CMPE 281-01 Cloud Technologies

Project System Design Document

Spring-2016

SAN JOSÉ STATE UNIVERSITY

Submitted to:

Dr. Jerry Gao

By

Aditya Chauhan(010816676)

Ashay Argal(010822513)

Divyakumar Patel(010819822)

Shalin Amin(010823826)

Table of Contents

Table of Figures

Acknowledgement

*We would like to thank **Dr. Jerry Gao** and his assistants **Jinal Shah**, **Deepesh Eliyot** and **Roopa Vedagiri** without whom this project would not have been completed. We are very grateful to be part of those few students and sincerely appreciate your guidance throughout the course. Learning under the guidance of such humble and guiding professor was an amazing experience and express our gratitude to him.*

1. Introduction

Mobile phones and Smartphones are at fast pace becoming the dominant form of communication. The Smartphones that are used today comes with the low cost built in sensors. The Smartphones do have sensors such as gyroscope, microphone, camera, compass and accelerometer. The Smartphones that are equipped with sensors will be helpful in private sector such as Healthcare, environmental monitoring and transportation industry.

The objective of developing the app is to implement, develop and validate a mobile sensor cloud engine for supporting and managing sensors and providing an on demand sensor cloud service. The features of the app other than sensor and sensor controller management and provisioning are load balancing and scalability management, Dashboard user interface, Sensor Monitoring.

The key features that the engine would consist are: -

1. **Provision and Management of Mobile Sensor**
   This feature will allow user to manage, register and configure different type of sensors. The user can even manage connectivity of sensor with mobile sensor server.
2. **Provision and Management of Sensor Controller**
   This feature will allow user to manage different type of sensor controllers and even check their connectivity with server.
3. **Provision and Management of Scalability and Load Balancing**
   This feature will provide functioning that it can support scalable number of sensors when required to do so
4. **Virtualization of sensor clouds**
   This feature will simulate a mobile sensor by providing sensing data. Similarly virtual sensor controller will be controlling sensors.
5. **Monitoring of Mobile Sensors**
   This will help user to monitor the sensing data from a sensor.
6. **Dashboard for providing user interface to the system**
   The user will do all the provisioning using Dashboard. The user will communicate to cloud using this feature.
7. **Billing**
   In this feature we are implementing ‘pay’ feature for the user. The user will be charged according to number of sensors that he is adding and also by the area in which he wants to add.
8. **Admin Panel**
   In this feature the admin can login and check Sensors with their Area, City, Sensor Type, Owner, Owner Id and health of that particular sensor.

