**Personal Assitance For Independent Senior Citizens**

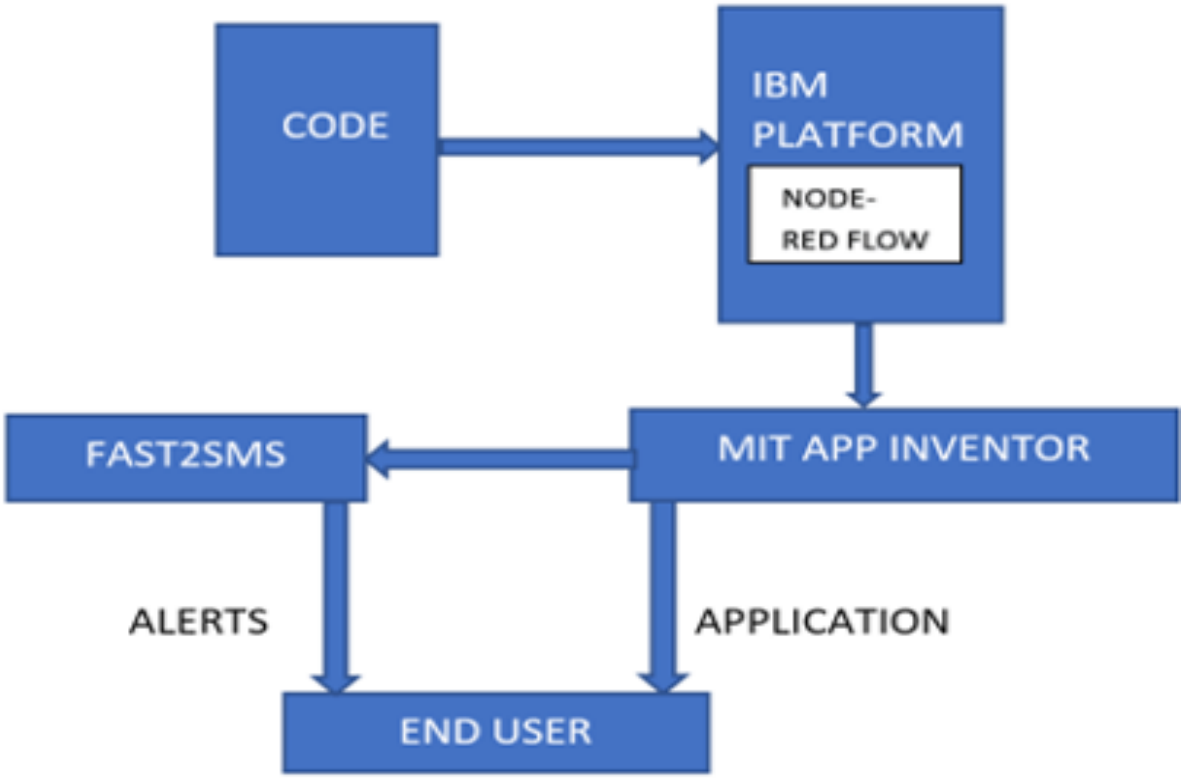
**Overview:** This project will guide you in developing a personal assistance for independent senior citizens using the IBM cloud services and MIT app inventer under IoT platform.

**Purpose:** To collect and store the information of the medicines of the user and thereby alerting the user to take medication from time to time without fail through an application.

**Existing problem:** Many old people forget about their medication either because of aging or memory related issues,which inturn affects their longterm health.

**Proposed solution:** Using the data we stored,we integrate the IBM NODE RED platform with MIT app inventor and Fast2Sms to send alerts to the user about the medicines.

**Block diagram:**



**Hardware-Software required:**

**IBM:** Acts as a platform for creating services like IoT,speech to text,text to speech etc.., and software applications like NODE RED and also for storing and subscribing data.

**NODE RED:**Software available within the IBM platform and helps us in using the MIT app inventor and also it has the UI option to display the data and required commands in the python.Nodes act as the back-bone here.

**Python(IDLE):** Helps us in writing the code for sending values to the IBM IoT platform and also for calling back them into the python shell.

**Fast2Sms:** It helps us in sending the alerts/messages to multiple users at same instance of time.

**MIT app inventor:** This helps us in building the model apk's/applications to simulate the data from IBM platform and also for sending alerts through Fast2Sms.Blocks acta as a back-bone in this platform.

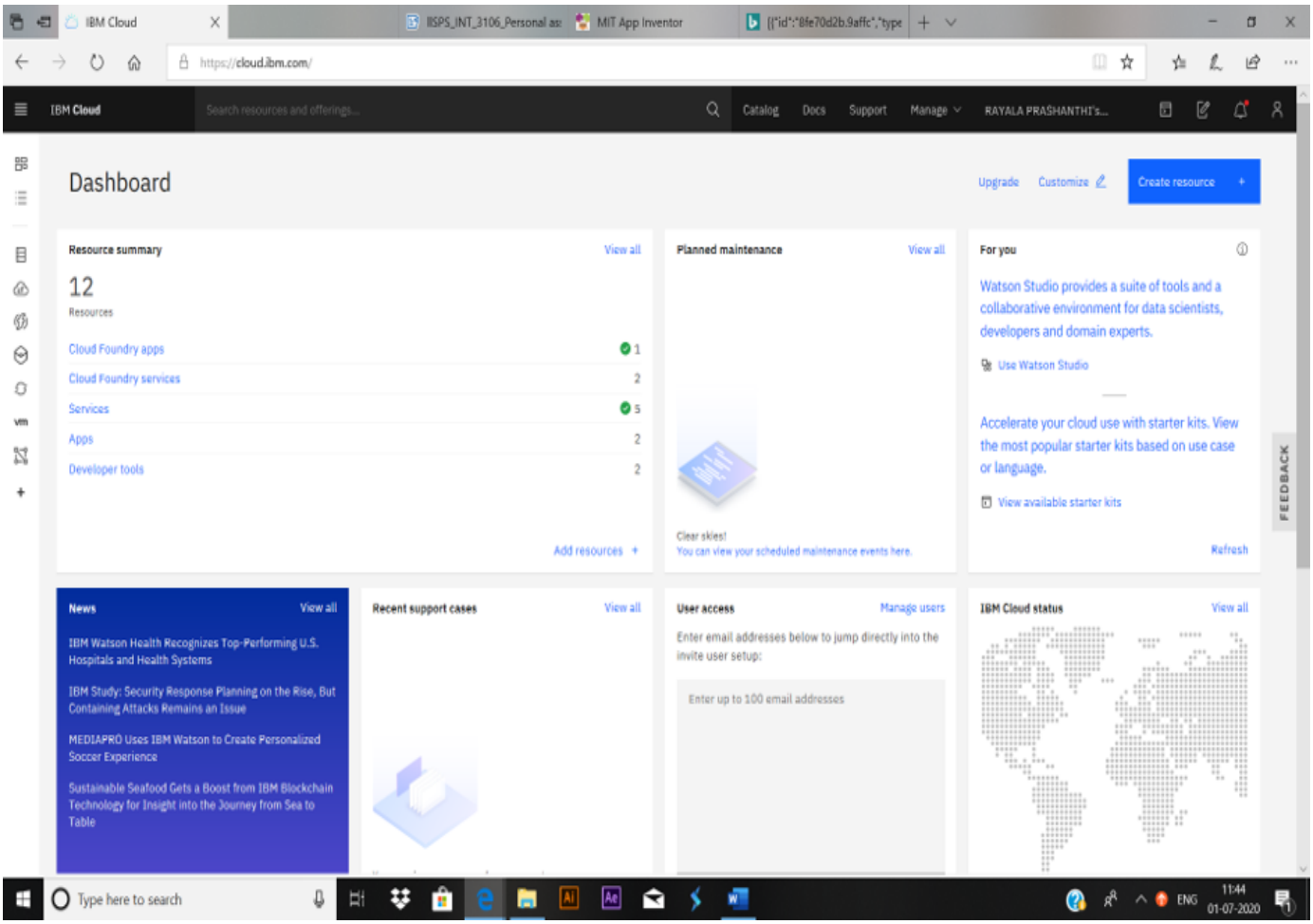
**Application:**It displays the data related to medication and alerts the user to take the medicine.

