

Testing

TEAM ID: PNT2022TMID16036

CODING AND SOLUTIONING

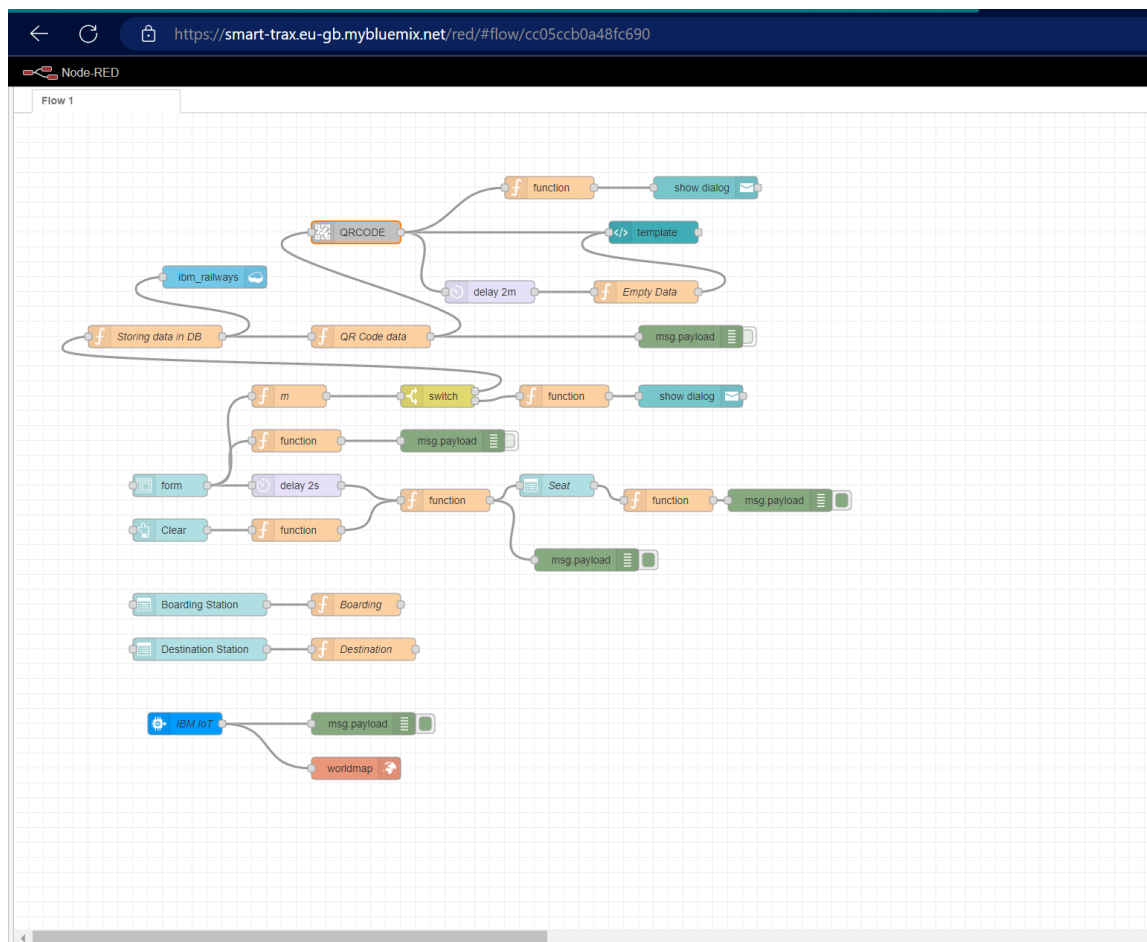
Feature 1

NODE RED SERVICE

Main application

- Connect to cloud
- View map
- Searching for trains
- Storing database
- Booking
- QR code generation

FLOW CHART



IBM IOT

This screenshot shows the Node-RED web interface. The main workspace displays a flow with several nodes, including a blue 'IBM IoT' node. The 'Edit ibmiot in node' dialog is open, showing the following properties:

- Authentication:** API Key
- API Key:** 0b4f321812a3e0d0
- Input Type:** Device Event
- Device Type:** All or h2
- Device Id:** All or h2
- Event:** All or +
- Format:** All or json
- QoS:** 0
- Name:** IBM IoT
- Service:** registered

Below the properties, there is a note: "Use the Input Type property to configure this node to receive Events sent by IoT Devices, Commands sent to IoT Devices, Status Messages referring to IoT Devices, or Status Messages referring to IoT Applications. Check the info tab, to get more information about each of the fields."

