

**College of Engineering**

**Department of computer Engineering**

**OpenUAE Competition**

**Official Report**

## **Title of the Project:** IoT- Based Smart Utility Meter

**Team Members:** Mais Haj Hassan, Mohammad Abdelsalam, Mustapha Ezzeddine and Mohannad Baseet

**Faculty Members:** Dr. Abdulrahman Al-Ali & Dr. Taha Landolsi

**Table of Contents**

Problem Statement**5**

Proposed Solution**6**

Why Raspberry Pi ?**7**

Software and Hardware System Requirements **8**

Software Requirements 8

Functional Requirements9

Non-Functional Requirements9

Hardware Requirements 10

Hardware Equipment Figures11

Block Diagram & State Diagram**13**

Block Diagram13

State Diagram14

Implementation**15**

Prototype Description15

Coding 16

Python16

HTML19

Home Page19

Water and Gas Page21

Power Page24

Consumption Page26

Future work**31**

References**32**

# List of Figures

[**Figure [1].** Utility Manual Prosses 5](#_Toc417902838)

[**Figure [2].** The expected number of IoT devices up to the year 2020 6](#_Toc417902839)

[**Figure [3].** Gas Meter. 11](#_Toc417902840)

[**Figure [4].** Water Meter (Top View). 11](#_Toc417902840)

[**Figure [5].** Water Meter (Side View). 11](#_Toc417902840)

[**Figure [6].**Raspbery Pi 3. 12](#_Toc417902840)

[**Figure [7].** ADC. 12](#_Toc417902840)

[**Figure [8].** ADC(Connections). 12](#_Toc417902840)

[**Figure [9].** Block Diagram 1. 13](#_Toc417902840)

[**Figure [10]** Block Diagram 2. 14](#_Toc417902840)

[**Figure [11].** State Diagram. 15](#_Toc417902840)

[**Figure [12].**Home HTML Page. 21](#_Toc417902840)

[**Figure [13].** Water and Gas HTML Page. 24](#_Toc417902840)

[**Figure [14].** Power HTML Page. 26](#_Toc417902840)

[**Figure [15].** Consumptio HTML Page. 30](#_Toc417902840)

[**Figure [16].**Connections. 30](#_Toc417902840)

[**Figure [17].**Google map markers. 31](#_Toc417902840)

# List of Tables

[**Table [1].** Comparison between a typical Arduino and the Raspberry Pi 3 7](#_Toc417902857)

[**Table[2].**Hardware Requirnments 10](#_Toc417902858)

# Problem Statement

We have reached an era where a lot of devices are becoming smart nowadays. A smart device – usually an electronic device – is one capable of connecting with other devices via wireless protocols such as Wi-Fi, 4G, or Bluetooth. One field which desperately needs an upgrade to the smart devices category is the field of utility meters. Utility meters are the meters installed in residential areas and in individual houses in order to measure and record the consumption of the utilities used up by the house. The three main types of meters for the three utilities are: a water meter, a gas meter, and a power meter.

The traditional setup for the utility meters dictates that the three different meters are installed in three different locations depending on the utility in question. This setup has a whole range of problems arising from the fact that a utility labor has to visit each meter for every house, note down the consumptions for each utility respectively, and calculate the bill accordingly. This process is illustrated in the figure [1] below.

Figure [1]

# Proposed Solution

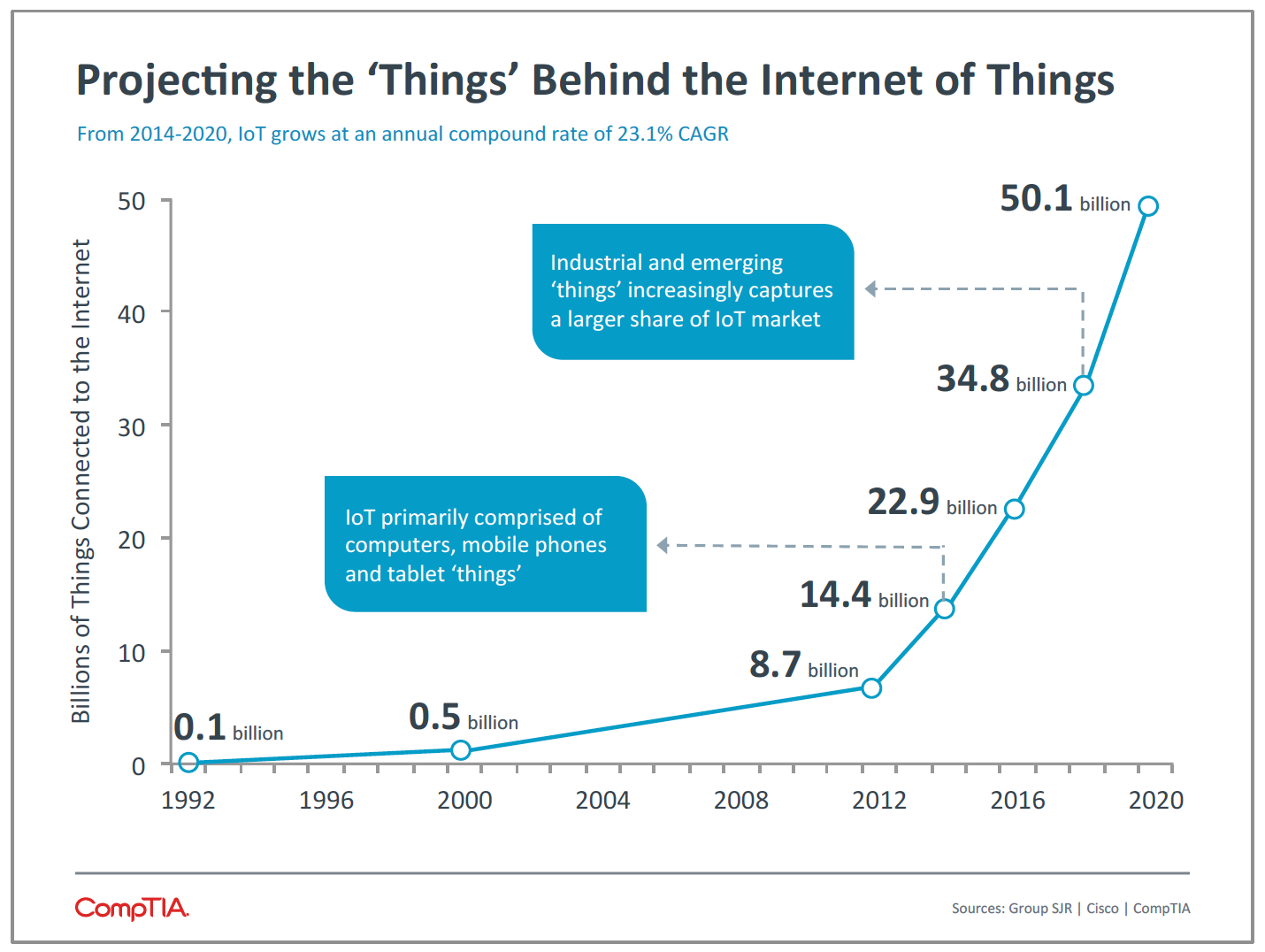
As mentioned before, smart devices greatly increase efficiency and reliability as the devices themselves can communicate with each other without manual operations. These smart devices are vastly growing in numbers year by year because of the efficiency and convenience they provide. Therefore, we proposed the idea of an Internet of Things (IoT) – based smart utility meter [1]. IoT devices are ones which are capable of interconnectivity with each other through the internet. The following statistic in Figure 2 shows the expected number of IoT devices up to the year 2020 [2].

Figure [2]

The smart utility meter we are building is a single meter which cumulates the functions of the three separate utility meters; Electricity [3], water [4] and gas through the utilization of the microcomputer and microcontroller capabilities provided by Raspberry Pi 3. Sensors will be set in place for every utility where they will generate a square wave signal depending on the consumption. These signals will be fed into the Raspberry Pi to calculate the consumption based on the number of pulses generated using a formula built from the resolutions of each sensor. When the consumption is computed, the bill is calculated accordingly and then uploaded to a server. This allows home owners as well as utilities providers to get data about the consumptions and due bills in real time through the internet. This greatly improves the efficiency of the process of inspecting the meter readings as well as paying due bills online through the use of an android application interface.