2. Deployment Architecture Design

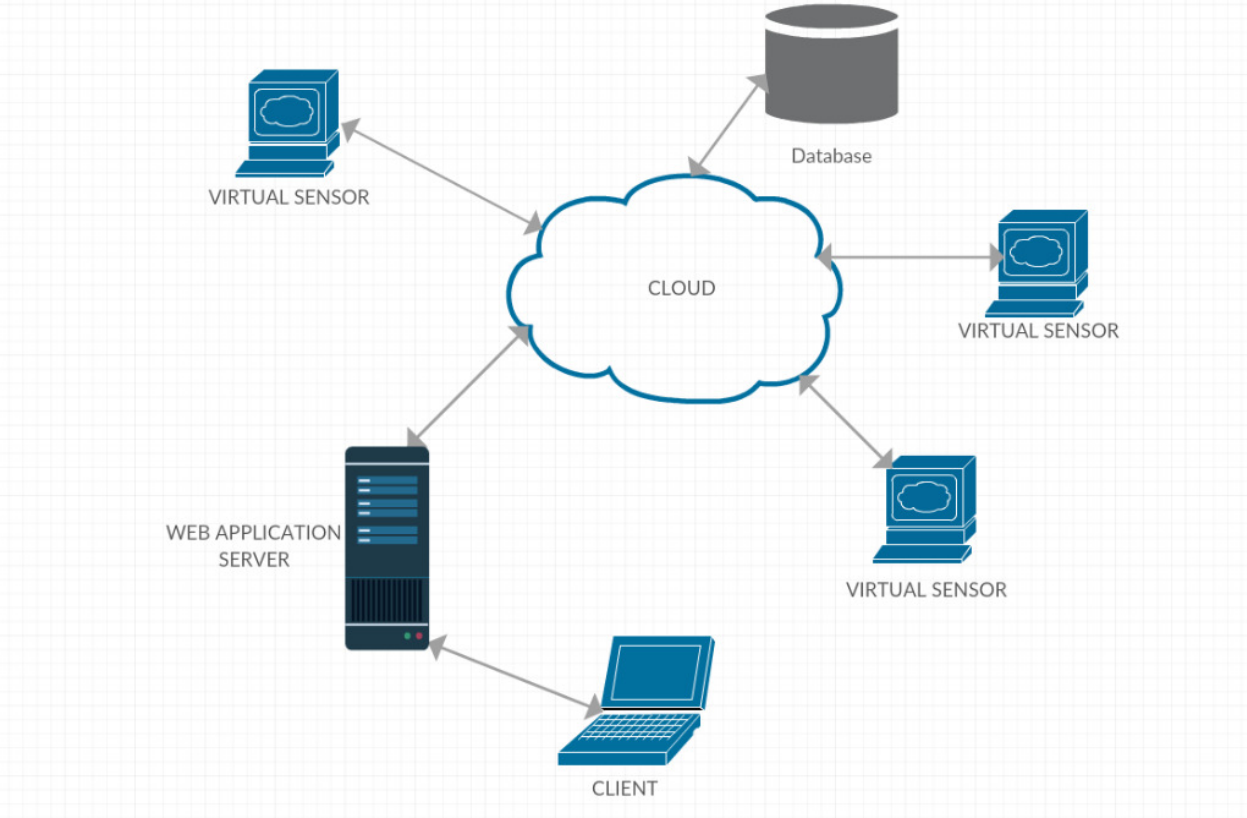


Figure 1 : Deployment Architecture

3. Application Architecture

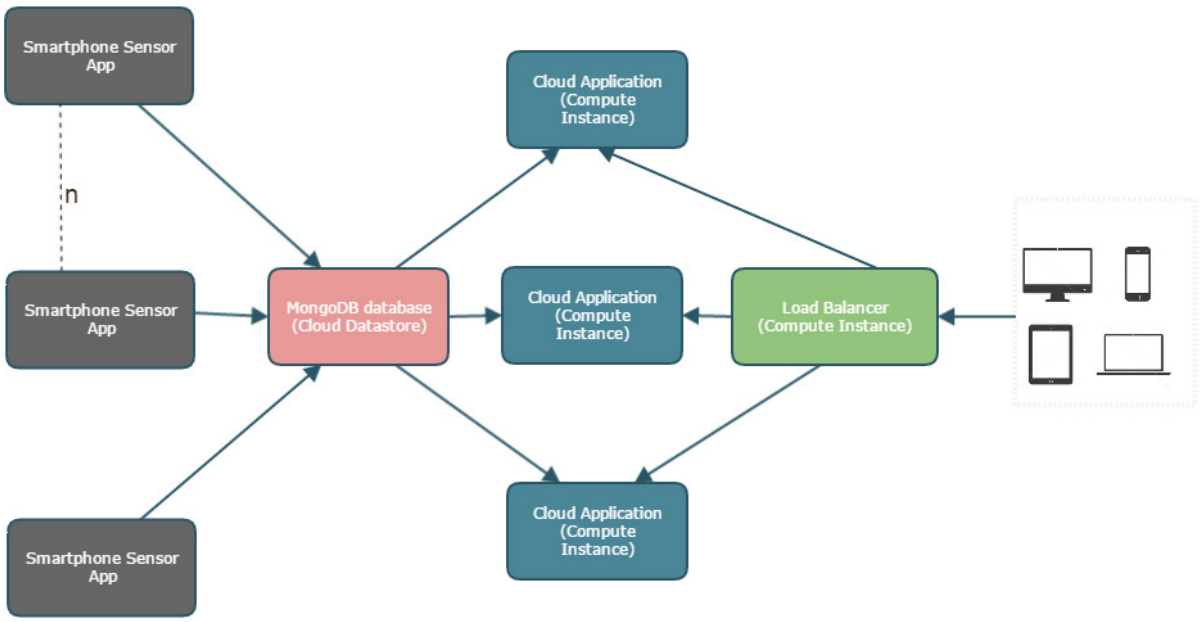


Figure 2 : Application Architecture

4. Load Balancing Algorithm

To manage the load on the server, effective load balancing is needed. The mechanism of dispersing the traffic across multiple hubs is Sensor Cloud load balancing. This mechanism lessens costs and maximizes availability of resources. Through an optimized load balancing, the resources can be better and easily be administered and can be better allocated and the scheduling can also be effective. This will help us handle the resources and will also reduce cost too.

