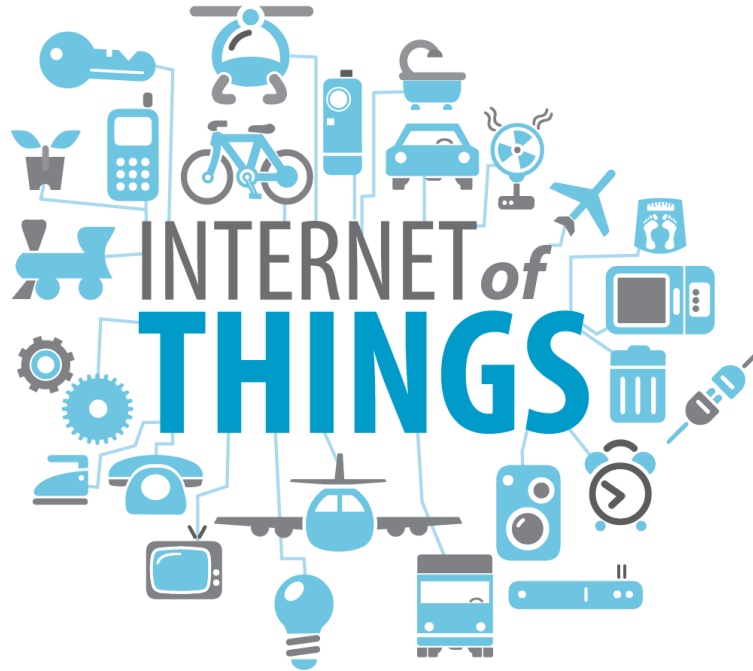


# USERS MANUAL

## *FRONT END PROGRAMMING OF IoT SMART OBJECTS*



Submitted By:

Kaustav Ghosh

Jay Shah

Rohit Makhija

Ping He

Ketan Rajput

Fall 2014-COEN 296 Project 2

## INTRODUCTION

The software application developed is a front end graphical user-interface (GUI) for programming smart objects in the Internet of Things (IoT) domain. We have developed this GUI using the SCALA programming language. The project is a user-friendly and scalable GUI which provides users to select different types of sensors, decide its individual threshold value and input/update its values. We can also select different colors for different nodes which will help in tracing large number of nodes selected simultaneously. We have also integrated the Twitter API with the GUI. The GUI serves as a platform to program the logic and control the connected devices with social networks for building new IoT applications.

The Home page of the application is shown in Figure 1.

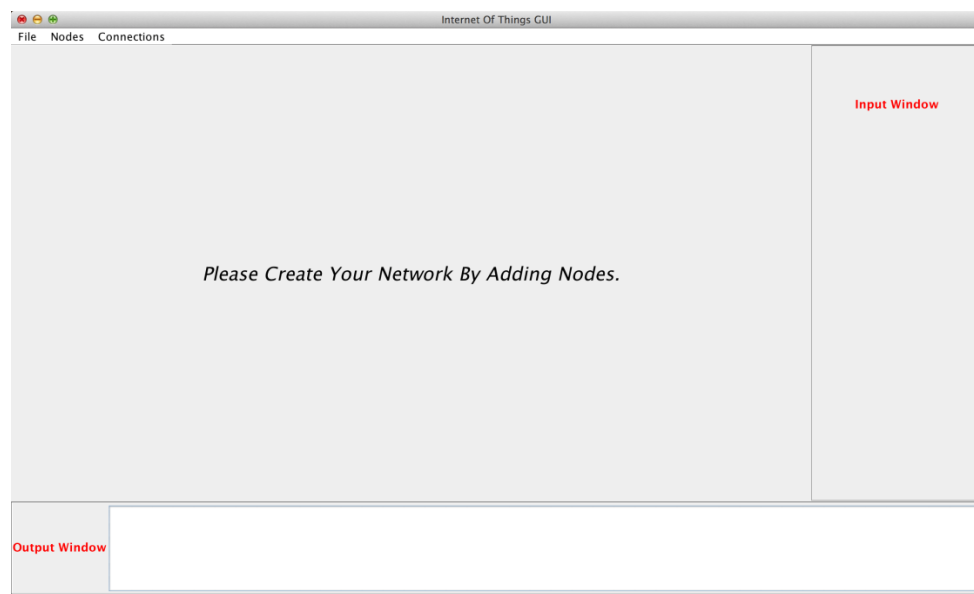


Figure 1: GUI Startpage.

### Implemented Functions:

1. File: This menu item consists of a drop down list with the options listed below
  - New: This option allows user to create a new file in the application.
  - Save: This option allows user to save their work to a specific folder on a local system.
  - Open: This option allows users to open an existing file on the system.
  - Exit: This option exits the application.

