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**

In supply chain managment ,For cargo positioning related tasks, the proposed cargo-level tracking system,which was supported by contigious monitoring cargo.The work integrated the benifits of these to achieve a low-cost and low

power scheme.A thermal sensor was adopted to monitor the temperature of the environment .The monitored environment temperature can be sent via bluetooth communication to android based mobile devices.the monitored environment temperature ,its status is uploded to server to monitor a status of the refregirated cargo.

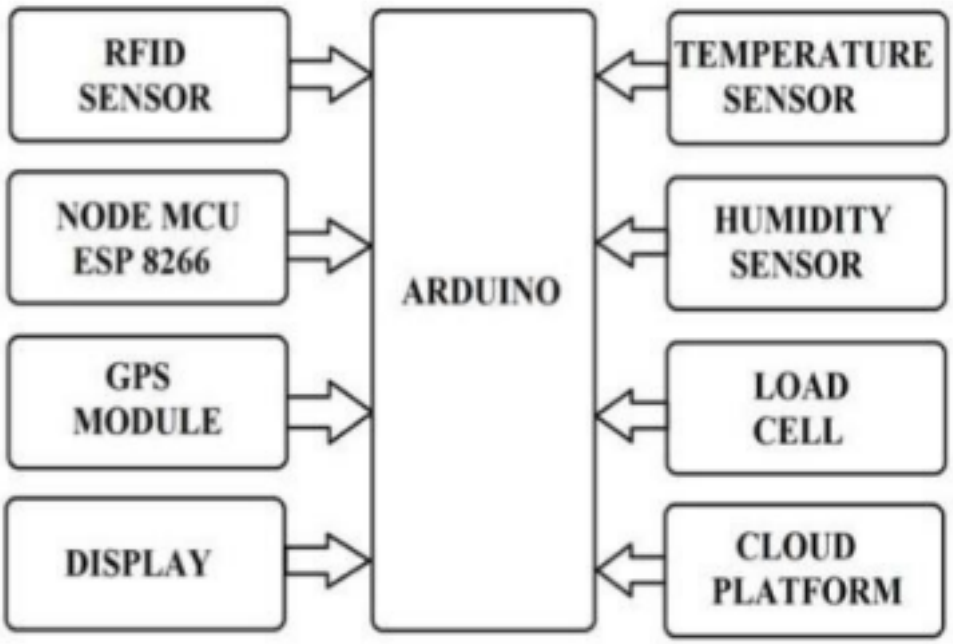
2.LITERATURE SURVEY

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 system but many etiquettes .W. htey ,E. L. Tan ,EW Lee 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 product in the warehouse.

These above mentioned system are great for references .However, none of them operate a system such as a refrigerated cargo tracking system to supply environment data .Therefore, we will propose a flexible solution to solve this issue.

