

PROJECT DEVELOPMENT – DELIVERY OF SPRINT 2

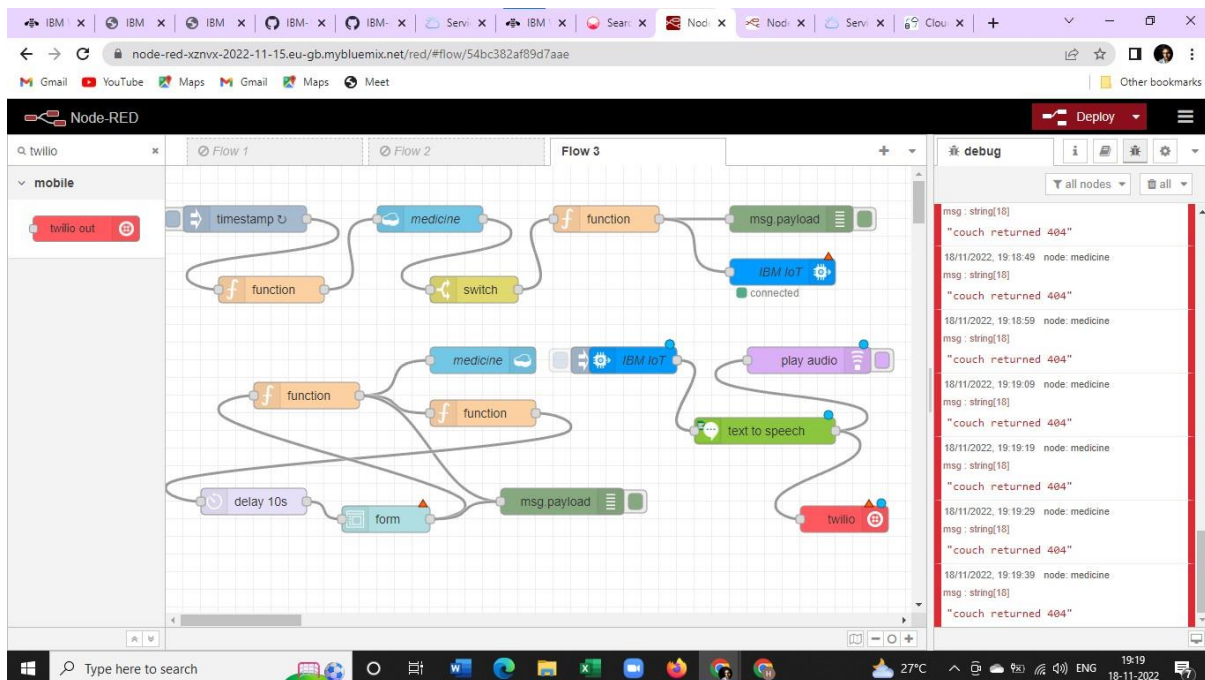
(CREATING A WEB APPLICATION USING NODE-RED)

TEAM ID	PNT2022TMID26620
PROJECT NAME	Personal Assistance for Seniors Who Are Self-Reliant