## EXPERIMENTAL INVESTIGATIONS AND THE PROJECT

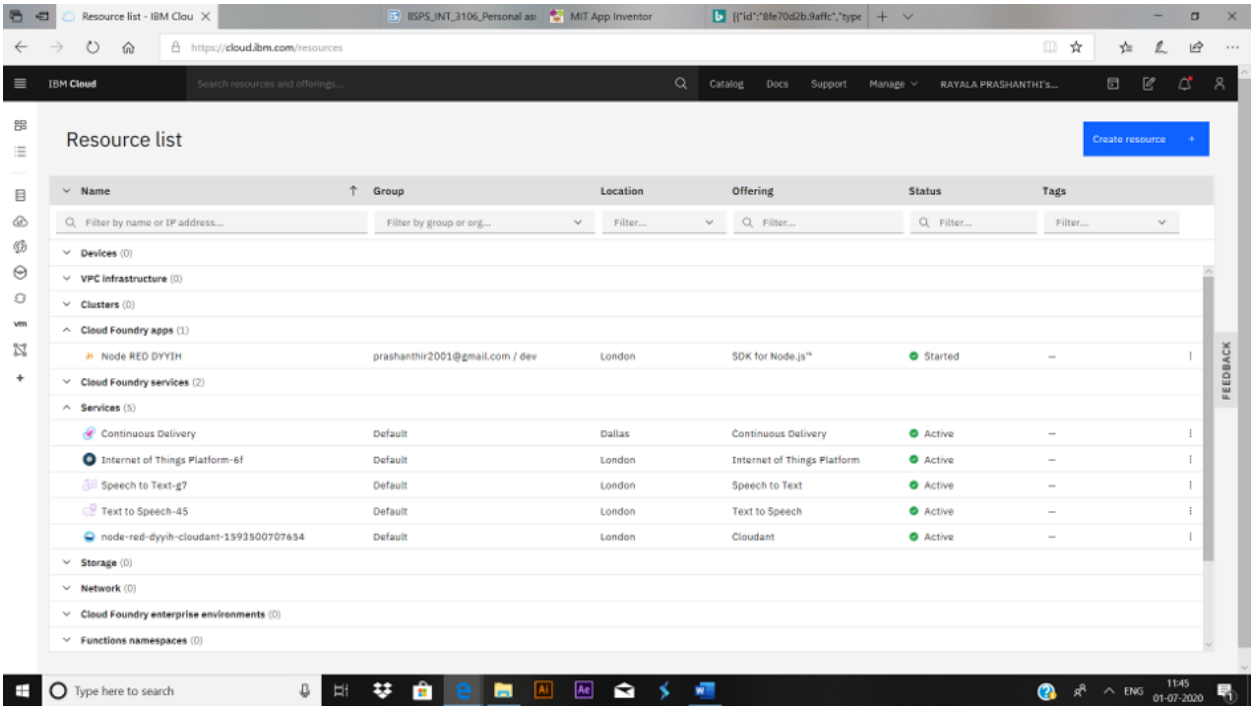
IBM is a platform which is used to create several services and softwares.It helps us to store the data and retrieve it whenever required.Initially we should create IBM IoT service, speech to text service and text to speech services in the IBM cloud platform. Using the python code, we need to send the details of time and the medicine to the IBM IoT service by providing the required credentials like device type, device ID, token and the organisation.This data can be seen in the device status in the IoT platform.This data is in the form of JSON.We should also give the text to speech and speech to text python codes in IDLE to understand what actually are those services.Credentials of speech to text and text to speech can be accessed through the services created earlier.NODE-RED is a software inside the IBM platform, which uses nodes to define and run a specified task/process.Initially the data from IBM IoT platform is viewed in the debug section of the node red platform.This can be done using the "IBM IoT IN NODE" and the debug node.We can also create some ui dashboards in the node red to print back the data and alerts into the python codes using buttons.Here, we used the "ALERTS" and "MEDICINES' buttons to get the warning and data into the python code

respectively.Buttons will be displayed in the url/ui page and clicking on them and running the code helps us to visualize data and alerts in the python.Now,an http "get" request is given for visualizing the medicine data into an url.We can now get a json data of medicines in the url using http response node.For understanding, we can also use text to speech nodes to let the node red platform speak out tablet names to just simulate how basically the text to speech service can be used in node red and in general applications.Now,using this url and the blocks provided in the MIT app inventor,we can build an application to see the time and medicine data and also send alerts to the user for using tablets using Fast2SMS Service.Finally, the end user is able to take medicines in time without any fail because of the data available in the application and the alerts we send.

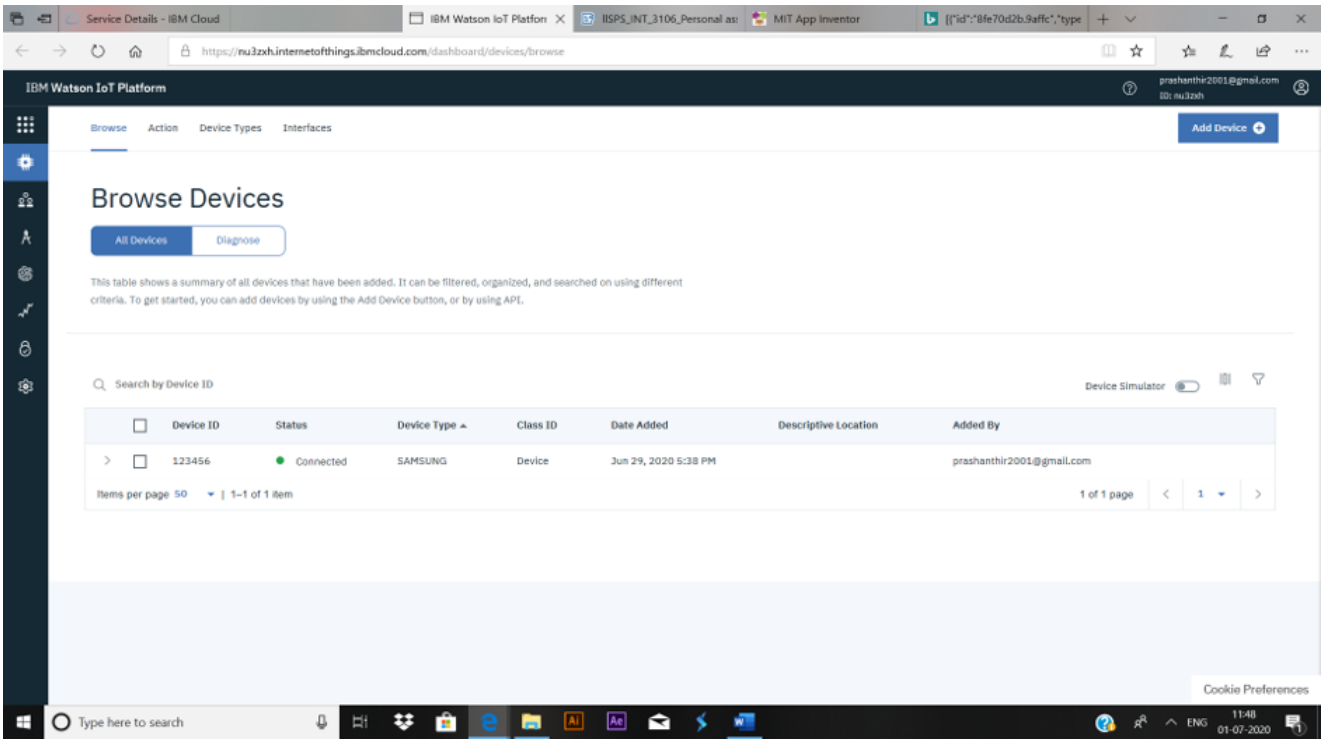
IBM CLOUD PLATFORM :