3.Theoretical Analysis

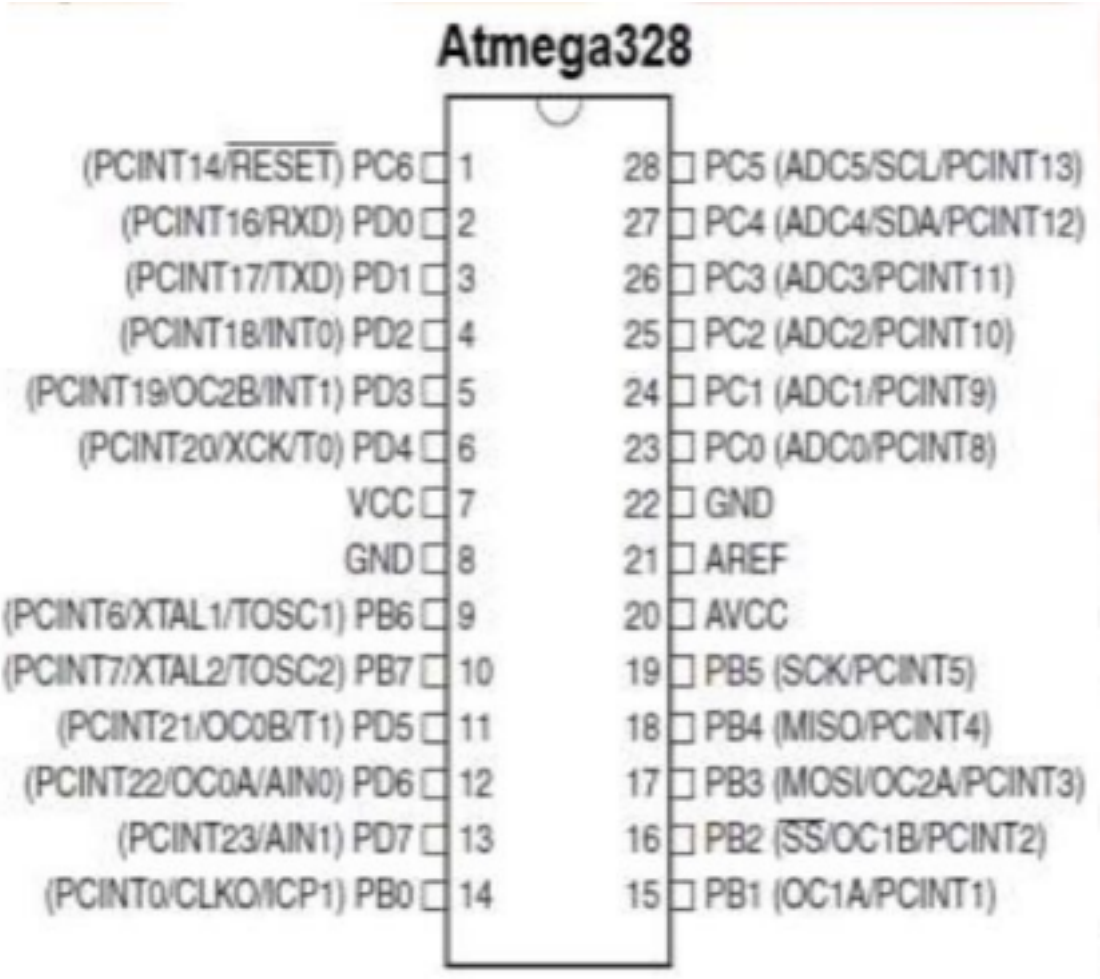
3.1 Block Diagram



3.2 Hardware/Software Designing

The Software designing involves genera We used IBM Cloud Services to create

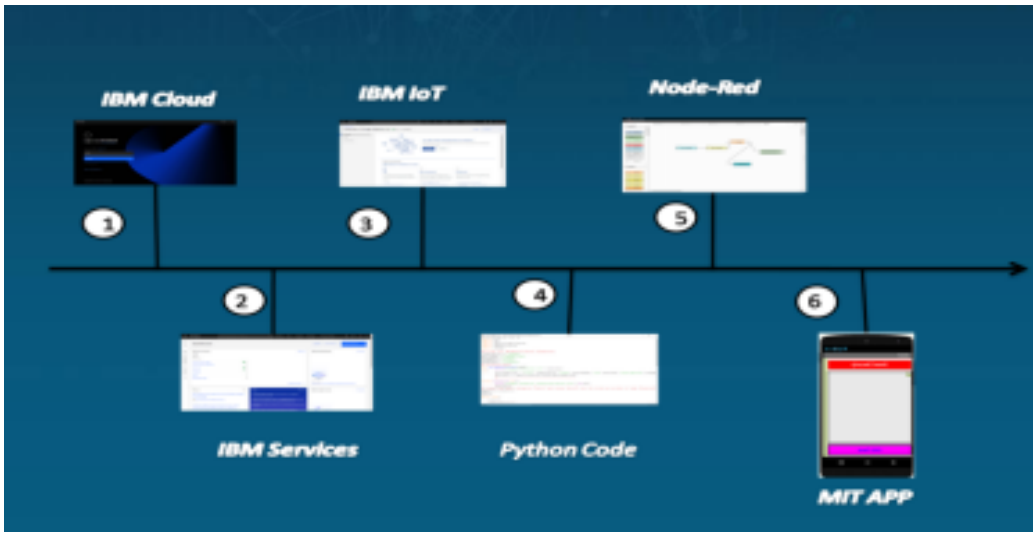
Internet of Things platform. In IoT platform we create a virtual Raspberry Pi device. After creating the design we get the device credentials. We use these credentials in Python program then we integrated the Node-Red platform with IoT. With the help of MIT APP Inverter we designed the app & integrated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

