**SPECIFICATION**

**2022-23 Trimester B - Internet of Things MMI126818-21-B (15 Credits)**

**Coursework 2**

[1 Coursework overview 2](#_Toc131765743)

[2 The sensor data 2](#_Toc131765744)

[3 Task 1 – Creating and deploying a Google Core IoT device for use with the sensor data 2](#_Toc131765745)

[4 Task 2 - Using the sensor data on Google Cloud 4](#_Toc131765746)

# Coursework overview

This coursework makes use of environmental data sourced from a set of two sensors located at a specific geographical location. The data is to be sent to Google Cloud, via a single IoT device representation.

This section of the specification provides an overview of what is to be delivered. Later sections detail the requirements.

The IoT configuration is to be as follows:

* Your own PC running Node-RED.
* A newly created Google Core IoT device representation for the two sensor devices.

The Google Cloud architecture includes the following:

* A Node-RED application on a VM that provides the following features:
  + Data analysis that utilises data stored in a Firebase database.
  + Visualisation of data.
  + A Google Map with a marker representing the sensor location.
    - Some data is to be made available in the marker InfoWindow.
  + A Slack messaging interface for the transmission of regular messages relating to environmental conditions.
* A Google Cloud function that writes data to a Firebase database.

# The sensor data

* The data is provided in a spreadsheet.
* In the spreadsheet each row of data represents sets of data for two sensors, with each set of sensor data given a timestamp in two formats:
  + A human readable format:
    - Date of the sensor readings.
    - Time of the sensor readings.
  + A Unix timestamp that matches the Date and Time data.
    - You will already be familiar with this as used within Node-RED.
* Node-RED code is provided to read from a spreadsheet and emit one set of sensor readings for each unit of time. In this way we can use the code to simulate individual sets of sensor readings, delivered over time.

# Task 1 – Creating and deploying a Google Core IoT device for use with the sensor data

Part (a) of this task represents a common challenge, interpreting the functionality of some new code that you are given.

* You are provided with the Node-RED code and a spreadsheet.
  + The spreadsheet represents a range of sensor data (two sensors) for specific dates/times.
* **The code should be deployed to your PC Node-RED.**
* The code requires the following Node-RED library to be installed first:
* node-red-contrib-spreadsheet-in
* Reference: <https://flows.nodered.org/node/node-red-contrib-spreadsheet-in>

The Node-RED import code and the spreadsheet are available at:   
<https://caledonianac-my.sharepoint.com/:f:/g/personal/pba4_gcu_ac_uk/EioDrlLHGTRIlBMy3i7sVt8B6eBqIeJ2Bx3wI0H0SpIM_w?e=dtuN6p>  
  
The code to read the spreadsheet is in the file: ‘IoT\_CW2\_IoT\_Device\_Code.txt’.

1. Explain the purpose of each part of the supplied code. Your description should start with the first node and follow through in-sequence to the final node. You can refer to the ‘Help’ information provided for each node-type.

**Note: Use narrative and screenshots as required. Ensure that you discuss all node configurations.**

|  |
| --- |
|  |

1. Within the Google Cloud Console create:
   1. A new Google Core IoT registry.
   2. Within that registry create a new Google Core IoT device representation. You should create a relevant name for the device representation. You should include Pub/Sub topics for Telemetry and State.
   3. Pub/Sub subscriptions for the Telemetry topic and the State topic.

To evidence this, provide screenshots.

* Each screenshot must contain the browser URL bar and contain the full width of the browser window.
* The following screenshots must be shown:
  + ‘Registry Details’, including the configured topics.
  + The registry entry for the new device that you have created.

**Note: Use narrative and screenshots as required. Ensure that you discuss all aspects of the configuration, including security.**

|  |
| --- |
| ‘Registry Details’ screenshot:  Screenshot of the ‘DETAILS’ tab of the registry entry for the new device:  Screenshot of the ‘AUTHENTICATION’ tab of the registry entry for the new device, demonstrating that a key is present:  Screenshot evidencing the Pub/Sub subscription for Telemetry:  Screenshot evidencing the Pub/Sub subscription for State: |

1. Describe how you set up this new Google Core IoT device on your PC Node-RED implementation for sending both Telemetry and State messages. Discuss all configurations, including authentication.   
   Show a screenshot that demonstrates that the device is connected.