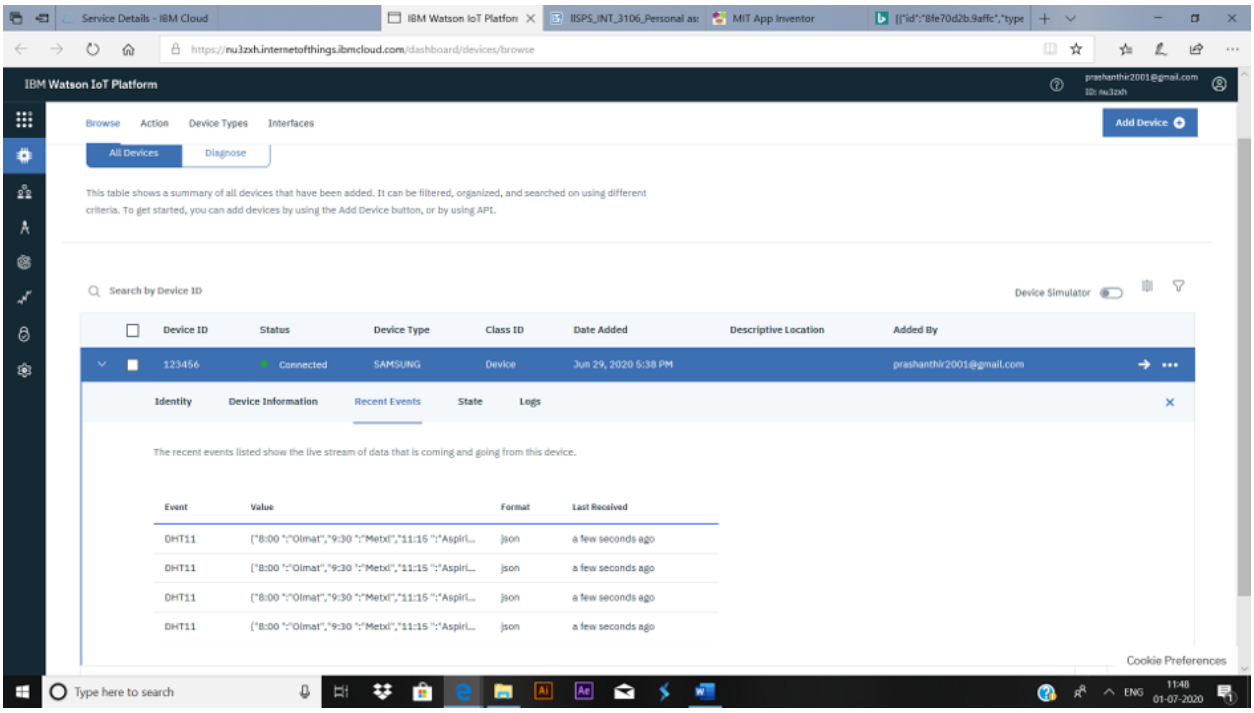


LIST OF SERVICES AND SOFTWARES USED IN IBM  CLOUD :

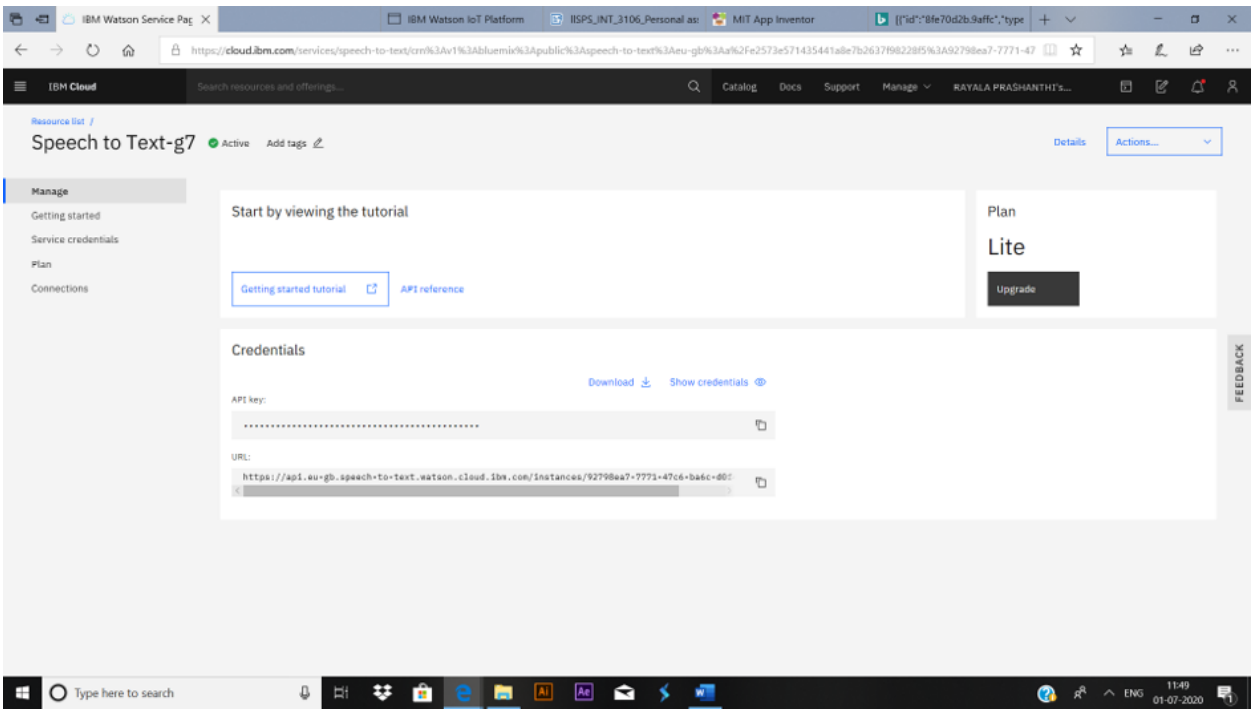



IOT PLATFORM :