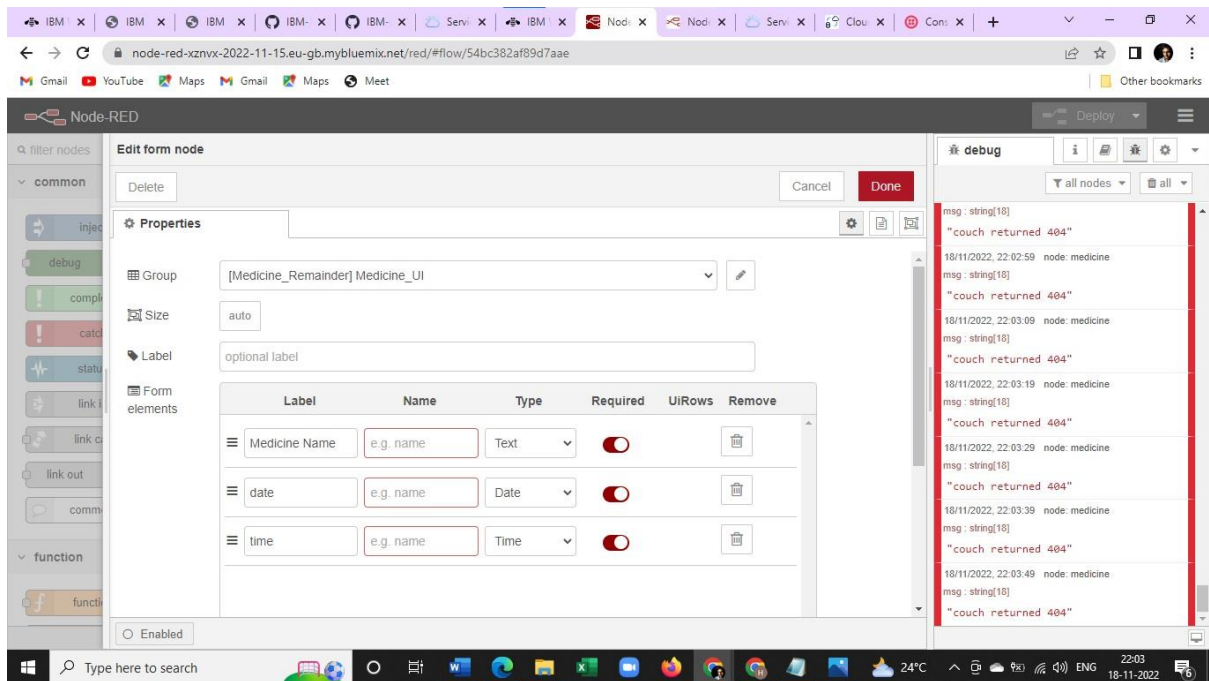
AIM : TO DESIGN A NODE-RED TO DEVELOP A WEB APPLICATION

STEPS TO BE FOLLOWED :

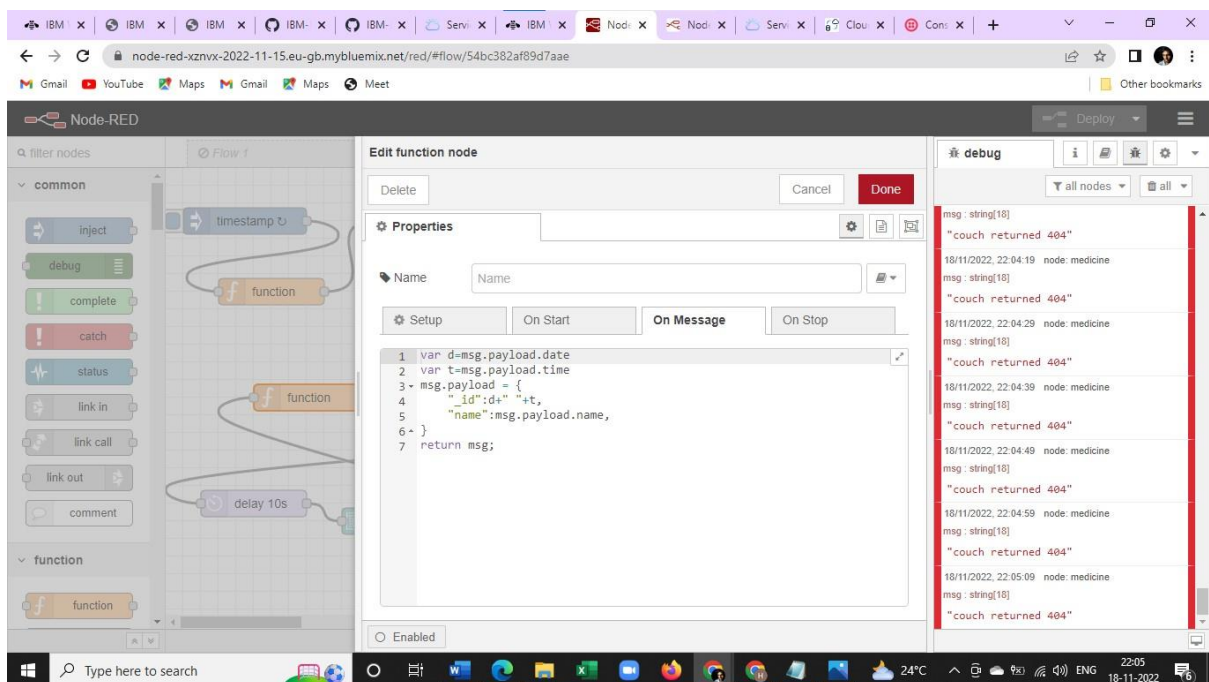
STEP 1 : TO DESIGN A NODE-RED WITH REQUIRED COMPONENTS (CLOUDANT,IOT DEVICE,FORM,FUNCTIONS,INPUT,OUTPUT,SWITCH).



