

## PROJECT DEVELOPMENT PHASE - SPRINT 2

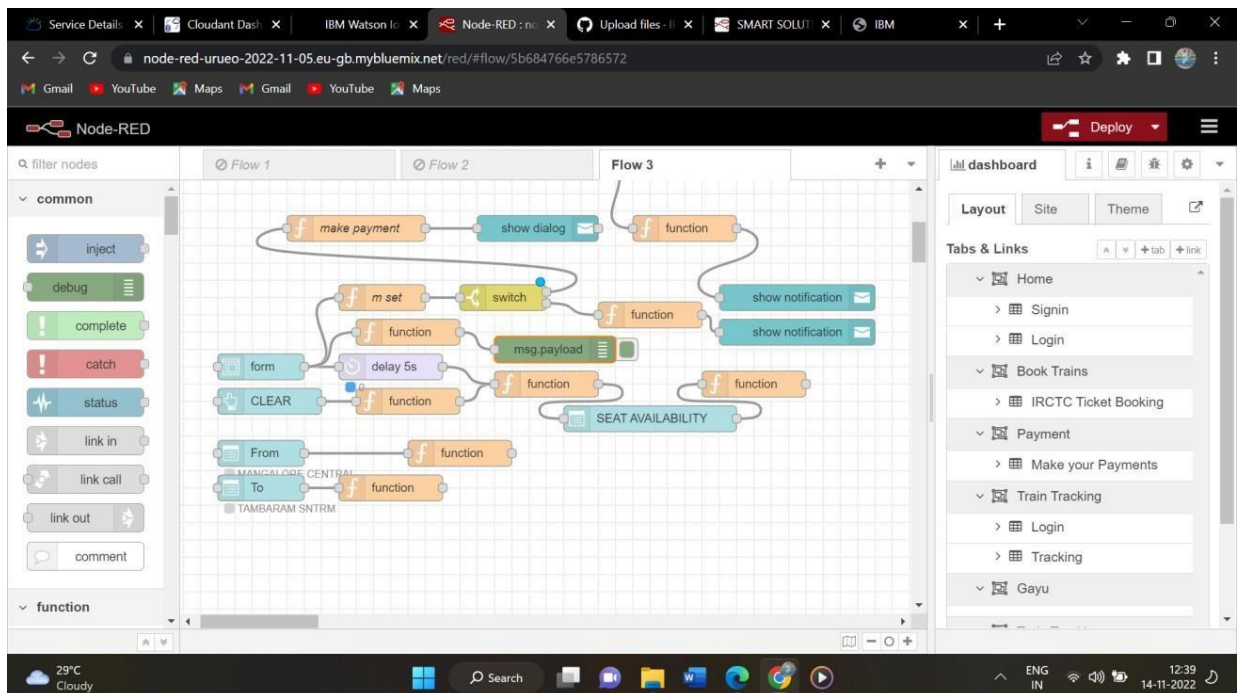
Date	8NOVEMBER 202
Team ID	PNT2022TMID45586
Project Name	Smart Solutions for railways

### SPRINT 2 – Train Booking and Payments

After logging into the website, the user should enter the Travelling details (Boarding and destination status). The user must also enter their personal details. After providing the required details, they are directed to the payment tab.

In the payment section, the user should enter their card details such as NAME, CARD NUMBER and CVV. Once they complete the payment, the ticket will be generated. The user will receive an unique identification (QR code) which will be used for ticket verification.

### NODE-RED FLOW FOR TICKET BOOKING



## NODE-RED FLOW FOR TICKET BOOKING

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow with a function node being edited. The left sidebar contains a palette of nodes, including 'common' (inject, debug, complete, catch, status, link in, link call, link out, comment) and 'function'. The right sidebar shows a dashboard with tabs and links for a ticket booking system, including Home, Signin, Login, Book Trains, IRCTC Ticket Booking, Payment, Make your Payments, Train Tracking, and Gayu.