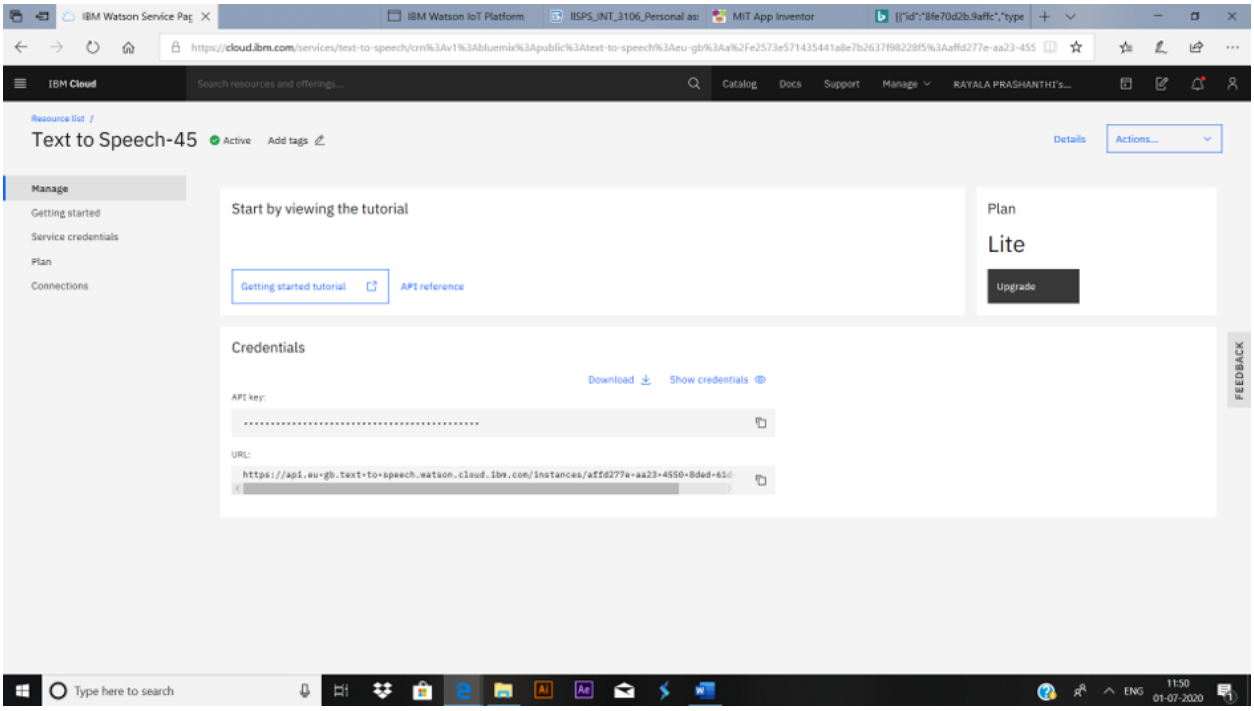




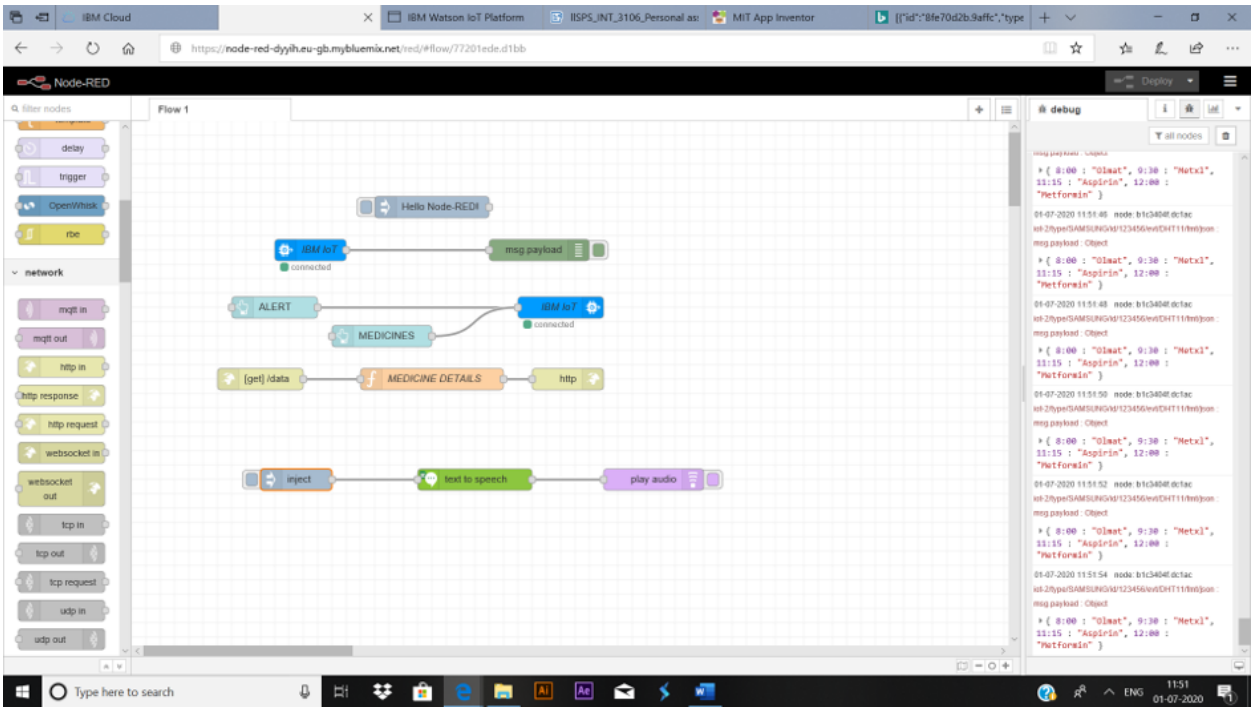

IBM SPEECH TO TEXT SERVICE :



