Project Development Phase

Sprint-4

Team ID	PNT2022TMID35843
Project Name	Smart Solutions for Railways

Contents

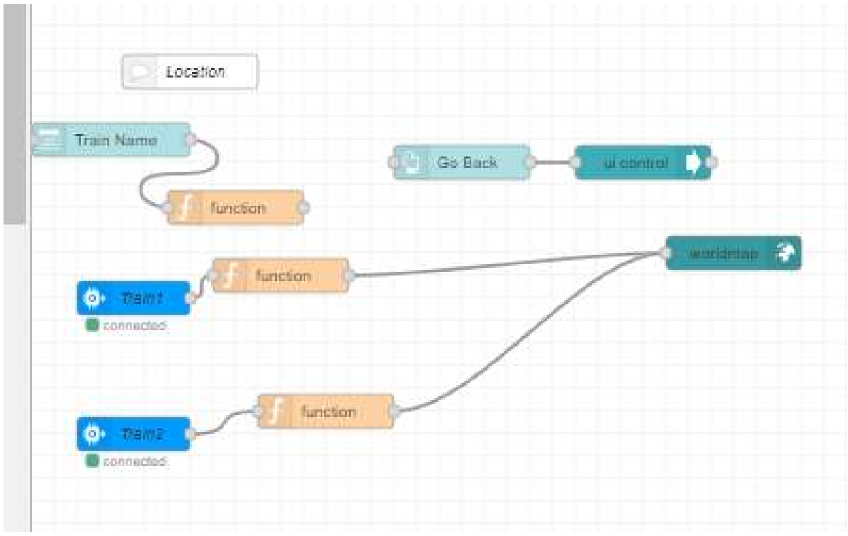
- Live location Tracking
- Live weather Update
- Check Travel History

Node-Red Flow

Location Tracking

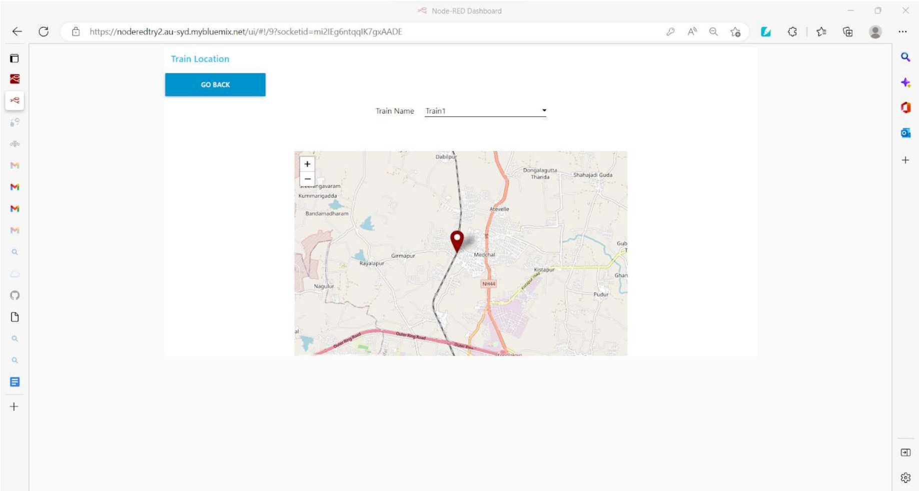
- We have created two IOT devices "Train1" and "Train2" which gives the Name,Latitude and Longitude of the particular Train.
- This data is generated using python Code and sent to the IOT device we created in IBM IOT platform(Python Code will be attached as a separate file).