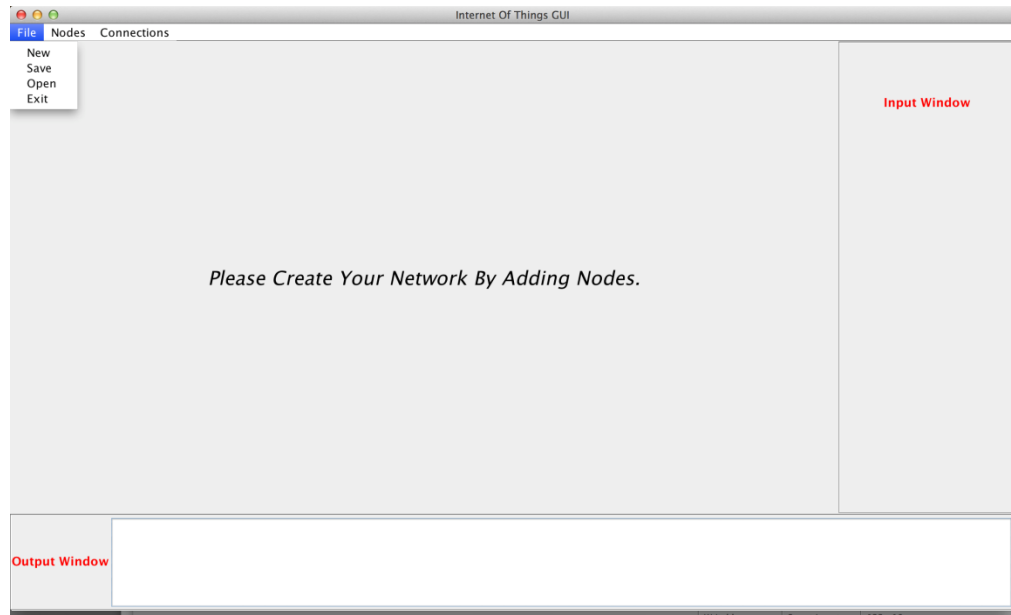


Figure 2: File Menu drop down.

2. Node: This menu item allows user to add multiple nodes to the interface. Each node represents a sensor. When you click on the 'Add Nodes', a dialogue will pop up asking for the type of sensor from a drop down list.

The available sensor types are:

- Temperature Sensor
- Light Sensor
- Buzzer
- Sound Sensor
- Twitter.

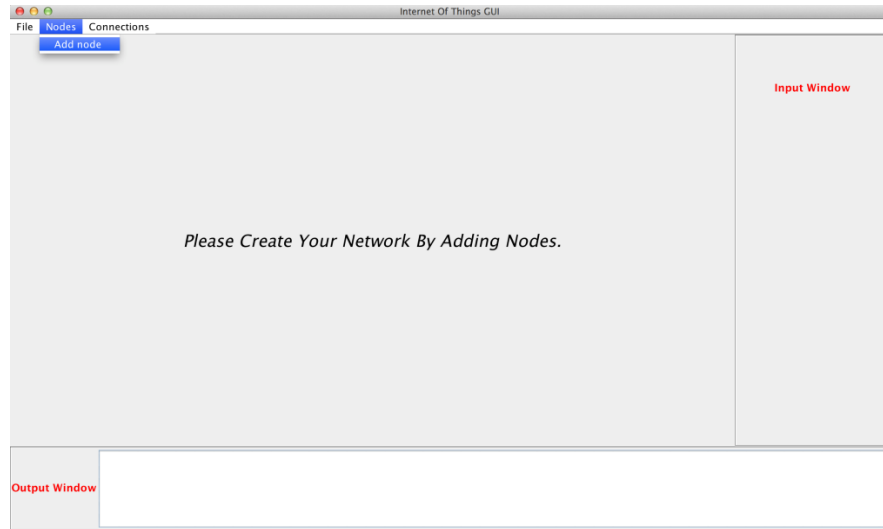


Figure 3: Adding a node using 'Nodes' menu item.

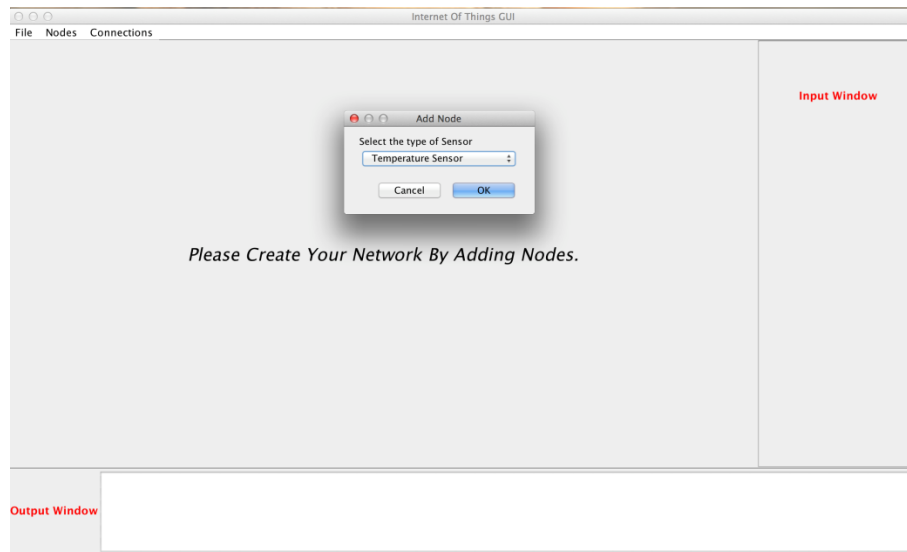


Figure 4: Selecting the type of sensor

After you click 'OK' to confirm the sensor type, another dialogue will pop up and let you choose the color of sensor from a drop down list.