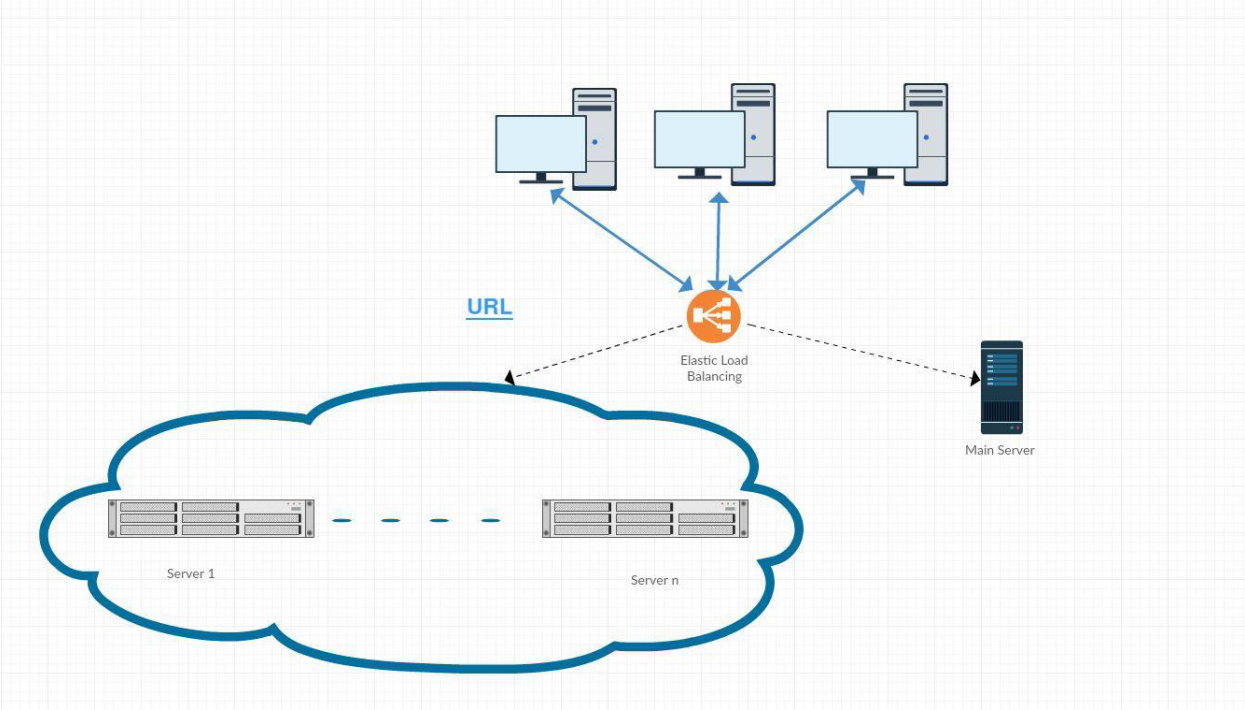
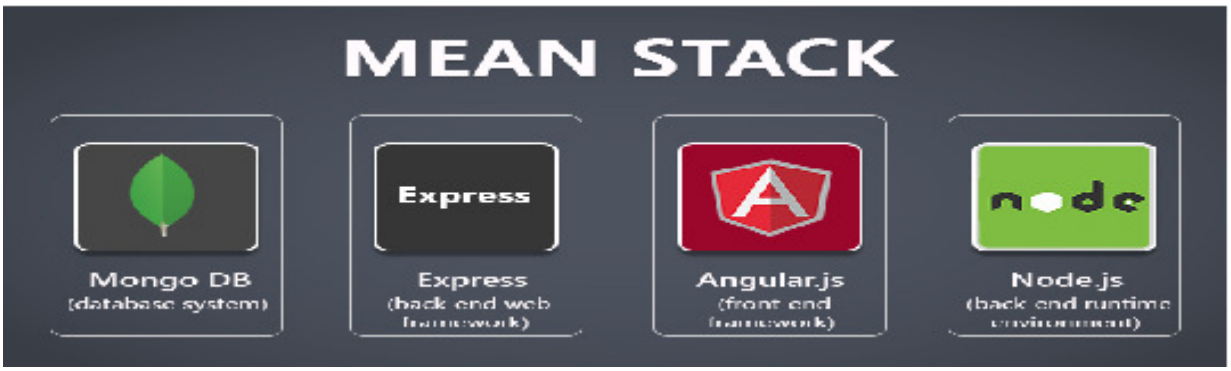


