

# PROJECT DEVELOPMENT PHASE-SPRINT 4

Date	19 <sup>th</sup> November 2022
Team ID	PNT2022TMID47535
Project Name	Personal assistance for Seniors who are self-reliant
Delivery	Sprint-4

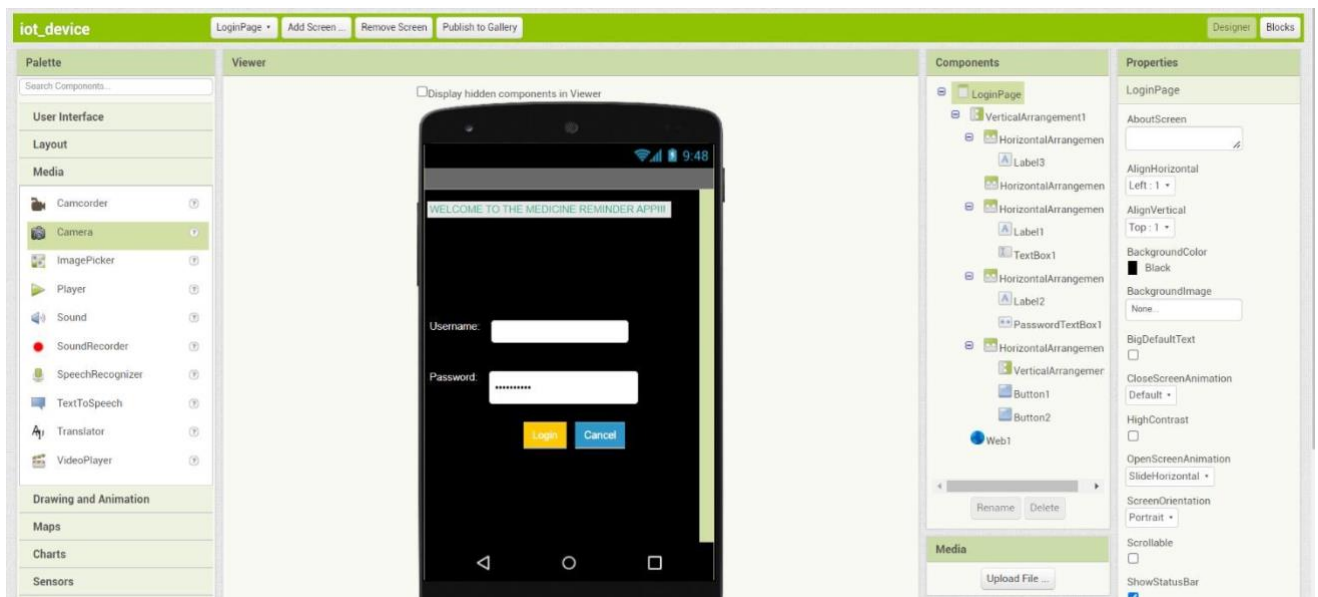
**SPRINT 3** -*Creating a Mobile application using MIT App inventor to add medicine and monitor the medicines.*

## 1.APP SETUP

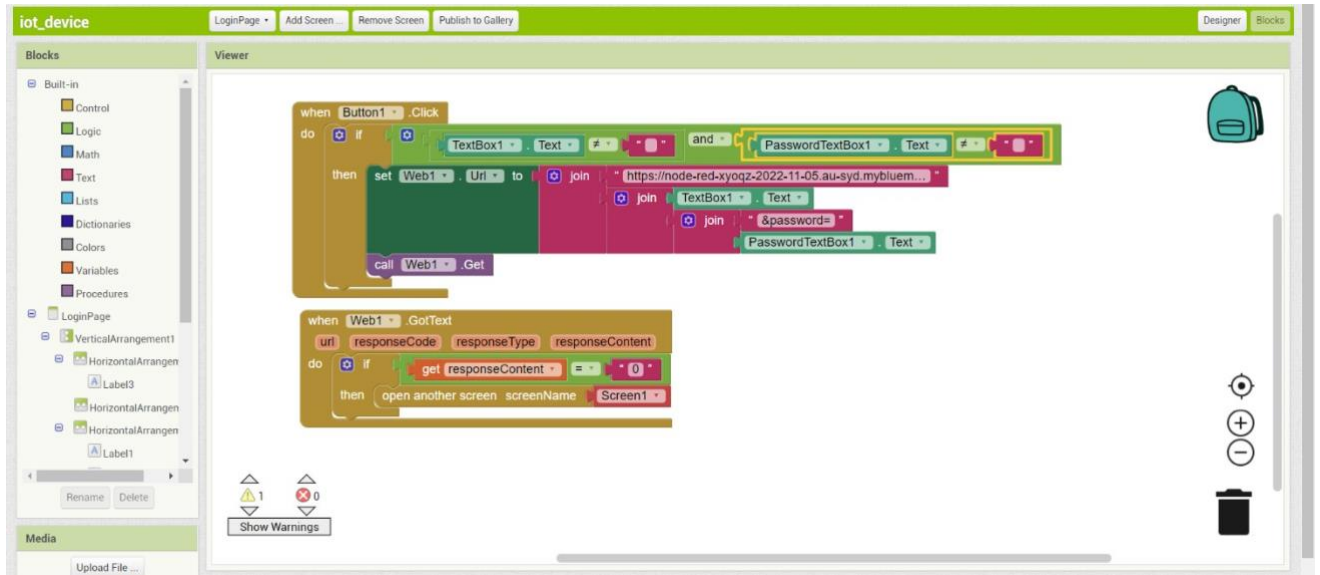
The app consists of two screen.First screen is the login page and the second is the medicine details page.