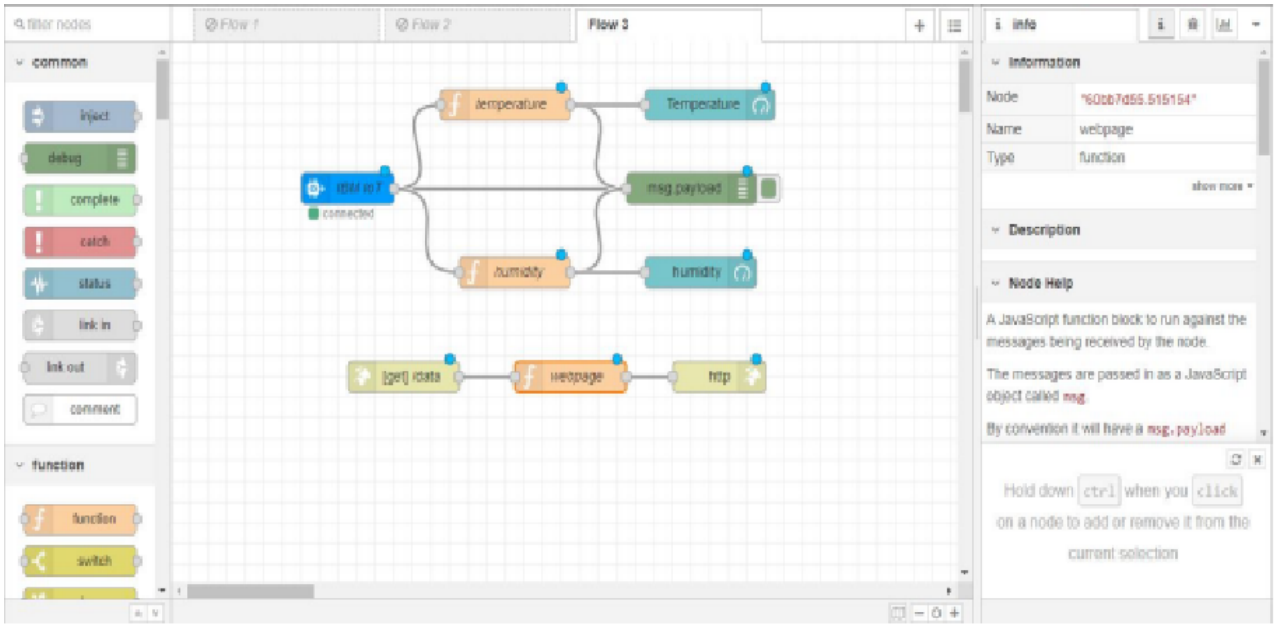


4.Experiment Investigation

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.FLOWCHAR




filter nodes
Flow 1
Flow 2
Flow 3
common
inject
debug
complete
catch
status
link in
link out
comment
function
function
switch
temperature
Temperature
msg.payload
connected
humidity
humidity
[get] rdata
webpage
http
info
Information
Node
"60bb7d55.515154"
Name
webpage
Type
function
show more
Description
Node Help
A JavaScript function block to run against the messages being received by the node.
The messages are passed in as a JavaScript object called msg.
By convention it will have a msg.payload
Hold down ctrl when you click on a node to add or remove it from the current selection

6 RESULT

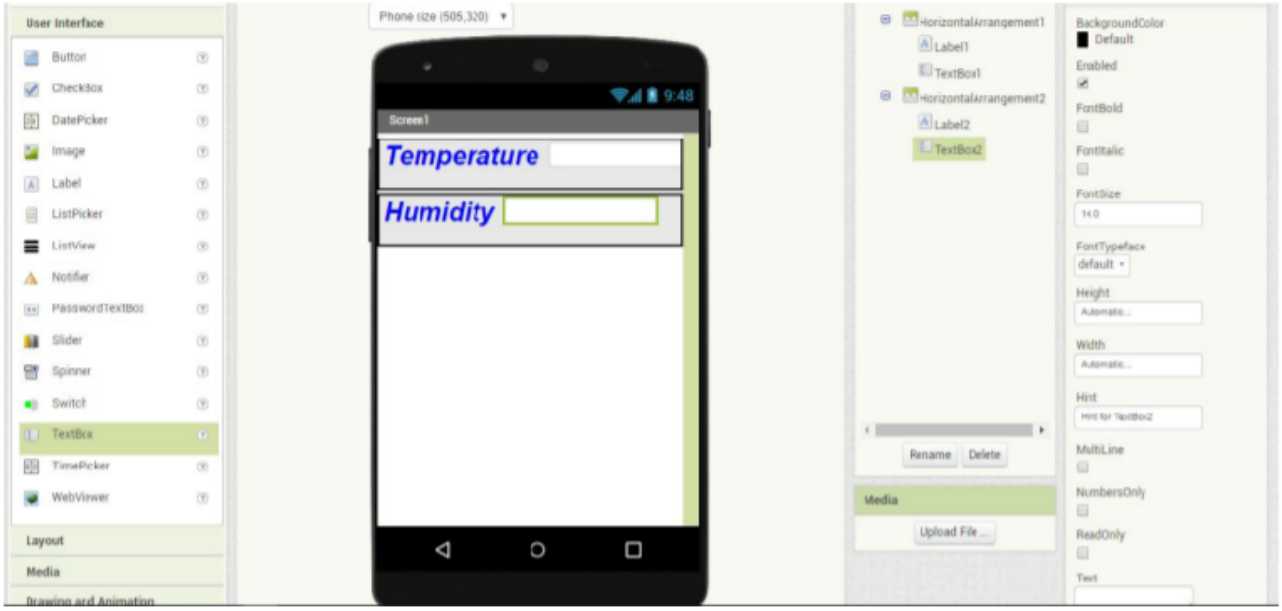
Python Code:

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
import json

#Provide your IBM Watson Device Credentials
organization = "gz7i80"
deviceType = "IotDevice"
deviceId = "1001"
authMethod = "token"
authToken = "12345678900"


# Initialize the device client.
T=0
H=0

def myCommandCallback(cmd):
        print("Command received: %s" % cmd.data['command'])


        if cmd.data['command']=='lighton':
                print("LIGHT ON IS RECEIVED")


        elif cmd.data['command']=='lightoff':
                print("LIGHT OFF IS RECEIVED")

        if cmd.command == "setInterval":
                if 'interval' not in cmd.data:
                        print("Error - command is missing required information: 'interval'")
                else:
                        interval = cmd.data['interval']
        elif cmd.command == "print":
                if 'message' not in cmd.data:
                        print("Error - command is missing required information: 'message'")
                else:
                        print(cmd.data['message'])

try:
        deviceOptions = {"org": organization, "type": deviceType, "id": deviceId, "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #..............................................

except Exception as e:
        print("Caught exception connecting device: %s" % str(e))
        sys.exit()

# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()

while True:
        T=23
        H=45
        #Send Temperature & Humidity to IBM Watson
        data = {"d":{ 'temperature' : T, 'humidity': H }}
        #print data
        def myOnPublishCallback():
```


Activate Windows
Go to Settings to activate Wind

```
61          def myOnPublishCallback():
62              print ("Published Temperature = %s C" % T, "Humidity = %s %%" % H, "to IBM Watson")
63
64          success = deviceCli.publishEvent("Data", "json", data, qos=0, on_publish=myOnPublishCallback)
65          if not success:
66              print("Not connected to IoTF")
67          time.sleep(1)
68
69          deviceCli.commandCallback = myCommandCallback
70
71  # Disconnect the device and application from the cloud
```

MIT APP:



7 ADVANTAGES & DISADVANTAGES

Advantages:

1) Manipulation of cargo are often easily traced

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 equipments carriers traditional cargo system

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

3) Its used for temperature sensitivity 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.

10. FUTURE SCOPE

The future work goes through the implementation of the solution in larger scales where more people would use it. Until then, the training of new neural networks using the preprocessing techniques is presented, and the study of new alternatives for cameras is on the agenda.

11. BIBLIOGRAPHY

https://cloud.ibm.com/registration

https://cloud.ibm.com/catalog/services/watson-studio

http://Ai2.appinventor.mit.edu

https://flows.nodered.org/node/node-red-dashboard

https://appinventor.mit.edu/

APPENDIX

A. Source Code

```
import sys
import ibmiotf.application
import ibmiotf.device
import random
import json
#Provide your IBM Watson Device Credentials
organization = "gz7i80"
deviceType = "iotdevice"
deviceId = "1001"
authMethod = "token"
```

```
authToken = "1234567890"
# Initialize the device client.
T=0
H=0
def myCommandCallback(cmd):
 print("Command received: %s" % cmd.data['command'])  if
cmd.data['command']=='lighton':
 print("LIGHT ON IS RECEIVED")



 elif cmd.data['command']=='lightoff':
 print("LIGHT OFF IS RECEIVED")

 if cmd.command == "setInterval":
 if 'interval' not in cmd.data:
 print("Error - command is missing required information:  'interval'")
 else:
 interval = cmd.data['interval']
 elif cmd.command == "print":
 if 'message' not in cmd.data:
 print("Error - command is missing required information:  'message'")
 else:
 print(cmd.data['message'])
try:
       deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,
"auth-method": authMethod, "auth-token": authToken}
       deviceCli = ibmiotf.device.Client(deviceOptions)
       #.....................................

except Exception as e:
       print("Caught exception connecting device: %s" % str(e))
       sys.exit()
# Connect and send a datapoint "hello" with value "world" into the cloud as an
event of type "greeting" 10 times
deviceCli.connect()
while True:
 T=23
 H=45
 #Send Temperature & Humidity to IBM Watson
 data = {"d":{ 'temperature' : T, 'humidity': H }}
 #print data
 def myOnPublishCallback():
```

```
 print ("Published Temperature = %s C" % T, "Humidity = %s %%" % H, "to  IBM
Watson")
 success = deviceCli.publishEvent("Data", "json", data, qos=0,
on_publish=myOnPublishCallback)
 if not success:
 print("Not connected to IoTF")
 time.sleep(1)

 deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()
```