Node-flow

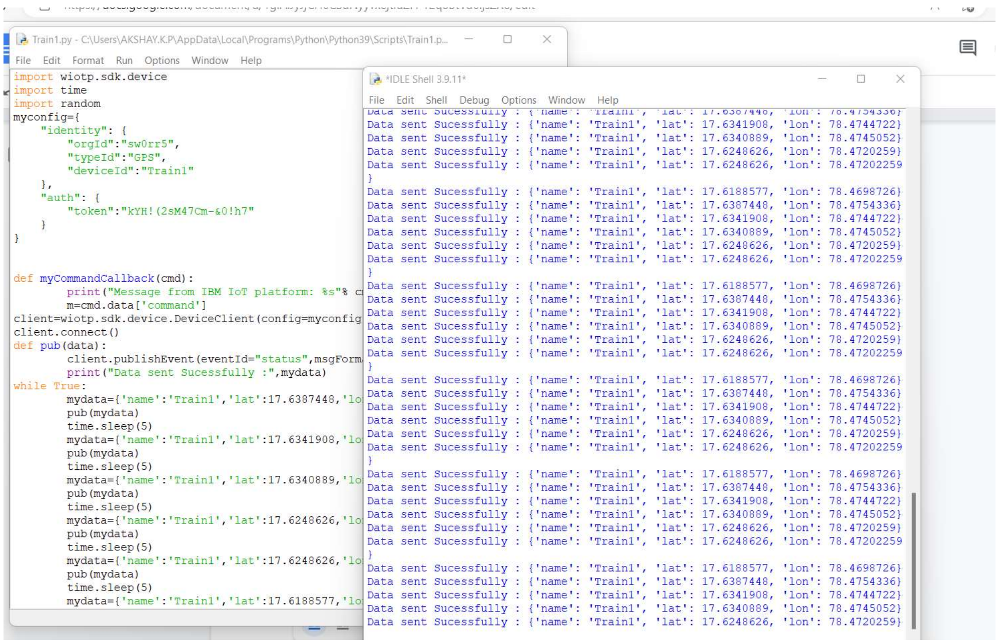


UI of Location Page

- You can select the which you want to locate and it’s location will be shown in the map.