SCREEN 1:

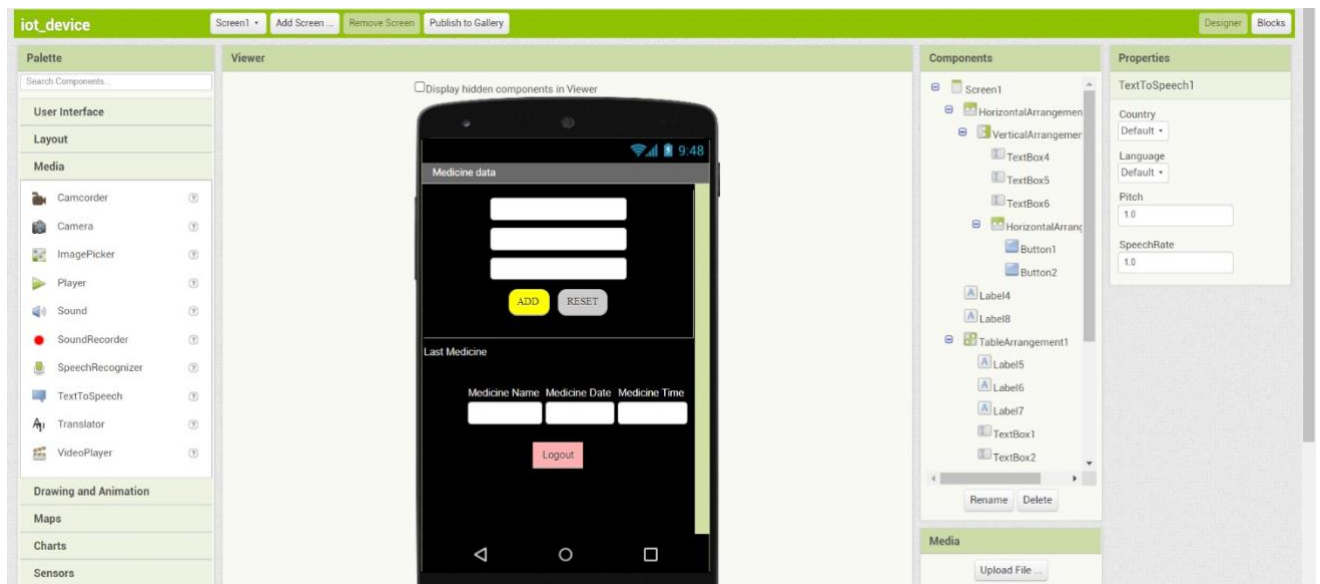
LOGIN SCREEN(DESIGN):