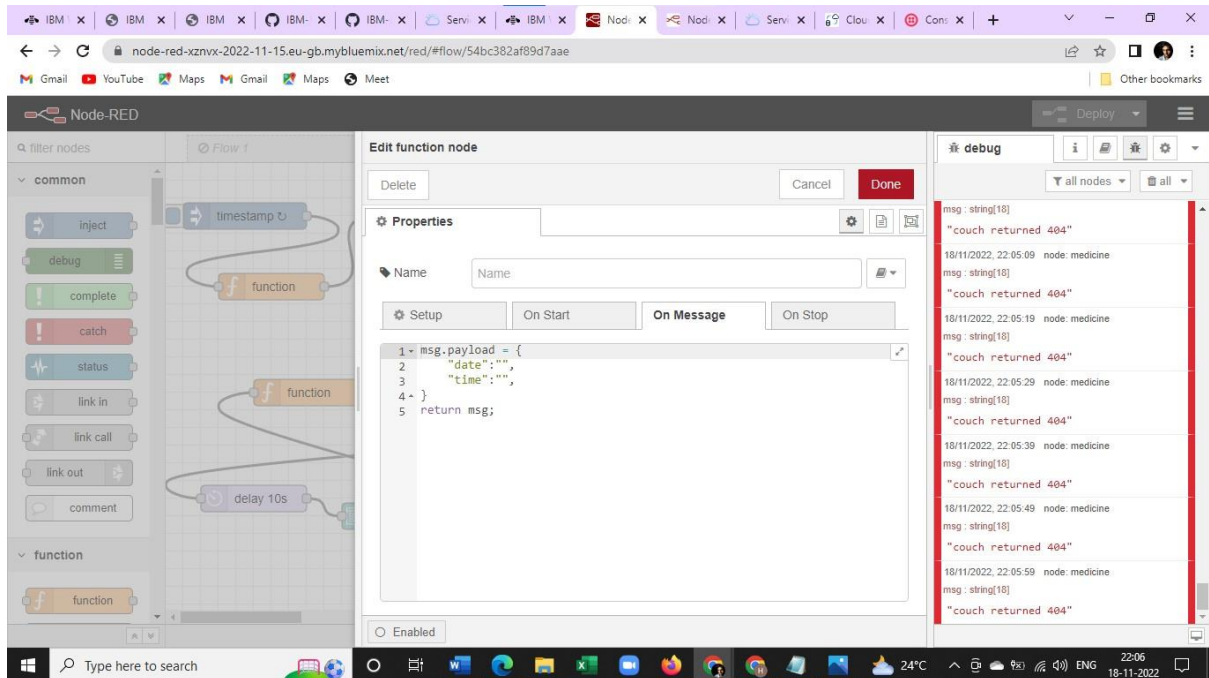
STEP 2 : IN CLOUDANT NODE ADD DATABASE,NAME,OPERATION.