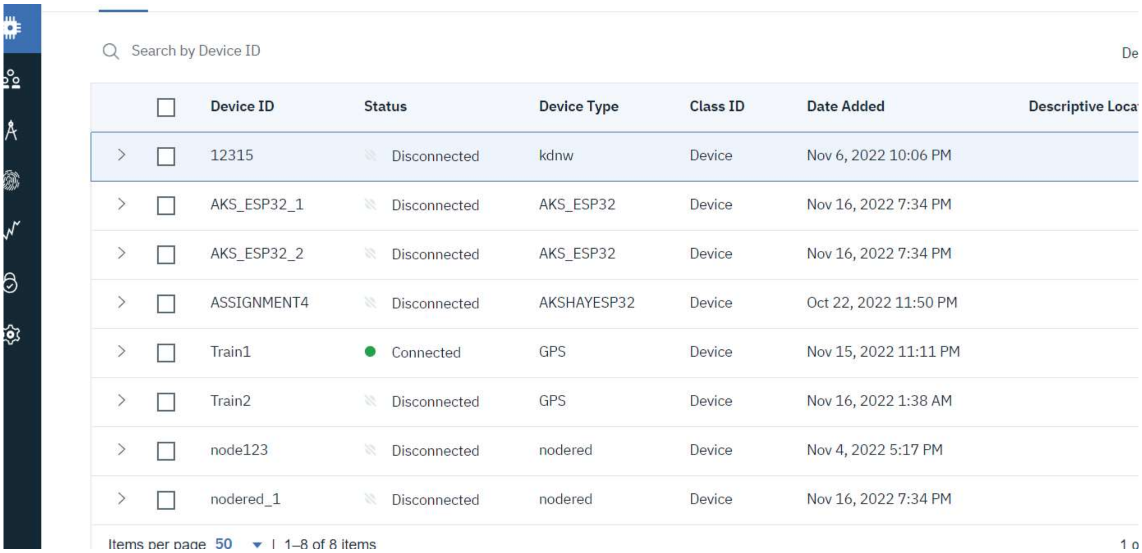
Python Script Output


Train1.py - C:\Users\AKSHAY.K.P\AppData\Local\Programs\Python\Python39\Scripts\Train1.p...
File Edit Format Run Options Window Help
IDLE Shell 3.9.11
File Edit Shell Debug Options Window Help

```
import wiotp.sdk.device
import time
import random
myconfig={
    "identity": {
        "orgId":"sw0rr5",
        "typeId":"GPS",
        "deviceId":"Train1"
    },
    "auth": {
        "token":"kYH!(2sM47Cm-&0!h7"
    }
}


def myCommandCallback(cmd):
        print("Message from IBM IoT platform: %s"% c
        m=cmd.data['command']
client=wiotp.sdk.device.DeviceClient(config=myconfig
client.connect()
def pub(data):
        client.publishEvent(eventId="status",msgForm
        print("Data sent Sucessfully :",mydata)
while True:
        mydata={'name':'Train1','lat':17.6387448,'lo
        pub(mydata)
        time.sleep(5)
        mydata={'name':'Train1','lat':17.6341908,'lo
        pub(mydata)
        time.sleep(5)
        mydata={'name':'Train1','lat':17.6340889,'lo
        pub(mydata)
        time.sleep(5)
        mydata={'name':'Train1','lat':17.6248626,'lo
        pub(mydata)
        time.sleep(5)
        mydata={'name':'Train1','lat':17.6248626,'lo
        pub(mydata)
        time.sleep(5)
        mydata={'name':'Train1','lat':17.6188577,'lo
```

```
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.47202259
}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6188577, 'lon': 78.4698726}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.47202259
}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6188577, 'lon': 78.4698726}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.47202259
}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6188577, 'lon': 78.4698726}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.47202259
}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6188577, 'lon': 78.4698726}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.47202259
}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6188577, 'lon': 78.4698726}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6387448, 'lon': 78.4754336}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6341908, 'lon': 78.4744722}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6340889, 'lon': 78.4745052}
Data sent Sucessfully : {'name': 'Train1', 'lat': 17.6248626, 'lon': 78.4720259}
```

Device in IOT Platform



Search by Device ID

De

	□	Device ID	Status	Device Type	Class ID	Date Added	Descriptive Loca
>	□	12315	Disconnected	kdnw	Device	Nov 6, 2022 10:06 PM	
>	□	AKS_ESP32_1	Disconnected	AKS_ESP32	Device	Nov 16, 2022 7:34 PM	
>	□	AKS_ESP32_2	Disconnected	AKS_ESP32	Device	Nov 16, 2022 7:34 PM	
>	□	ASSIGNMENT4	Disconnected	AKSHAYESP32	Device	Oct 22, 2022 11:50 PM	
>	□	Train1	● Connected	GPS	Device	Nov 15, 2022 11:11 PM	
>	□	Train2	Disconnected	GPS	Device	Nov 16, 2022 1:38 AM	
>	□	node123	Disconnected	nodered	Device	Nov 4, 2022 5:17 PM	
>	□	nodered_1	Disconnected	nodered	Device	Nov 16, 2022 7:34 PM	

