sending GET request (Client-side)**

After the form is checked and selected with the correct information of the user’s choice, the form will be sent using XMLHttpRequest to the server requesting “/reqJson” request handler.

**Receiving the data from client (Server-side)**

After receiving the data from the client, the server will route to trigger “/reqJson” request handler and it will run the process.

**“/reqJSON” request handler**

## Request Error

When a non-existing path is requested from the client the error request handler will be called and send a “404 error not found” message back to client to indicate no such path.

# Description of data structures

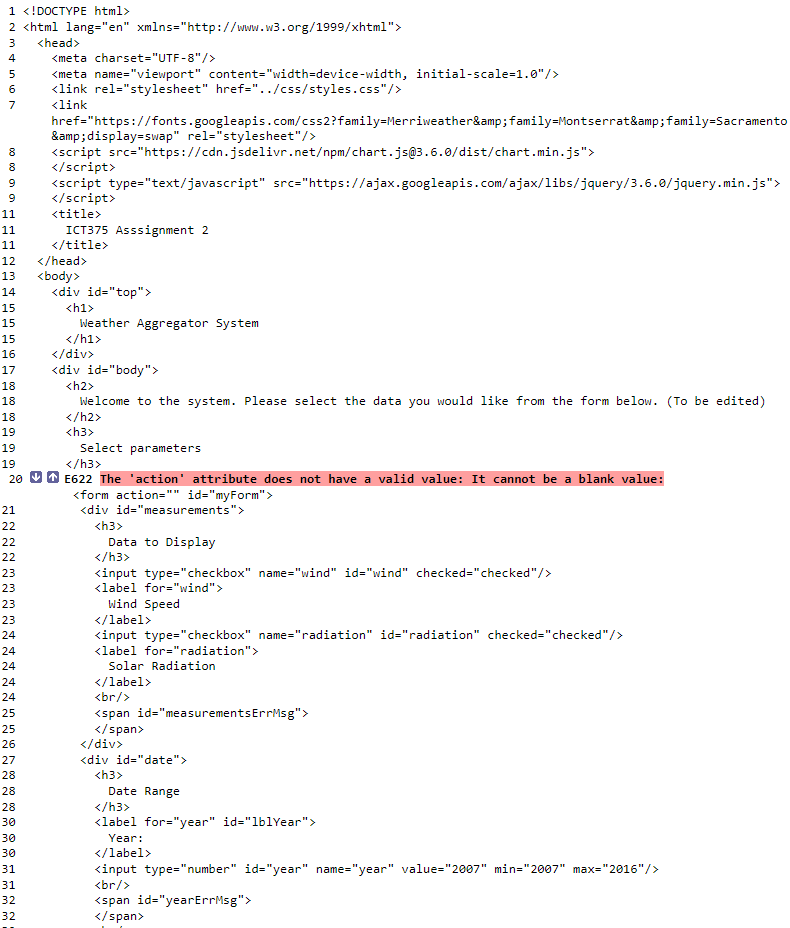
# Application test (Screenshots)

## HTML TotalValidator

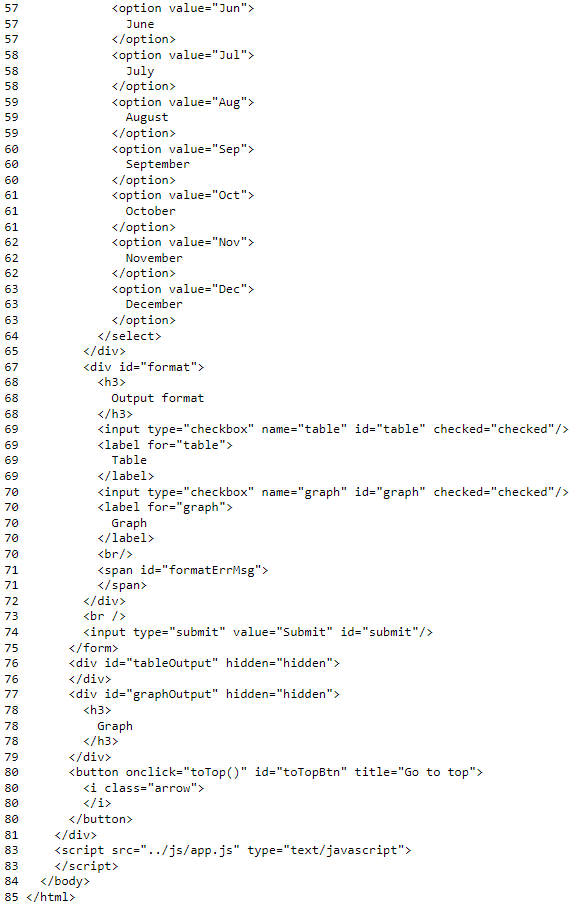
The result of HTML TotalValidator, there is an error found, but it is for the functionality of the website that causes this error: Graphical user interface, application, Teams

Description automatically generated

The only error of the HTML was the action was blank as the form needs to be validate before submitting it to the server, I have leave the action attribute blank to prevent it to submit to server first before validation and changed the action of the form in the JavaScript after validation is completed, just to make sure user entered the required fields in the form so that it is valid to submit to server:



A picture containing graphical user interface

Description automatically generated

## CSS validation

Graphical user interface, text, application, email

Description automatically generated

## Form validation

# Conclusion