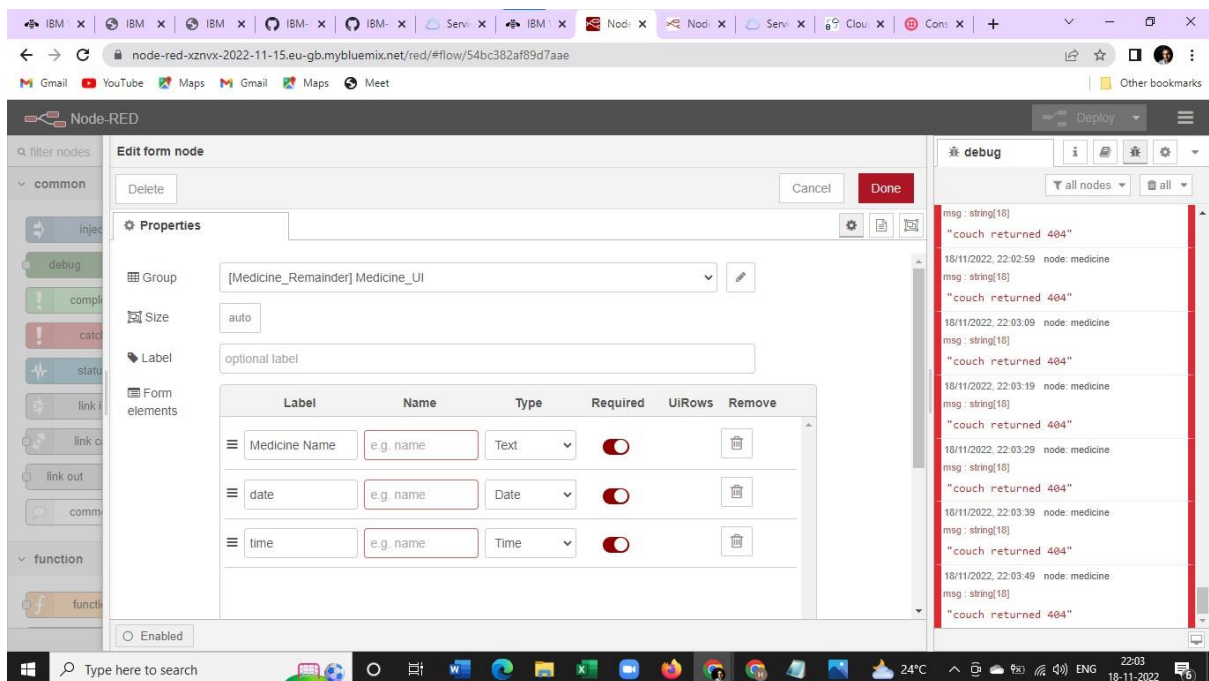
STEP 3 : ADD THE PROGRAM FOR TIME AND DATE IN FUNCTION NODE



STEP 4 : ADD THE PROGRAM FOR PAYLOAD IN THE FUNCTION NODE.



STEP 5 : FILL THE DETAILS FORMAT IN FORM NODE



STEP 6 : INJECT THE TIMESTRAMP IN INJECT NODE

The screenshot shows the Node-RED web interface. The main workspace contains a flow with an 'inject' node, a 'timestamp' node, and a 'function' node. The 'inject' node is selected, and the 'Edit inject node' dialog is open. The dialog has a 'Properties' section with 'Name' set to 'Name'. The 'msg.payload' is set to 'timestamp' and 'msg.topic' is set to 'a_z'. The 'Inject once after' checkbox is checked, with a value of '0.1' seconds. The 'Repeat' section is set to 'Interval' with a value of '5' minutes. The 'Enabled' checkbox is checked. The 'debug' console on the right shows a series of messages: 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:06:29 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:06:39 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:06:49 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:06:59 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:09 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:19 node: medicine', 'msg: string[18]', '"couch returned 404"'. The bottom status bar shows the time as 22:07 on 18-11-2022.

