Xamarin application to Read Sensor Data

1. Project Description :

The motivation of the project is to demonstrate the use of xamarin application to monitor various IOT devices connected to raspberry pi. Here, backend API is written in python-flask and client app is written in xamarin.

Xamarin makes it easy to develop cross platform applications. Thus, the application developed using xamarin is compatible with Android, IOS and Windows phone.

2. Github Repository:

The repository contains all code containing Xamarin, Azure Middleware Server and RaspberryPi Client Server Code and documentation.

URL : https://github.com/DhyeyMoliya/XamarinRasPiApp

3. Backend Technology :

The backend code is written in python-flask. It is python mini-framework, and supports APIs over HTTP. Thus we have developed a small API as an interface to control various IOT devices. The developed flask server is hosted on Azure. Following are the route defined on flask server.

1. To check if the server is reachable or not :
    a. URL :- http://raspi.azurewebsites.net/api/check
    b. Type :- GET
    c. Return value :- {“error” : False}

2. Get Temperature:
    a. URL :- http://raspi.azurewebsites.net/api/temperature
    b. Type :- GET
    c. Return value :- {“timestamp”: CUR_DT_TIME, “temperature”: CUR_TEMP}

3. Set Temperature :

   a. URL :- http://raspi.azurewebsites.net/api/temperature
   b. Type :- POST
   c. Body :- {“temperature”: CUR_TEMP}
   d. Return value :- {“timestamp”: CUR_DT_TIME, “temperature”: CUR_TEMP}

4. Get Humidity:
   a. URL :- http://raspi.azurewebsites.net/api/humidity
   b. Type :- GET
   c. Return value :- {“timestamp”: CUR_DT_TIME, “humidity”: CUR_TEMP}

5. Set Humidity:
   a. URL :- http://raspi.azurewebsites.net/api/humidity
   b. Type :- POST
   c. Body :- {“temperature”: CUR_TEMP}
   d. Return value :- {“timestamp”: CUR_DT_TIME, “humidity”: CUR_TEMP}

4. Frontend Technology :

Xamarin is used for client application. It consists of mainly two parts.

1. Main page :

   The main page requires two main inputs. First is IP address of the server and second is PORT on which the server is listening.

2. Control page :

   The control page consists of 2 main sections, for Ultrasonic Sensor and Temperature Sensor. Each section consists of a button to refresh and get current values from Raspberry Pi.

5. Raspberry Pi Code :

Here, A simple code to monitor sensor data and update it to server using httpclient module in python is used. The file is present at “~/Desktop” with filename “XamarinAppClient.py”

6. Components & Pin configuration:

To interface with sensors and calculate we need following sensors and components with appropriate libraries.

1. DHT11 Temperature and Humidity Sensor X 1

To interface with temperature sensor we have used open source library from github :

To install Flask Server Library:

```
# pip install Flask
```

PIN Configuration for Temperature Sensor:

1. VCC : Pin-2 (5V)
2. GND : Pin-6
3. DQ : Pin-7 (GPIO-4)