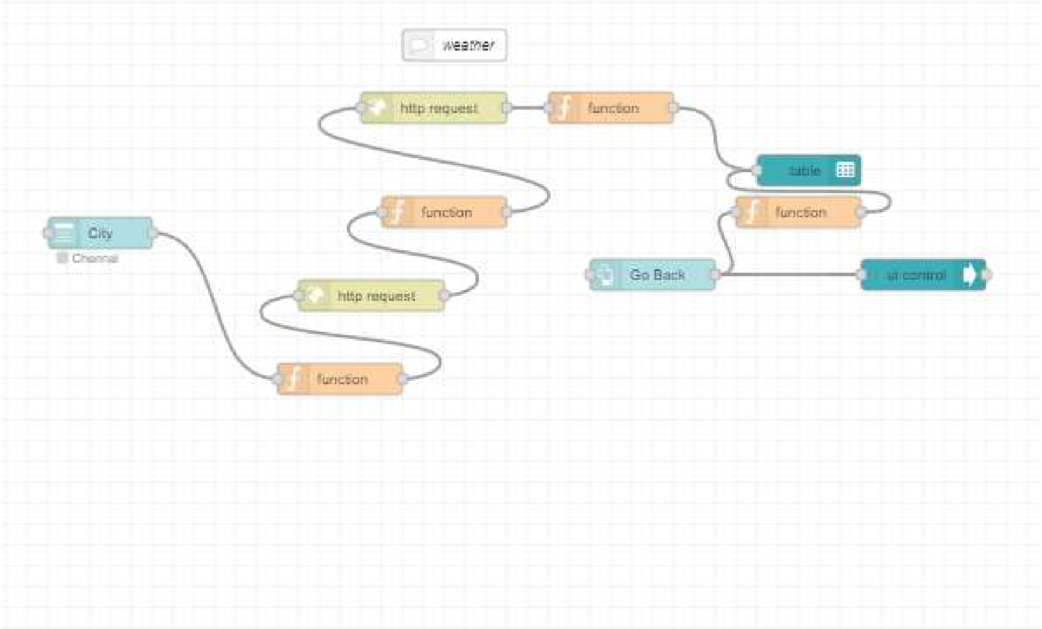
Items per page 50 ▾ | 1–8 of 8 items

1 o

Weather

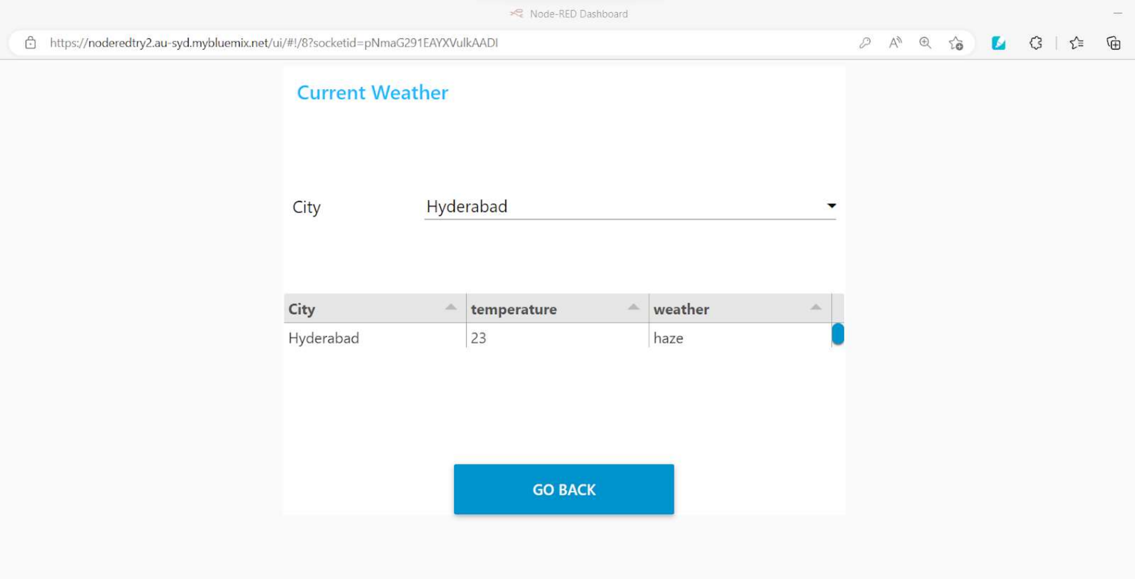
- You can select the place where you want to know the weather condition.(**We have used openweather API**).

Node-Red Flow



- We have used two http requests one for getting the Latitude and Longitude and another for getting the weather at that corresponding Lat and Lon.