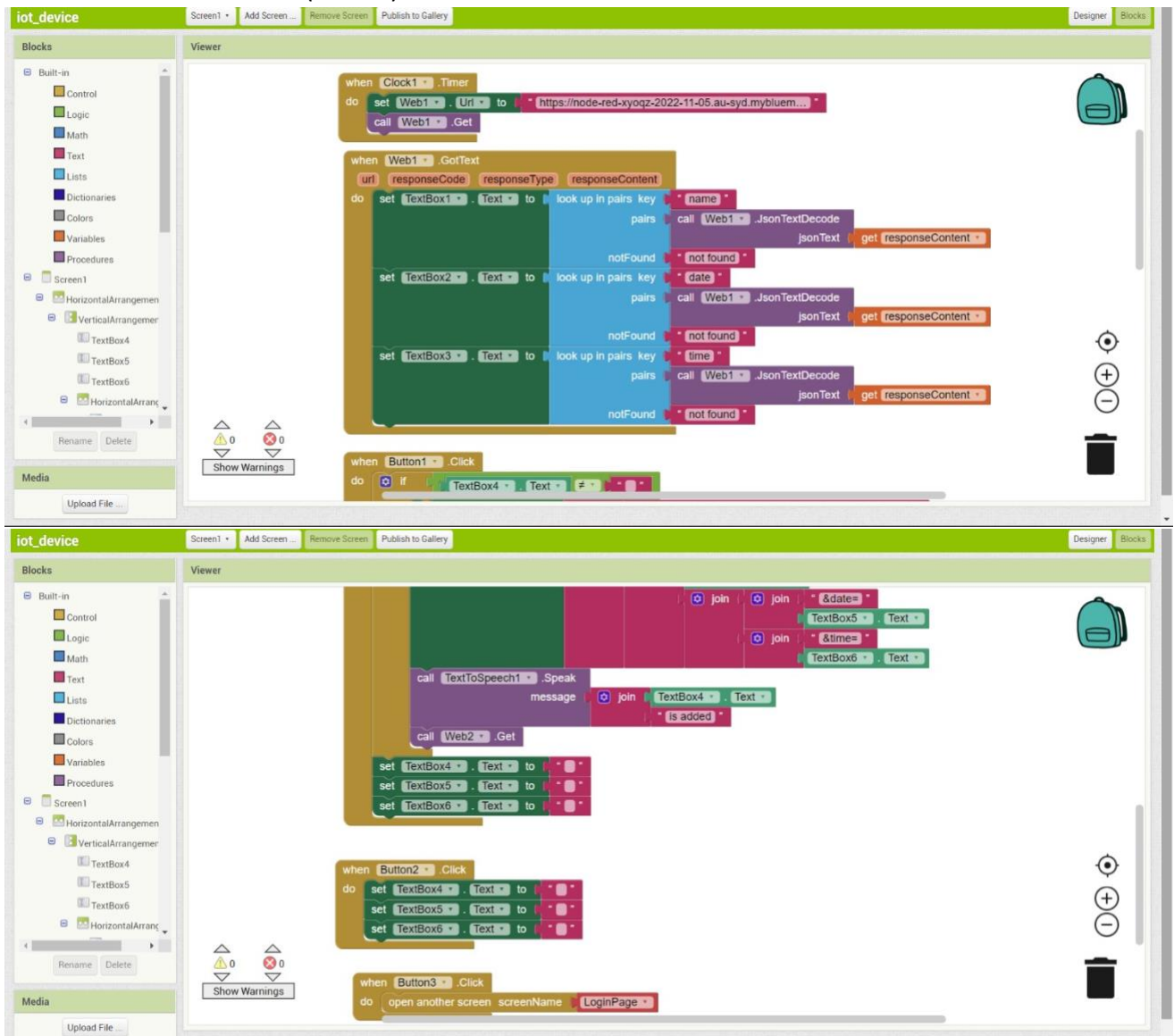
## LOGIN SCREEN(BLOCK):