The available colors are:

- Red
- Blue
- Green

- Yellow and
- Orange.

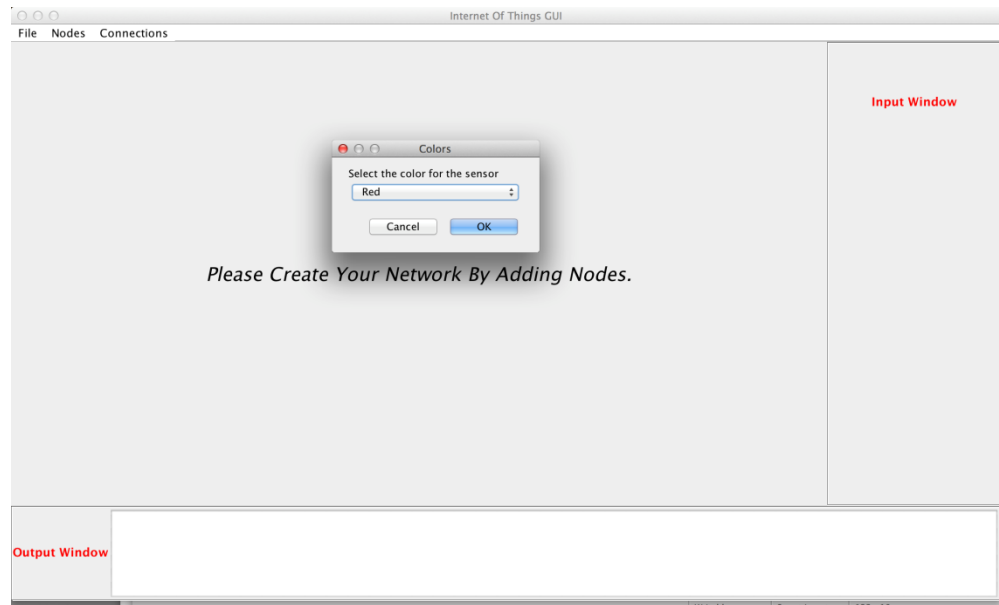


Figure 5: Selecting the color for the sensor.

After confirming the color type, another dialogue will pop up and allow you to enter initial value of the sensor selected (by default 20). After that is done, the final pop up will ask you to enter the threshold value of the sensor (by default 0).

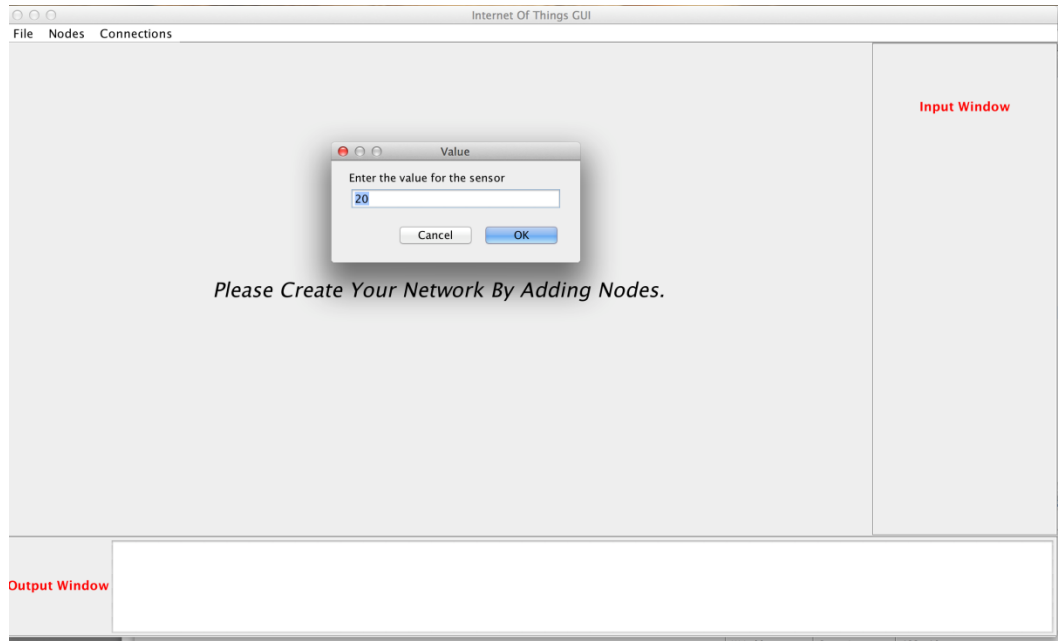


Figure 6: Taking input value of the sensor.