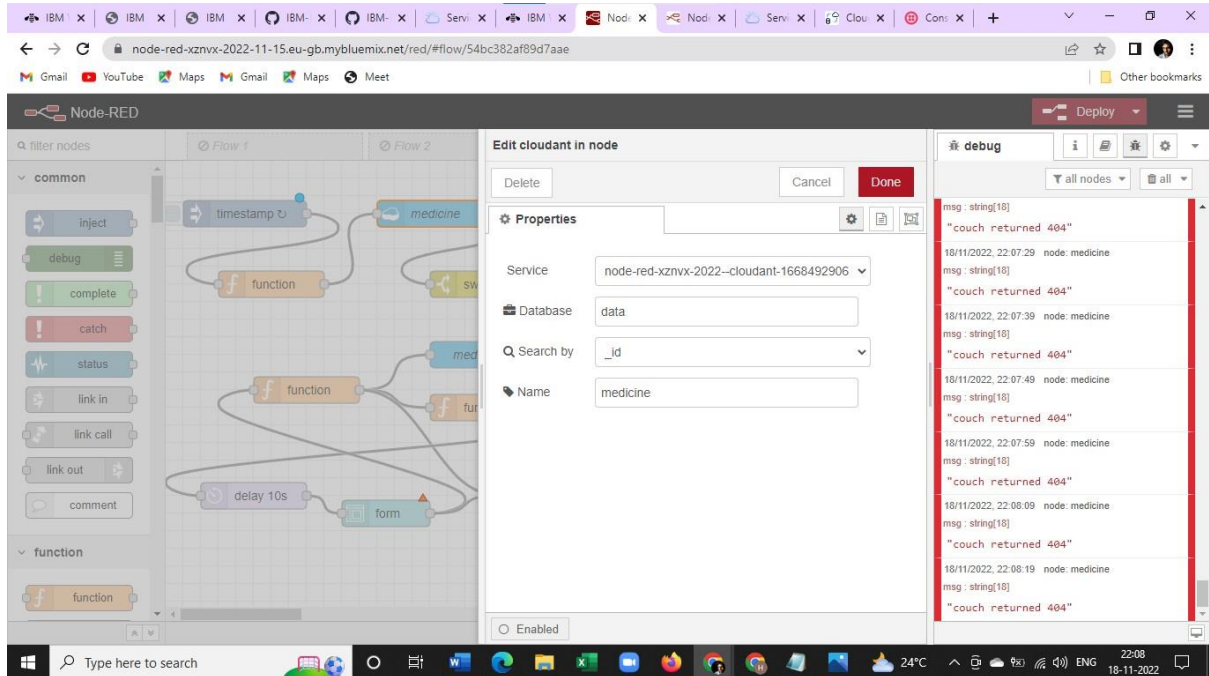
STEP 7 : ADD THE PROGRAM FOR COMPARE THE TIME IN THE FUNCTION NODE

The screenshot shows the Node-RED web interface. The main workspace contains a flow with an 'inject' node, a 'timestamp' node, and a 'function' node. The 'function' node is selected, and the 'Edit function node' dialog is open. The dialog has a 'Properties' section with 'Name' set to 'Name'. The 'Setup' tab is selected. The code in the 'Code' section is as follows:

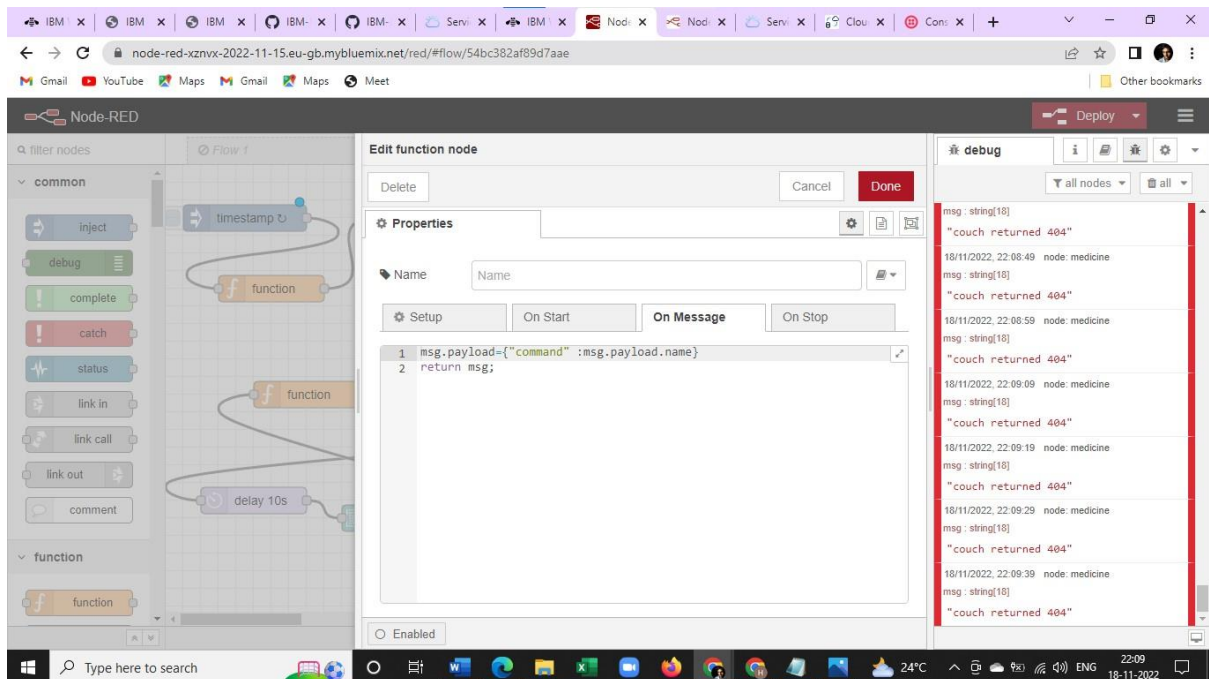
```
1 var d = new Date();
2 var utc = d.getTime() + (d.getTimezoneOffset() * 60000);
3 var offset = 5.5;
4 var newDate = new Date(utc + (3600000*offset));
5 var n=newDate.toISOString()
6 var date = n.slice(0,10)
7 var time = n.slice(11,16)
8 global.set('time',time)
9 msg.payload=date+" "+time
10 return msg;
```

The 'Enabled' checkbox is checked. The 'debug' console on the right shows a series of messages: 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:06:59 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:09 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:19 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:29 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:39 node: medicine', 'msg: string[18]', '"couch returned 404"', '18/11/2022, 22:07:49 node: medicine', 'msg: string[18]', '"couch returned 404"'. The bottom status bar shows the time as 22:07 on 18-11-2022.