Figure 3 : Load Balancing Algorithm

5. Technology Selection


MEAN STACK
Mongo DB
(database system)
Express
Express
(back end web
framework)
Angular.js
(front end
framework)
node
Node.js
(back end runtime
environment)


openstack™


HTML
JS
CSS


Passport


android


git


CHEF™


Bootstrap

6. Strategy

Figure 4 : Strategy Architecture

Virtual Sensor:

This simulates a sensor and provides sensing data. This sensor takes the data from actual physical sensor and sends it to the user for the completion of the request

Billing:

User needs to pay for the sensors that he adds according to area. Utility billing is the universal concept that is used in this project.

Scalable:

Sensors can be scaled up and down as per the user need. The rescaling done is the product of itself. With this feature we can remarkably diminish the risks correlated with traffic overflow which causes server overflow.

On Demand:

User can add sensors and monitor them as he wants real-time or hourly. This service enables the provisioning of sensors, on demand whenever they are required instead of waiting for the subs

7. Implementation Design

7.1 Sensor Management

The user has to login first. After the user is authenticated the user can add the sensors that he/she wants to see in his/her dashboard. The sensors he wants can be selected from the dashboard. The user can even view all the sensors that he has added and all that are deactivated and needs to be added.

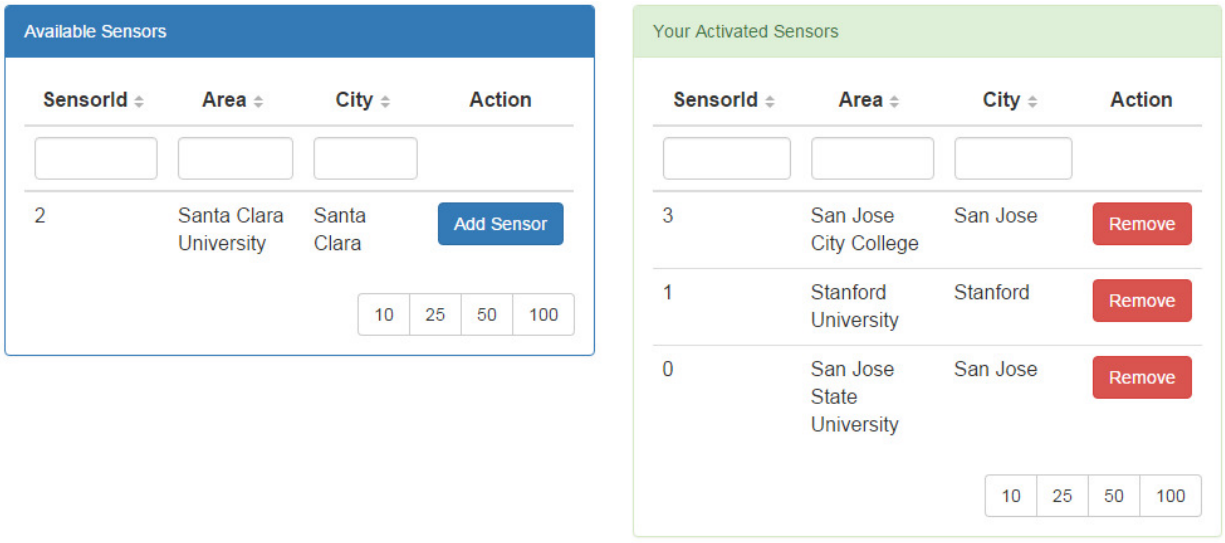


Figure 5 : Sensor Management Screenshot

7.2 Graphs

User can view temperature, humidity and pressure graphs in this menu. User can group graphs according to area and city also. By location also the user can sort the graphs that he wants to see.