The right sidebar shows the 'info' tab for the selected node, displaying its ID 'a2261fcaf9b05823' and type 'ibmiot in'.

This screenshot shows the Node-RED web interface with a different flow. The 'Edit function node' dialog is open, showing the following properties:

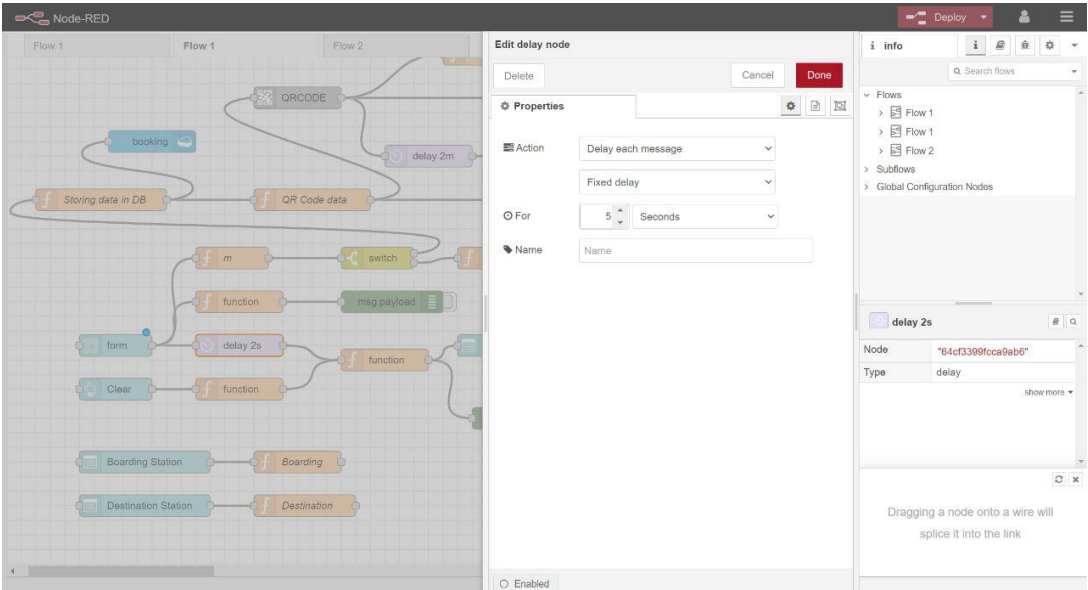
- Name:** m
- Setup:** On Start, On Message, On Stop

The function code is as follows:

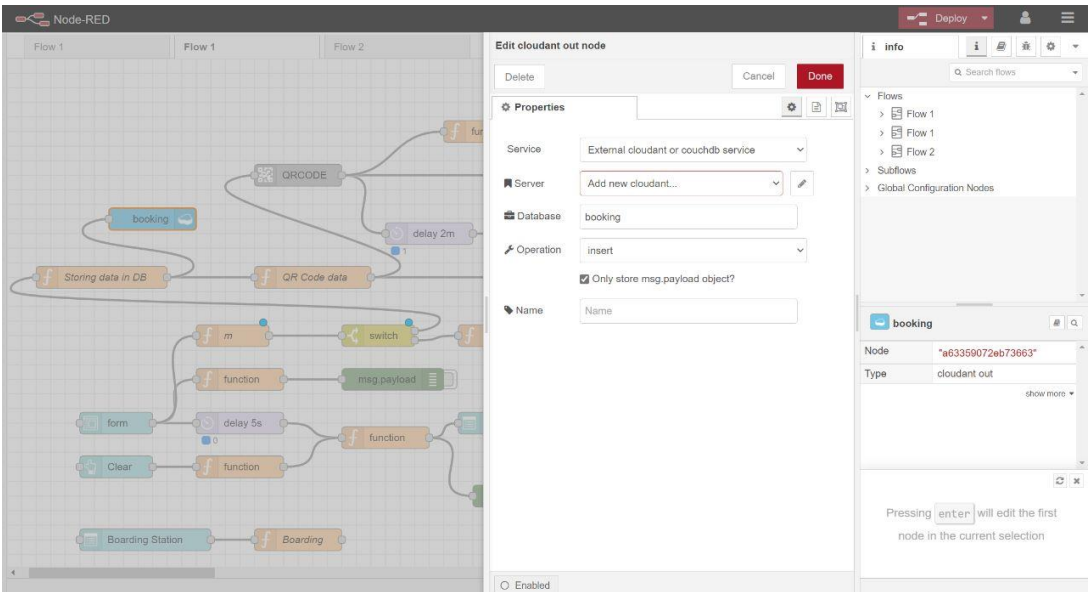
```
1 global.set('m',msg.payload)
2 var a = global.get('s')
3 if(a==1 || a==2 || a==3 || a==4 || a==5){
4   msg.payload = 0
5 }
6 else{
7   msg.payload = 1
8 }
9 return msg;
```