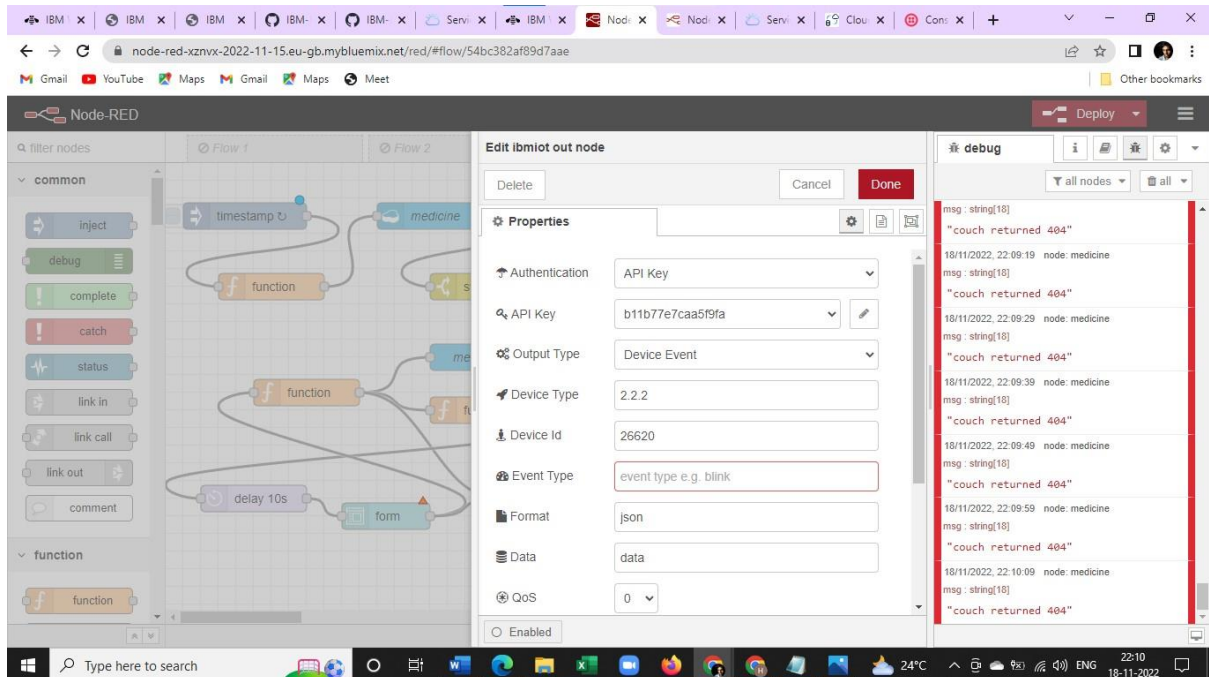
STEP 8 : ADD THE DATABASE,NAME,SEARCH BY INSTRUCTIONS IN CLOUDANT IN NODE



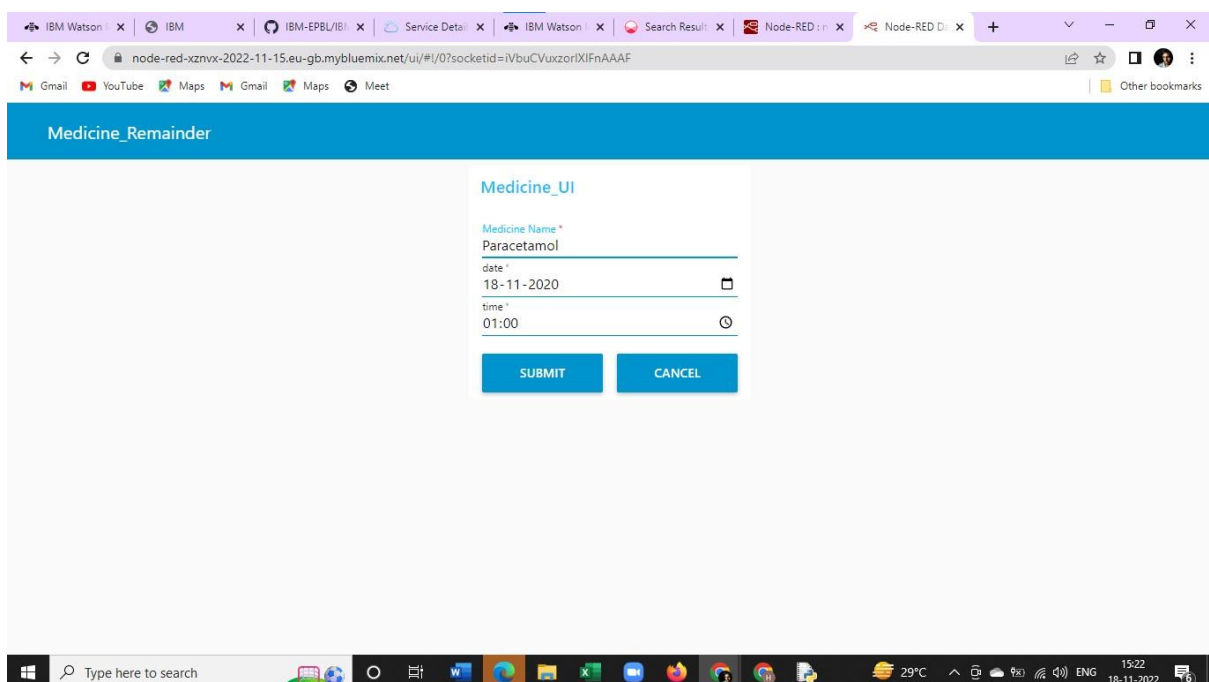
STEP 9 : ADD THE COMMAND FOR PAYLOAD IN FUNCTION NODE



