**1.INTRODUCTION**

1.1 **Overview**

* Purpose of the Project
* Existing Problem
* Proposed Solution
* Block Diagram
* Hardware/Software Designing
* Experimental Investigations
* Flowchart
* Result
* Advantages & Disadvantages
* Applications
* Conclusions
* Future Scope
* Bibliography

1.2 **Purpose**

* Continuous monitoring of Cargo temperature, humidity, and air quality level.
* Alert is generated when freshness becomes low
* Trucks will be traced during the transition between the source and destination
* Admins will be notified whenever the truck reaches the destination
* Less latency in communication from device to cloud using MQTT

**2.LITERATURE SURVEY**

2.1  **Existing Problem**

The problem with the present existing device is it cannot able to identifies the actual destination of cargo it only assume the destination of cargo.

2.2 **Proposed Solution**

Before starting with the analysis and elegance of project, we've got a bent to refer many analysis papers, manuals, documents

associated with the thought of project there are many paper concerns about cargo tracking systems but many etiquettes. W. They., E. L. Tan,

EW Lee, Ti Li, Integrated a Solutions for Integrated Track and Trace in provide Chains supported RFID and GPS Bottom of Form [1]. RFID

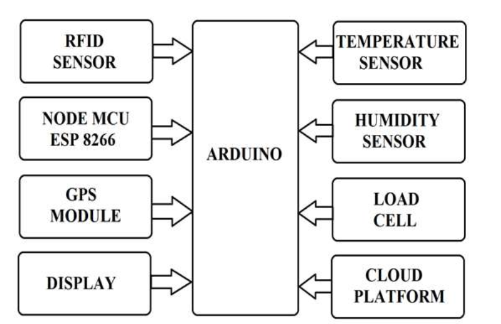
is used for the inventory and material handling method within the warehouse to control the dropping of products in the warehouse.

These above mentioned systems are great for references. However, none of them operate a system such as a refrigerated cargo tracking

system to supply environmental data. Therefore, we will propose a flexible solution to solve this issue.

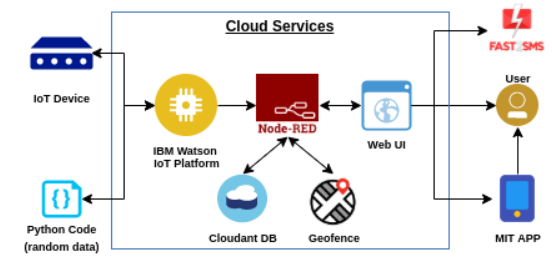
**3.Theoretical Analysis**

**3.1 Block Diagram**



**3.2 Hardware/Software Designing**

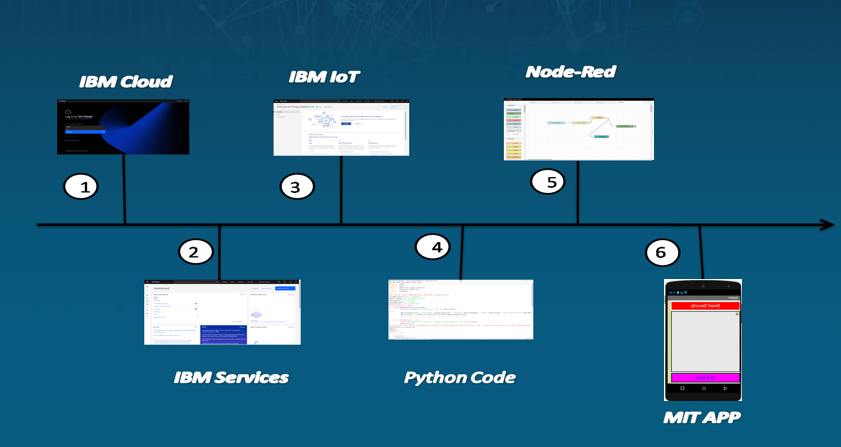
The Software designing involves genera We used IBM Cloud Services to create Internet of Things platform. In IoT platform we create a virtual Raspberry Pi device. After creating the design we get the device credentials. We use these credentials in Python program then we integrated the Node-Red platform with IoT. With the help of MIT APP Inverter we designed the app & integrated with the Node-Red to observe the values.