# Why Raspberry Pi3?

For our project, we decided to go with Raspberry Pi 3 over an Arduino microcontroller due to the overwhelming advantages of the Raspberry Pi in comparison to Arduino. In table [1] below is a comparison between a typical Arduino microcontroller and the Raspberry Pi 3.

|  |  |  |
| --- | --- | --- |
|  | **Arduino** | **Raspberry Pi 3** |
| **Price** | $30 | $35 |
| **Size** | 7.6 x 1.9 x 6.4 cm | 8.6 x 5.4 x 1.7 cm |
| **Memory** | 0.002 MB | 1 GB |
| **Clock Speed** | 16 MHz | 1.2 GHz |
| **On Board Network** | None | 10/100 wired Ethernet RJ45  802.11n Wireless LAN |
| **Multitasking** | No | Yes |
| **Input Voltage** | 7 to 12 V | 5 V |
| **Flash** | 32 KB | SD Card (2 to 32 GB) |
| **USB** | One, input only | Four, peripherals OK |
| **Operating System** | None | Linux distributions |
| **Integrated Development Environment** | Arduino  Table [1] | Scratch, IDLE, anything with Linus support |

As seen above, Raspberry Pi 3 costs about the same as the Arduino, but it incorporates network capabilities, wired Ethernet and Wi-Fi, much higher expandable storage, four different USB ports, and a Linux operating system

# Software and Hardware System Requirements

In our system, we have Hardware and software requirements. Software requirements will be divided into functional and non-functional.

Functional Requirement is any requirement which specifies **what** the system should do. However, Non-Functional Requirement is any requirement which specifies **how** the system performs a certain function (Eriksson, 2015).

Finally, the hardware requirements will present the used Equipment and Tools.

## Software Requirements

A software requirements is a comprehensive description of the intended purpose and environment for the developed software. Besides, it fully describes what the software will do and how it will be expected to perform.

### Functional Requirements

### 

The functional requirement will describe a particular behavior of function of the system when certain conditions are met.

In Our system the Functional requirements are;

FR 1.The system will allows utility authorizes members to login and enter their passwords.

FR1.1 the system will save the utility member ID and displays their name at the top of the website.

FR1.2 in case the user entered a wrong password a warning message will be displayed and a retrial is allowed.

FR1.3 in case the password was entered correctly the website will open the official page of the authority.

FR 2. The login functionality of the user.

FR2.1 Once the Raspberry Pi and the meters are implemented, the name of the home owner will be the username and a random password will be generated from the utility website.

FR2.2 The user will download the application on any smart phone, and trace their consumption at any period of time.

### Non-Functional Requirements

The non-functional requirement will describe how a system should behave and what limits there are on its functionality.

In our System, the Non-Functional requirements are;

NFR 1. Performance

NFR 1.1The system should be capable of operating requests within seconds.

NFR 2. Efficiency of use:

NFR 2.1 The system accomplish results quickly with accurately

NFR 3. Security

NFR 3.1 Login requirements; each user must enter his information

NFR 3.2 Password requirements; users must have a unique password for their login

NFR 4. Capacity:

NFR 4.1 The system can handle huge number of users at once.

NFR 5. Manageability:

NFR 4.1 The system provide the user with tools to manage his consumption.

## Hardware System Requirements

**Table [2] shows the full explanation of each hardware** Equipment/Tools.

|  |  |  |
| --- | --- | --- |
| Number | Tool Name | Usage |
| 1 | Power Meter | Power Meter will be generating pulses that will help us measure the power consumption. |
| 2 | Gas Meter | Gas Meter will be generating pulses that will help us measure the Gas consumption. |
| 3 | Water meter | Water meter will be generating pulses that will help us measure the Water consumption. |
| 4 | Raspberry pie 3 | The Raspberry Pie 3 will be used as a microcomputer to process all the calculations and as a microcontroller to read all the sensors |
| 5 | Potential Meter  Table [2] | Will be used for the simulation |

### Hardware Equipment Figures

The Gas meter and the Water meter are sponsored by: Dutco Tennant LCC.

**Gas Meter:**



Figure [3]

**Water Meter:**

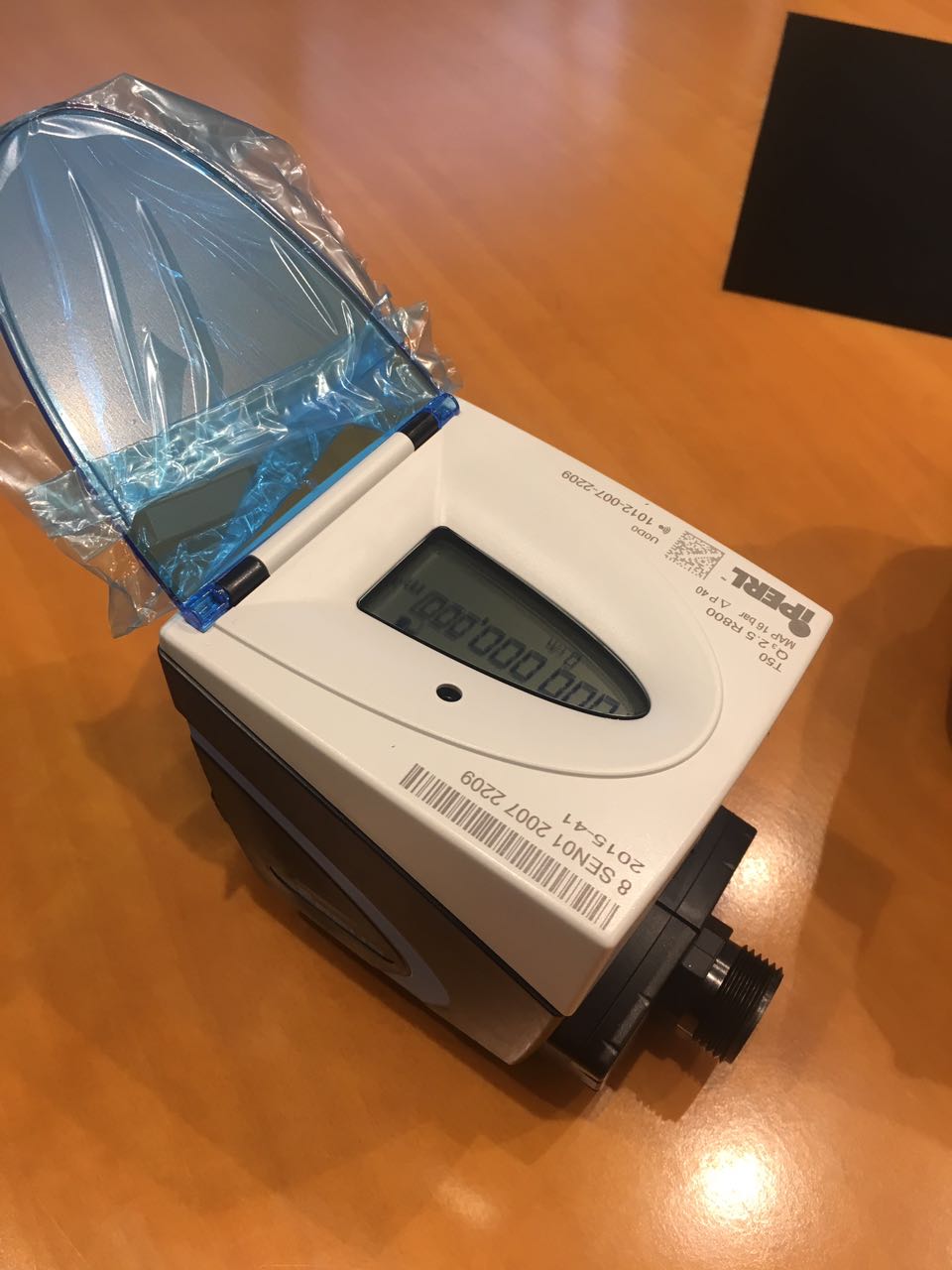
****

Figure [5]