UI of weather Page



Node-RED Dashboard

https://noderedtry2.au-syd.mybluemix.net/ui/#!/8?socketid=pNmaG291EAYXVulkAADI

Current Weather

City: Hyderabad

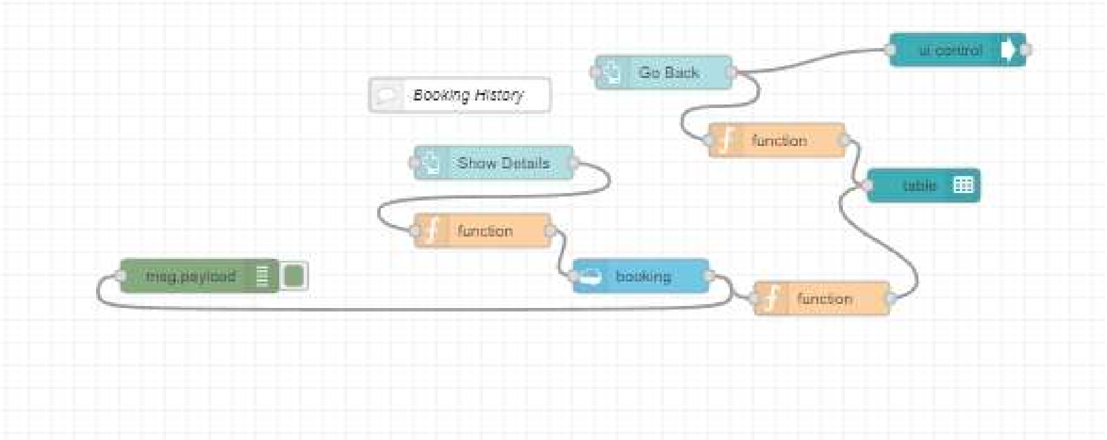
City	temperature	weather
Hyderabad	23	haze

Booking History

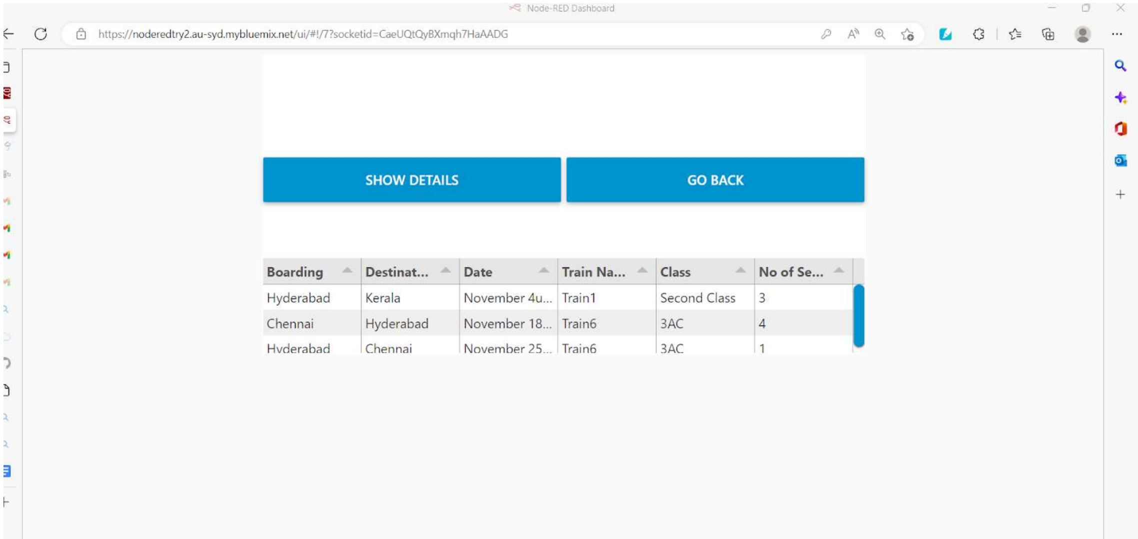
Key features

- We have created a database named “booking” where all the booking details will be stored.
- <u>But we have created an index in our Database with the username and used query to filter out the history of a particular user.</u>
- <u>So we have used Call by index option in cloudant node.</u>