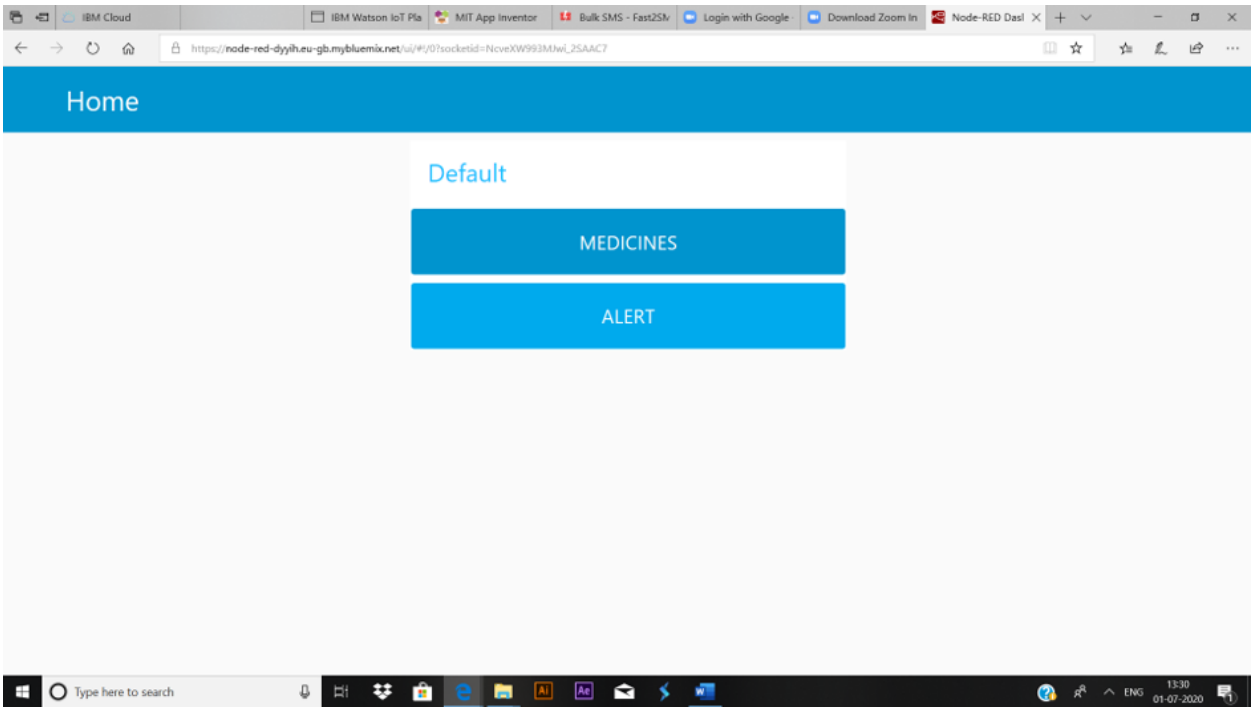
IBM TEXT TO SPEECH SERVICE :



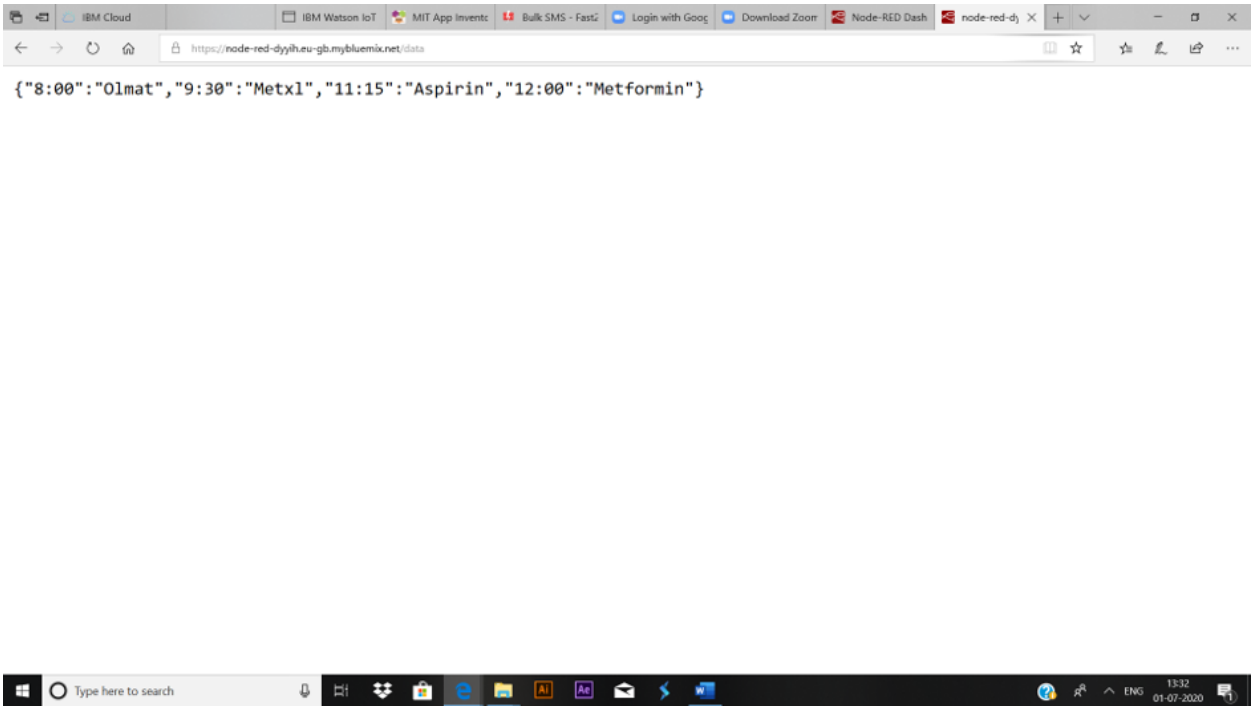

IBM NODE-RED :



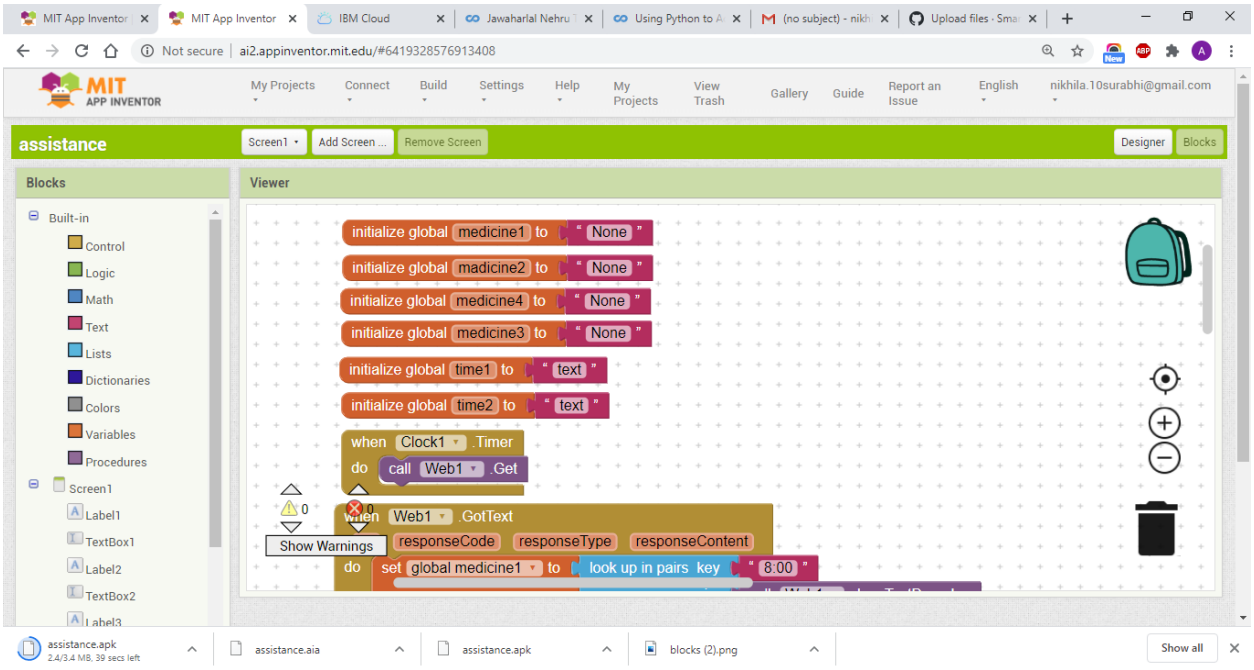
IBM NODE RED UI PLATFORM :