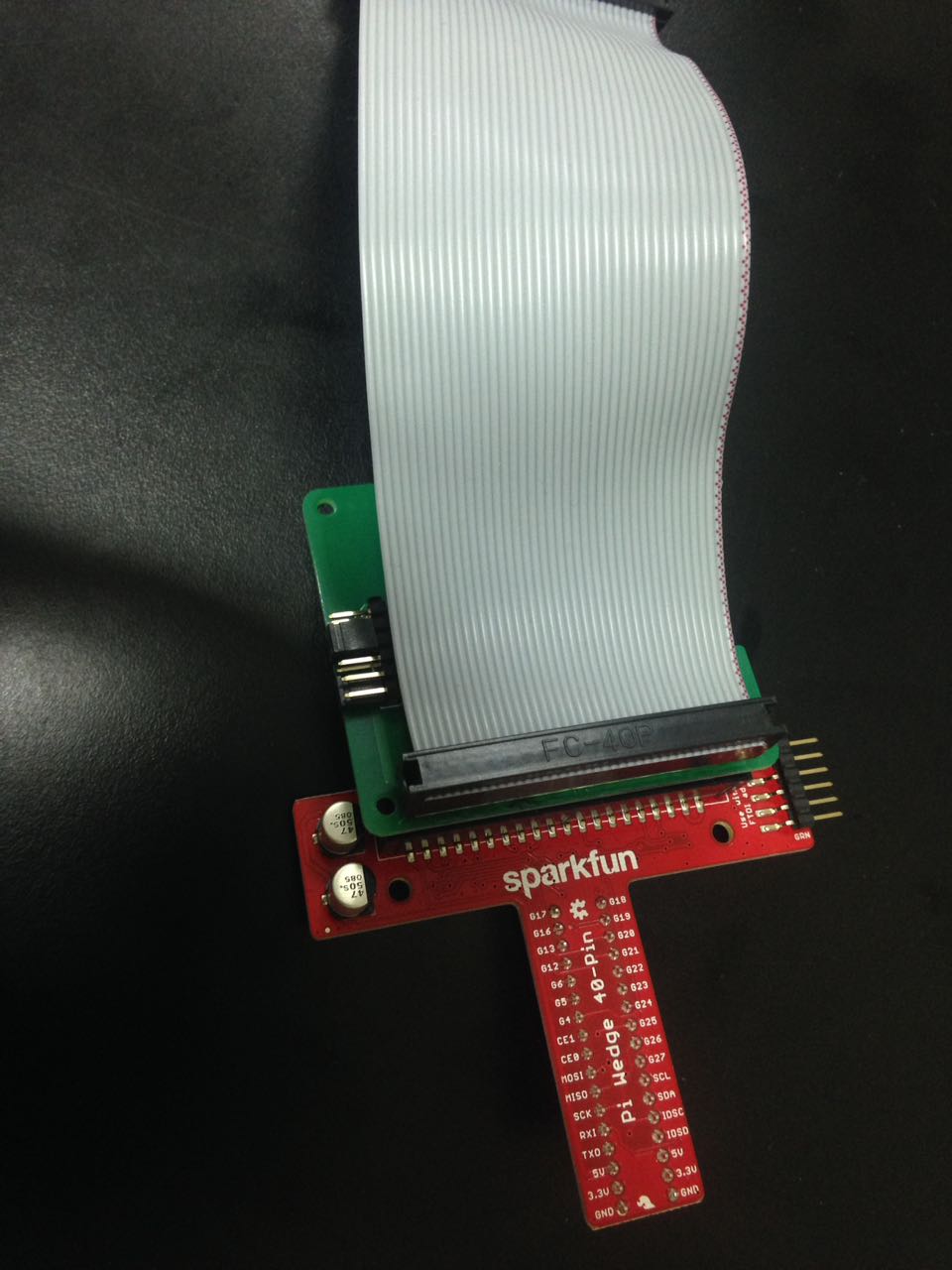
Figure [4]

**Raspberry Pi 3:**

# Image result for Raspberry pi 3

Figure [6]

**ADC:**

****

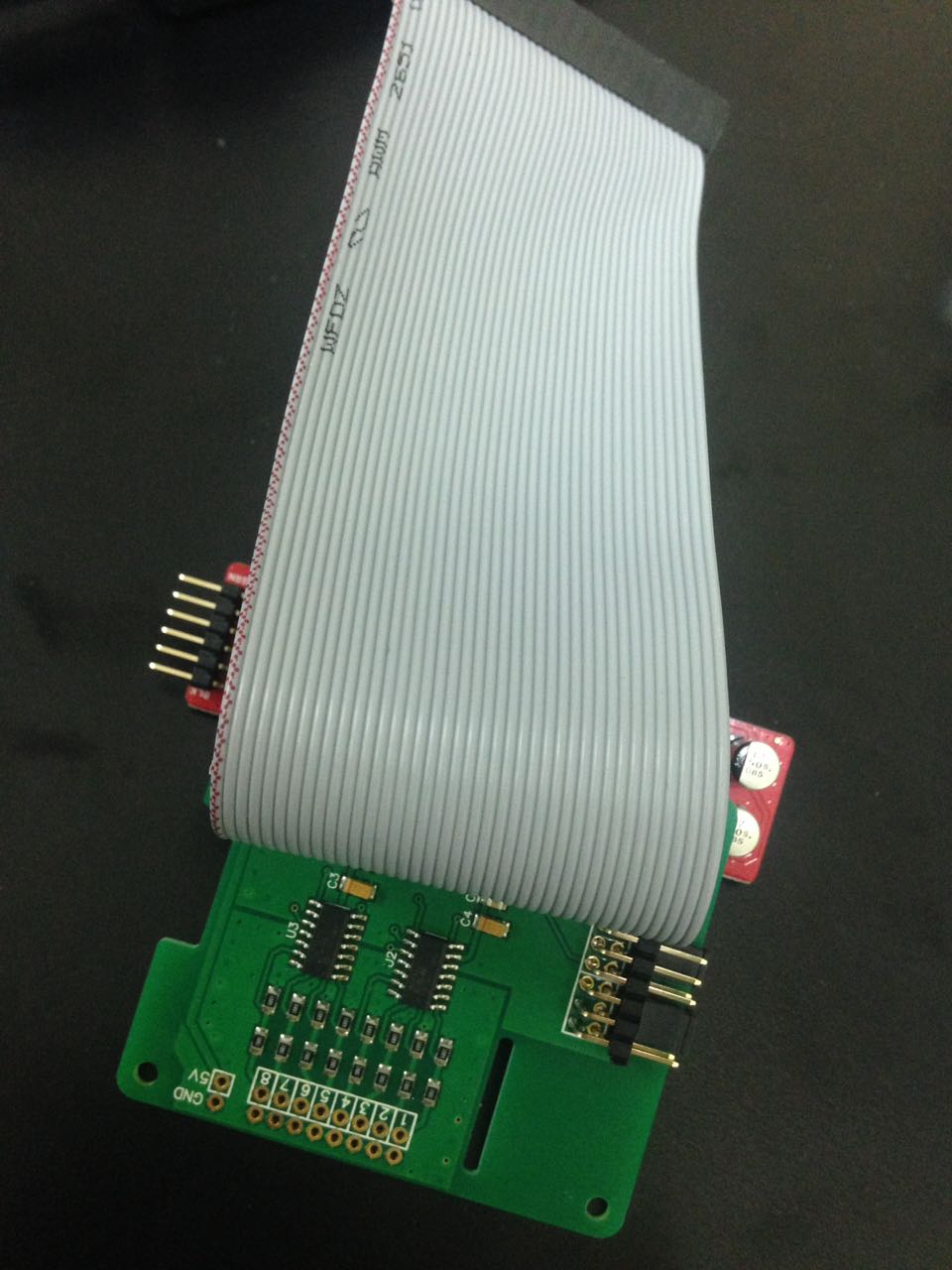
****

Figure [8]

Figure [7]

# Block Diagram & State Diagram

## Block Diagram

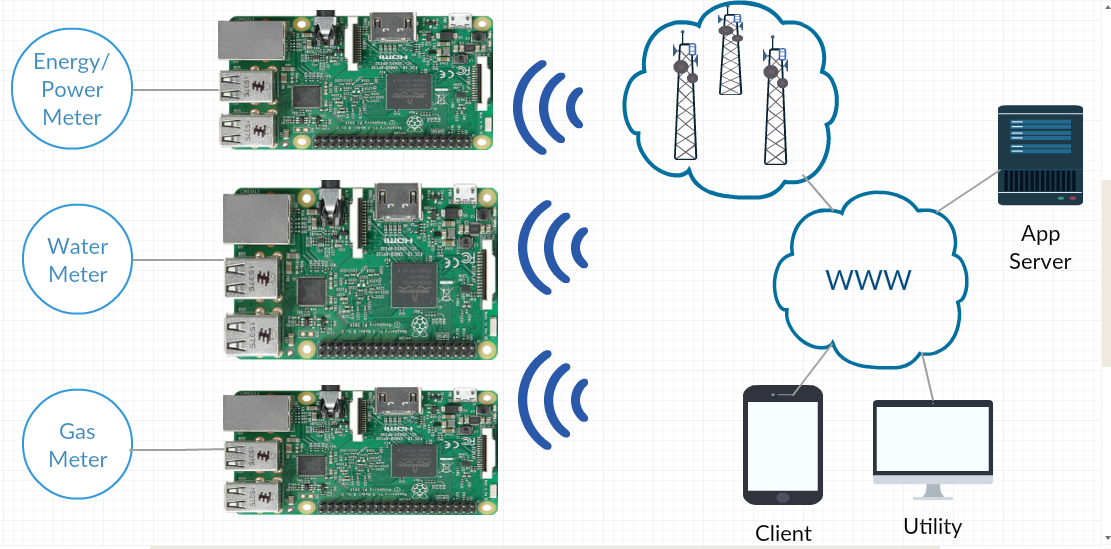
Block diagram is to show in schematic form the general arrangement of the parts or components of a complex system such as the inputs and outputs. Figure [9] will present the Block Diagram of our system that will illustrate the inputs and outputs. However, in our simplified system, we will be using one Raspberry Pi and three potential meters. Besides, the Utility website interface. As we are proposing two prototypes, that has different hardware structure but the same deliverables, Figure [10] will illustrate the connections of the alternative solution.