## SCREEN 2: MEDICINE SCREEN(DSIGN):

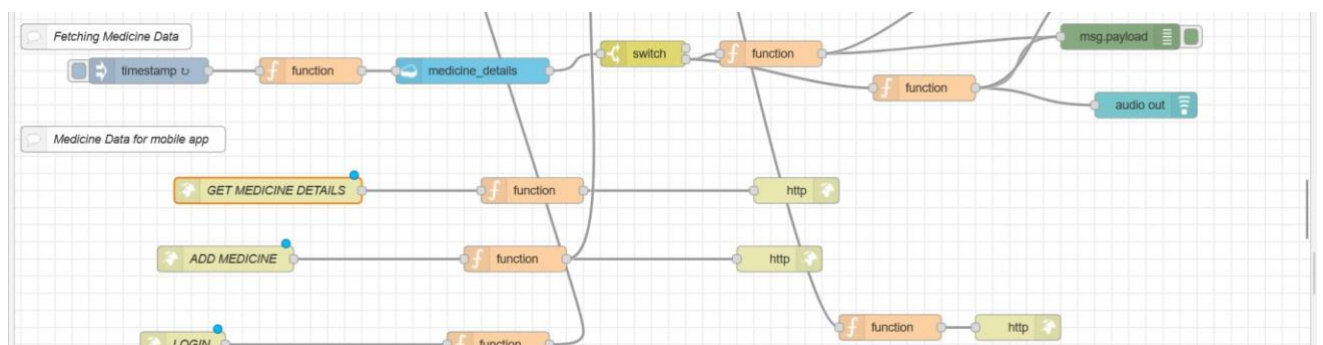


## MEDICINE SCREEN(BLOCK):



## 2. NODE RED FLOW

The flow has http get requests for logging in, adding medicine details and fetching medicine details.



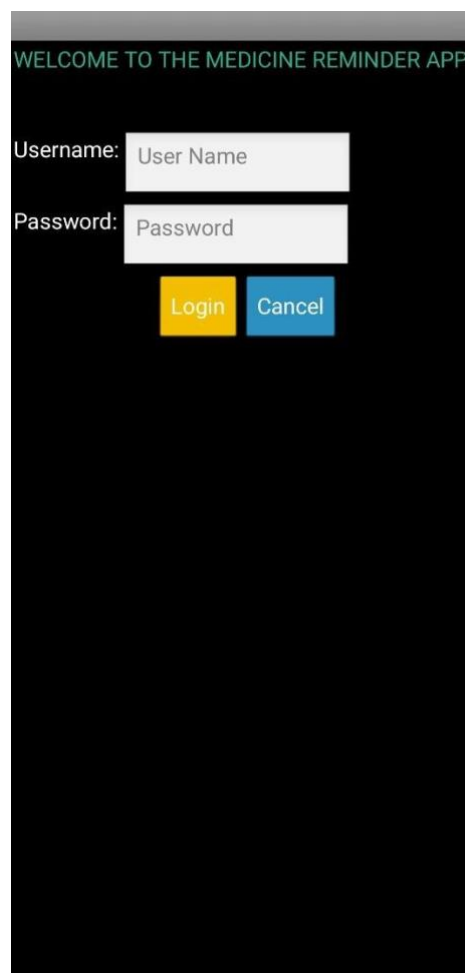
When the user logs in through the mobile application, the username and password is passed from the login url as json parameters which is then verified in the database.

When medicine details are added from the mobile application, the added details is passed using the add\_medicine url which contains medicine name, date and time as parameters in json format.

And the latest medicine that was taken is displayed in the mobile application using the med\_details url.

**3. Mobile application** The user can login into the mobile application and add medicine details or check the last medicine that was consumed.

*Login Screen:*



The image shows a mobile application login screen with a black background. At the top, there is a green header bar with the text "WELCOME TO THE MEDICINE REMINDER APP!". Below the header, there are two input fields: "Username:" with a placeholder "User Name" and "Password:" with a placeholder "Password". Below the password field, there are two buttons: a yellow "Login" button and a blue "Cancel" button.

*Medicine Screen:*

Medicine data

Paracetamol

2022-11-17

00:52

ADD

RESET

Last Medicine

Medicine Name

Medicine Date

Medicine Time

citrizen

2022-11-17

23:45

Logout

Medicine data

Enter Medicine Nam

Date(YYYY-MM-DD)

Time(HH:MM)

ADD

RESET

Last Medicine

Medicine Name

Medicine Date

Medicine Time

citrizen

2022-11-17

23:45

Logout

The screenshot displays a Node-RED interface for an IoT project. The main workspace shows a visual representation of the hardware: an ESP32 microcontroller board is connected to an LED (red wire to VCC, green to GND), a buzzer (green wire to VCC, black to GND), and an LCD display (red to VCC, green to GND, and a blue wire to a data pin). The LCD screen shows the text "It's time for PARACETAMOL". Below the workspace, a terminal window provides a log of MQTT communications:

```
Reconnecting client to byl8wl.messaging.internetofthings.ibmcloud.com
iot-2/cmd/test/fmt/string
subscribe to cmd OK

callback invoked for topic: iot-2/cmd/test/fmt/string
PARACETAMOLdata: PARACETAMOL
PARACETAMOL
```

