1. INTRODUCTION

1.1 OVERVIEW

A wearable device that contains sensors that are used for the monitoring of the temperature, blood pressure and the pulse of the individual who wears the device. The device monitors the parameters mentioned and the process that takes place gives an output to the user about their blood pressure, temperature and pulse values on a device and an alert is given once the parameters are abnormal.

1.2 PURPOSE

The main aim of the device is to detect the abnormalities in the health parameters that are essential for a human being in leading a healthy life.

2. LITERATURE SURVEY

2.1 EXISITING PROBLEM

During these busy days people do not have a better quality of time to monitor their basic health parameters, so problems arise from there. They may even be the root cause for certain diseases. And in such a pandemic situation you can not go out to just check your BP or pulse.

2.2 PROPOSED SOLUTION

A device that senses the health parameters and is displayed on the app that is the user interface for the user to know the data up to date. This is also featured by the alert of an sms when abnormal patterns are detected in the parameters sensed.

3. THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

Connection Gateway

Cloud Platform

Sensors

Display of the data collected

abnormal health pattern

Normal health pattern

SMS Alert to user

3.2 HARDWARE , SOFTWARE DESIGNING AND  EXPERIMENTAL INVESTIGATION

For the construction of this device, the services were created on the IBM cloud platform, the Internet of things platform, Node red, are the main services that are used for the development. The nodes are configured in such a way that a fluctuation in the normal values of these health parameters would intimate the user about it through SMS.

4. FLOW CHART

Sensors that are attached to the wearable devices

Flows are created using Node red to receive the data from the device

HTTP requests are made to communicate with the mobile application

The user interface is designed to display the data

Configuration of the application to report an sms in case of abnormal state in the data collected.

SMS alerts are sent to the mobile when abnormal state detected

5. RESULT

A device that is wearable is capable of sensing the blood pressure, temperature and the pulse, this displays the data on the mobile application developed and alerts the user once there are abnormal states detected.

6.ADVANTAGES AND DISADVANTAGES

People are free to access it in just a touch, that keeps you notified about your health.

These save time and provides increased mobility.

7. APPLICATIONS

These have a wide range application in the medical fields.

8. CONCLUSION

This provides a better path for the users to know themselves better. It also reports any abnormal changes detected.

9. FUTURE SCOPE

When the whole world is busy with heavy workloads they aren’t concerned about going to the hospital and getting their regular checks done. This situation increases with respect to the days that pass, so such a device can be useful for the maintenance of better health at an individuals level.

10.APPENDIX

SOURCE CODE FOR NORMAL HEALTH PATTERN

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "jfxbyq"

deviceType = "RASPBERRYPI"

deviceId = "654321"

authMethod = "token"

authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data)

print(type(cmd.data))

i=cmd.data['command']

if i=='lighton':

print("light is on")

elif i=='lightoff':

print("light is off")

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:

temp= random.randint(96,99)

#print(temp)

bp = random.randint(80,120)

#Send Temperature, Bloodpressure & Pulse to IBM Watson

pul = random.randint(60,100)

data = { 'Temperature' : temp, 'Bloodpressure': bp, 'Pulse':pul }

#print (data)

def myOnPublishCallback():

print ("Published Temperature = %s F" % temp, "Bloodpressure = %s " % bp, "Pulse = %s " % pul, "to IBM Watson")

success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(2)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()

SOURCE CODE FOR ABNORMAL HEALTH PATTERN

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "jfxbyq"

deviceType = "RASPBERRYPI"

deviceId = "654321"

authMethod = "token"

authToken = "12345678"

# Initialize GPIO

def myCommandCallback(cmd):

print("Command received: %s" % cmd.data)

print(type(cmd.data))

i=cmd.data['command']

if i=='lighton':

print("light is on")

elif i=='lightoff':

print("light is off")

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:

temp= 101

#print(temp)

bp = random.randint(80,120)

#Send Temperature, Bloodpressure & Pulse to IBM Watson

pul = random.randint(60,100)

data = { 'Temperature' : temp, 'Bloodpressure': bp, 'Pulse':pul }

#print (data)

def myOnPublishCallback():

print ("Published Temperature = %s F" % temp, "Bloodpressure = %s " % bp, "Pulse = %s " % pul, "to IBM Watson")

success = deviceCli.publishEvent("DHT11", "json", data, qos=0, on\_publish=myOnPublishCallback)

if not success:

print("Not connected to IoTF")

time.sleep(2)

deviceCli.commandCallback = myCommandCallback

# Disconnect the device and application from the cloud

deviceCli.disconnect()

SCREENSHOTS OF FLOWS, MIT APP DISPLAY AND BLOCKS

  