**4.Experiment Investigation**

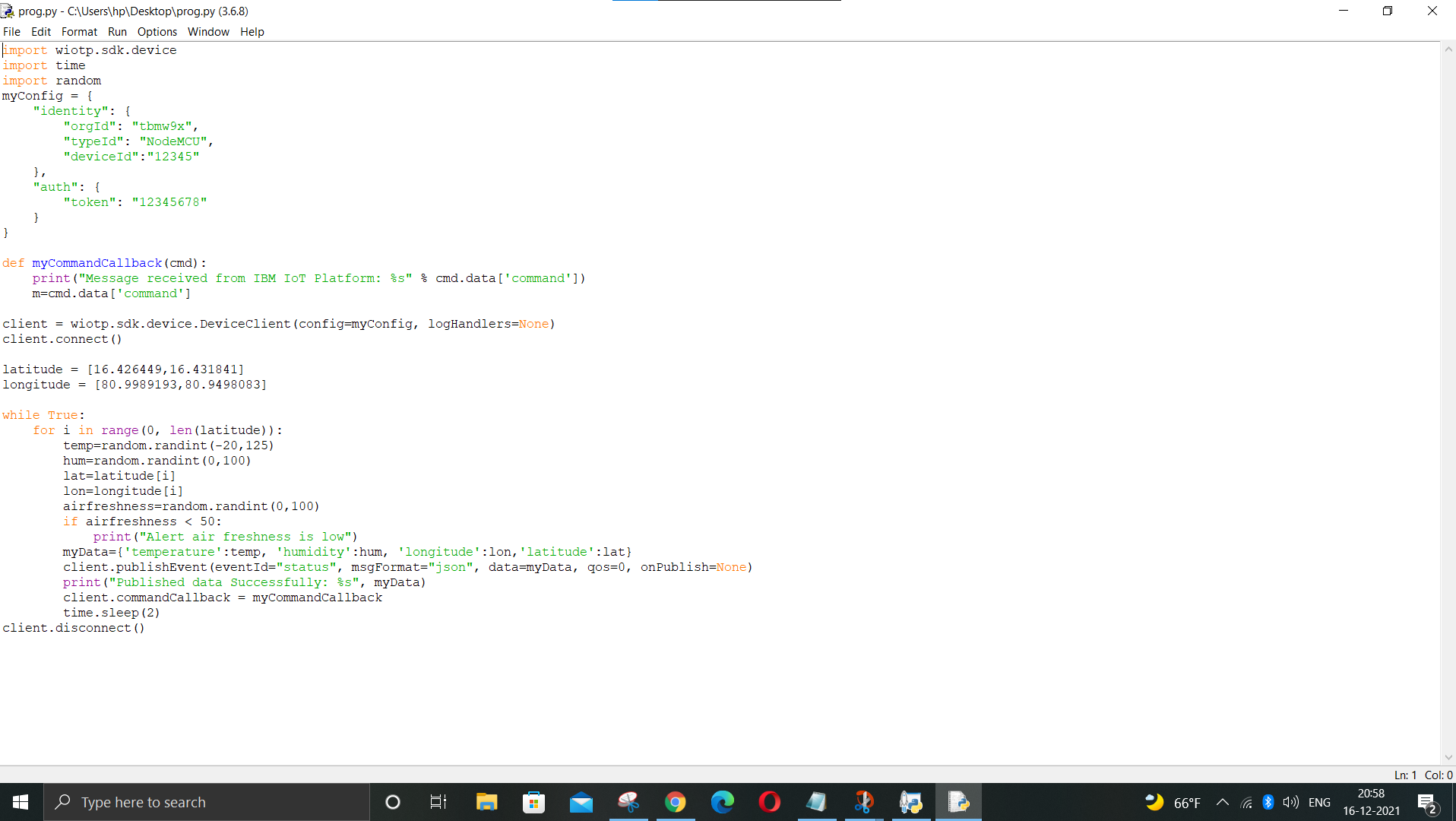
To complete our project work we collected the required data from Google & research papers. After getting the complete knowledge we work according to our roles in the project. At first we create the IBM Cloud account then we created the Internet of Things Platform after we wrote a python code in IDLE to connect IBM IoT Platform. Next we created the Node-Red Services. This service helps us to show virtual flow graphs. We connect Node-Red to IBM IoT to get the current, voltage and calculated bills. From Node-Red we send values to the MIT APP. From app we can view the details of the person .

