

**INTERNET OF THINGS LAB**

**BECE351E**

**TASK – 6**

*NODE-RED PLATFORM FLOWS*

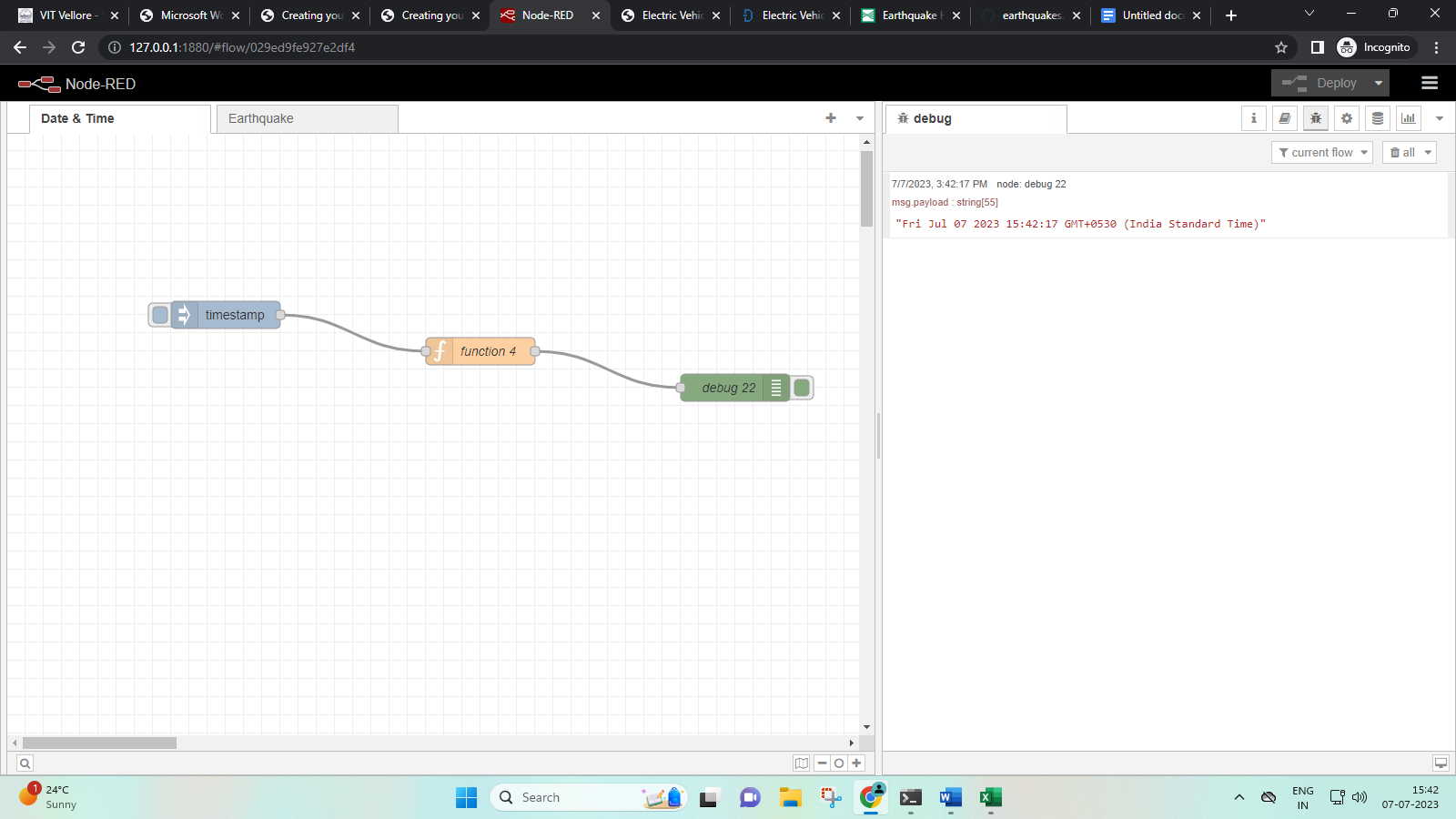
**NAME:** TARAKSHRAJ SINGH SHEKHAWAT

**REG NO:** 21BEC2150

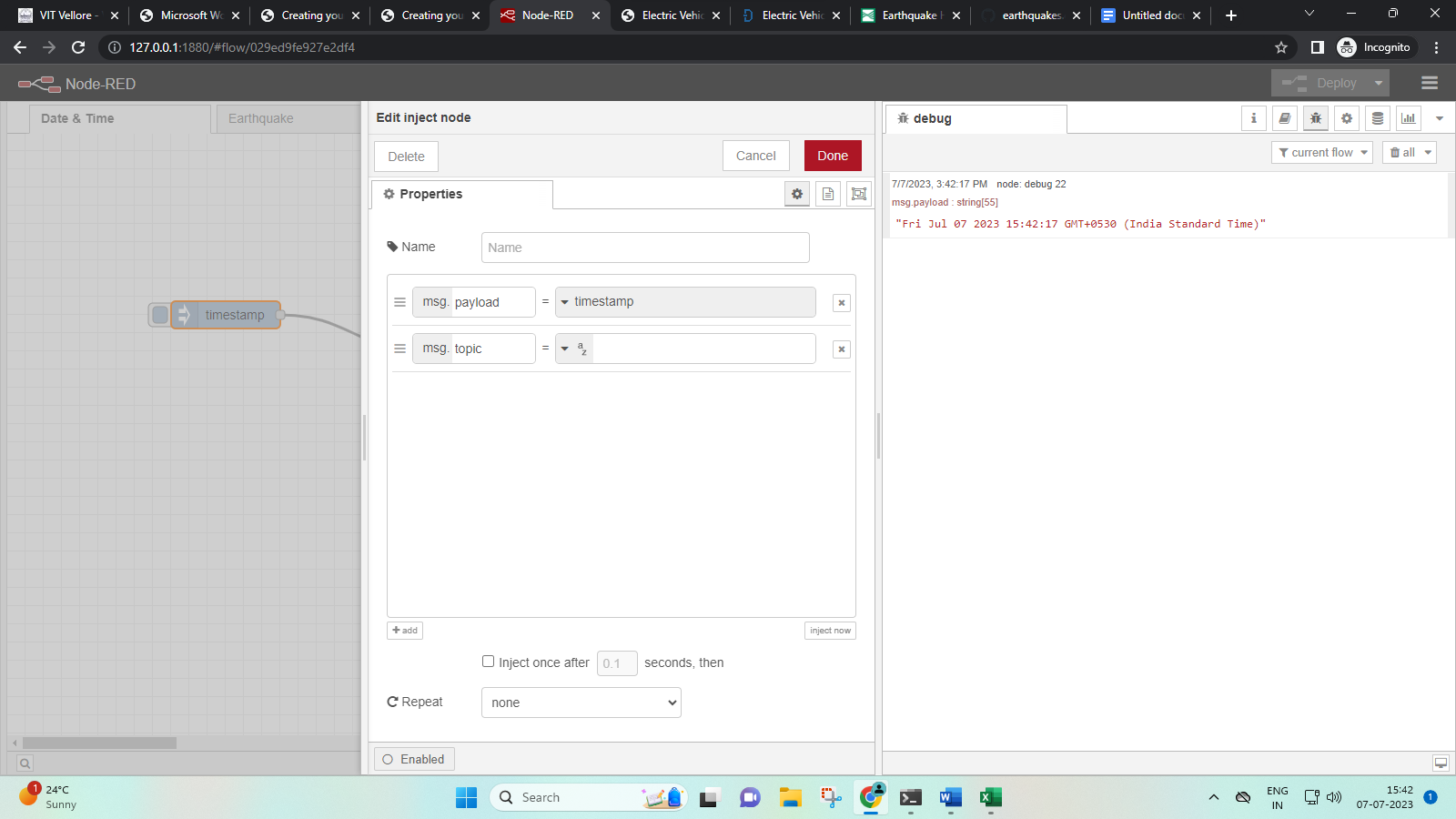