Node-Red Flow

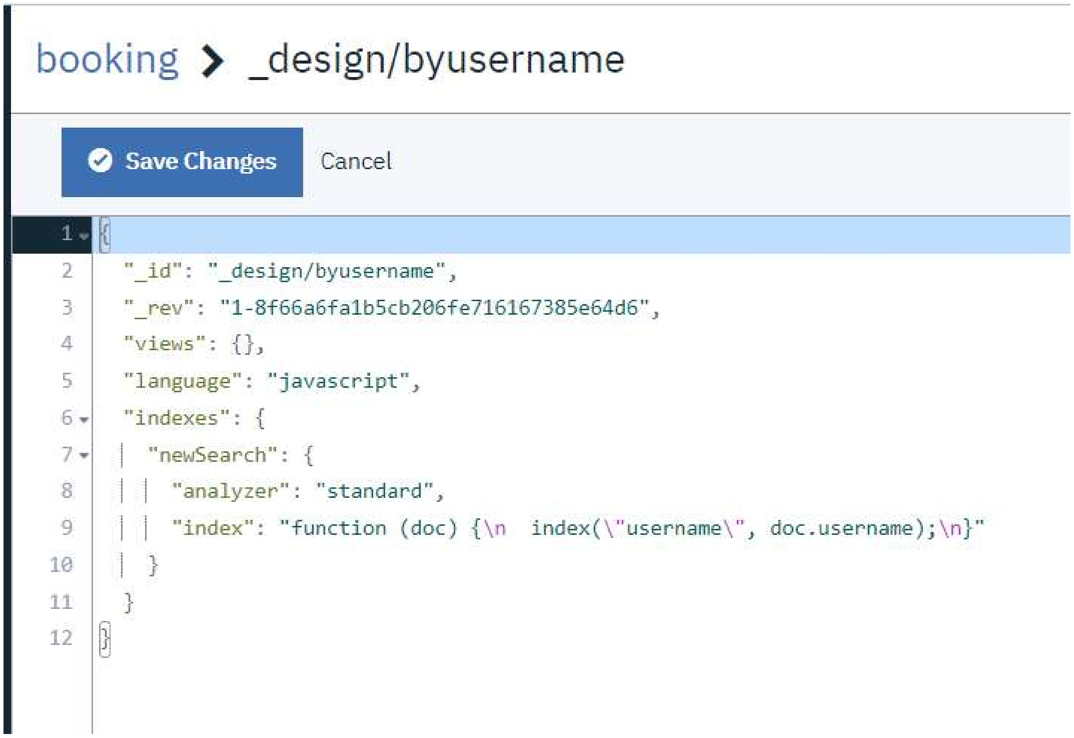


UI of Booking History Page


Node-RED Dashboard
https://noderedtry2.au-syd.mybluemix.net/ui/#!/7?socketid=CaeUQtQyBXmqh7HaADG
SHOW DETAILS
GO BACK

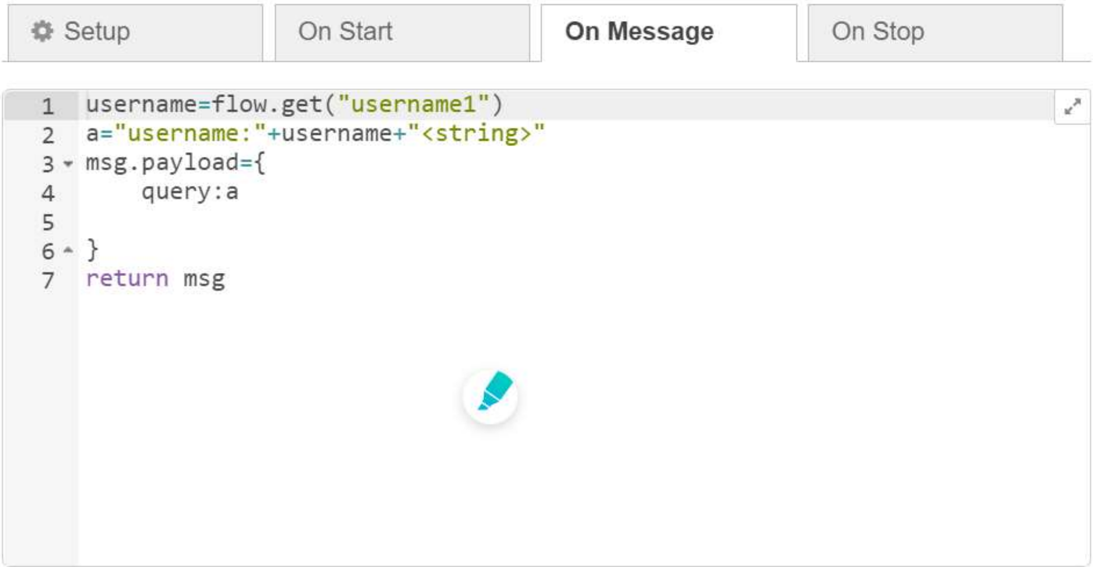
Boarding	Destinat...	Date	Train Na...	Class	No of Se...
Hyderabad	Kerala	November 4u...	Train1	Second Class	3
Chennai	Hyderabad	November 18...	Train6	3AC	4
Hyderabad	Chennai	November 25...	Train6	3AC	1