**Note: Use narrative and screenshots as required. Ensure that you discuss all aspects of the configuration, including security.**

|  |
| --- |
| Narrative and screenshots evidencing the configurations: |

1. Examine the spreadsheet from start to finish. You will see that for some rows in the spreadsheet, the sensor entries have values that are unusable.   
   Extend the original code so that it ignores the rows with unusable values, rather than emitting these values at the end of the flow.  
   Provide evidence of this extension to the code, as follows:

|  |
| --- |
| A screenshot showing how you have extended the flow: |

|  |
| --- |
| Configuration of any nodes that you use, and/or content of any function that you create: |

1. On your PC Node-RED, join the output of the existing sensor code (‘IoT\_CW2\_IoT\_Device\_Code.txt’) to both Telemetry and State nodes, to send sensor messages to Google Core IoT. These messages will be identical for Telemetry and State. Provide evidence that the messages have been received on Google Core IoT. You should see the messages within each of the subscriptions with the *Cloud Console pages for the Pub/Sub Service*. Once you are viewing a particular subscription (Telemetry or State), use the ‘Messages’ tab to view the incoming messages.

|  |
| --- |
| Screenshot evidencing reception of Telemetry messages in relevant Pub/Sub subscription: |

|  |
| --- |
| Screenshot evidencing reception of State messages in relevant Pub/Sub subscription: |

|  |
| --- |
| A complete screenshot of the IoT device flow: |

# Task 2 - Using the sensor data on Google Cloud

1. There is a requirement to view the sensor data on Google Cloud Node-RED. You should already have deployed code to view data on Dashboard charts, within the Week 7 laboratory Dashboard example.

Create a Cloud Node-RED flow that includes the following:

* Subscribes to the **IoT device Telemetry messages only**. Do **not** subscribe to State messages in your Node-RED application. The Telemetry data must come from the IoT device (on your PC), via the Google Core IoT service.
* Provides a Node-RED dashboard with two charts (one for each sensor) that contain the data arriving from the IoT device.
  + The charts should be correctly labelled and should each show the last 100 data points received.
  + You can ignore the timestamp received from the Telemetry messages.
* **Each screenshot must contain the browser URL bar and contain the full width of the browser window. Screenshots that do not deliver this will not be marked.**

|  |
| --- |
| A screenshot showing the working Node-RED flow for all aspects for the Cloud Node-RED implementation. You should take this screenshot once you have completed all tasks of the Cloud Node-RED section of the coursework: |

|  |
| --- |
| Screenshot evidencing the reception of any two Telemetry messages from relevant Pub/Sub subscription, displayed in the Node-RED debug panel: |

|  |
| --- |
| Screenshot evidencing the Node-RED dashboard with two charts, over the last 100 items received: |

|  |
| --- |
| Source code of any function node(s) to process the Telemetry messages and prepares these for display on the charts: |

|  |
| --- |
| Screenshots of configurations of any other nodes that are used within the flow: |

1. Provide a Google Maps web page within the Cloud Node-RED application. On that page you should provide a marker at the following location:  
   Latitude: 55.94532323435053, Longitude: -3.1843848117904385  
   Note: this location is: Edinburgh Nicolson Street.

Within that marker you should provide an InfoWindow that contains the latest sensor data from the two sensors, clearly labelled.  
  