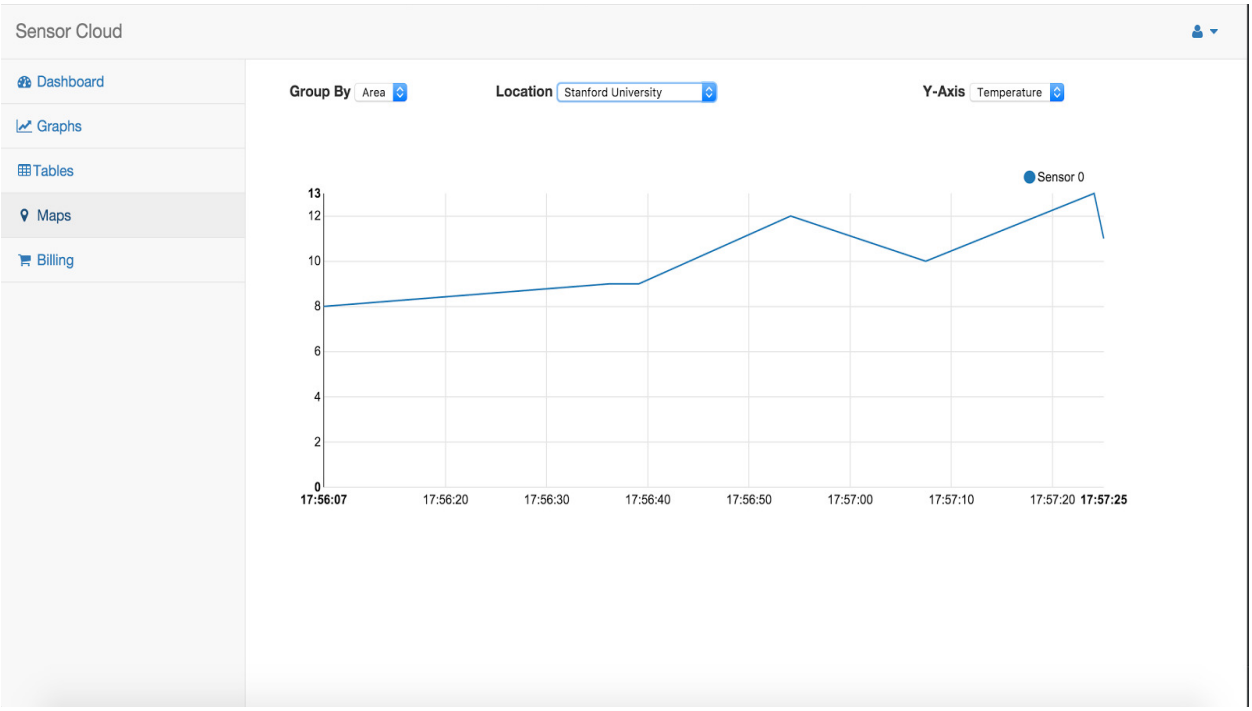


Figure 6 : Graphs Screenshot

7.3 Tables

In this the user can see all the sensor data in a table and can sort according to the sensor id, area, city, date, temperature, humidity and pressure.