**LAB SLOT:** L55+L56

**FACULTY:** MA’AM GERARDINE I. MARY

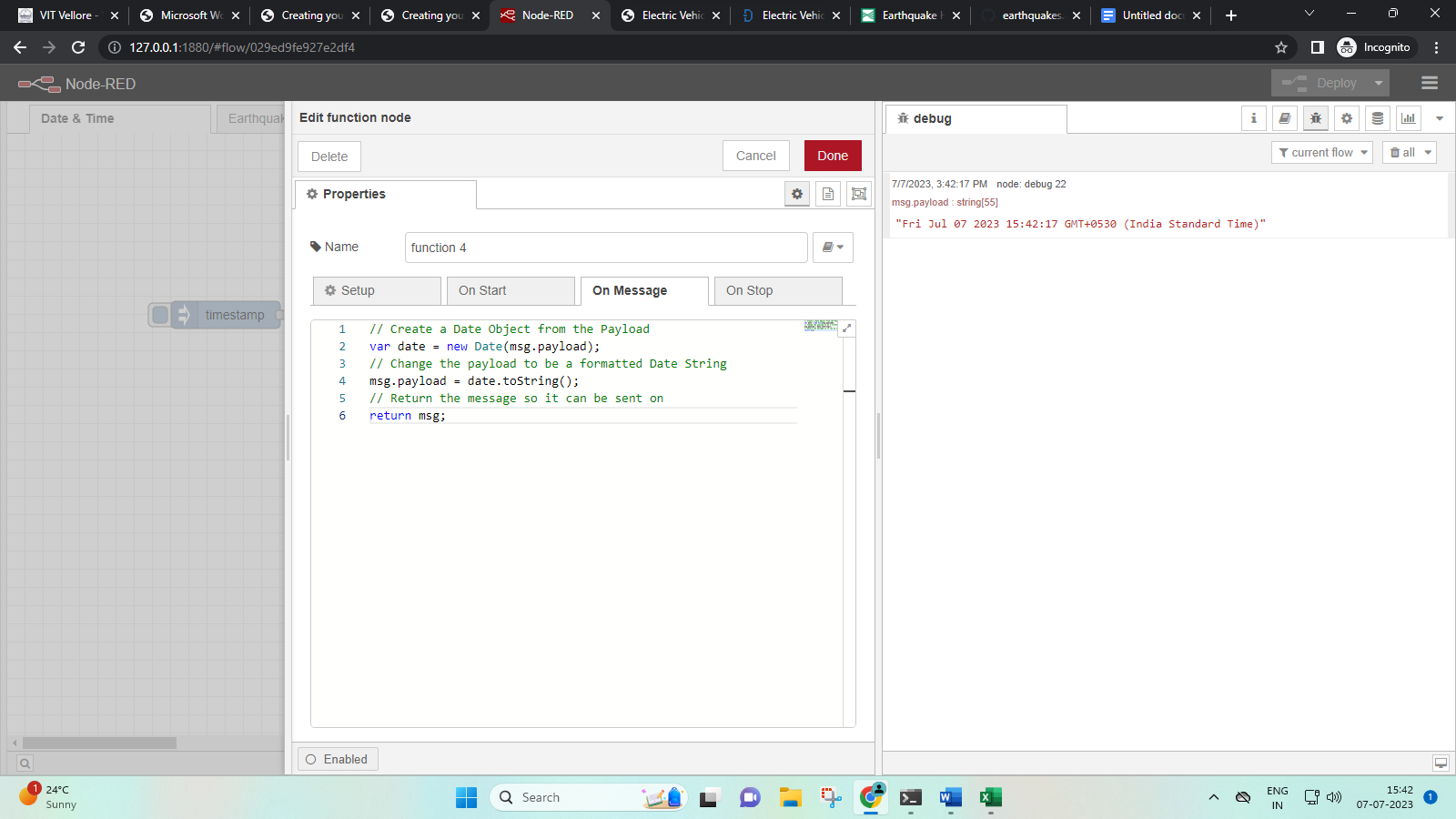
**FLOW 1:**



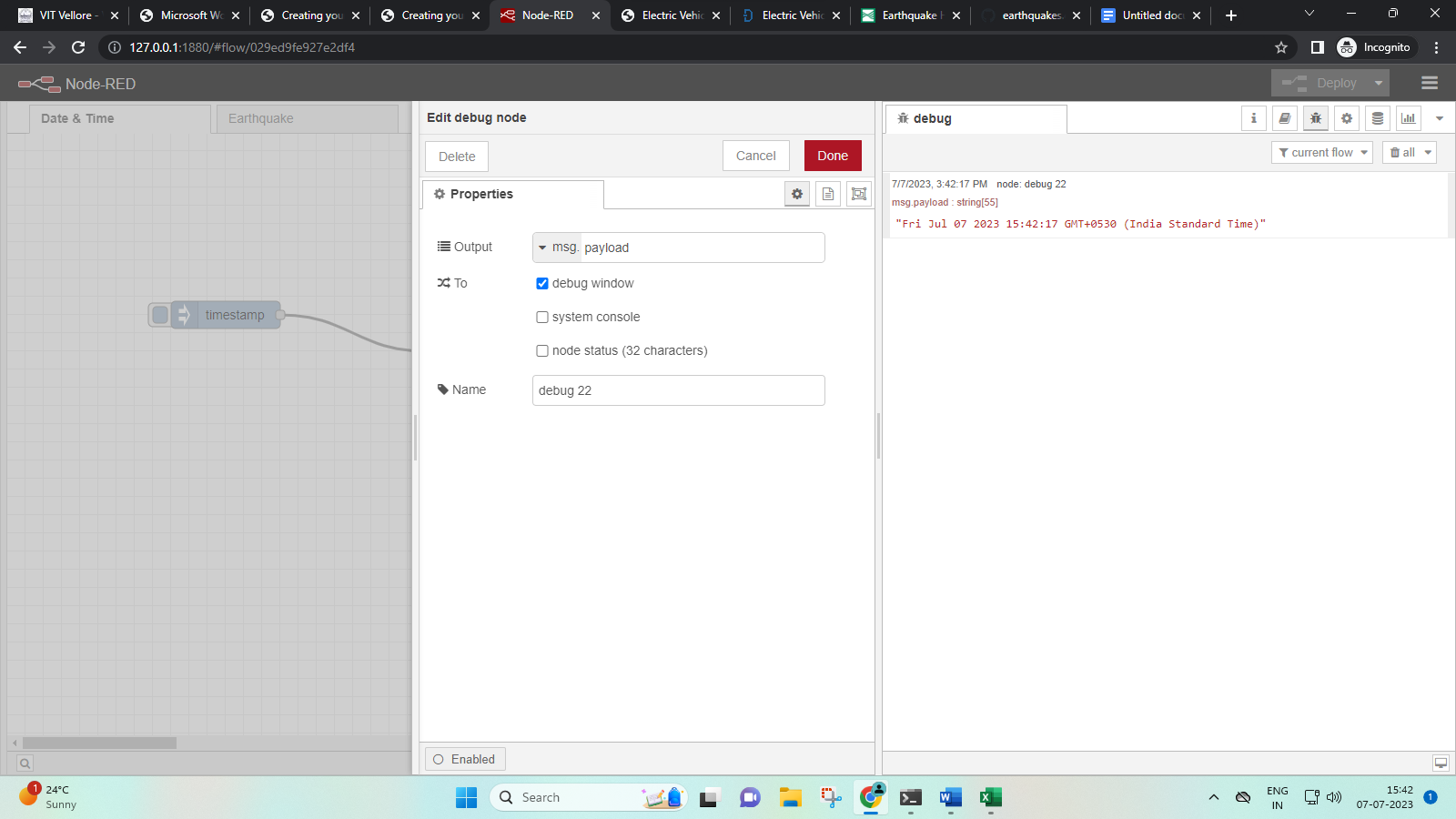
Inject Node Settings:



Function Node Settings:



Debug Node Settings:



Flow Explanation:

The inject node is set to send a payload once everytime the user injects once, after a delay of 0.1 seconds.

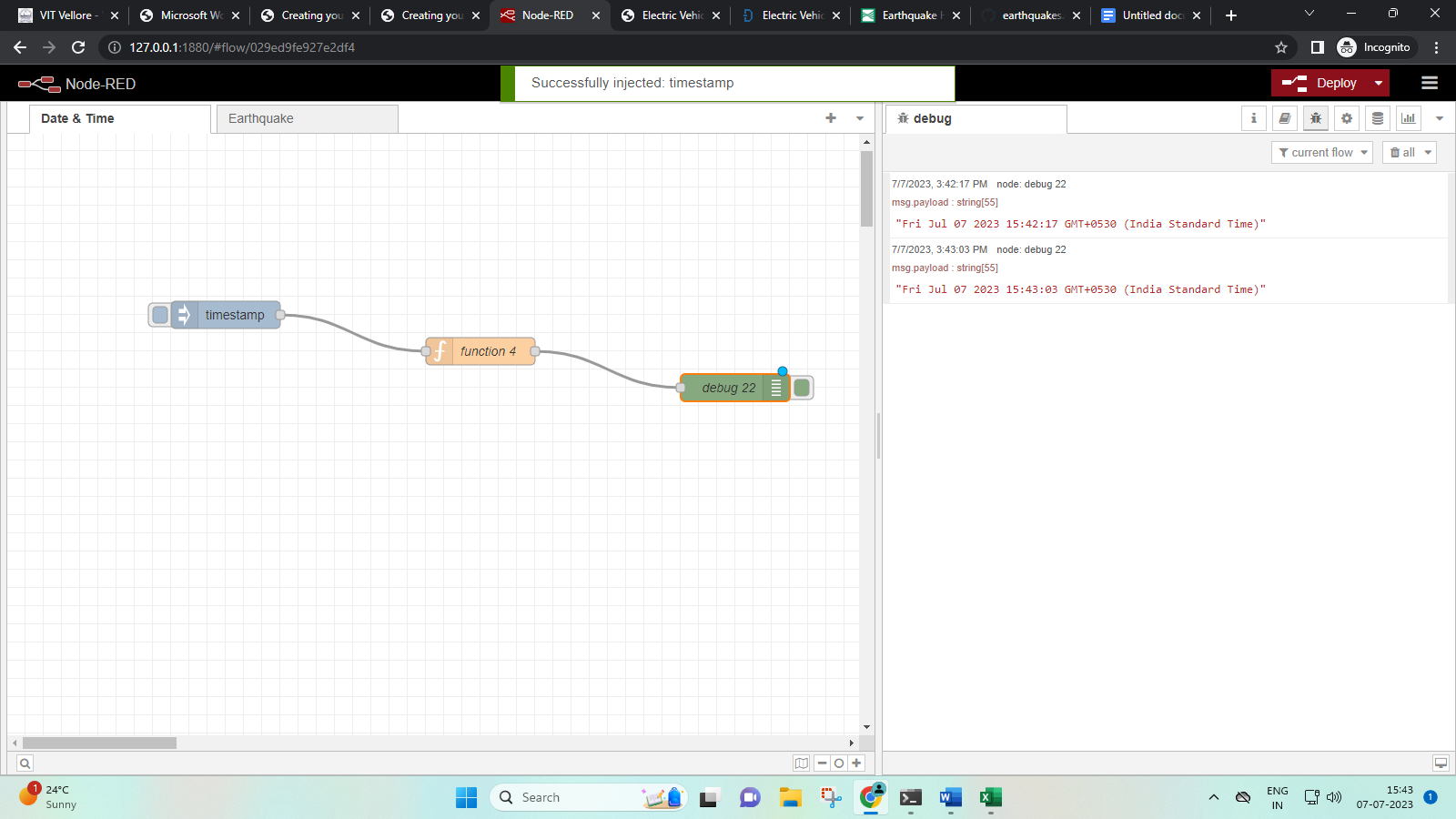
The function node is set to display day, date and time through a string of code.

The debug node is set to receive the message payload and print it on the debug window.

After clicking deploy, we can start testing out flow.

Once we inject a payload through the inject node, we should get the day, date, and time printed on the debug window after the respective delay.