Figure [9]

### C:\Users\reema_000\Desktop\IOT_Based_Utility_Meter_Diagram_3.PNG

Figure [10]

### Inputs:

The system will be able to read the input from three different devices/Meters. Starting from the Power/Energy meter that will provide the Digital input. The Raspberry Pi, will be counting the number of output pulses and calculate the power consumption per kw/h. The same goes to the Gas and Water meters. Hence, we will be able to calculate the user's bills in the system.

### Outputs:

The system will start by printing the User's consumption on the utility interface, it will display the amount of each of the utility measurements, the Fuel charges for Electricity and water and finally it will print a summary of the bill, with the total amount of power and water consumptions. In addition, the Users will have an Android Application that will display their consumption and billing at any time.

## State Diagram

State Diagram is a diagram of the sequence of actions of systems involved in a complex system. It presents a graphical representation of a computer simulation in relation to its sequence of functions.

The below diagram represent the state diagram of our system, from starting until the end.

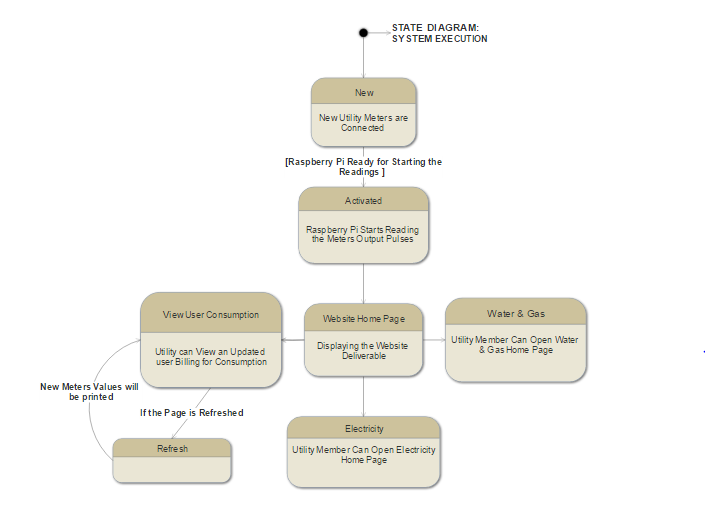


Figure [11]

# Implementation

## Prototype Description

In this section, we will present an implementation of a simplified model of our proposed solution. In this illustration we will use three different potential meters that will present the utility meters; Electricity, water and gas. The meters will output values in the (mV), and we will convert those values into kw/h for electricity, Cubic Feet for gas and Gallons for water according to a logical equations. The Raspberry Pi will read the values of the meters through a Python Code. And, once the Raspberry Pi starts reading the values, it will calculate the Users Billing and send them to the HTML file that will be displaying it to the utility interface. Two types of HTML pages are presented, dynamic and Static. The Static HTML pages will display the content when a call happens in the Python Code such as; Home Page, Electricity Page and the Water & Gas Page. However, the Dynamic Page will be changing according to the values of the meters. And when the utility member refresh this page it will send the most updated values.

This demonstration will help us visualizing the real work we are aiming for. The way the Raspberry Pi will perform, and the essential interface of the utility. Because of time concerns, we could not implement the Android Application. Yet, it will be part of our next stage plan.

## Coding

### Python

**Description:** Python allows us with the Raspberry Pi to connect our project to the real world as it is a very powerful programming language. Besides, it is very easy in writing or reading it and it has a very simple and clean syntax. Python syntax is readable and uses Standard English keywords. In our project, the Python code will be routing the HTML pages as required from the user. It will, additionally, read the values of the potential meters and send them to an HTML file. The code bellow is fully commented to explain the functions of each line.

Note: Any wording after "#" is a commented line.

**Code:**

|  |
| --- |
| from ABE\_ADCPi import ADCPi  #Import ADCPi library to use the ADC Pi from ABE\_helpers import ABEHelpers  #import the helper class import time #import the library to use delay in code import smbus     from flask import Flask, render\_template # to import the flask library with the render\_template inside it app = Flask(\_\_name\_\_, template\_folder='/home/pi/testing') # create a flask object called app and direct it to the mentioned distention where all the files of the code will at the same directory level  # setting up the ADC i2c\_helper=ABEHelpers() bus=i2c\_helper.get\_smbus()#initialize sumbs using i2c\_helper object adc=ADCPi (bus, 0x68,0x69)#initialize the ADC device using default address & sample rate  adc.set\_bit\_rate(12)# set bit rate to 12 adc.set\_conversion\_mode(1)#set continuous conversion mode by sending 1 time.sleep(0.2)  Potential meter# def getPotentiometerInvoltage1():     return(adc.read\_voltage(2))#reads voltage value form selected channel 2 on the raspberry pi, range between 0v-5v  def getPotentiometerInvoltage2():     return(adc.read\_voltage(3)) ))#reads voltage value form selected channel 3 on the raspberry pi, range between 0v-5v  def getPotentiometerInvoltage3():     return(adc.read\_voltage(1)) ))#reads voltage value form selected channel 1 on the raspberry pi, range between 0v-5v  @app.route('/') #Run the code below this function when someone accesses the root URL (‘/’) or default web page of the server, navigate to http://0.0.0.0:5090 def home():          return render\_template('Main\_3.html', title='Test Page') #Send a static HTML home page to the client's web browser   @app.route('/power') #Run the code below this function when someone accesses the root URL (‘/power’) of the server, navigate to http://0.0.0.0:5090 def power():         return render\_template('Electricity\_3.html', title='Test Page') #Send a static HTML Electricity page to the client's web browser   @app.route('/water') #Run the code below this function when someone accesses the root URL (‘/water) of the server, navigate to http://0.0.0.0:5090 def water():         return render\_template('Water&Gas\_3.html', title='Test Page') #Send a static HTML water & gas page to the client's web browser  #@app.route('/consumption') #Run the code below this function when someone accesses the root URL (‘/consumption) of the server, navigate to http://0.0.0.0:5090 #def consumption():  @app.route('/consumption') def Consumption():#Send a dynamic HTML page to print the values of the consumptions (after billing) the client's web browser     potentiometerVoltage1=getPotentiometerInvoltage1()#copying the return value of the functions and saves them in a string, this string will have the mv output of the power meter    potentiometerVoltage2=getPotentiometerInvoltage2()#copying the return value of the functions and saves them in a string, this string will have the mv output of the water meter     potentiometerVoltage3=getPotentiometerInvoltage3()()#copying the return value of the functions and saves them in a string, this string will have the mv output of the gas meter             water = (potentiometerVoltage2 / 0.015) \* 0.04 #the water parameter will have the converted mv value to Gallon     power = (potentiometerVoltage1 / 0.010) \* 0.23 #the power parameter will have the converted mv value to kw/h     gas = (potentiometerVoltage3 / 0.020) \* 2.90 #the gas parameter will have the converted mv value to cubic feet      WaterFuelCharge = (potentiometerVoltage2 / 0.015) \* 0.04 #the parameter will have the fuel charge of water according to the consumption (mv)      PowerFuelCharge = (potentiometerVoltage1 / 0.010) \* 0.01 #the parameter will have the fuel charge of power according to the consumption (mv)          totPower = power + PowerFuelCharge #the total power consumption will be calculated as the addition of the charges and the consumption     totWater = water + WaterFuelCharge +gas #the water total cost will be the water consumption added with the Fuel charges and the gas      Bill =  totWater+ totPower # total bill will include all the charges on the user           author = "IOT-BASED SMART UTILITY METER WEBSITE" #the HTML page title      return render\_template('test1.html', author=author, power=power, water=water, gas=gas, PowerFuelCharge=PowerFuelCharge , WaterFuelCharge=WaterFuelCharge, totPower= totPower,totWater=totWater,Bill=Bill)# here we are sending each value to the corresponding place on the Dynamic HTML page.   if \_\_name\_\_=='\_\_main\_\_': #If this script was run directly from the command line     app.run(debug=True, host='0.0.0.0', port=5090) #Have the server listen on port 5090 and report any errors. The addition of 0.0.0.0 implies that the web server is going to listen on all network devices, so the web site can be browsed from anywhere on the same local network |

### HTML Pages

### Home Page

**Description:** The home HTML page will consist of a description of our prototype system. And the methods we are using in implementing such system. In addition, it will have a brief introduction about the Electricity and Water Authority.