The right sidebar shows the 'info' tab for the selected node, displaying its ID '773c6ccad80fa4f2' and type 'function'.

DELAY NODE



CLOUDANT



FUNCTION NODE

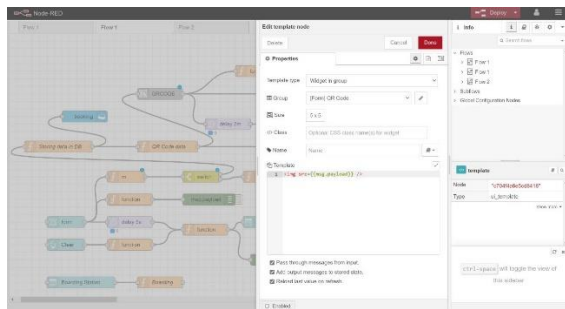
The screenshot shows the Node-RED web interface. On the left, a flow diagram is visible with nodes like 'booking', 'Storing data in DB', 'QR Code data', 'form', 'delay 5s', 'Clear', 'Boarding Station', and 'Destination Station'. The central panel is the 'Edit function node' dialog for the 'Storing data in DB' node. The 'Name' field is set to 'Storing data in DB'. The 'On Message' tab is selected, showing the following JavaScript code:

```
1 var msg=global.get('m')
2 var d=new Date();
3 var utcnd.getTime()=(d.getTimezoneOffset()*60000);
4 var offset=1;
5 newDate=new Date(utc+(3600000*offset));
6 var n=newDate.toISOString()
7 var date=n.slice(0,10)
8 var time=n.slice(11,19)
9 var d1=date+'-'+time
10
11 msg.payload={
12   "_id":d1,
13   "Name":m.Name,
14   "Age":m.Age,
15   "Mobile":m.Num,
16   "Gender":m.Gen,
17   "boarding":global.get('b'),
18   "destination":global.get('d'),
19   "Seat":global.get('s')
20 }
21 return msg;
```

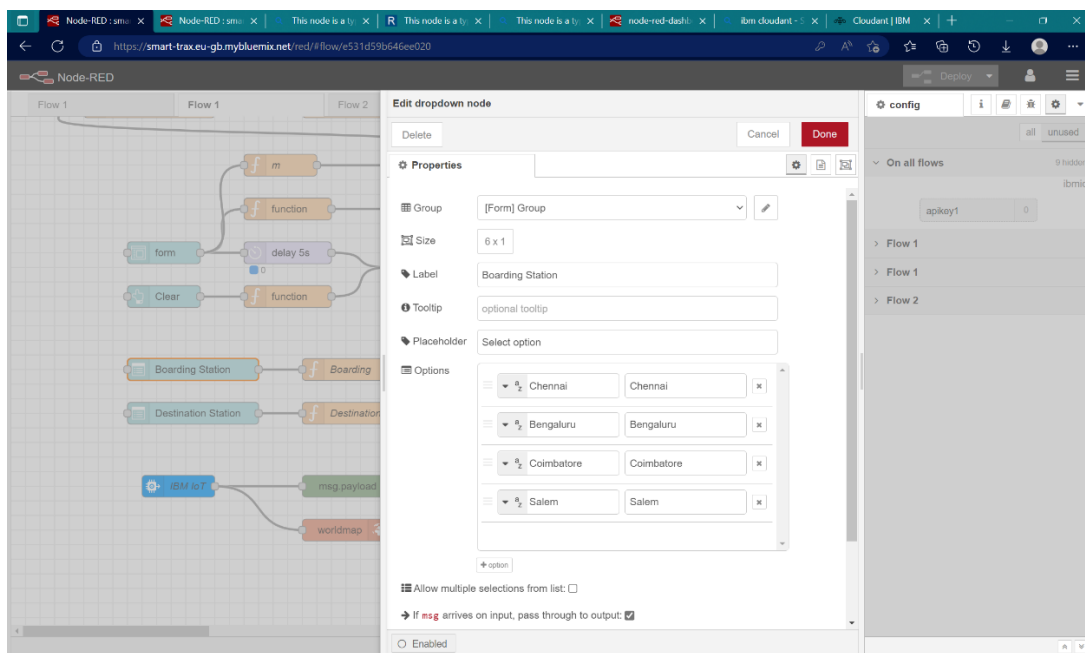
The right panel shows the 'info' sidebar with a search bar and a list of flows. The 'Storing data in DB' node is highlighted in the list, showing its ID '717bc5e641df34a' and type 'function'.