**Edit function node**

**Properties**

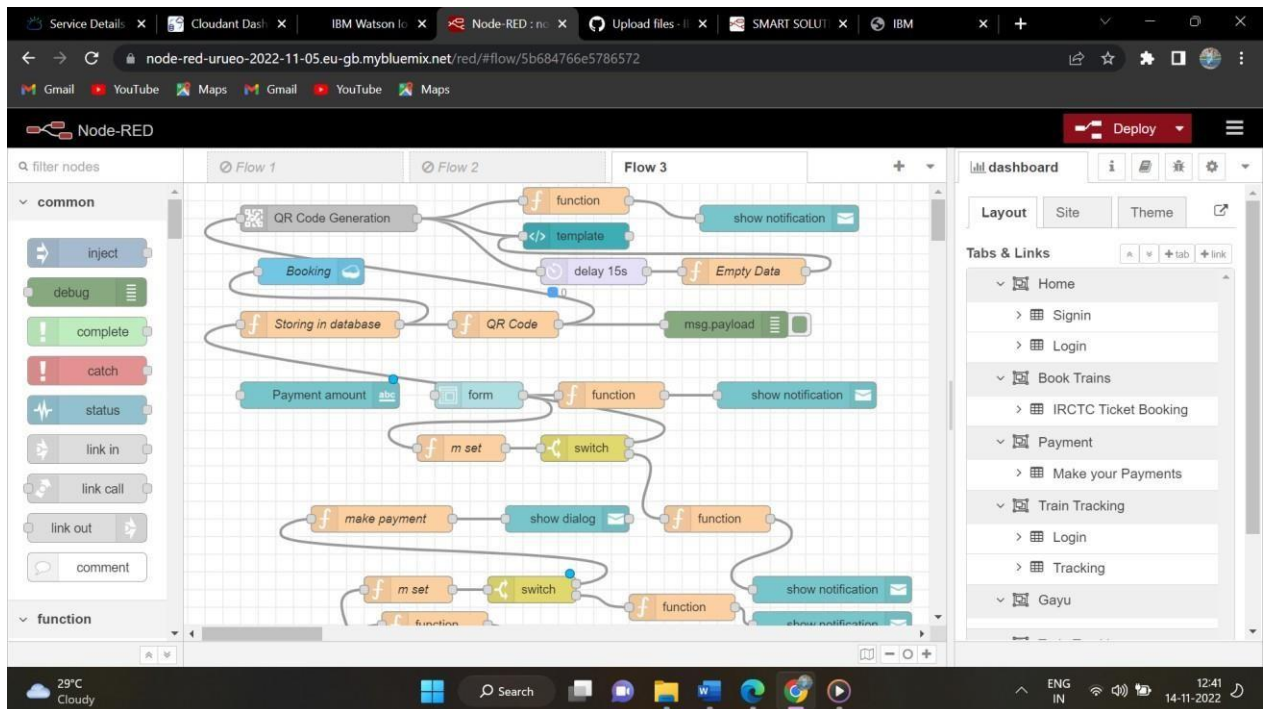
Name:

Setup | On Start | **On Message** | On Stop

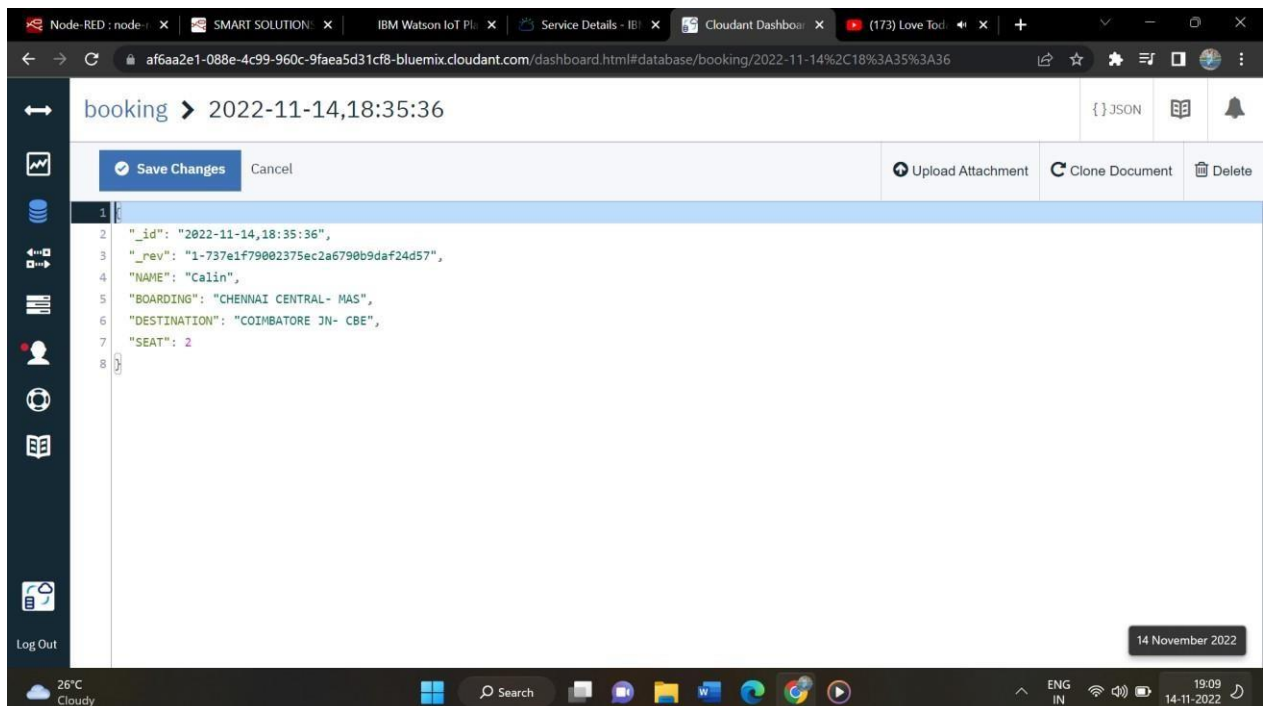
```
1 var a = global.get('a')
2 var s = []
3 for (let i=0; i<a.length; i++){
4   s.push(a[i])
5 }
6 if(s.length==0){
7   msg.options=[{"NO SEATS AVAILABLE":0}]
8 }
9 else{
10  msg.option=s
11 }
12 msg.payload = s
13 return msg;
```

☐ Enabled

## NODE-RED FLOW FOR PAYMENT



## IBM CLOUDANT SIGNIN DATABASE



## PYTHON CODE FOR GPS:

```

import
wiotp.sdk.device
import time
import
random
myConfi
g = {
    "identity": {
        "orgId": " s91n0t",
        "typeId": "
        MyDeviceType",
        "deviceId": " 12345"
    },
    "auth": {
        "token": " @)fRE3fdiTS!MaT3F_"
    }
}

def myCommandCallback (cmd):

    print ("Message received from IBM IoT Platform: %s" %
    cmd.data['command'])m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig,
logHandlers=None)client.connect()

def pub (data):

```

```
client.publishEvent(eventId="status", msgFormat="json",
data=myData, qos=0,onPublish=None) print ("Published
data Successfully: %s", myData)
```

```
while True:
```

```
    myData={'name': 'Train1', 'lat':13.08363 ,
'lon': 80.27080}pub (myData)
    time.sleep (2)
    myData={'name': 'Train2', 'lat': 12.40797,
'lon': 79.81410}pub (myData)
    time.sleep (2)
```

```
    myData={'name': 'Train1', 'lat': 11.83331,
'lon': 79.37465}pub(myData)
    time.sleep(6)
```

```
    myData={'name': 'Train1', 'lat': 11.59664,
'lon': 78.69899}pub (myData)
    time.sleep (6)
```

```
    myData={'name': 'Train1', 'lat': 11.63431,
'lon': 78.11122}pub (myData)
    time.sleep (6)
```

```
    myData={'name': 'Train1', 'lat': 11.32207,
'lon': 77.61684}pub (myData)
    time.sleep (6)
```

```
    myData={'name': 'Train1', 'lat': 11.03107,
'lon': 76.96864}pub (myData)
    time.sleep (6)
```

```
    client.commandCallback =
myCommandCallbackclient.disconnect ()
```