**5.FLOWCHART**

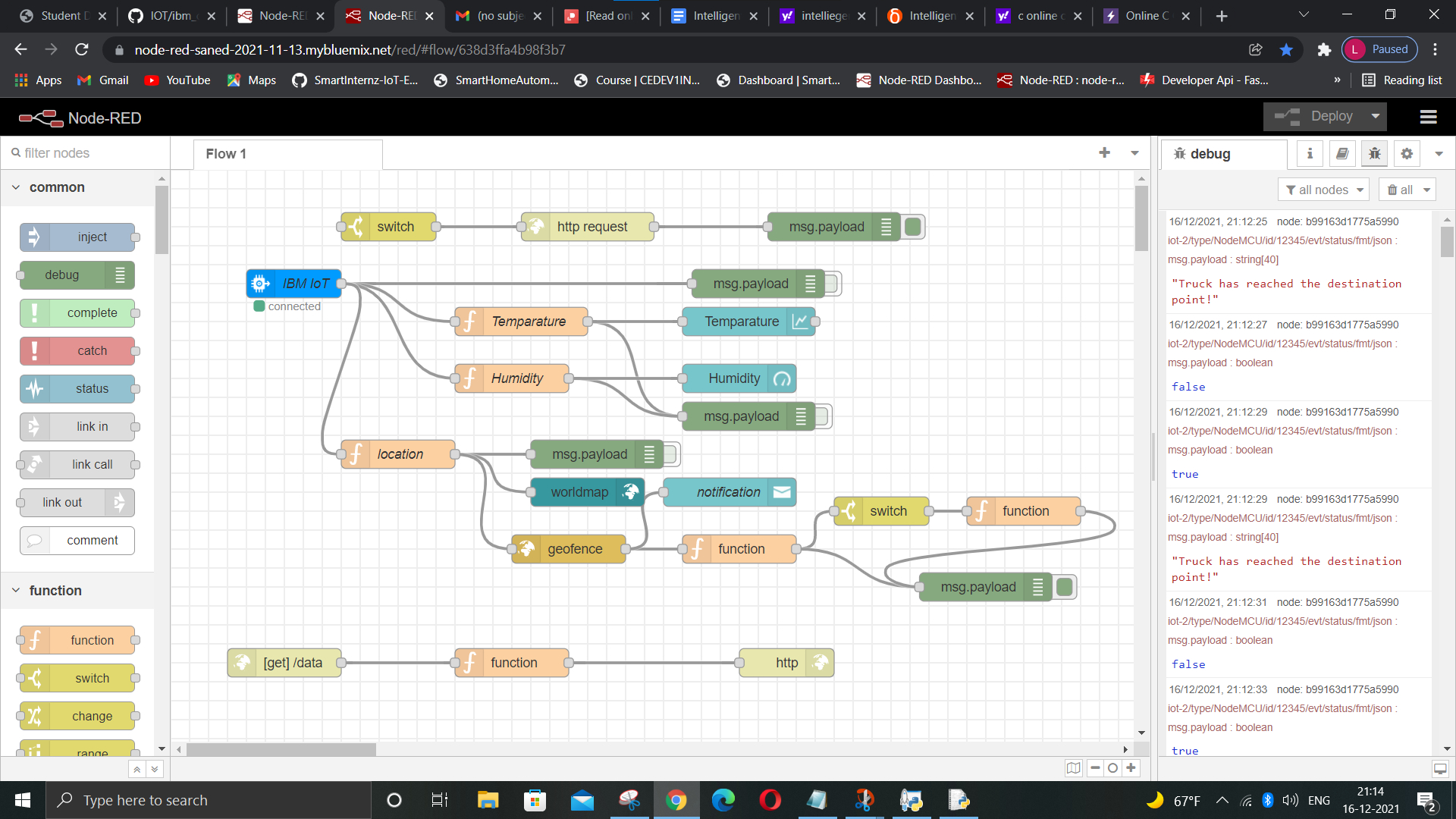
****

**6 RESULT**

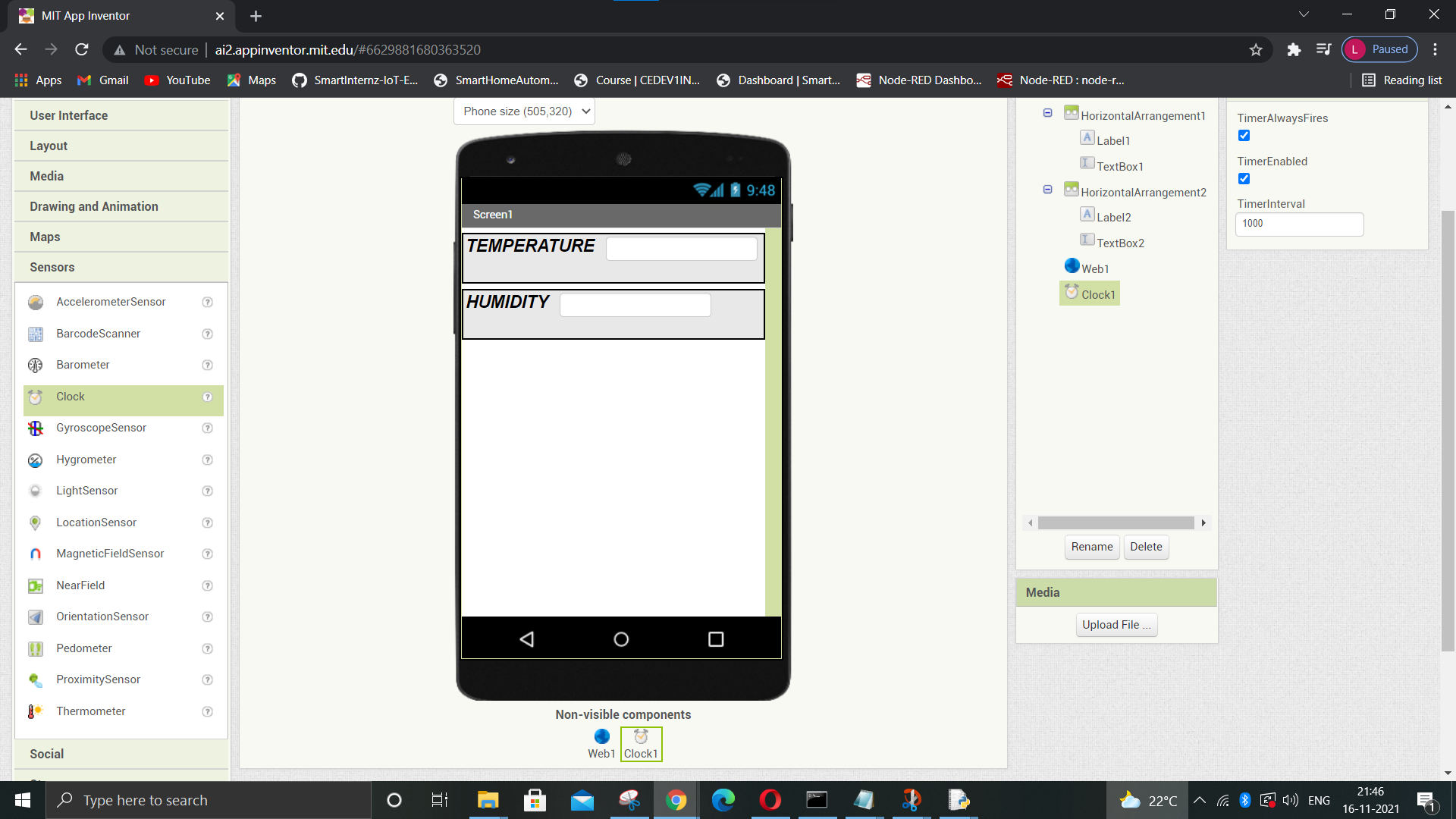
**Python Code:**

****

**Node-Red:**

****

**MIT APP:**

****

**7 ADVANTAGES & DISADVANTAGES**

**Advantages:**

1) Manipulation of cargo are often easily tracked.

2) Cargo can be in surveillance during journey

3) Easy to manage all the parameter data securely and easily.

4) Centralized database helps in avoiding conflicts between different branches.

5) Due to cloud based automatic system is used the data is more error free.

6) Can generate required reports easily.

**Disadvantages:**

1) Internet connectivity is mandatory.

**8. APPLICATIONS**

1) Industrial applications:-These system are often used for transportation of products equipment’s carriers in industries replacing

traditional cargo systems.

2) It's used for transportation of perishable Agricultural products.

3) It's used for temperature sensitive Medicine's transportation

**9. CONCLUSIONS**

Some well-known and fashionable wares management system are antecedently developed that are classified on completely