QR CODE GENERATION NODE

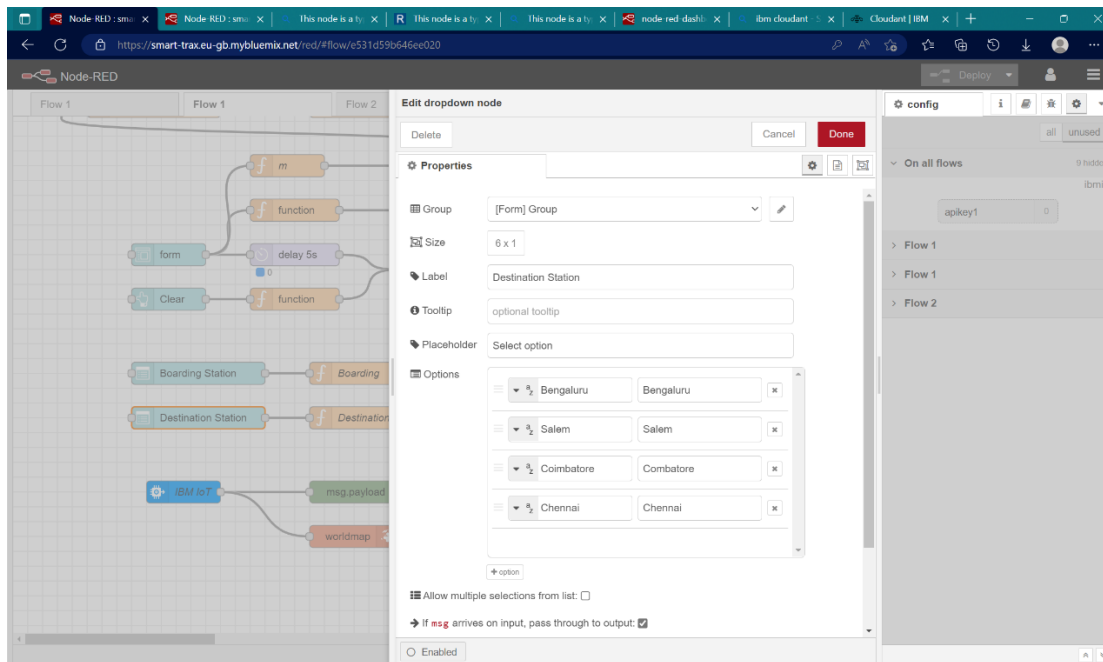
The screenshot shows the Node-RED web interface. On the left, the same flow diagram is visible, but with an additional 'QRCODE' node. The central panel is the 'Edit qrcode gen node' dialog for the 'QRCODE' node. The 'Name' field is set to 'QRCODE'. The 'Show actionable string in Status' checkbox is unchecked. The 'Colors' section shows 'Background' as white and 'QRcode' as black. The 'QR Type' is set to 'Html-Link or Text'. The 'Text or URL' field contains 'https://example.com'. The right panel shows the 'info' sidebar with the 'QRCODE' node highlighted, showing its ID '771e00ab9c084301' and type 'qrcode-generator'.



BOARDING DETAILS



DESTINATION DETAILS




TICKET BOOKING:

Form

Boarding StationChennai

DestinationCombatore

QR Code



Clear

CLEAR

SeatSelect option

Details

Name*

Age*

Mobile.No*

CANCEL

QR Code



Clear


CLEAR

Form

Boarding StationChennai

DestinationCombatore


QR Code



Clear

CLEAR

QR Code



Age*

Mobile.No*

CANCEL

Ticket Booked

OK