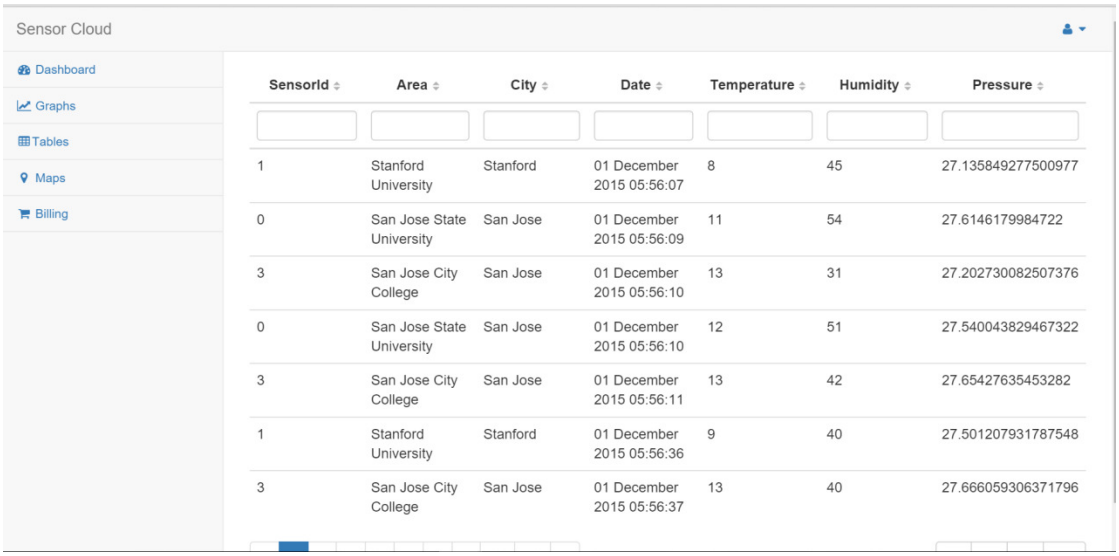


Figure 7 : Tables Screenshot

7.4 Maps

In this the user can see the sensors on the map and can even see real time sensor data on it. User can select the sensor and sensor with the area on maps would be visible.

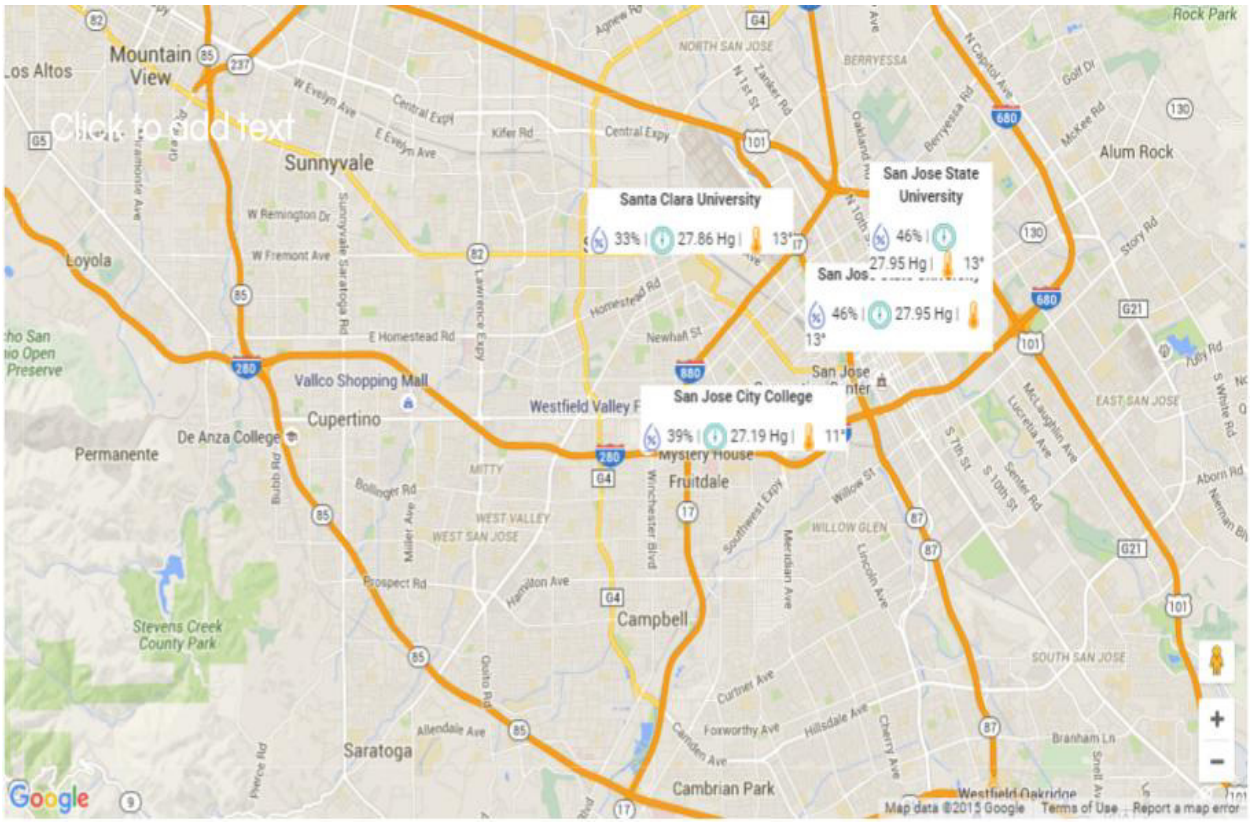


Figure 8: Maps Screenshot

7.5 Billing

User will be charged $10.00/Sensor and can add as many as he wants and will have to pay per sensor only.