The evidence required is:

|  |
| --- |
| A copy of the Node-RED function code that sets up the request to the Google Maps subflow: |

|  |
| --- |
| Screenshot showing the map with marker and active InfoWindow: |

1. One of the laboratory sessions deals with creating a Cloud Function to store IoT data into a Firestore database, on a particular database ‘Collection’. Based on that exercise you should do the following:

* *Create a new Cloud Function* that is triggered by messages from the Pub/Sub ‘State’ topic that you have created (Note: we are **not** triggering from the Telemetry messages).
* This Cloud Function should save IoT State messages that arrive. This Cloud Function code is very similar to the functionality of the previous Cloud Function within the lab session, and you will need a new database *collection*. Please ensure that you store the entire contents of the payload of the received message.
* Send a set of ‘State’ messages from the IoT device to be processed by the Cloud Function and stored in the database collection. These will be used for subsequent database queries.
  + - The messages should have dates between:
      * Start date/time: 01/01/2022 01:00:00 (this is the first data item in the spreadsheet)
      * End date/time: 21/01/2022 20:00:00

Provide evidence as described below.  
[**Note: Each screenshot must contain the browser URL bar and contain the full width of the browser window. Screenshots that do not deliver this will not be marked**]:

Evidence: Provide a screenshot of Google Cloud Log that shows the execution of the Cloud function, accessed from the three-dot menu on the Cloud Function entry on page: <https://console.cloud.google.com/functions/list>  
You should specifically show within the Log output the IoT message object content that will create the document in the database.  
Please ensure that you scroll the log entry to show all content, and include evidence that the Cloud Function has actually executed without error.

|  |
| --- |
| Screenshot of the Cloud Function ‘Source’ tab: |

|  |
| --- |
| Screenshot of the Cloud Function ‘Trigger’ tab: |

Evidence: From the *Firebase Console*, provide two screenshots, showing the contents of two different Collection entries that have been created from the Cloud Function being triggered by IoT ‘State’ data.

|  |
| --- |
| Collection entry example 1: |

|  |
| --- |
| Collection entry example 2: |

1. A set of analytic operations is required within the Cloud Node-RED application, to deliver insights into the received sensor data. The interface to these operations must be on the same Node-RED dashboard as delivered for Section 4 Part (a).

The specific requirements are:

* Delivery of a set of Node-RED dashboard buttons that trigger relevant operations to deliver the following analysis, as dashboard text outputs. The analytical operations are listed below.   
  **Note:** The analysis must be based upon database queries, followed by any other analytical operations (if required).
* There should be a separate *dashboard button* and *dashboard output* for each of the following:
  + The highest value of nitric\_oxide within the entire set of stored data.
  + The highest value of nitrogen\_dioxide between two dates.
    - The dates should be selected on the dashboard using date-selector code.
      * This code is available in the import file : ‘Node-RED\_dates.txt’ supplied in the folder linked on Page 2 of this document.
  + The most recent value of nitric\_oxide.
    - This value should also be sent in a message to a Slack channel. The message should describe what data is being sent.
  + The median value of nitrogen\_dioxide between two dates.
    - Use the date-selector code to set the date.
    - Use an appropriate node that will yield the ‘median’ statistic.

The following evidence is required:

|  |
| --- |
| The function code that implements the database query for the highest value of nitric\_oxide within the entire set of stored data:    An example of debug output that provides evidence that you have received the correct result for the database query: |

|  |
| --- |
| Function code that implements the database query for the highest value of nitrogen\_dioxide between two dates:  An example of debug output that provides evidence that you have received the correct result for the database query: |

|  |
| --- |
| Function code that implements the database query for the most recent value of nitric\_oxide:  An example of debug output that provides evidence that you have received the correct result for the database query:  Code/node-configurations for sending a Slack message:  A screenshot of the received Slack message, showing the entire Slack window, with the same sensor data as the debug example shown in the first part of your response: |

|  |
| --- |
| The median value of nitrogen\_dioxide between two dates. Provide an example of debug output that provides evidence that you have received the correct result for the database query part of this problem:  Any code/node-configuration(s) that you use to yield the ‘median’ outcome: |

|  |
| --- |
| The complete export of your Node-RED **device** flow, Courier font, font size 8 |
| REPLACE THIS with device code. |

|  |
| --- |
| The complete export of your **Cloud** Node-RED flow, Courier font, font size 8 |
| REPLACE THIS with Cloud code. |