**Code:**

|  |
| --- |
| <html lang="en-AE">  <head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <style>  .flex-container {  display: -webkit-flex;  display: flex;  -webkit-flex-flow: row wrap;  flex-flow: row wrap;  text-align: center;  }  .flex-container > \* {  padding: 15px;  -webkit-flex: 1 100%;  flex: 1 100%;  }  .article {  text-align: left;  }  header {background: black;color:white;}  footer {background: #aaa;color:white;}  .nav {background:#eee;}  .nav ul {  list-style-type: none;  padding: 0;  }    .nav ul a {  text-decoration: none;  }  @media all and (min-width: 768px) {  .nav {text-align:left;-webkit-flex: 1 auto;flex:1 auto;-webkit-order:1;order:1;}  .article {-webkit-flex:5 0px;flex:5 0px;-webkit-order:2;order:2;}  footer {-webkit-order:3;order:3;}  }  p {  font-family:Calibri Light;  }  h1 {  font-family:Calibri Light;  }  li {  list-style: none;  float:left;  font-size: 30px;  font-family:Calibri Light;  margin-left: 15px;  padding-top: 10px;  }  </style>  </head>  <body>  <div class="flex-container">  <header>  <h1>IOT-BASED SMART UTILITY METER</h1>  </header>  <nav class="nav">  <ul>  <li><a href="http://www.w3schools.com/html/tryit.asp?filename=tryhtml\_layout\_flexbox">Home</a></li>  <li><a href="http://www.w3schools.com/html/tryit.asp?filename=tryhtml\_links\_w3schools">Electricity</a></li>  <li><a href="http://www.w3schools.com/html/tryit.asp?filename=tryhtml\_layout\_flexbox">Water and Natural Gas</a></li>  </ul>  </nav>  <article class="article">  <h1>Power, Water and Gas Orgnizations</h1>  <p>The three important utility meters in each house presenst; power, water and gas. They are discrete and installed in various residential, industrial and commercial sectors. Most of them, are not engaged to any mean of communication that enables the utility to manage their consumption. All the current methods of connecting or disconnecting are done manually. However, the main objective of Power, Water ans Gas Orgnizations is providing the fastest, most effcient and the most practicle methods. </p>  <p>To provide such services, Smart metering is proposed to calculate customrs bills and to control the switching On/Off utilities throught the internet. This communication will decrease the manual operations that might hold large error percentages. </p>  <p>In our simplified demonistration, we will read three Potentioal meters for Power, Water and Natural Gas. Hence, we will convert the number of pulses that should represent (kw/h, Cubic feet and Gallons) by voltages. </p>  </article>  <footer>Copyright © AUS.edu</footer>  </div>  </body>  </htm> |

**Sample of the Page:**

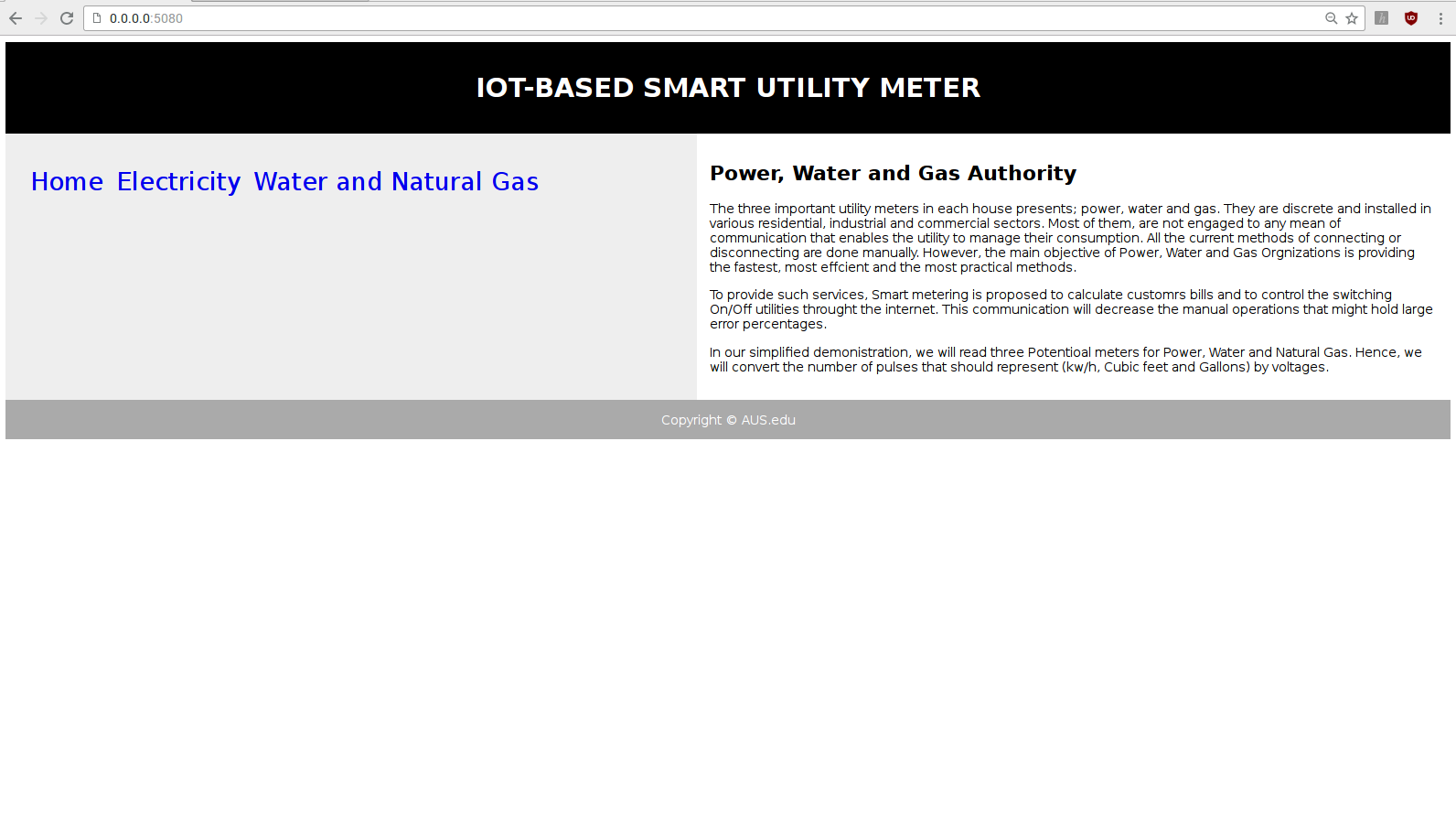


Figure [12]

### Water and Gas Page

**Description:** The Water and Gas HTML page will consist of a description of how the water and gas will be calculated in our system and the billing of the costumer. The prices are according to Dubai Electricity and Water Authority.