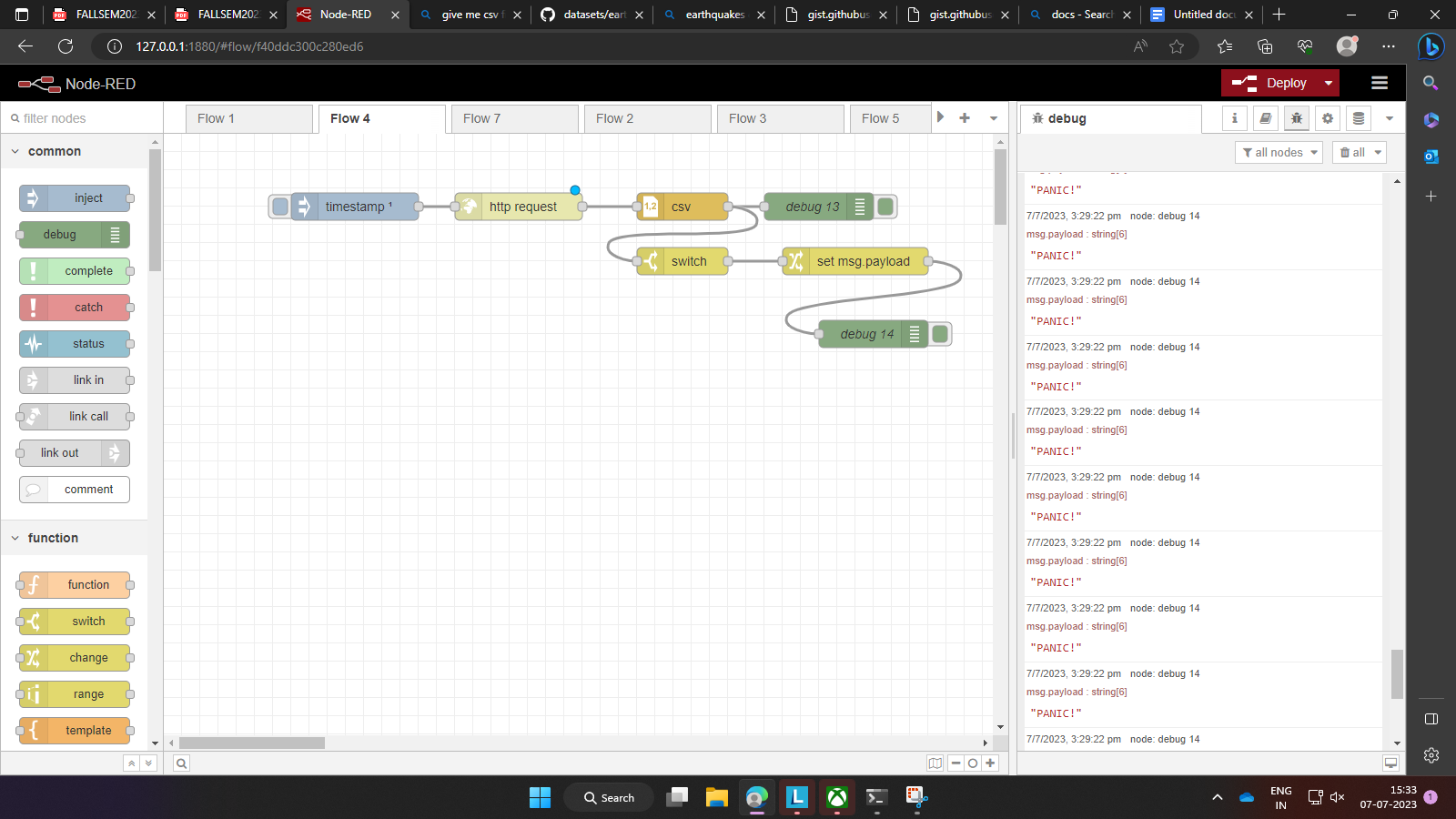
Output:



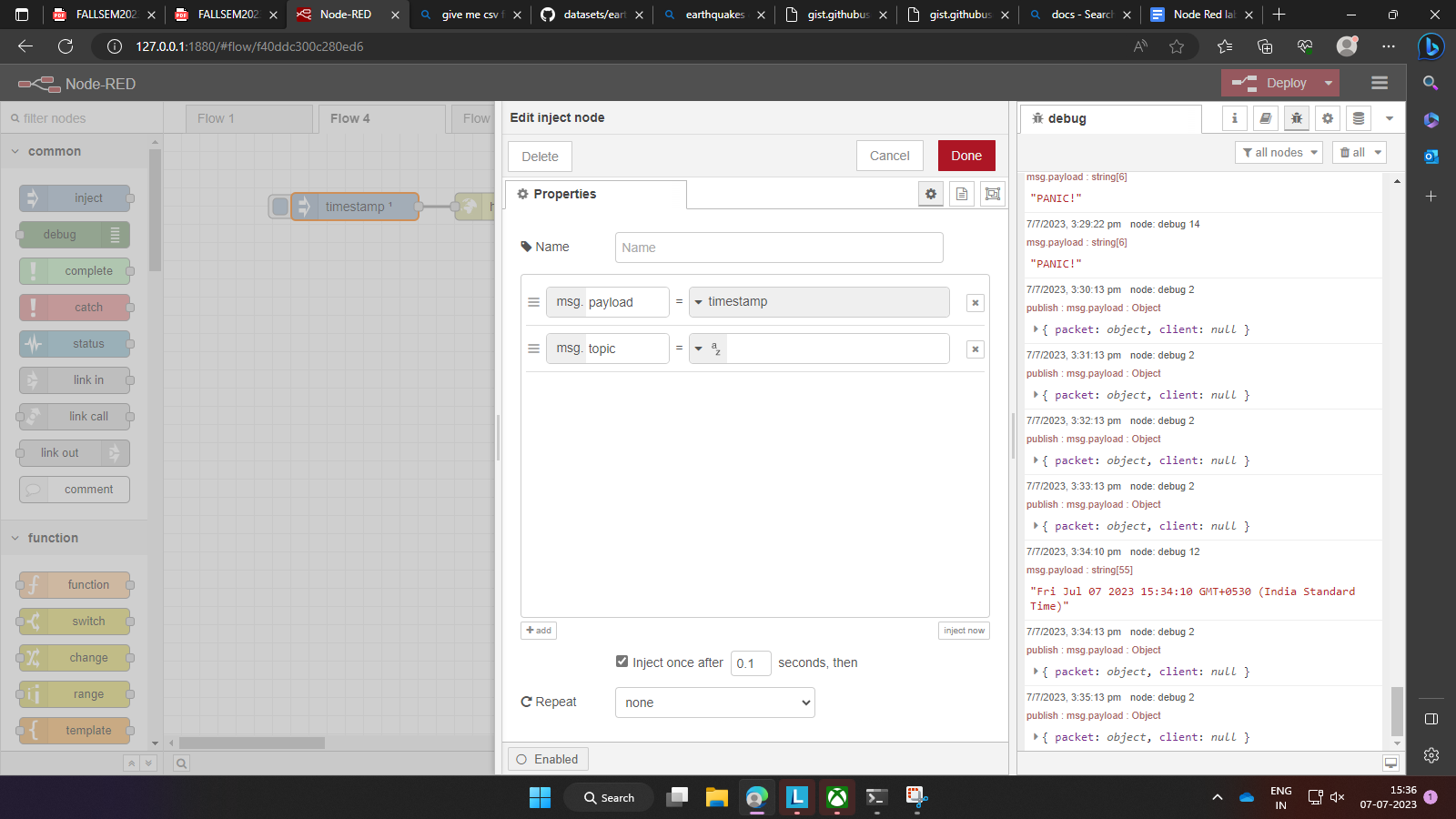
We will get the day, date and time printed once for each time we inject the payload according to our chosen settings for the inject node.

