GROUP 5G

**Smart Meter Smart Recepticle PROJECT**

# Software

The software system would be composed of 4 programs. This includes an android application, program for the smart receptacle device, the smart meter device and the online website.

**Smart Receptacle Program**

**Programming Language:** Python 2.7

**Device:** Raspberry Pi 2

**IDE:** IDLE

This program would be designed to take current and voltage readings from the ADC’s and computer the power consumption of each socket that is connected to it. This device would take readings from the voltage and current ADC’s every 30 seconds and computer the power. This power value would be then sent to the smart meter device. This device would log on to the smart meter device using the secure shell (ssh) protocol. This device would then access the database stored on the smart meter device and add the new power entry to the table. When the power is computed, it should be added to the previous power value to generate a new cumulative power value. This value should be displayed on the LCD screen for the receptacle. Before this value is displayed the device would log into the smart meter table containing all the IP addresses and names associated with the IP addresses. This would be used to verify the name hasn’t changed. Once this is verified, the device name and current consumption would be updated. The power would be displayed by cycling through the power consumption per socket every 3 seconds.

**Smart Meter Program**

**Programming Language:** Python 2.7

**Device:** Raspberry Pi B+ / Raspberry Pi 2

**OS:** Raspbian

**IDE:** IDLE

This program would be designed to have a table of all of the power consumed by the individual receptacles in the home. This program would also keep track of all of the power consumed by all of the outlets. This figure would be computed by adding up the values in the power consumption table in the database every 60 seconds. This table would then be exported into a file and uploaded to the Dropbox storage. The total power consumed should be displayed as well as the outlet that has consumed the most power.

## Android Application

**Android Application**

**Programming Language:** Java

**Device:** Any Android Device

**OS:** Android 4.4 (KitKat) to 6.0 (Marshmallow)

**IDE:** Android Studio 2.2

This android application allows the user (home owner) to access the stored data on their home. This app allows them to view the data graphically.

The first screen seen by the user is a login screen, this would allow a user to enter their credentials. Once the credentials are correct the main menu is loaded. The android application is designed so that when it is turned on it checks for a meter key and also for the current electricity rate. If either of these settings are not found the user is prompted. The meter key is obtained from the smart meter; it would be shown on the LCD screen of the meter once it has detected that it is not currently associated with an android app. The current electricity rate would have to be obtained from the electricity commission which the user is currently under. This is the cost per watt for electricity. For TTEC this is approximately $0.26. (<https://ttec.co.tt/default/wp-content/uploads/2015/08/Tariffs.pdf> ) .

### View Stats

#### Overview

This Activity allows the user to see the current total consumption for the life time of the meter and also the consumption of the power for each outlet on a bar graph. There are also calculations done to estimate bill cost for the month, highest consuming outlet and the number of watts used for the month. There is also a refresh button to re download data. From the dropdown menu at the top the consumption for each outlet can be computed. This list is auto generated for every file based on the readings in the file.

#### Details

This Activity works by initially downloading the file for the meter connected to the app containing all of the readings. This file would be selected based on the hash of the meter key stored on the device and also the file name would be in the format “Meterhash\_Readings.txt” where “Meterhash” would be the MD5 hash of the key entered for the meter. Once this file is being downloaded a toast saying “loading” would be shown. Once the file has been successfully downloaded a toast saying “completed” would be shown. Once the file is downloaded it is read line by line and the data is extracted. The lines starting with “CumPower” contain the cumulative total power consumed, i.e. the current meter reading at that time and date for the entire home. The other lines contain receptacle readings at that date and time specified. To compute the “Watts Used This Month”, first, the application pulls the highest cumulative value for that month. Then it looks for the highest value from all previous months and subtracts the two readings. To compute the bill, estimate the “Watts Used This Month Value” is then multiplied by the “Cost Per Watt” value set by the user. The “Highest Consumer” value is computed by selecting the highest reading that is not a cumulative reading and getting the plug name for which is associated with it.

### Change Key

#### Overview

This activity is used to add or change the key currently associated with the android application. This key is obtained from the meter and entered into the text box. If a key is already associated with the account it would be shown.

#### Details

When the change key button is pressed the MD5 Hash of the key is generated. This hash is used to determine the files associated with the meter. The meter would also generate the same hash when uploading files and append it to the file name. All files associated with the meter would have the hash in the file name. Therefore, the app would only download a file that contains the correct hash onto the device, this would prevent one user from accessing another user’s data.

### My Outlets

This activity is used to associate individual receptacles to the android application. This part of the app would be used to turn receptacles on and off.

#### Details

When the activity is loaded the list of outlets are generated based on the current configuration file. If no file is found the list would be blank. The user must first add a receptacle to the list of outlets. This is done by clicking the plus in the top right hand corner. They then must enter the address associated with the outlet, this would appear on the LCD screen of the outlet. They must also enter the number of the outlet; it would be either “1” or “2” since a receptacle only has two outlets. They can then name the outlet for easier identification when reviewing usage data. They can also turn the individual outlets on an off using the toggle button in the outlet settings. Once the settings are saved the list entry left corner would turn either red or green. If the outlet is off it would be red and if the outlet is on it would be green. The configuration file would be updated and uploaded to the dropbox. This file would then be downloaded by the meter and sent to the receptacle to turn on or off the receptacle depending on the setting. Each time a setting is changed in the android app the configuration file is regenerated and reuploaded to the dropbox overwriting the currently uploaded version. The hash of the meter key is appended to the configuration file name before it is uploaded so it would be found by the correct meter. Selecting an already created outlet allows the user to modify its properties. Long pressing an entry would allow a user to remove it.

### Settings

#### Overview

This activity allows a user to change their electricity rate setting so that the estimated bill can be computed accurately. If a setting is already stored it would be shown to the user.

**Website**

**Programming Language:** Python 2.7/HTML/JavaScript

**Device:** Online Server

**OS:** N/A

**IDE:** N/A

This website would be targeted to the utility companies. This would allow them to read the power consumed for the entire home. This would not give them access to power consumed per receptacle. This can be used for them to see trends in power consumption as well as data such as percentage change to previous readings and estimate bill cost.

# Accounts Credentials

Email Credentials

**Email :** [team5G@outlook.com](mailto:team5G@outlook.com)

**Password :** ThisTeamRocksAlot!

Dropbox Credentials

**Email :** [team5G@outlook.com](mailto:team5G@outlook.com)

**Password :** ThisTeamRocksAlot!

Android Application

**Email :** [team5G@outlook.com](mailto:team5G@outlook.com)

**Password:**testpass