**Code:**

|  |
| --- |
| <html lang="en-AE"> <head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <style>  .flex-container {  display: -webkit-flex;  display: flex;  -webkit-flex-flow: row wrap;  flex-flow: row wrap;  text-align: center;  }  .flex-container > \* {  padding: 15px;  -webkit-flex: 1 100%;  flex: 1 100%;  }  .article {  text-align: left;  }  header {background: black;color:white;}  footer {background: #aaa;color:white;}  .nav {background:#eee;}  .nav ul {  list-style-type: none;  padding: 0;  }    .nav ul a {  text-decoration: none;  }  @media all and (min-width: 768px) {  .nav {text-align:left;-webkit-flex: 1 auto;flex:1 auto;-webkit-order:1;order:1;}  .article {-webkit-flex:5 0px;flex:5 0px;-webkit-order:2;order:2;}  footer {-webkit-order:3;order:3;}  }  p {  font-family:Calibri Light;  }  h1 {  font-family:Calibri Light;  }  li {  list-style: none;  float:left;  font-size: 30px;  font-family:Calibri Light;  margin-left: 15px;  padding-top: 10px;  }  </style>  </head>  <body>  <div class="flex-container">  <header>  <h1>IOT-BASED SMART UTILITY METER ⁠⁠</h1>  </header>  <nav class="nav">  <ul>  <li><a href="https://www.google.ae/\_/chrome/newtab?espv=2&amp;ie=UTF-8#">Home</a></li>  <li><a href="http://www.w3schools.com/css/tryit.asp?filename=trycss\_default">Electricity</a></li>  <li><a href="https://www.google.ae/\_/chrome/newtab?espv=2&amp;ie=UTF-8#">Water and Natural Gas</a></li>    </ul>  </nav>  <article class="article">  <h1>Water and Natural Gas</h1>  <p>According to Dubai Electricity and Power Authority, each 1 IG (Imperial Gallon) costs 0.04 fills. There are in addition to the power consumption a Fuel charges. For each 1kwh it cost 0.01 fills.</p>  <p>And for Gas Consumption, we will consider each 1 Therm (that is eqaul t0 100 cubic feet) to cost 2.90 AED according to national standerds. With no additional feeses. </p>  <p> For Water, for each 15mv input it will represent 1 Gallon. Hence, for each 15mv it will cost 0.04 fills. And, for each 15mv input, it will present an 100 cubic feet of Natural Gas. So, 2.90 AED for each cubic feet </p>  </article>  <footer>Copyright © AUS.edu</footer>  </div>  </body>  </html> |

**Sample of the Page:**

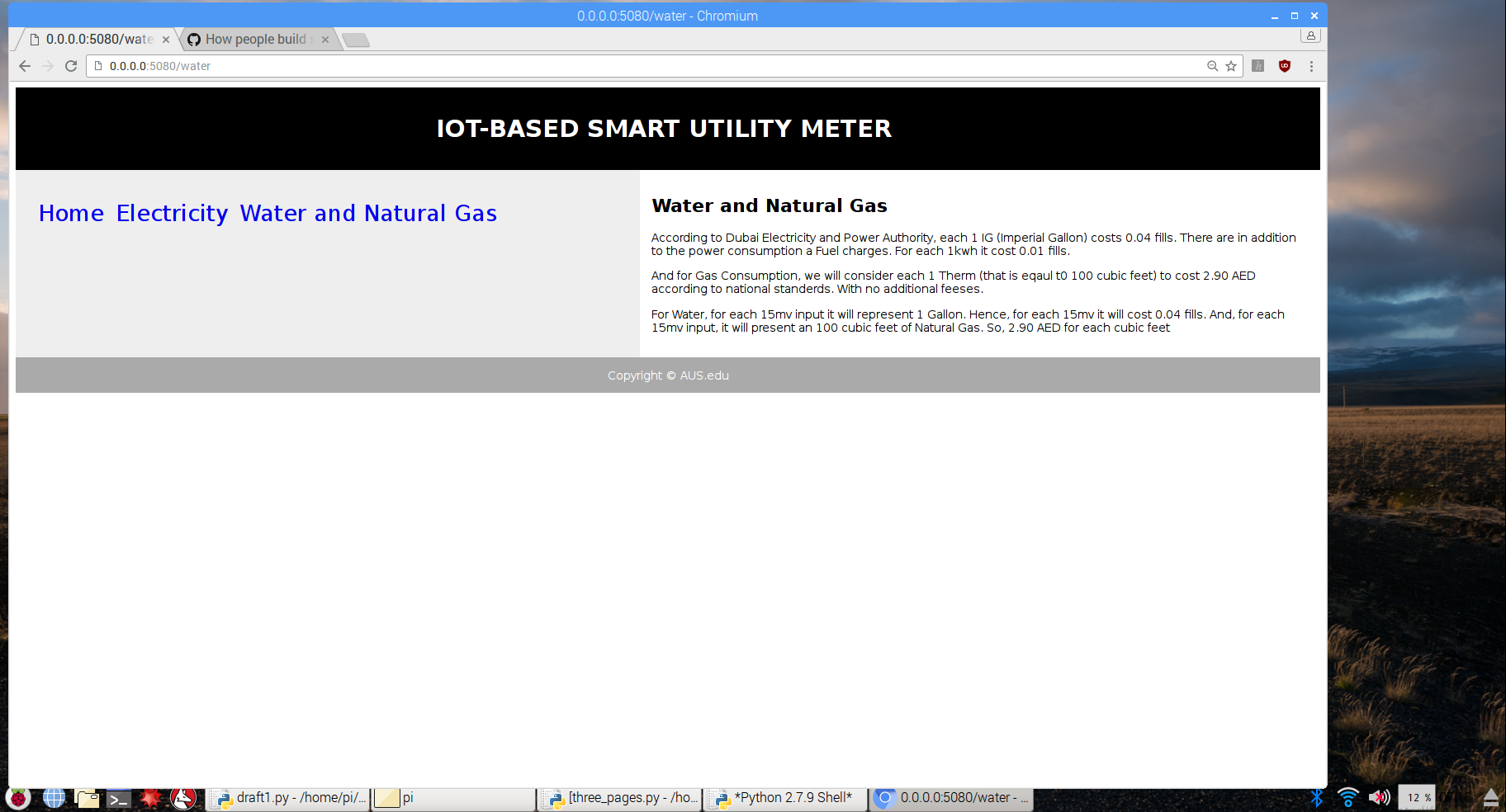
****

Figure [13]

### Power Page

**Description:** The Power HTML page will consist of a description of how the power will be calculated in our system and the billing of the costumer. The prices are according to Dubai Electricity and Water Authority.

**Code:**

|  |
| --- |
| <html lang="en-AE">  <head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <style>  .flex-container {  display: -webkit-flex;  display: flex;  -webkit-flex-flow: row wrap;  flex-flow: row wrap;  text-align: center;  }  .flex-container > \* {  padding: 15px;  -webkit-flex: 1 100%;  flex: 1 100%;  }  .article {  text-align: left;  }  header {background: black;color:white;}  footer {background: #aaa;color:white;}  .nav {background:#eee;}  .nav ul {  list-style-type: none;  padding: 0;  }    .nav ul a {  text-decoration: none;  }  @media all and (min-width: 768px) {  .nav {text-align:left;-webkit-flex: 1 auto;flex:1 auto;-webkit-order:1;order:1;}  .article {-webkit-flex:5 0px;flex:5 0px;-webkit-order:2;order:2;}  footer {-webkit-order:3;order:3;}  }  p {  font-family:Calibri Light;  }  h1 {  font-family:Calibri Light;  }  li {  list-style: none;  float:left;  font-size: 30px;  font-family:Calibri Light;  margin-left: 15px;  padding-top: 10px;  }  </style>  </head>  <body>  <div class="flex-container">  <header>  <h1>IOT-BASED SMART UTILITY METER ⁠⁠</h1>  </header>  <nav class="nav">  <ul>  <li><a href="https://www.google.ae/\_/chrome/newtab?espv=2&amp;ie=UTF-8#">Home</a></li>  <li><a href="http://www.w3schools.com/css/tryit.asp?filename=trycss\_default">Electricity</a></li>  <li><a href="https://www.google.ae/\_/chrome/newtab?espv=2&amp;ie=UTF-8#">Water and Natural Gas</a></li>    </ul>  </nav>  <article class="article">  <h1>Electricity- Power</h1>  <p>According to Dubai Electricity and Power Authority, each 1kwh (killo what per hour) costs 0.23 fills. There are in addition to the power consumption a Fuel charges. For each 1kwh it cost 0.07 fills. </p>  <p> For Power, for each 20mv input it will represent 1 Kw/h. Hence, each 20mv will cost 0.23 fills. </p>  </article>  <footer>Copyright © AUS.edu</footer>  </div>  </body>  </html> |