*The database is also updated when the user enters the data into from the mobile application.*

medicine\_details

Document ID

Options

{ } JSON

All Documents

Query

Permissions

Changes

Design Documents

Table

Metadata

{ } JSON

Create Document

_id	date	name	time
2022-11-13 17:05	2022-11-13	PARACETAMOL	17:05
2022-11-13 17:24	2022-11-13	Amoxylin	17:24
2022-11-13 17:30	2022-11-13	Amoxylin	17:30
2022-11-13 17:32	2022-11-13	Paracetamol	17:32
2022-11-13 17:40	2022-11-13	Dolo 360	17:40
2022-11-13 18:55	2022-11-13	Amoxylin	18:55
abhi 2022-11-14 18:35	2022-11-14	Amoxylin	18:35
abhi 2022-11-14 18:39	2022-11-14	Dolo 365	18:39
abhi 2022-11-14 18:41	2022-11-14	Dolo 360	18:41
swathi 2022-11-14 18:18	2022-11-14	Amoxylin	18:18
swathi 2022-11-14 18:32	2022-11-14	Amoxylin	18:32
swathi 2022-11-14 18:34	2022-11-14	Amoxylin	18:34
swathi 2022-11-14 18:03	2022-11-14	Paracetamol	18:03

#### 4. TEXT TO SPEECH SERVICE

The text to speech service from IBM Watson IOT platform is used to give voice commands to the user. This feature is implemented using PYTHON IDE .It is connected to the IBM cloud and when command arrives it converts the medicine name that was sent into speech so as to give command to the user.

**CODE:**

```
import time import sys import ibmiotf import ibmiotf.device import
random from ibm_watson import TextToSpeechV1 from
ibm_cloud_sdk_core.authenticators import IAMAuthenticator

#Provide your IBM Watson Device Credentials organization = "by18wl" # repalce it with
organization ID deviceType = "IOT_DEVICE" #replace it with device type deviceId =
"12345" #repalce with device id authMethod = "token" authToken = "123456789"#repalce
with token authenticator =
IAMAuthenticator('0HE1EXoPggPLiwdq6jbLkw7qJuBkSilP8TYOvNK7XXpk') text_to_speech
= TextToSpeechV1( authenticator=authenticator
)

text_to_speech.set_service_url('https://api.eu-de.text-to-
speech.watson.cloud.ibm.com/instances/e24ca6f6-496b-463b-b56f-4a0e79e76362')

def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data)
    if cmd.data['n']!= 'undefined':
        print(cmd.data['n'])          x="its time to take
"+ cmd.data['n']          with open('hello_world.wav',
'wb') as audio_file:
```

```
audio_file.write(text_to_speech.synthesize(x,voice='enUS_AllisonV3Voice',accept='audio/wav').  
get_result().content)
```

```
else:
```

```
    print("LIGHT 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()
```

```
deviceCli.connect()
```

```
while True:
```

```
    #T=random.randint(0,100);
```

```
    #H=random.randint(0,100);
```

```
    #Send Temperature & Humidity to IBM Watson
```

```
    #data = { 'temperature' : T, 'humidity': H }
```

```
    #print data
```

```
    #def myOnPublishCallback():
```

```
    # print ("Published Temperature = %s C" % T, "Humidity = %s %" % H, "to IBM Watson")
```

```
    #success = deviceCli.publishEvent("event", "json", data, qos=0,  
on_publish=myOnPublishCallback)
```

```
    #if not success:
```



```
# print("Not connected to IoT")  
time.sleep(1)
```

```
deviceCli.commandCallback = myCommandCallback
```

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

*“Thus the Personal Assistance for Seniors Who Are Self-Reliant was successfully developed. It makes use of node-red for web UI and adding details to cloudant and it is also responsible for issuing command to the IBM IOT Watson platform. IBM IOT Watson platform gets the command from the node red flow when medicine has to be taken and sends it to the IOT device which signals when the medicine time arrives. This communication is done using MQTT and HTTP protocol. IBM text to speech service is also used to orally tell the tablet name to the user. Finally, the mobile application developed with the help of MIT App inventor is used for adding and monitoring the medicines which is user friendly and convenient. ”*

**Future improvement:** *IOT DEVICE can consist of sensors to monitor number of medicines and separate racks for each medicine and provide that medicine when the time arrives and also speech to text service to recognize the user voice rather than using push buttons to snooze or turn off the alarm.*