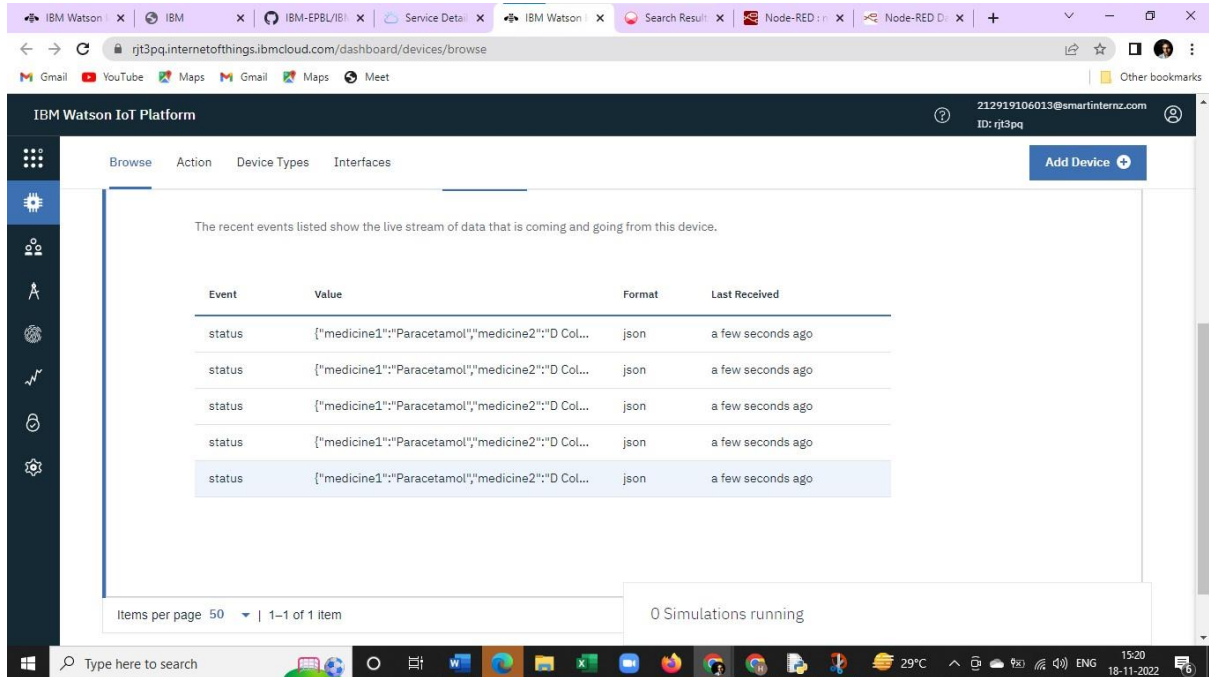
STEP 10 : ADD THE API KEY, DEVICE TYPE, DEVICE ID , AUTHENTICATION TOKEN IN IBM IOT NODE



STEP 11 : COPY THE NODE-RED URL AND BROWSE IT WITH ADDING /ui/ ALONG WITH THAT LINK. YOUR WEB APPLICATION WILL OPEN AND ADD THE DATA YOU WANT THE SERVICE .



STEP 12 : NOW ADDED DATA WILL BE STORED IN THE IBM IOT WATSON PLATFORM'S DEVICE SUCCESSFULLY



The screenshot displays the IBM Watson IoT Platform interface. The top navigation bar includes tabs for 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains icons for various IoT functions. The main content area shows a table of recent events for a device. The table has four columns: 'Event', 'Value', 'Format', and 'Last Received'. The events are listed as 'status' with a JSON value containing 'medicine1' and 'medicine2' data. The format is 'json' and the last received time is 'a few seconds ago'. At the bottom, there is a status bar indicating '0 Simulations running' and a search bar.

Event	Value	Format	Last Received
status	{"medicine1":"Paracetamol","medicine2":"D Col..."}	json	a few seconds ago
status	{"medicine1":"Paracetamol","medicine2":"D Col..."}	json	a few seconds ago
status	{"medicine1":"Paracetamol","medicine2":"D Col..."}	json	a few seconds ago
status	{"medicine1":"Paracetamol","medicine2":"D Col..."}	json	a few seconds ago
status	{"medicine1":"Paracetamol","medicine2":"D Col..."}	json	a few seconds ago

RESULT : DATA WILL BE STORED IN IOT DEVICE BY NODE-RED WEB APPLICATION SUCCESSFULLY