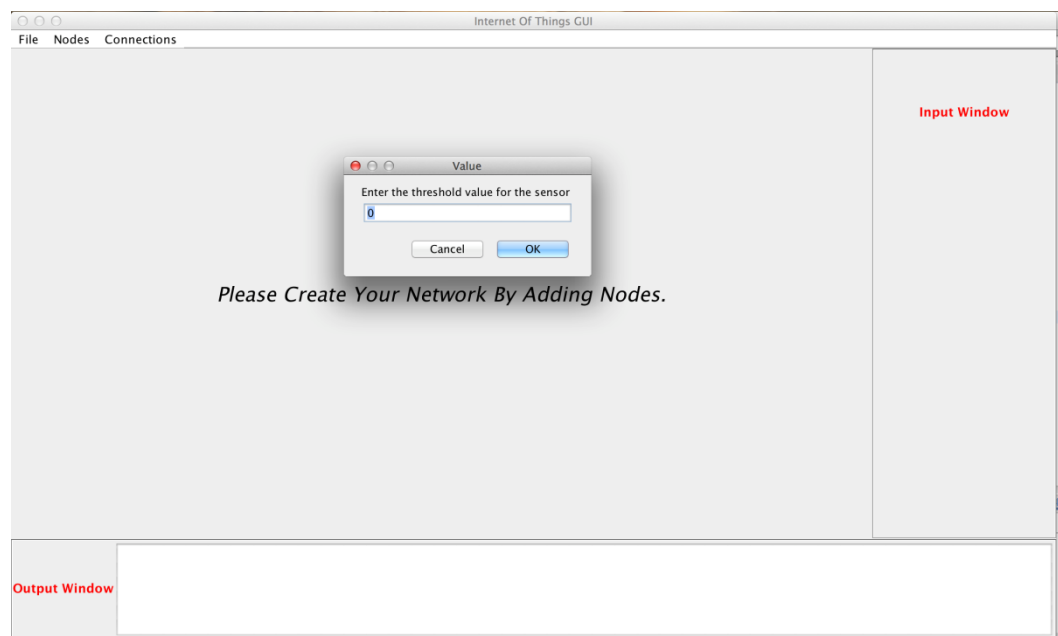


Figure 7: Taking threshold value.

The above steps will add one sensor on the screen, in order to add multiple sensors we need to repeat the above steps. Once that is done the application screen will look like given in figure: 8 given below.

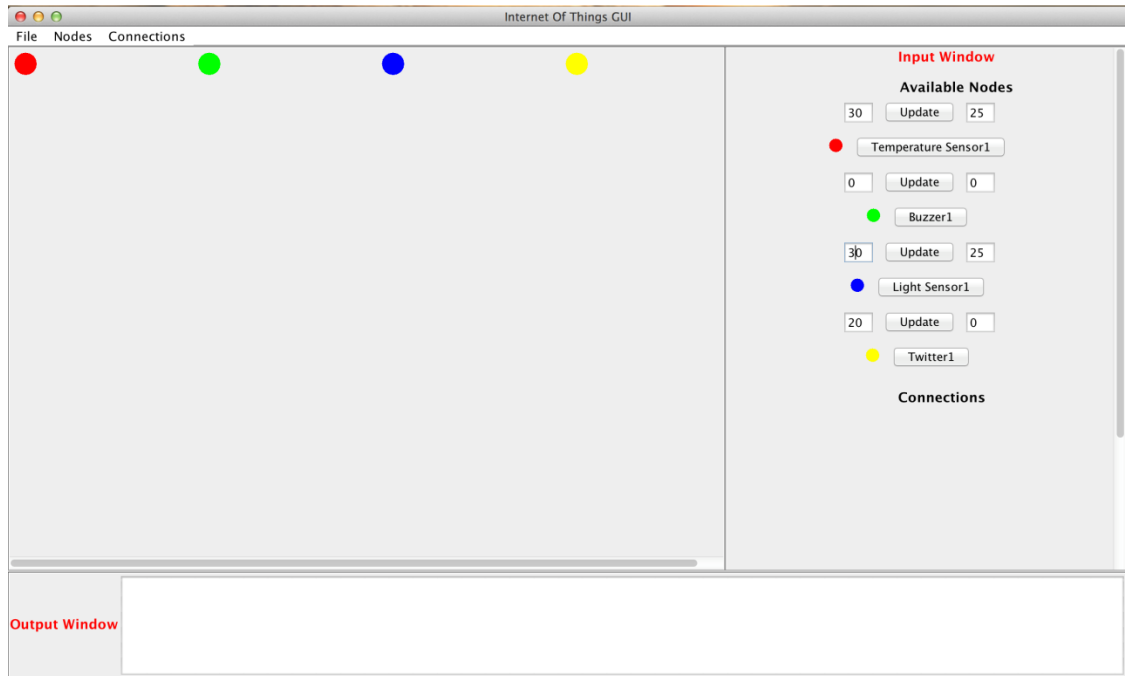


Figure 8: Screen after displaying 4 nodes.