different technologies. however this project is used IOT based mostly system for wares management long with cloud-based services and

cargo load management, RFID secured access technology so as that the protection of the door are typically managed by remote location.

Since our planned system is created over wireless sensors network. it's a simply be put in and friendly for the users with none overheard

difficult work and it conjointly doesn’t would like a lot of coming up with and wired association. Real time detector knowledge are typically

obtained exploitation varied routing protocols over a wireless medium.

**10. FUTURE SCOPE**

The future work goes through the implementation of the solution in larger scales where more people would use it.

**11. BIBLIOGRAPHY**

<https://cloud.ibm.com/>

<https://cloud.ibm.com/services/iotf-service/crn%3Av1%3Abluemix%3Apublic%3Aiotf-service%3Aus-south%3Aa%2Fd48841940b8347ae82a67d853621d13e%3A43a31a3e-78e6-4667-9d57-738d176edcf8%3A%3A?paneId=manage>

<http://ai2.appinventor.mit.edu/#6629881680363520>

<https://node-red-saned-2021-11-13.mybluemix.net/red/#flow/638d3ffa4b98f3b7>

<https://node-red-saned-2021-11-13.mybluemix.net/ui/#!/0?socketid=YAMBjHmVrfqj7imJAAAk>

**APPENDIX**

**A. Source Code**

import wiotp.sdk.device

import time

import random

myConfig = {

"identity": {

"orgId": "tbmw9x",

"typeId": "NodeMCU",

"deviceId":"12345"

},

"auth": {

"token": "12345678"

}

}

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()

latitude = [16.426449,16.431841]

longitude = [80.9989193,80.9498083]

while True:

for i in range(0, len(latitude)):

temp=random.randint(-20,125)

hum=random.randint(0,100)

lat=latitude[i]

lon=longitude[i]

airfreshness=random.randint(0,100)

if airfreshness < 50:

print("Alert air freshness is low")

myData={'temperature':temp, 'humidity':hum, 'longitude':lon,'latitude':lat}

client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0, onPublish=None)

print("Published data Successfully: %s", myData)

client.commandCallback = myCommandCallback

time.sleep(2)

client.disconnect()