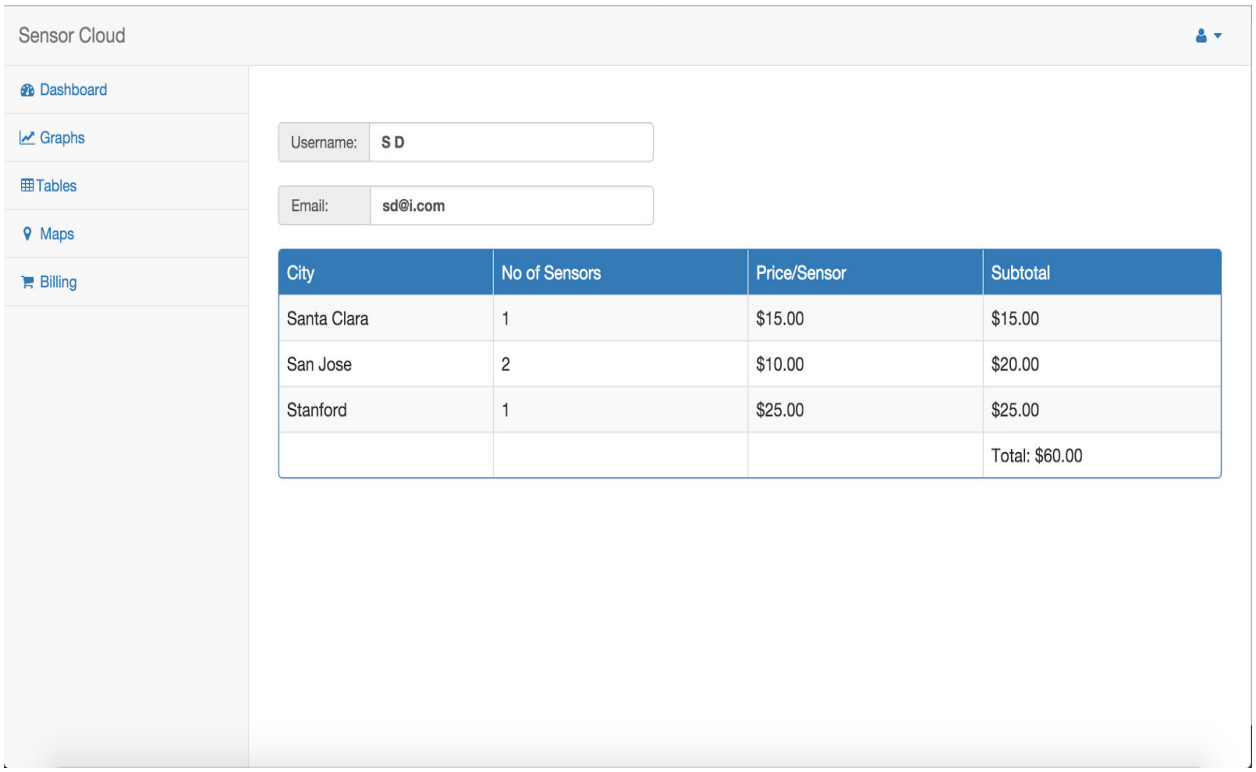


Figure 9 : Billing Screenshot

7.6 Admin Panel

The Admin can login using his User Id and Password and can keep a check on all the sensors that are being added to be monitored. The admin can view Sensor Type whether it is temperature, humidity or pressure sensor, Admin can also see the Sensor Area, City, Owner Name and Owner Id. Along with all these Admin can see the health of all the sensors also on his screen.