After providing these input values the screen will look similar to the one below. The colored circles represent the different selected sensors.

3. Connections: This file menu item allows users to select the connection between the two sensors selected in the above steps. When you click on 'Connections' it will provide you with an 'Add connections' option. When you click the "Add Connections," a dialogue will pop up and let you choose the first sensor from the dropdown list of sensors added in the previous steps. After you click "OK" to confirm the type of first sensor, another dialogue will pop up and let you choose the second sensor from the drop down list. A message will be displayed on the bottom right of the screen stating the two sensors are connected successfully.

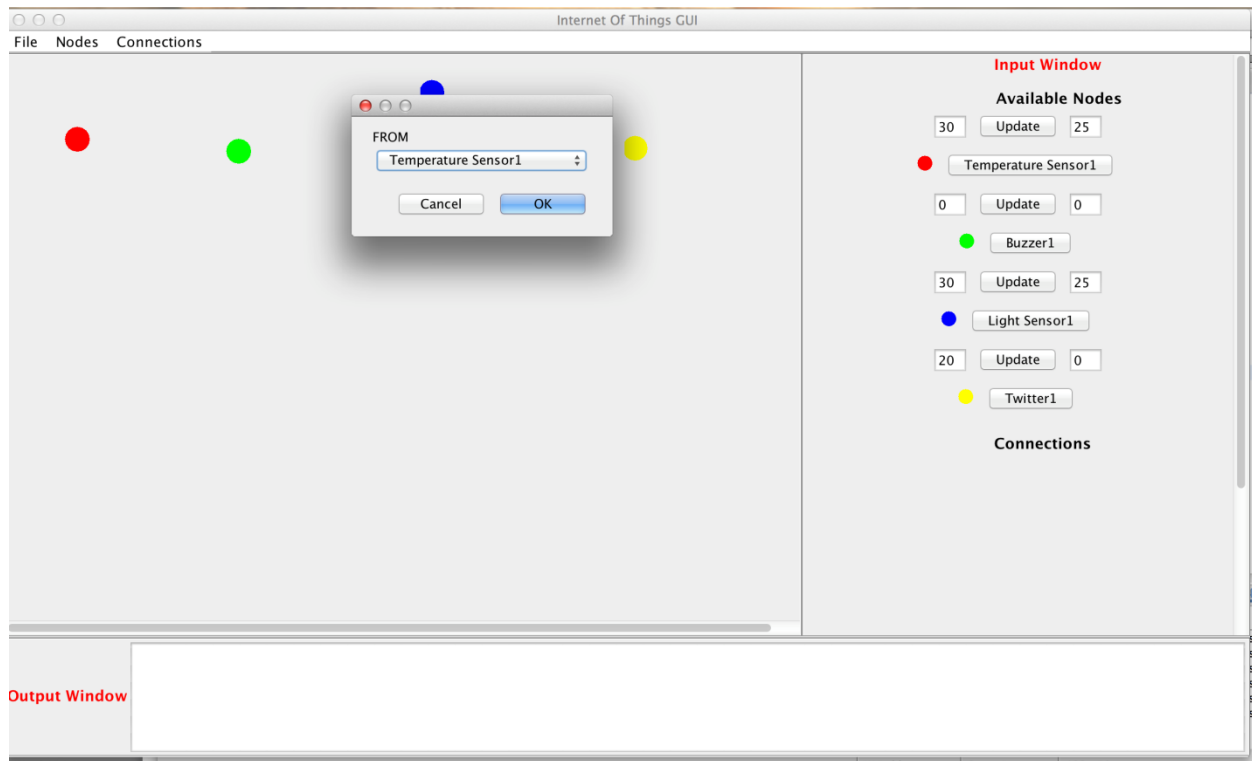


Figure 8: Selecting the first sensor to be connected.