Database with username added as index



```
{
  "_id": "_design/byusername",
  "_rev": "1-8f66a6fa1b5cb206fe716167385e64d6",
  "views": {},
  "language": "javascript",
  "indexes": {
    "newSearch": {
      "analyzer": "standard",
      "index": "function (doc) {\n  index(\"username\", doc.username);\n}"
    }
  }
}
```

Query code passed to the cloudant node



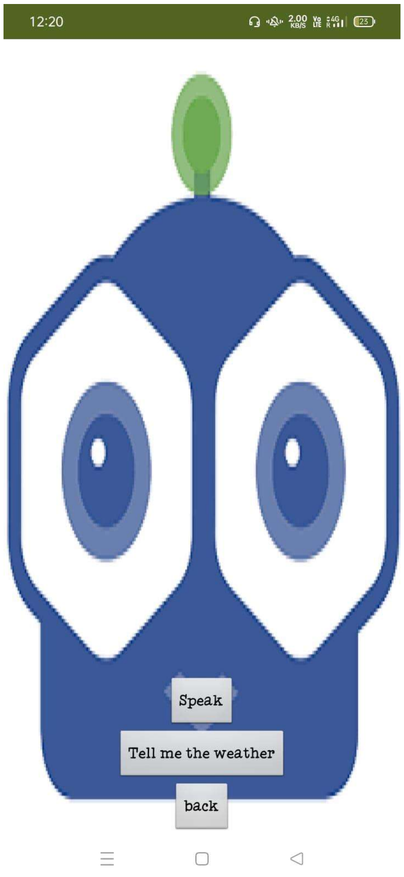
```
username=flow.get("username1")
a="username:"+username+"<string>"
msg.payload={
    query:a

}
return msg
```

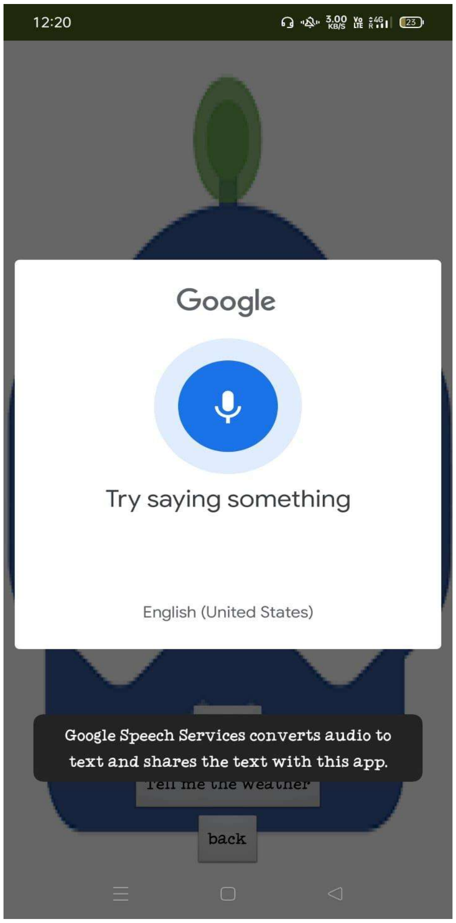
MIT APP INVENTOR ANDROID APPLICATION:-

WORKFLOW OF SPEAKING BOT TO GET WEATHER REPORT OF THE CITY

Weather detail of particular city will be notified :-
Here we have created a **speaking bot ,**which will take input city names from the user ,by clicking the **speak button.**
After clicking the speak button ,user need to tell the city via speaking i.e. a **google speaking assistant** will be invoked on the screen.
Then, the place will be stored and using **weather API ,** weather of particular location will be retrieved and will be intimated to the user via voice speaking bot.

{
THE INTERNAL WORKINGS ARE PROGRAMMED IN THE BACKEND USING BLOCKS IN THE MIT APP INVENTOR
}





Google Speech Services converts audio to text and shares the text with this app.