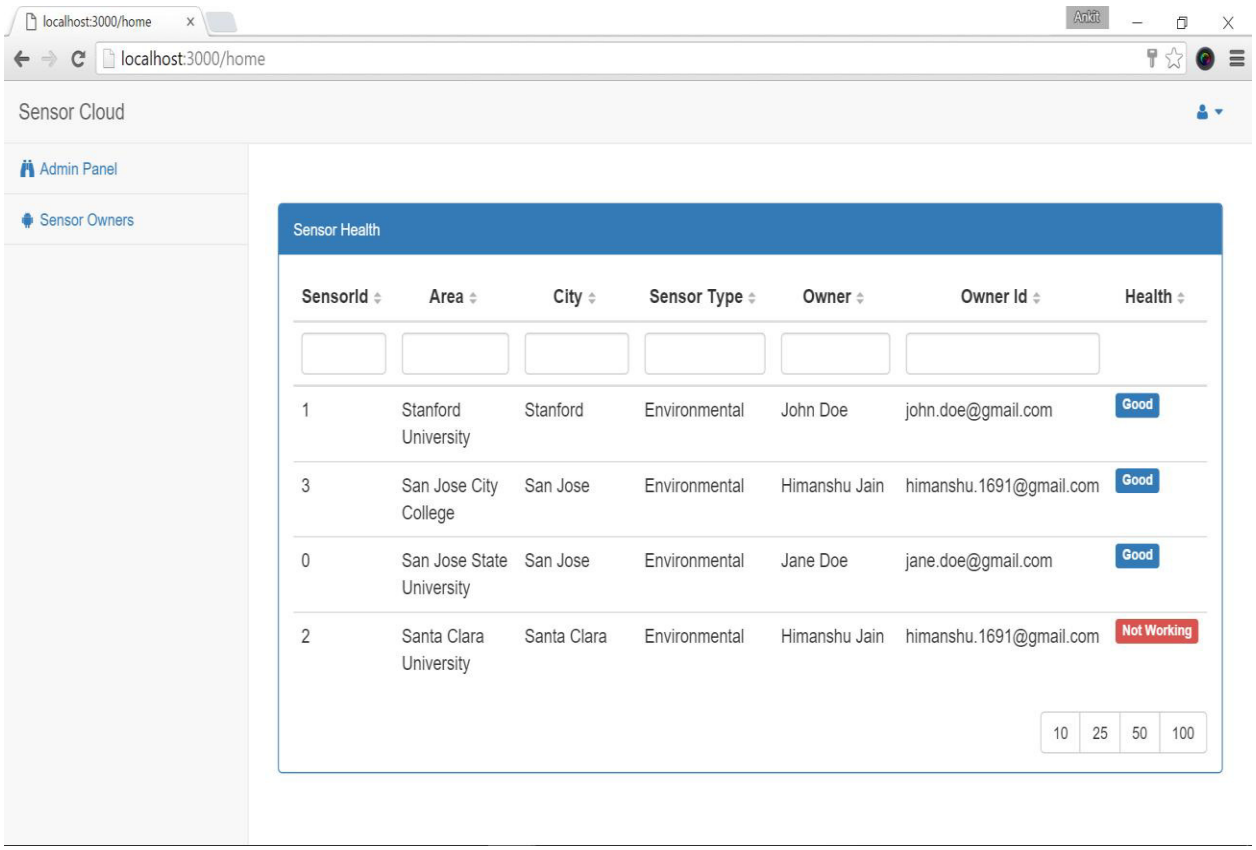


Figure 10: Admin Panel Screenshot

8. Virtual Sensors

This component works as a simulator of actual physical sensor residing on different location. This virtual sensor gets the data from actual physical sensor in appropriate format as required by the client. This component acts as a physical sensor which will provide all the sensor data, health data and status data and will be able to communicate as well with sensor hub through sensor data hub.

Input = physical sensor data.

Output = sensor data

Figure 11 : Virtual Sensors website screenshot

9. Conclusion

The user can use this web app to be updated about all the sensors data that he wants to know about. User can see historical graphs to know the change in temperature, pressure and humidity. While user is also given provision to add sensor that he wants to monitor and remove sensor that he doesn't want to.

While this serves as an ease to the user it also has high business value. The app also functions with the feature of scalability.

10. References

1. https://sjsu.instructure.com/courses/1162965/files/?preview=40037457
2. http://ieeexplore.ieee.org/xpl/login.jsp?tp=&arnumber=5560598&url=http%3A%2F%2Fieeexplore.ieee.org%2Fxpls%2Fabs_all.jsp%3Farnumber%3D5560598
3. http://en.wikipedia.org/wiki/Mobile_cloud_computing
4. http://en.wikipedia.org/wiki/Cloud_computing_architecture