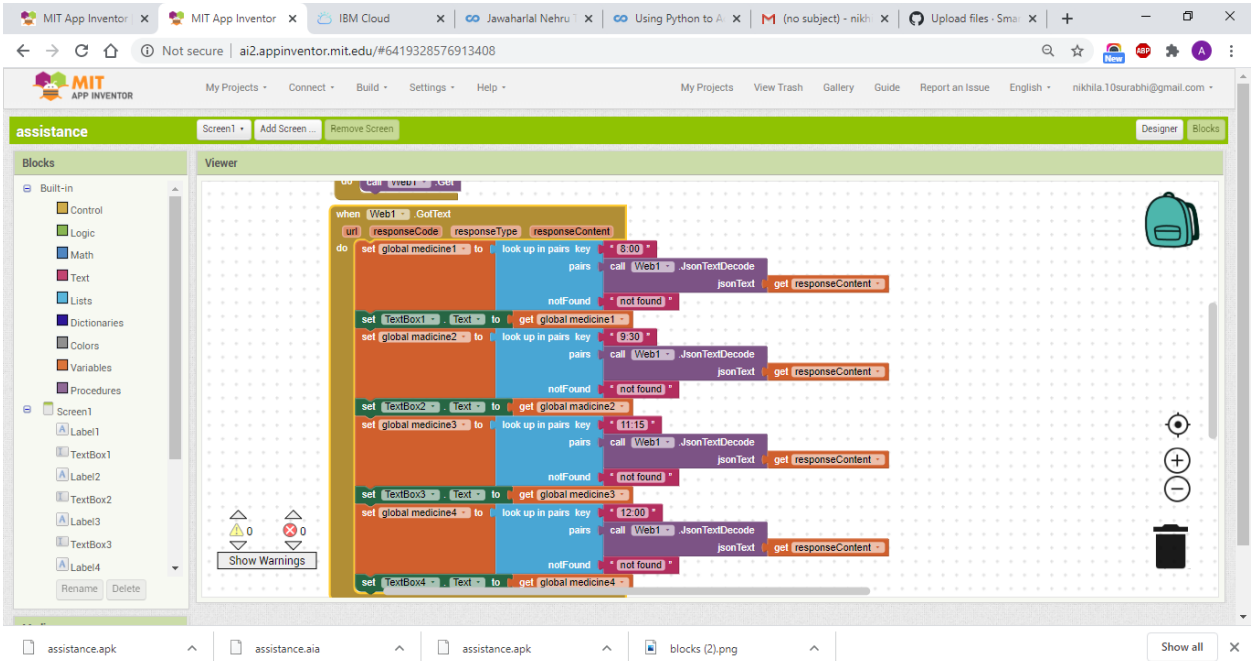


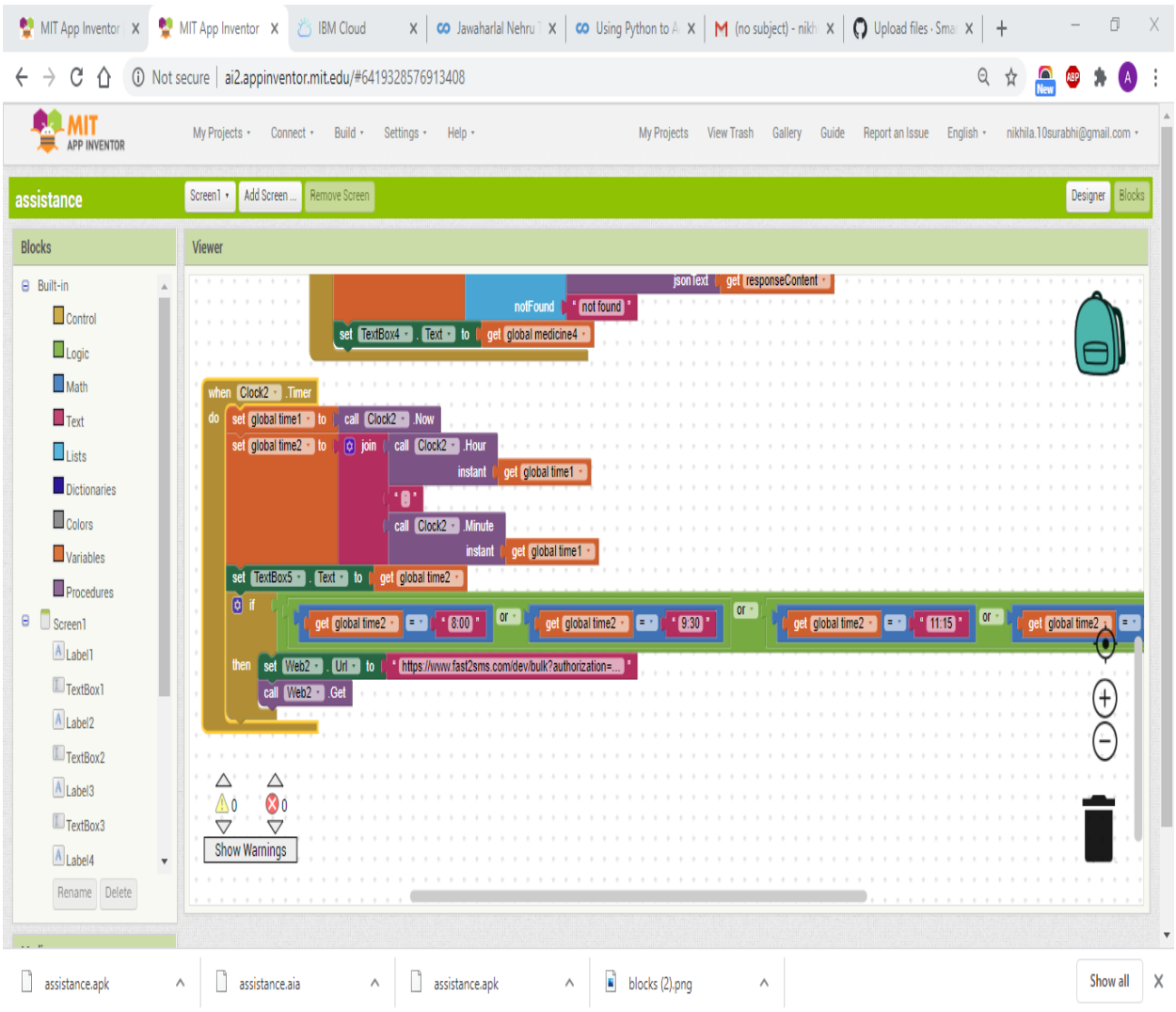

IBM NODE HTTP DATA WINDOW :