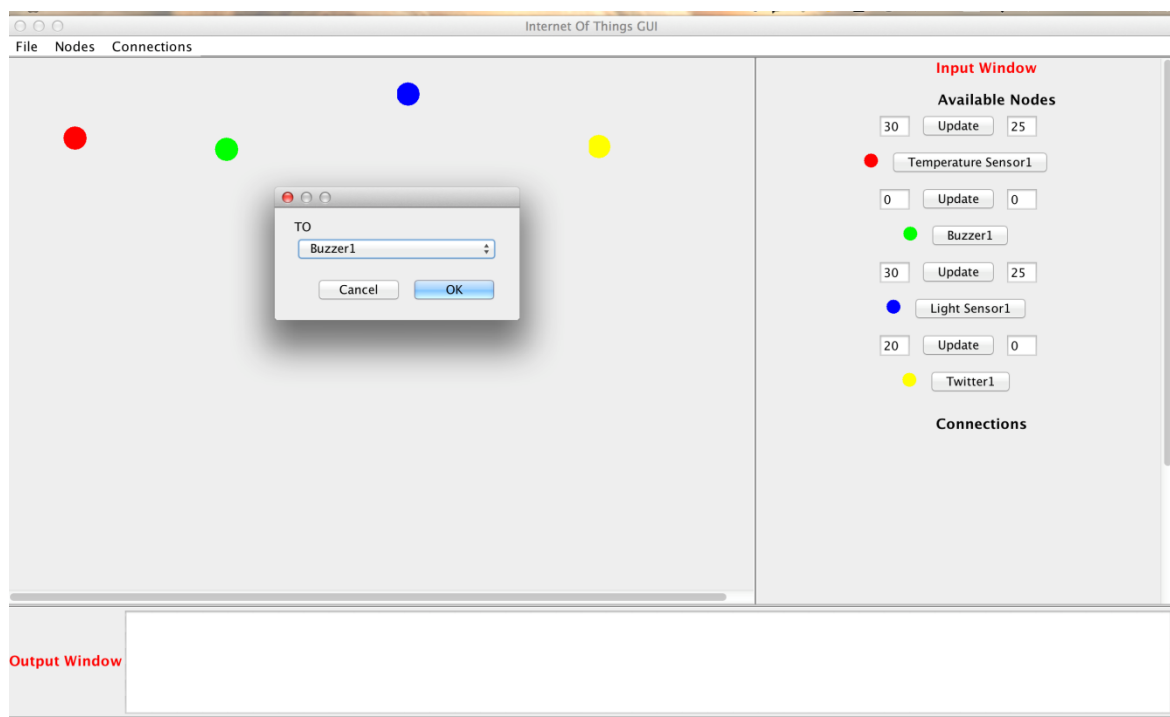


Figure 9: Selecting the target sensor to be connected.



There are three types of pre-defined sensor combinations, they are as follows

- a) Temperature sensor with Buzzer
  - b) Light sensor with twitter account
  - c) Sound sensor with light sensor
4. Updating values and threshold: The right hand side (Input window) of the screen will have a partition displaying the details of the available nodes. It will have the initial value field which will show the initial value set by the user, and then will be an update button followed by a threshold text field where the user can set a threshold limit for his sensor and then perform the desired functionalities. The initial value text field can get a new value from the user and then on clicking the update button that new value will be set to that sensor. So basically the value to the left hand side of the 'update' button displays the value and the one on the right side displays the threshold value.

The screenshot shows a software interface titled "Input Window" in red text. It contains a section titled "Available Nodes" with a list of five nodes, each with a colored circular icon, a name in a rounded rectangle, and a row of three input fields (two text boxes and one button). The nodes are: Temperature Sensor1 (red icon), Buzzer1 (green icon), Light Sensor1 (blue icon), and Twitter1 (yellow icon). Each node's row includes a text box with a value (20, 0, 20, 20 respectively), an "Update" button, and a text box with a threshold value (0, 0, 0, 0 respectively). Below this section is a "Connections" section with the text "Temperature Sensor1 is connected to Buzzer1".

Node Name	Current Value	Update Button	Threshold Value
Temperature Sensor1	20	Update	0
Buzzer1	0	Update	0
Light Sensor1	20	Update	0
Twitter1	20	Update	0

**Connections**  
Temperature Sensor1 is connected to Buzzer1

Figure 10: Editing the Threshold values

5. Simulation: The bottom right of the screen has a 'Simulation' button, which will start the simulation process and then display the desired output in the output area at the bottom on the screen as shown below. This helps the user to check if the functionality has been performed and view that output.

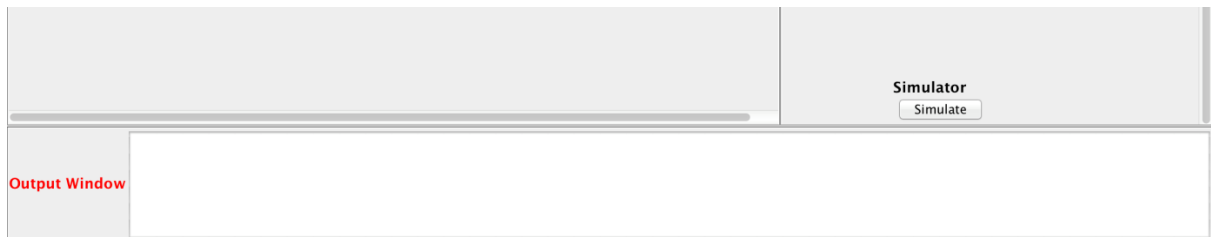


Figure 11: Simulator and the Output window.

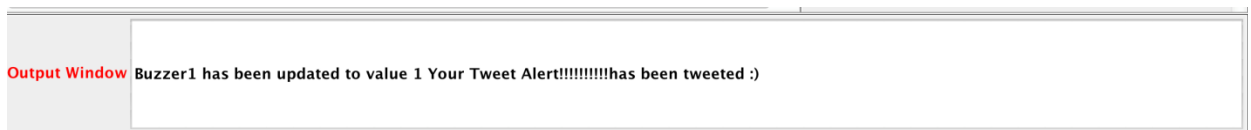


Figure 12: Message displayed in the output window.