**Sample of the Page:**

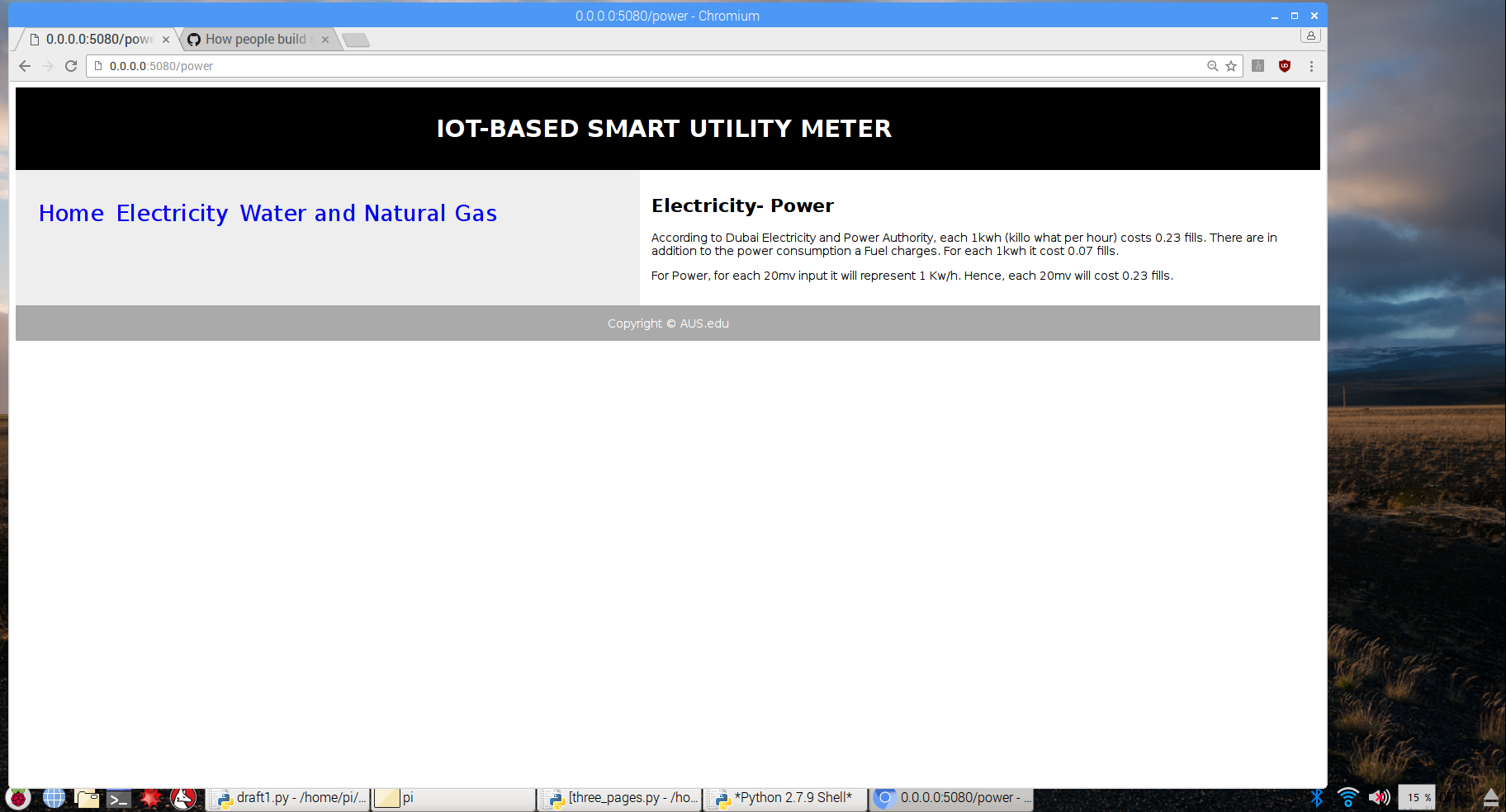
****

Figure [14]

### Consumption Page

**Description:**

The Consumption page will consist of the user consumption information and the total billing information.

**Code:**

|  |
| --- |
| <html><!-- Copyright 2015 The Chromium Authors. All rights reserved.  Use of this source code is governed by a BSD-style license that can be  found in the LICENSE file. --><head><meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <style>  .flex-container {  display: -webkit-flex;  display: flex;  -webkit-flex-flow: row wrap;  flex-flow: row wrap;  text-align: center;  }  .flex-container > \* {  padding: 15px;  -webkit-flex: 1 100%;  flex: 1 100%;  }  .article {  text-align: left;  }  header {background: black;color:white;}  footer {background: #aaa;color:white;}  .nav {background:#eee;}  .nav ul {  list-style-type: none;  padding: 0;  }    .nav ul a {  text-decoration: none;  }  @media all and (min-width: 768px) {  .nav {text-align:left;-webkit-flex: 1 auto;flex:1 auto;-webkit-order:1;order:1;}  .article {-webkit-flex:5 0px;flex:5 0px;-webkit-order:2;order:2;}  footer {-webkit-order:3;order:3;}  }  p {  font-family:Calibri Light;  }  h1 {  font-family:Calibri Light;  }  li {  list-style: none;  float:left;  font-size: 30px;  font-family:Calibri Light;  margin-left: 15px;  padding-top: 10px;  }  table, td, th {  border: 1px solid black;  }  table {  border-collapse: collapse;  width: 50%;  }  td {  height: 50px;  vertical-align: bottom;  text-align: center;  }  th {  text-align: left;  }  </style>  </head>  <body>    <div class="flex-container">  <header>  <h1>Utility meter Website</h1>  </header>  <nav class="nav">  <ul>  <li><a href="chrome-search://local-ntp/local-ntp.html#">Home</a></li>  <li><a href="http://www.w3schools.com/css/tryit.asp?filename=trycss\_default">Electricity</a></li>  <li><a href="chrome-search://local-ntp/local-ntp.html#">Water and Natural Gas</a></li>    </ul>  </nav>  </div>  <h2> User Consumption Details </h2>  <p> Consumption </p>  <table border="1">  <tbody><tr>  <td><b> Utility</b> </td>  <td> <b>User Consumption</b> </td>  </tr>  <tr>  <td>Electricity</td>  <td> {{power}}</td>  </tr>  <tr>  <td>Water</td>  <td> {{water}}</td>  </tr>  <tr>  <td>Gas</td>  <td> {{gas}}</td>  </tr>    </tbody></table>  <p> The Fual charges</p>    <table border="1">    <tbody><tr>  <td> <b>Electricity Fual Charges</b> </td>  <td> {{PowerFuelCharge}}</td>  </tr>  <tr>  <td><b>Water Fuel Charges</b></td>  <td> {{WaterFuelCharge}}</td>  </tr>  </tbody></table>    <p> Summary </p>    <table border="1">  <tbody>  <tr>  <td><b> Electricity</b></td>  <td> {{totPower}}</td>  </tr>  <tr>  <td><b>Water</b></td>  <td> {{totWater}}</td>  </tr>  <tr>  <td><b>Total Bill(AED)</b></td>  <td> {{Bill}}</td>  </tr>    </tbody></table>  <br>    <footer>Copyright © AUS.edu</footer>  </body></html> |