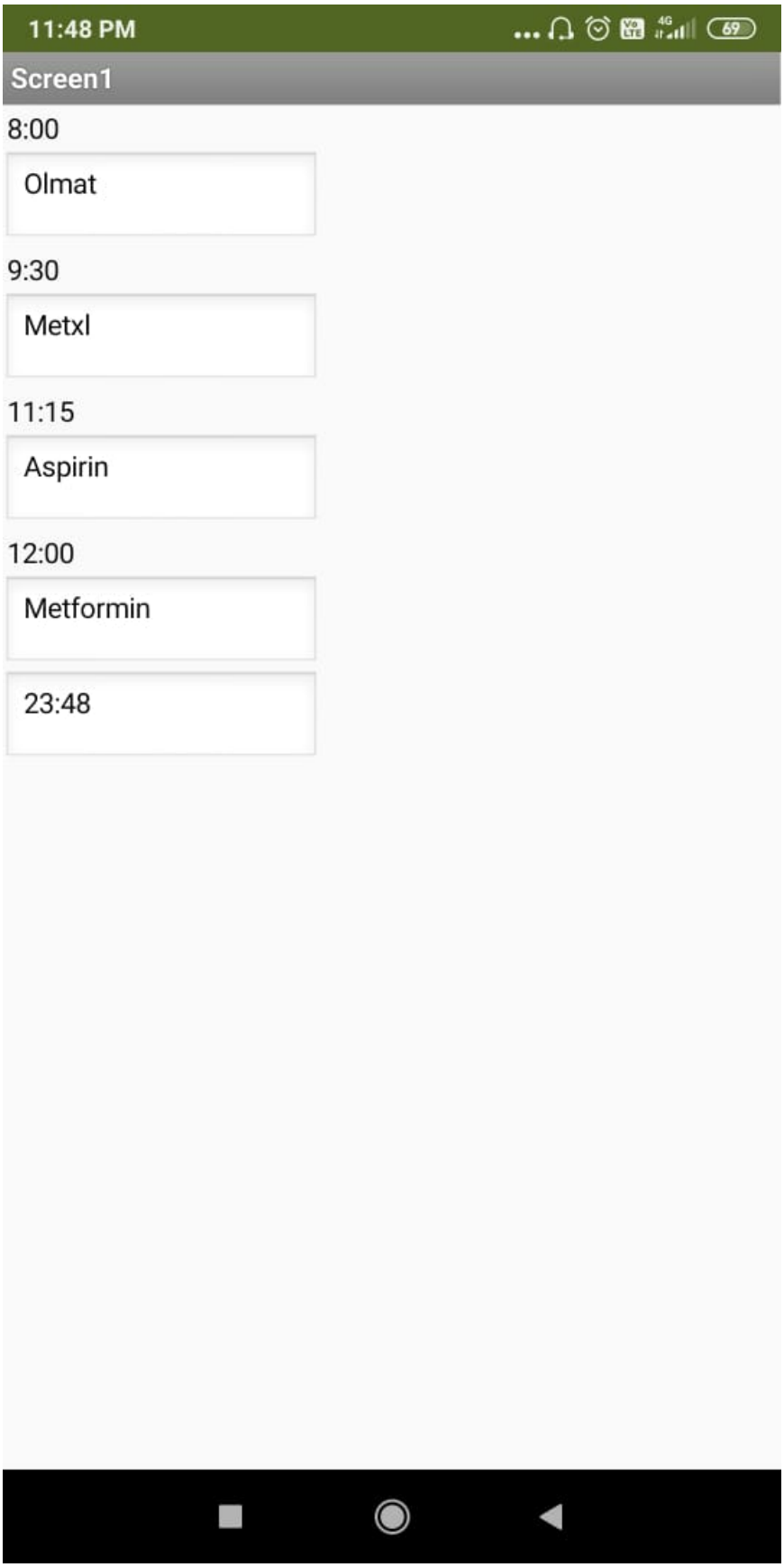


MIT APP INVENTOR :