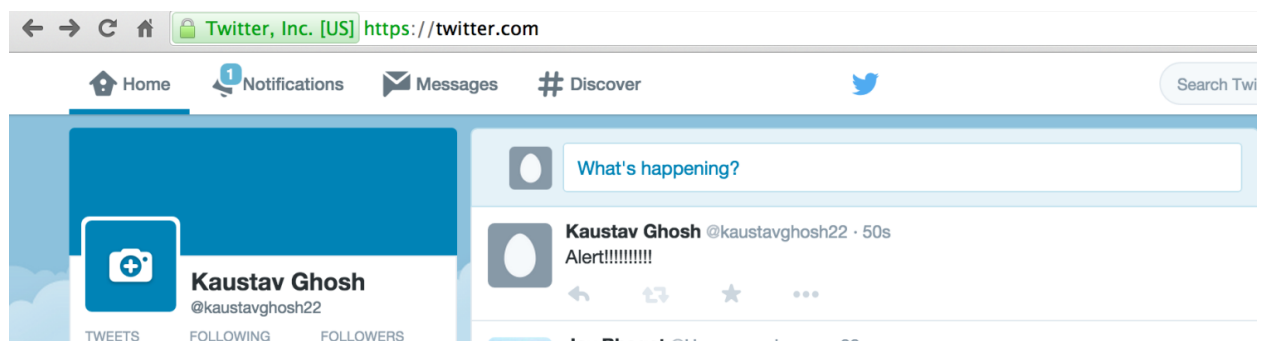
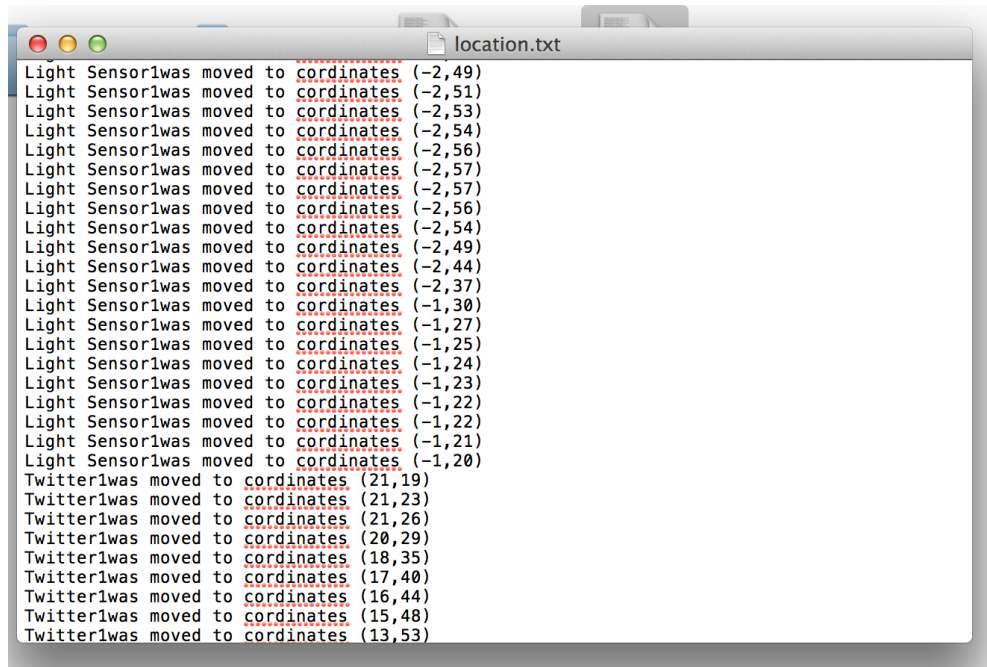


Figure 13: Tweet generated after the simulation

6. Tracking location of the nodes: Another functionality of project is it allows user to track and log of the position of the sensors in the map. As soon as we drag a node from its position to another, its co-ordinates are updated. The co-ordinates are logged into a file that is stored in the directory as the file system with the name Location.txt



```
location.txt
Light Sensor1was moved to cordinates (-2,49)
Light Sensor1was moved to cordinates (-2,51)
Light Sensor1was moved to cordinates (-2,53)
Light Sensor1was moved to cordinates (-2,54)
Light Sensor1was moved to cordinates (-2,56)
Light Sensor1was moved to cordinates (-2,57)
Light Sensor1was moved to cordinates (-2,57)
Light Sensor1was moved to cordinates (-2,56)
Light Sensor1was moved to cordinates (-2,54)
Light Sensor1was moved to cordinates (-2,49)
Light Sensor1was moved to cordinates (-2,44)
Light Sensor1was moved to cordinates (-2,37)
Light Sensor1was moved to cordinates (-1,30)
Light Sensor1was moved to cordinates (-1,27)
Light Sensor1was moved to cordinates (-1,25)
Light Sensor1was moved to cordinates (-1,24)
Light Sensor1was moved to cordinates (-1,23)
Light Sensor1was moved to cordinates (-1,22)
Light Sensor1was moved to cordinates (-1,22)
Light Sensor1was moved to cordinates (-1,21)
Light Sensor1was moved to cordinates (-1,20)
Twitter1was moved to cordinates (21,19)
Twitter1was moved to cordinates (21,23)
Twitter1was moved to cordinates (21,26)
Twitter1was moved to cordinates (20,29)
Twitter1was moved to cordinates (18,35)
Twitter1was moved to cordinates (17,40)
Twitter1was moved to cordinates (16,44)
Twitter1was moved to cordinates (15,48)
Twitter1was moved to cordinates (13,53)
```