**Sample of the Page:**

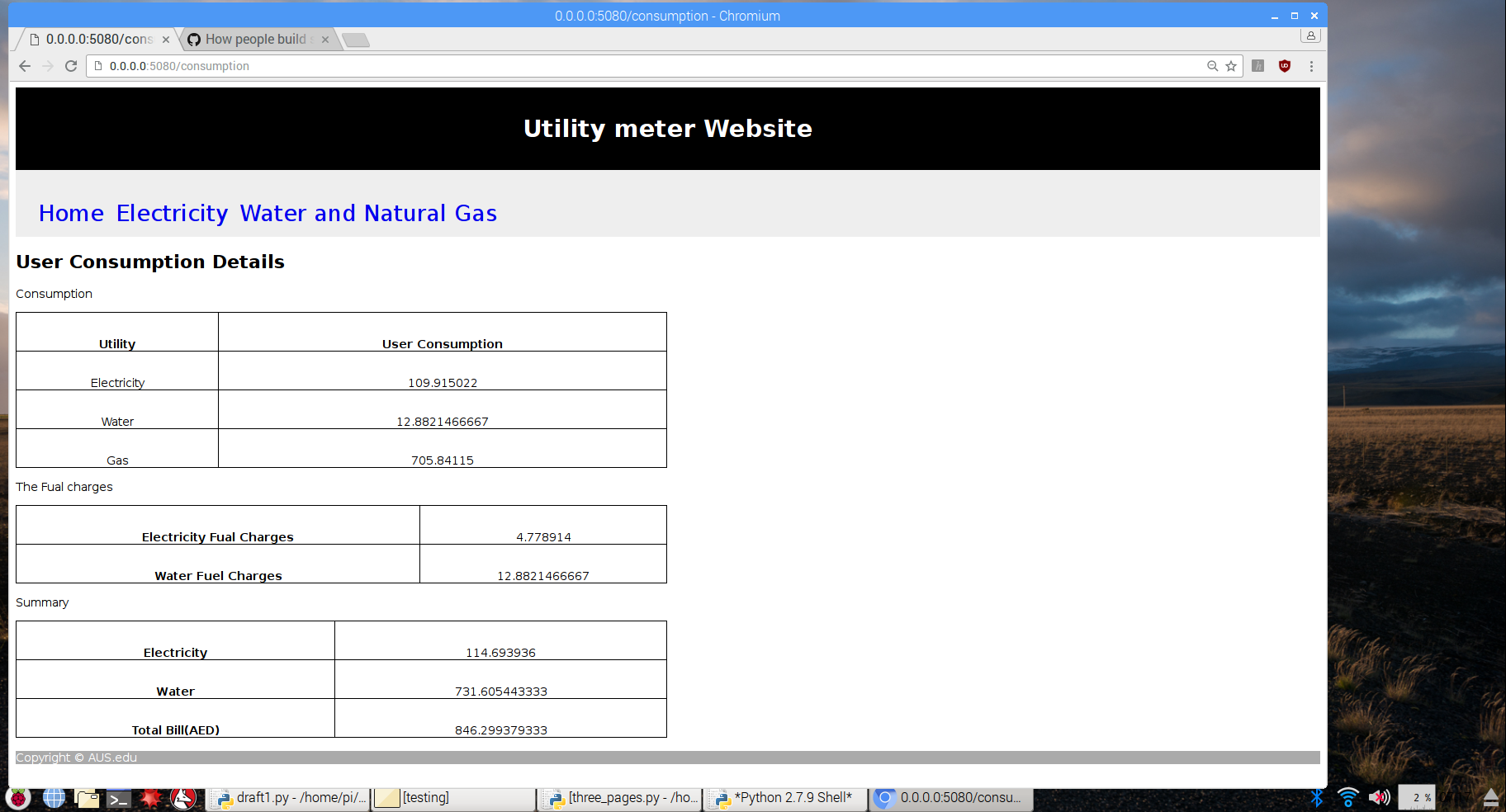
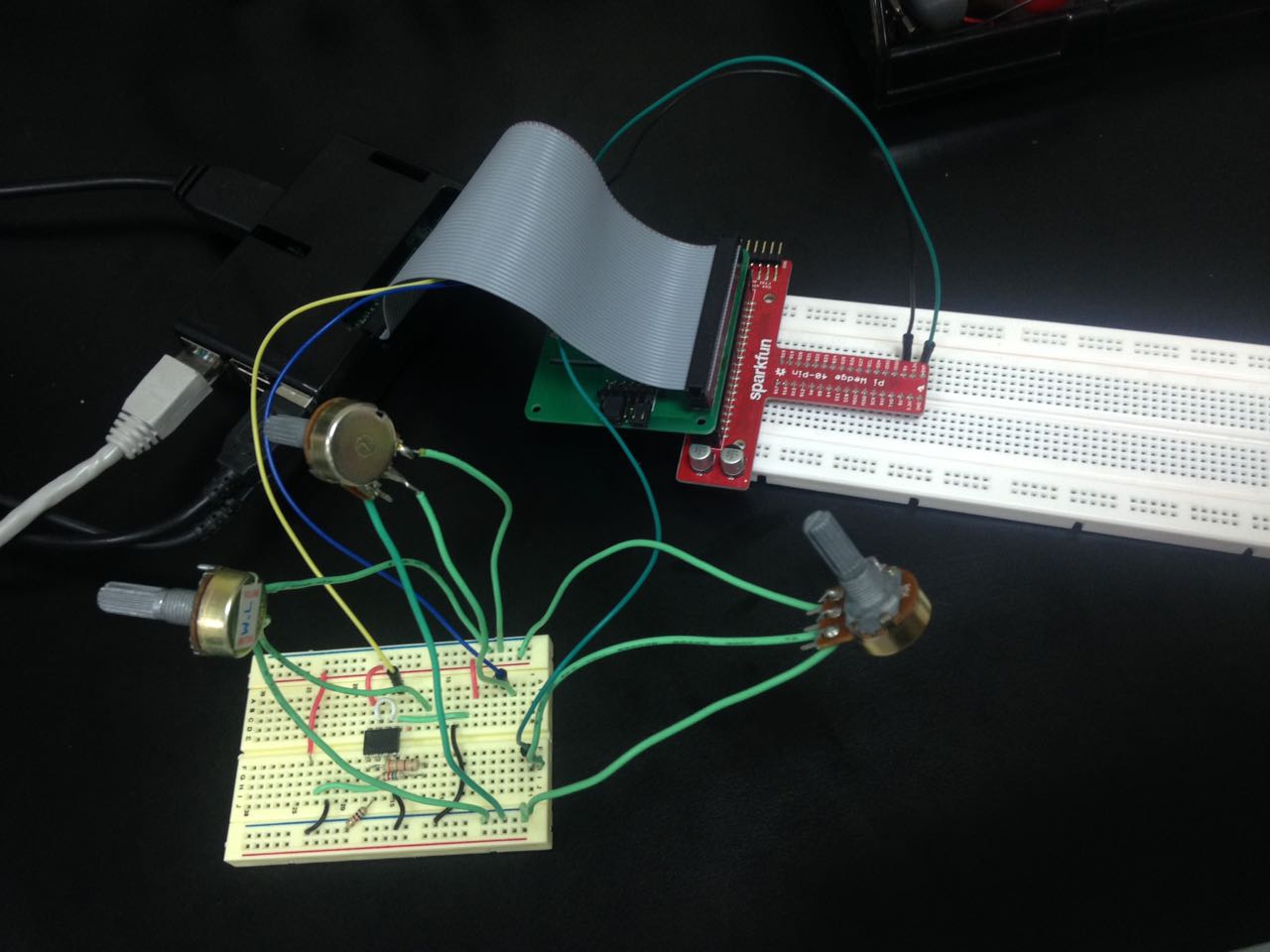


Figure [15]

### Our Implantation Connections:

Raspberry Pi 3

Potential meter 3



Potential meter 2

ADC

Potential meter 1

Figure [16]

# Future work

In the future, we will be extending on the system using google maps, and Android Application. Google maps will contain the location of the meters, their status, and the consumption rate. We will include markers in order to visualize the connectivity and the billing of each house of the three meters as shown in a previous work done by

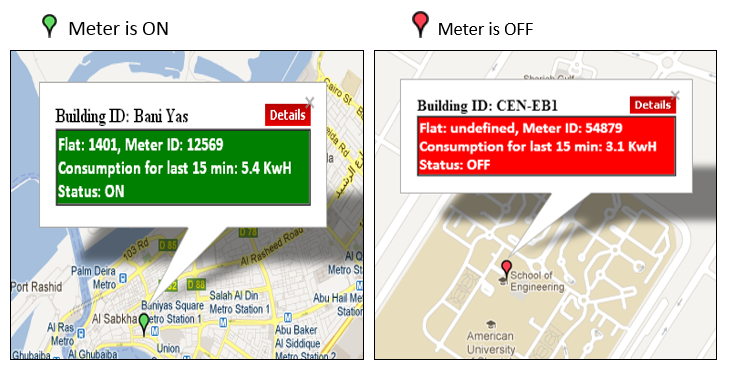
F .Ahmad and B. Mohammed El-Kurdi [5] about this topic, Figure 15 and 16. In addition, we will implement the system to be used through an Android application where the users could view the consumption using their smart phones. Also, our system will perform switching on the meters in any place through the internet, it will be able to disconnect the meters of to connect them. Finally, our system will approach an algorithm to manage the electricity billing during night and day hours in a more efficient way [6]. 

Figure [17]

Figure [16]

# References

[1] L. Wang, V. Devabhaktuni and S. Depuru, "Smart meters for power grid: Challenges, issues, advantages and status", *Sciencedirect.com*, 2011. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1364032111000876> .

[2] "CompTIA | Sizing Up the Internet of Things", *Comptia.org*, 2016. [Online]. Available: <https://www.comptia.org/resources/sizing-up-the-internet-of-things>

[3] F. Benzi, L. Frosini, E. Bassi and N. Anglani, "Electricity Smart Meters Interfacing the Households - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2011. [Online]. Available: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5699358>.

[4] G. Hauber-Davidson and E. Idris, "SMART WATER METERING", *amazonaws.com*, 2016. [Online]. Available: <http://s3.amazonaws.com/academia.edu.documents/6239032/wcg-smartwater-article.pdf?AWSAccessKeyId=AKIAJ56TQJRTWSMTNPEA&Expires=1479379995&Signature=7SycjB36u%2FBycl%2BqBdbutMPQFvo%3D&response-content-disposition=inline%3B%20filename%3DSmart_water_metering.pdf>

[5] GIS Bases: Smart Energy Meter Monitoring System by Fazel Ahmad Bashir

Mohammed El-Kurdi (2012).

[6] M. Erol-Kantarci and H. T. Mouftah, "Wireless Sensor Networks for Cost-Efficient Residential Energy Management in the Smart Grid - IEEE Xplore Document", *Ieeexplore.ieee.org*, 2016. [Online]. Available: <http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5734885>

# 