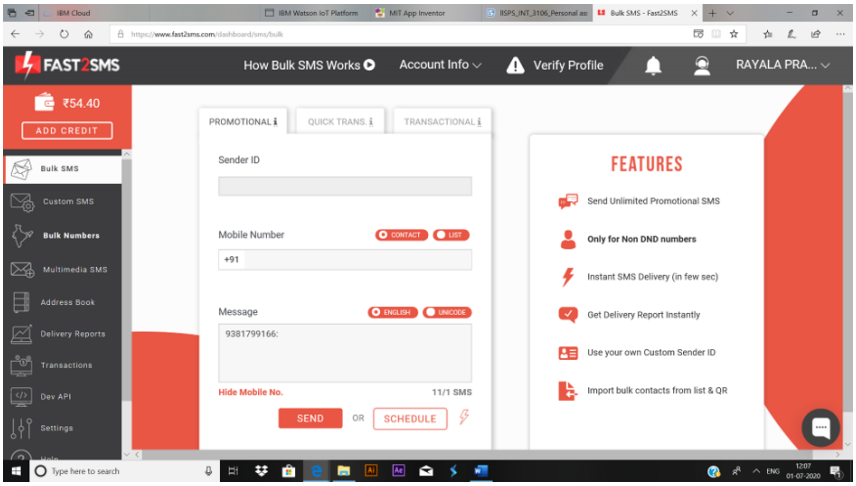


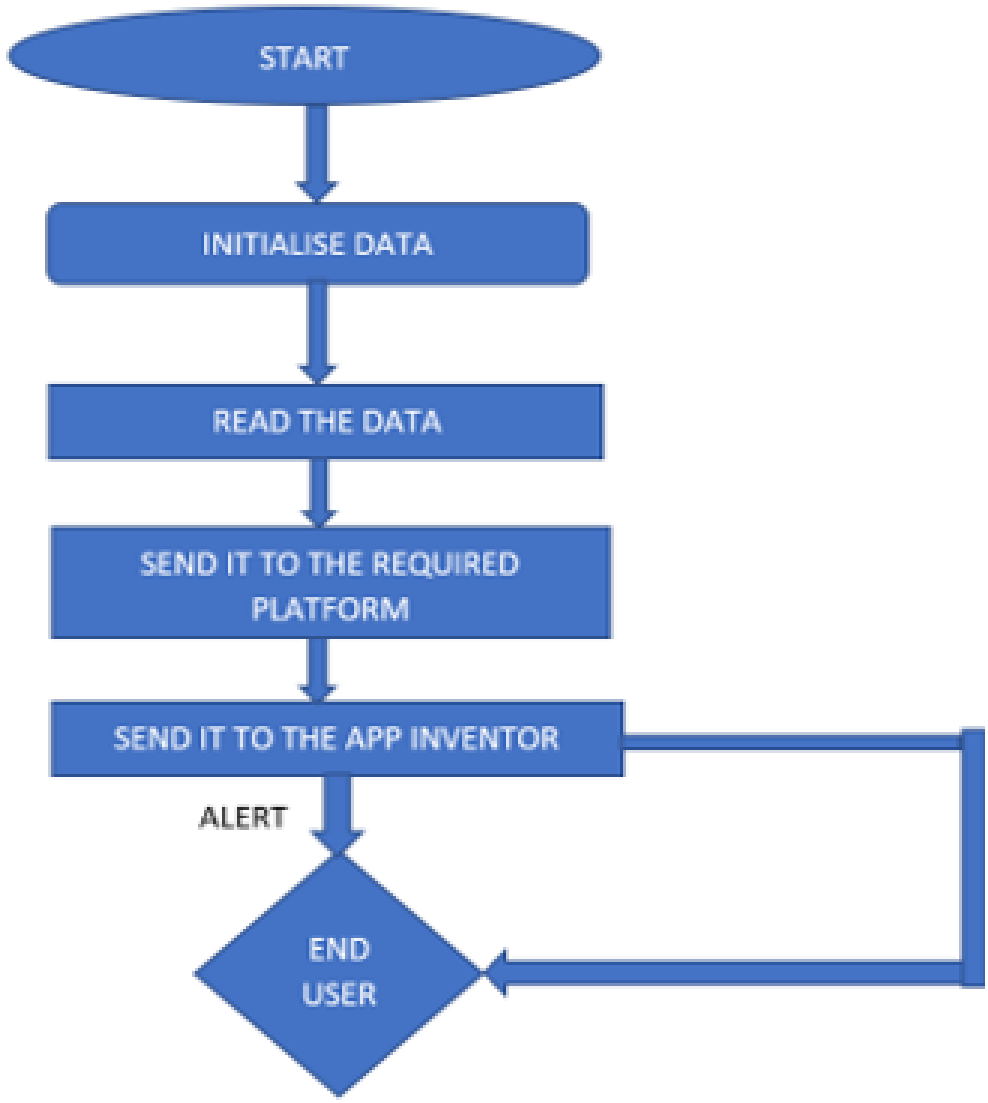

MIT App Inventor
IBM Cloud
Jawaharlal Nehru
Using Python to A
(no subject) - nikh
Upload files - Sma
Not secure
ai2.appinventor.mit.edu/#6419328576913408
MIT APP INVENTOR
My Projects
Connect
Build
Settings
Help
My Projects
View Trash
Gallery
Guide
Report an Issue
English
nikhila.10surabhi@gmail.com
assistance
Screen1
Add Screen ...
Remove Screen
Designer
Blocks
Blocks
Viewer
Built-in
Control
Logic
Math
Text
Lists
Dictionaries
Colors
Variables
Procedures
Screen1
Label1
TextBox1
Label2
TextBox2
Label3
TextBox3
Label4
Rename
Delete
when Clock2 .Timer
do set global time1 to call Clock2 .Now
set global time2 to join call Clock2 .Hour
instant get global time1
" : "
call Clock2 .Minute
instant get global time1
set TextBox5 . Text to get global time2
if get global time2 = " 8:00 " or get global time2 = " 9:30 " or get global time2 = " 11:15 " or get global time2 =
then set Web2 . Url to " https://www.fast2sms.com/dev/bulk?authorization=... "
call Web2 .Get
json text get responseContent
notFound " not found "
set TextBox4 . Text to get global medicine4
Show Warnings
assistance.apk
assistance.aia
assistance.apk
blocks (2).png
Show all


11:48 PM
Screen1
8:00
Olmat
9:30
Metxl
11:15
Aspirin
12:00
Metformin
23:48

FAST2SMS :




**FLOW CHART :**



**RESULT** :

Hence, using the data of the medicines provided, the timings of the medication will be provided to the user and alerts will be sent to the mobile using fast2sms.Thereby it ensures the safe medication and timely medication to the individual senior citizen.

**ADVANTAGES :**

>> Ensures the proper health of the elderly people.

>> It also helps the elderly users who have less memory power by creating alerts/remainders and makes them self dependent.

**DIS-ADVANTAGES** :

>> Internet availability or app working may be a problem in some of the remote areas.

>> Wrong feeding of the data affects the user's health.

**APPLICATIONS** :

>> In old age homes

>> For all the elderly people who either live individually or with their children.

>> Also for people with little memory problems.

**CONCLUSION :**

The Personal Assistance system ensures the proper health of the individual senior citzens and also for the people who need to take medication(and are sufferiing with memory problems) .Though there are some disadvantages with this system,it still has the ability to properly ensure proper care of senior citizens.

**FUTURE SCOPE**:

Elderly people play a very important role in the development of society.Though, they cannot contribute much financially, they play a major role by teaching ethical values and morals to their children and grand children.So,their well being will be prsent generation's well being.Hence,we need to understand their needs and serve them properly.Main need for any elderly citizen is "HEALTH".So,this "PERSONAL ASSISTANT FOR INDIVIDUAL SENIOR CITIZENS APPLICATION" has an ability of playing a major role in maintaining elderly people's health.Every person on ths earth, have a right to live. So,by adapting this system in a larger scale ensures the proper development of ethical values in future generations and in increasing the life span of elderly people and thereby decreasing the mortality rate.

**BIBLIOGRAPHY :**

Some of the websites like cloud.ibm.com,www.fast2sms.com and appinventor.mit.edu were used.

Also the readily available "PYTHON IDLE" was used for the coding part in the project.

**PROJECT DELIVERABLES** :

WEB APPLICATION
PROJECT REPORT

PROJECT VIDEO

***"RESPECT AND SERVE ELDER PEOPLE WITH ALL FACILITIES TO ENSURE YOUR ETHICS"***

**APPENDIX :**

**CODE FOR TEXT TO SPEECH :**

```
from ibm_watson import TextToSpeechV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator

from playsound import playsound



authenticator =
IAMAuthenticator('yci1-exDJ1UrPVWahNl478_0qq3G_9D5XUagkYUqjfCj')
text_to_speech = TextToSpeechV1(
    authenticator=authenticator
)

text_to_speech.set_service_url('https://api.eu-gb.text-to-speech.watson.cloud.ibm.com/
instances/f4fc8363-62dc-4947-9293-9822fdb56590')

with open('medicine.mp3', 'wb') as audio_file:
    audio_file.write(
        text_to_speech.synthesize(
            'Olmat ',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
        ).get_result().content)

    audio_file.write(
        text_to_speech.synthesize(
            '  Met xl',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
        ).get_result().content)
    audio_file.write(
        text_to_speech.synthesize(
```

```
            ' Aspirin  ',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
        ).get_result().content)
    audio_file.write(
        text_to_speech.synthesize(
            '  Metformin  ',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
        ).get_result().content)
    audio_file.write(
        text_to_speech.synthesize(
            'Linagliptin',
            voice='en-US_AllisonV3Voice',
            accept='audio/mp3'
        ).get_result().content)

playsound('medicine.mp3')
```

**CODE FOR SPEECH TO TEXT:**

```
import json
from os.path import join, dirname
from ibm_watson import SpeechToTextV1
from ibm_cloud_sdk_core.authenticators import IAMAuthenticator

authenticator =
IAMAuthenticator('DhIN3LWVXzaNTbEZ1X2Shm12o12-bxUSNZgCQ3QbfVPI')
speech_to_text = SpeechToTextV1(
    authenticator=authenticator
)
```

```
speech_to_text.set_service_url('https://api.eu-gb.speech-to-text.watson.cloud.ibm.com/
instances/d8ae2efd-eedd-4749-a50d-e7a35b49e52e')

with open(join(dirname(__file__), './.', 'medicine.mp3'),
            'rb') as audio_file:
   speech_recognition_results = speech_to_text.recognize(
      audio=audio_file,
      content_type='audio/mp3',

   ).get_result()
print(json.dumps(speech_recognition_results, indent=2))
```

**FINAL CODE:**

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "nu3zxh"
deviceType = "SAMSUNG"
deviceId = "123456"
authMethod = "token"
authToken = "99121499"

# Initialize GPIO

def myCommandCallback(cmd):
      if cmd.data:
            print("Time to take medicine")#for commands

try:
```

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

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()#connect client to platform

while True:



        data = { '8:00 ' : 'Olmat' , '9:30 ' : 'Metxl' , '11:15 ': 'Aspirin' , '12:00 ': 'Metformin' }
        #print (data)
        def myOnPublishCallback():
            print ('8:00  : Olmat' , '9:30  : Metxl' , '11:15 : Aspirin', '12:00 : Metformin' )
        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()
```

**_____THE END_____**