Figure 14: Log file tracking the sensor co-ordinates

## Block Diagram

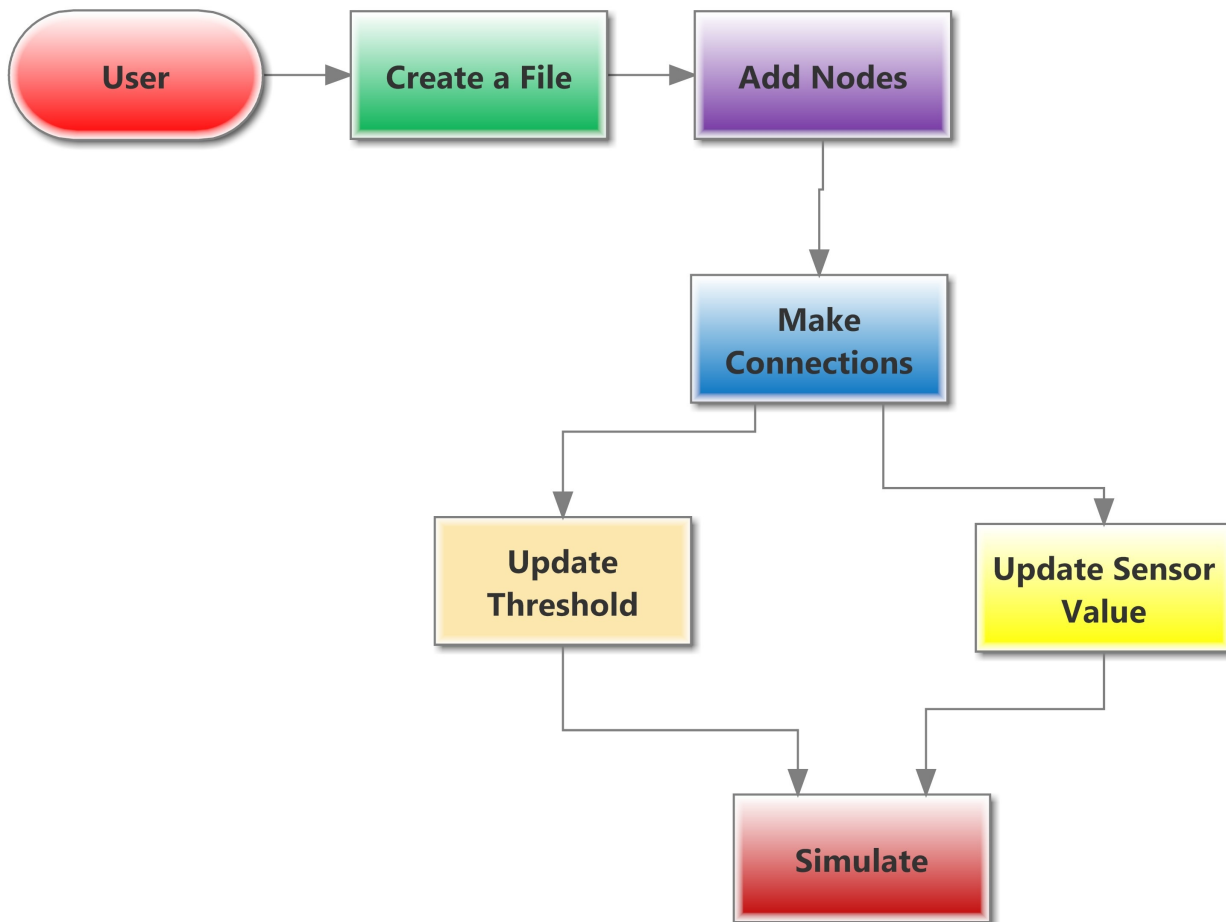


Figure 15: Depicts Block Diagram of the Application

## Conclusion

The GUI developed using Scala programming language is a simple application to interconnect various sensors and understand their interoperability. The developed project is scalable to the extent that it can be extended to include smart objects transmitting and receiving real time data. The future of IoT looks promising and the number of IoT based applications is bound to increase. This project will aide those who intent to begin the basics of logic designing of the connected devices for building efficient IoT applications