(PTO)

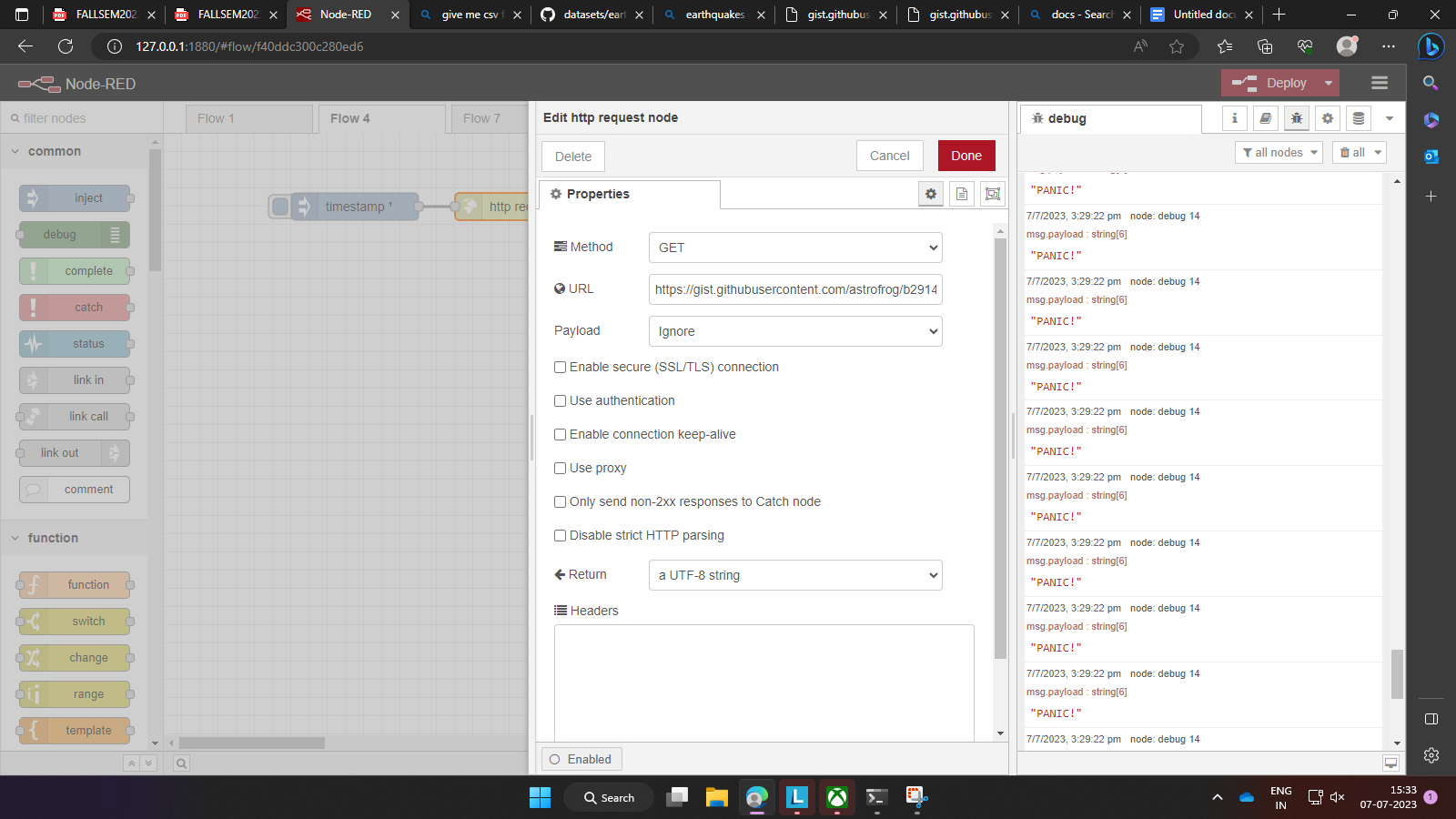
**FLOW 2:**



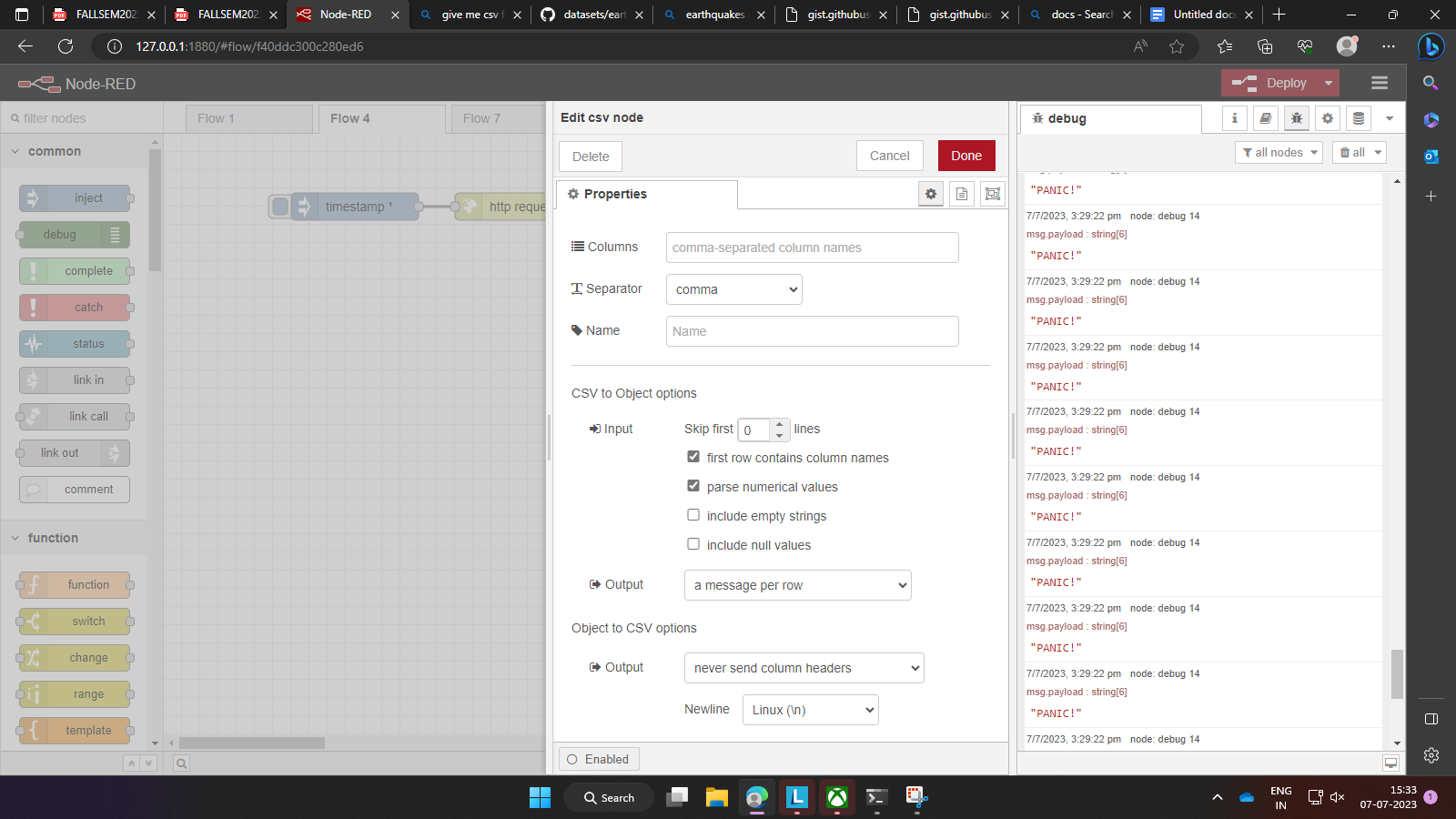
Inject Node Settings:



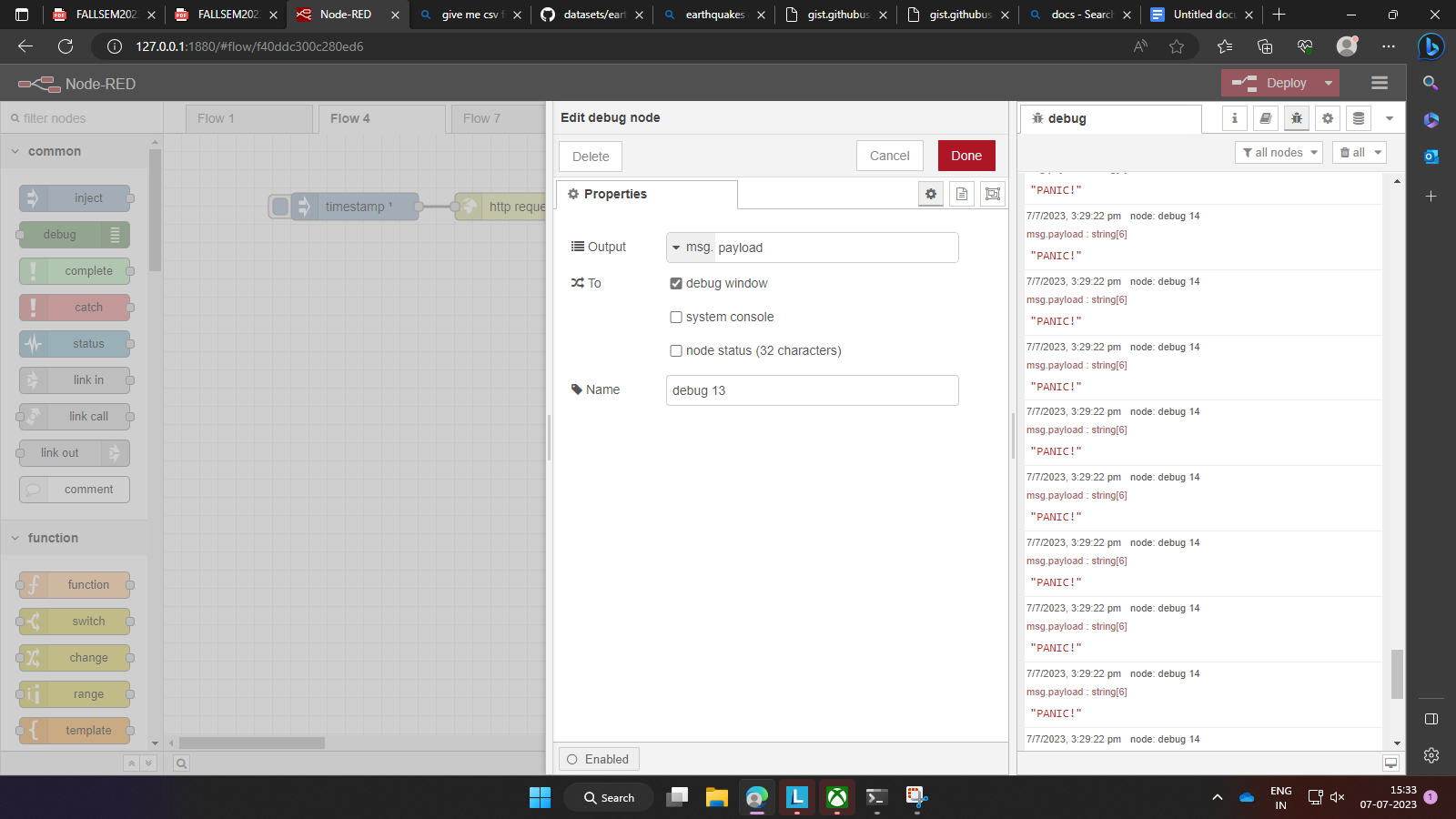
HTTP Request Node Settings:



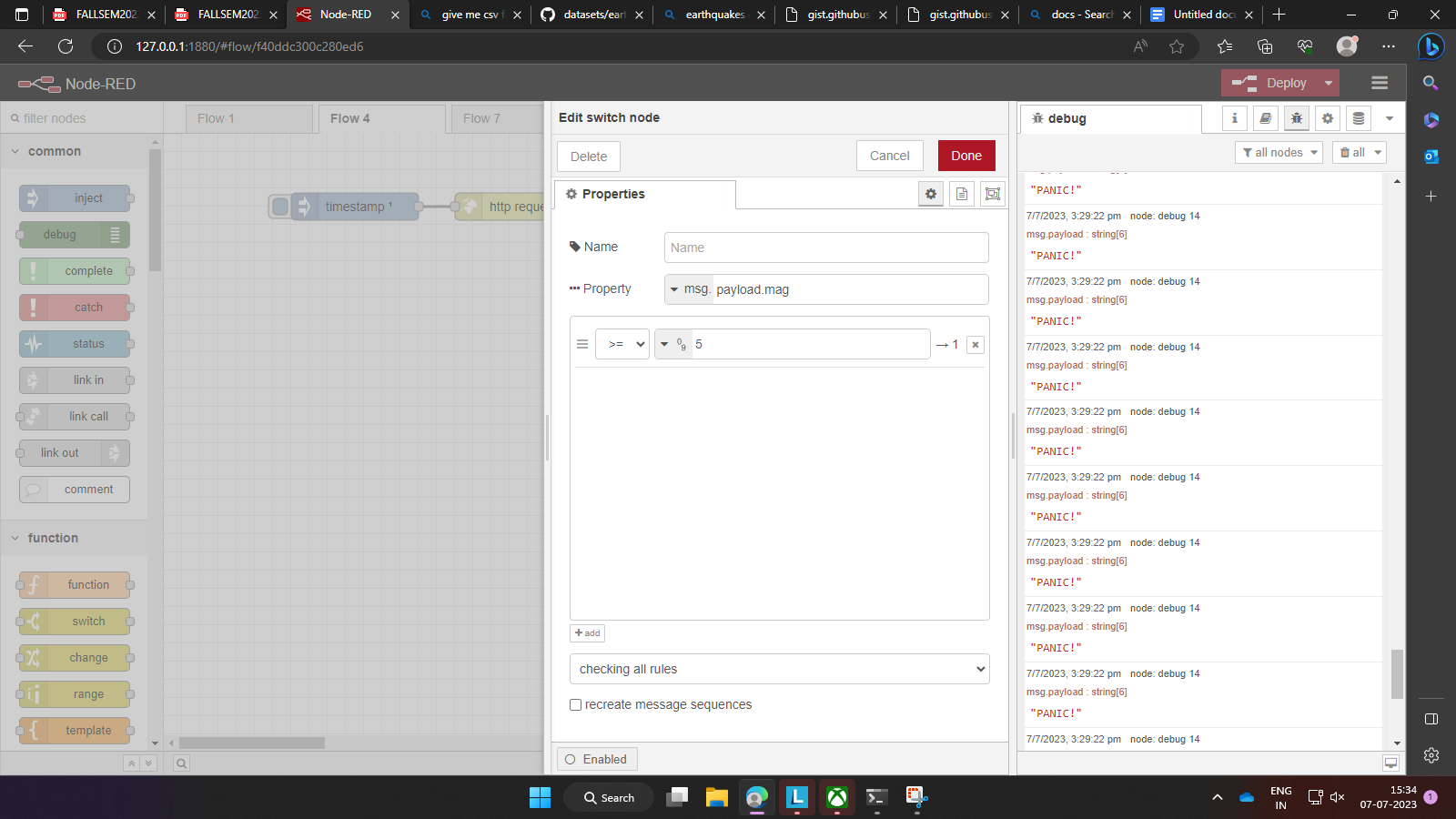
CSV Node Settings:



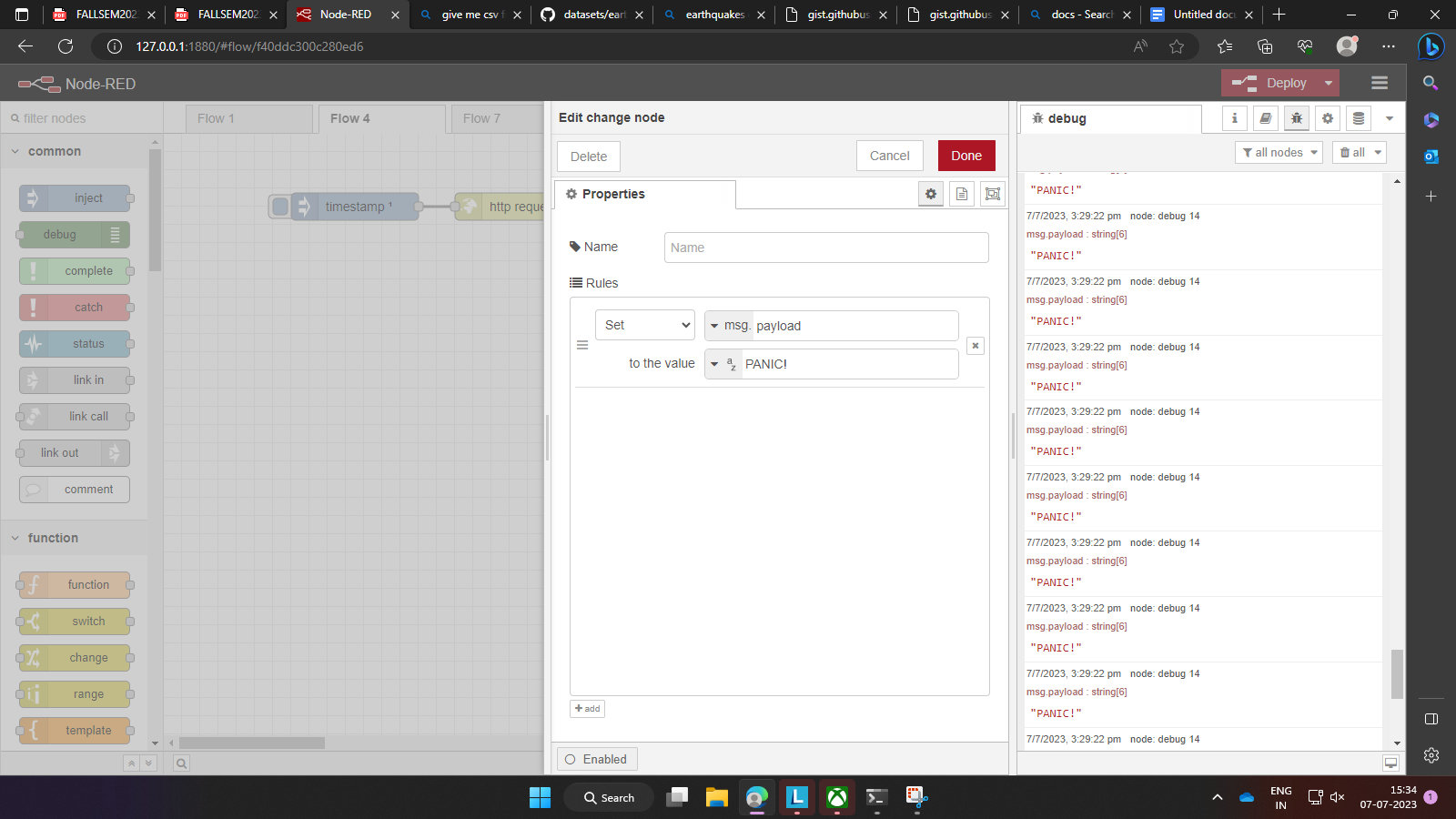
First Debug Node Settings (Debug 13):



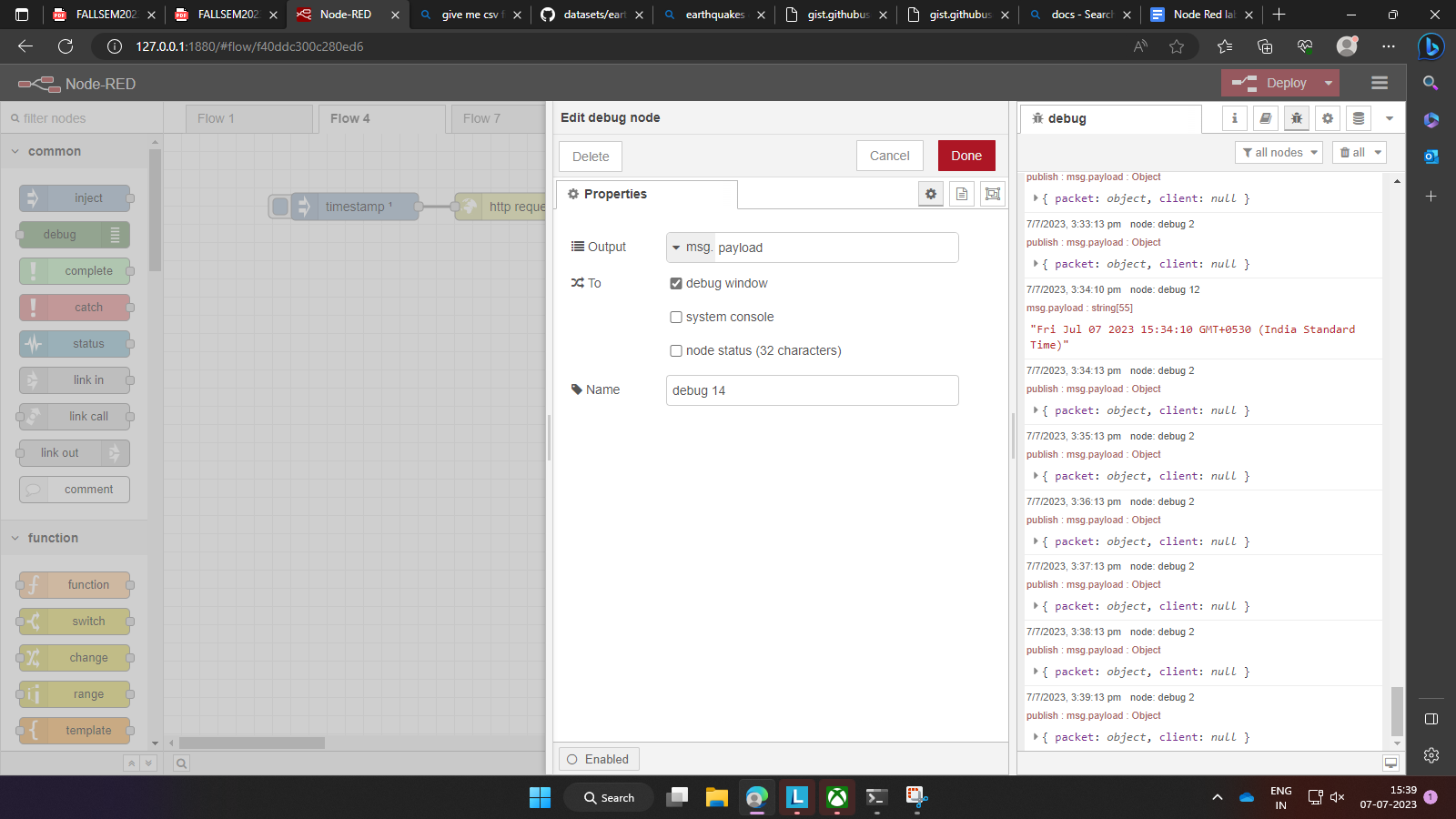
Switch Node Settings:



Change Node Settings:



Second Debug Node Settings:



Flow Explanation:

The **inject node** is set to trigger a payload transmission once for each time you inject it (after first deploying the flow).

Once you inject, the **http request node** will get data from the link that we have attached in its URL section. In our case, it is a csv file of a data table of earthquake records from GITHUB.

This is then sent to the **csv node** that is set to read the csv and parse numerical values (the most notable ones being the magnitudes of the earthquakes from the data table).

This is then sent two ways.

The first **debug node (debug 13)** will simply print all the values that are sent by the csv node (parsed in packets) on the debug window.

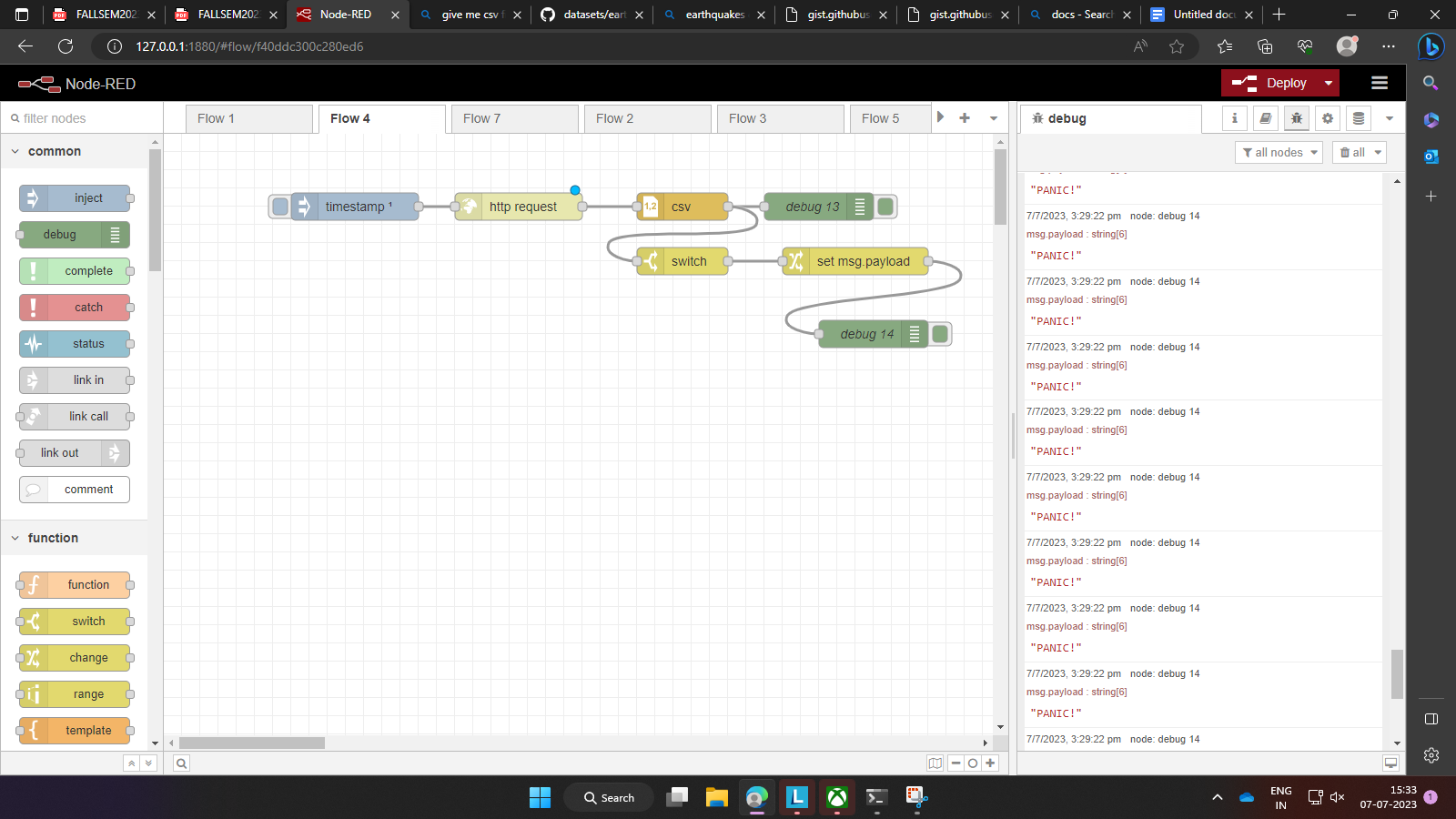
The **switch node** will receive the parsed values from csv node, then look at magnitudes of each ones (since we have set it as such) and then select sort which ones are equal to or greater than 5 (according to our settings).

This is then forwarded to the **change node** which will then print a message for each value that the switch node has sorted as equal to or greater than 5. In our case, the message is ‘PANIC!’.

Finally, the second **debug node (node 14)** will print ‘PANIC!’ for each time the magnitude value is equal to or greater than 5.

**In conclusion,** this flow will retrieve data from a csv of earthquake records, print the data on the debug window, then pick out the magnitudes and compare them to a specific threshold (in our case, 5) and for each time the magnitude is equal to or greater than the threshold, it’ll transmit ‘PANIC!’ which will then be printed on the debug window.

Output:



CSV File Used:

<https://gist.githubusercontent.com/Felienne/00d73c032f4ad14d701aecac4c0245fa/raw/0de7cff62e96a0ead4b586274479f7713be76833/earthquakes.csv>