**CSE 570: Project Report**

**Cooperative Spectrum Sensing on Mobile Platforms**

**Submitted By:**

**Daljeet Singh Chabbra (), Nikita Goyal (), Isha Khanna (109771959)**

**I. INTRODUCTION**

Spectrum sensing involves sensing RF spectrum to determine what power is present at what frequency. The main motivation behind spectrum sensing is to study utilization of radio spectrum and/or to discover underutilized spectrum (e.g., TV white space). Spectrum sensing on mobile platforms, e.g., on smartphones, can enable large-scale, fine-grain spectrum sensing. There has already been some work done in this direction and we have the wifi data available from smart phones. Our goal in this project is to contribute to this research by providing a UI platform similar to a dashboard to enable the users to perform particular tasks and analysis on this data.

**II. INSTALLATION & STARTING THE APP**

We suggest to use a virtualenv for running the app since it will independently keep the required packages at the versions used in this project. Following are the steps to get started with the installation:

* pip install virtualenv
* virtualenv env
* pip install –r SpectrumSensingDashboard/requirements.txt
* ./SpectrumSensingDashboard/manage.py runserver

If you are working on local host the above application will run on 127.0.0.0:8000/

Tech Stack Used:

* Python 2.7.10
* Django 1.8.6
* Mongodb 3.0.7
* HTML5
* JQuery

**III. TECHNICAL OVERVIEW**

Database used: MongoDb

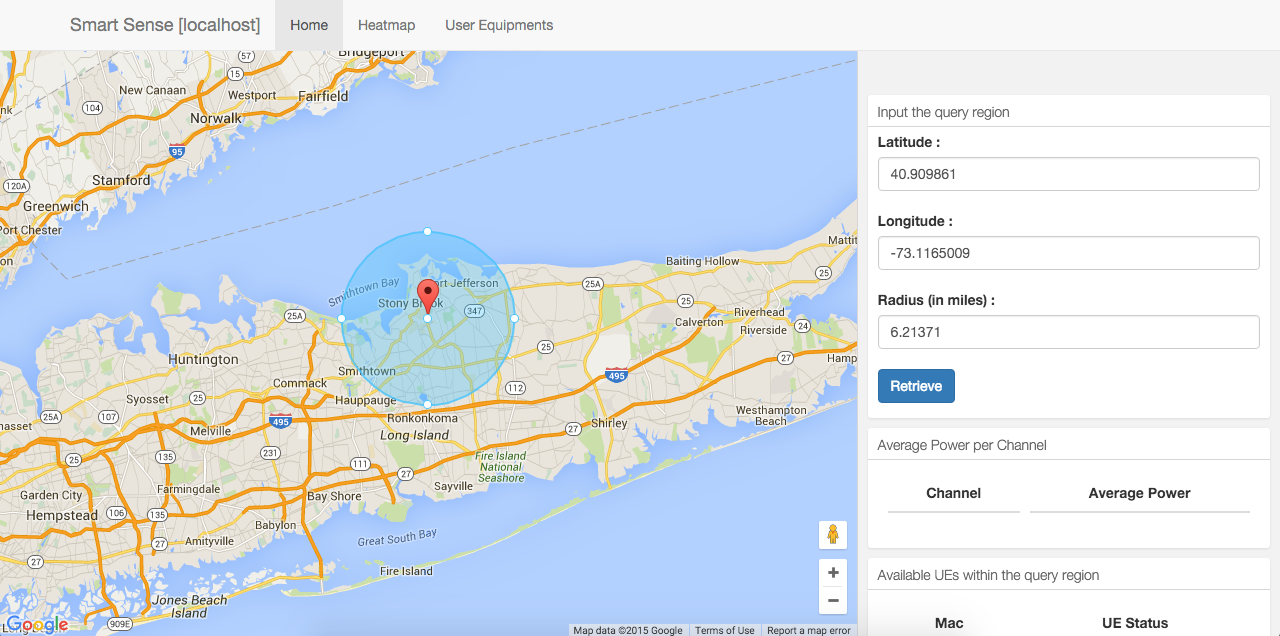
Backend: Python, Django Framework

Frontend: HTML5, JQuery, AJAX, CSS

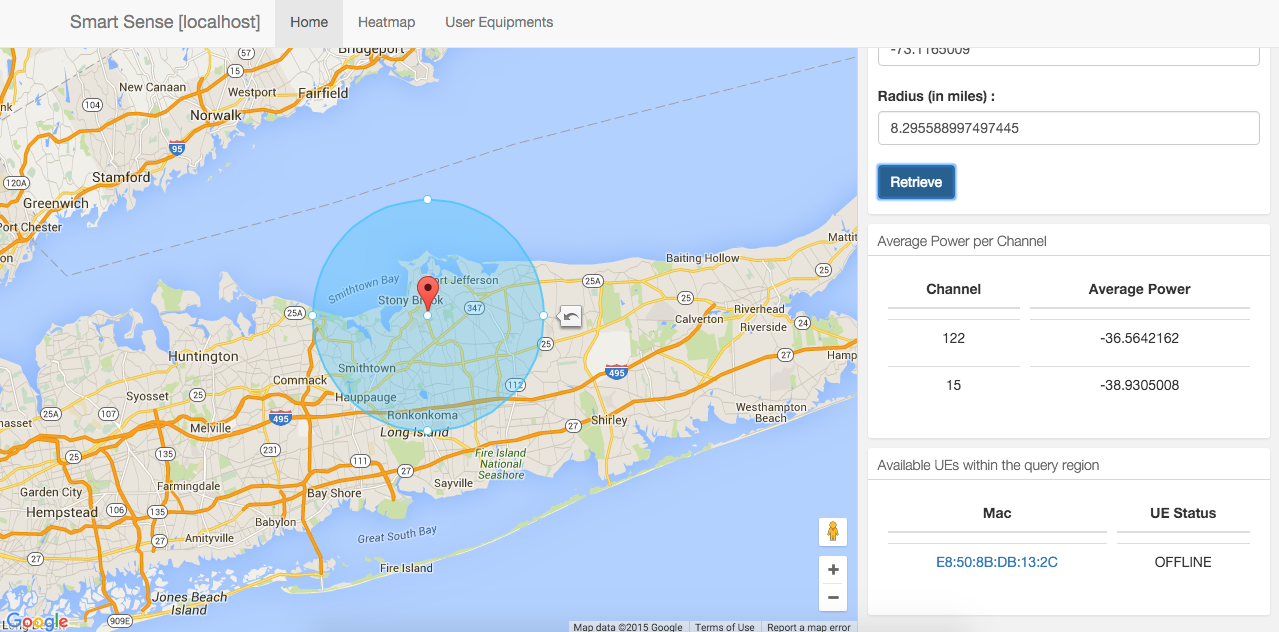
**Files Submitted:**

1. **db\_adapter.py:** The file contains the python integration with MongoDB (using pymongo). Different functions in the file serve different client calls from the views by quering and returning the data from the database. Following is a brief description for all functions in the file.
   1. getReadingsinRadius(lat, lon, radius): gives back the ue measurements within the radius range of given latitude and longitude point.
   2. getChannels(): returns all the distinct channels for which the measurements exist in the database.
   3. getUEMeasurements(channel): Gives list of all measurements that exist for a particular channel
   4. getAllRegisteredUEData(): Gives the entire data for all the registered UEs in the database.
2. **views.py:** This serves as the controller between the front-end and back-end code which renders the requested templates (htmls)
3. **templates/** folder contains all the templates for different screens of the dashboard: homepage.html, heatmap.html, ue\_details.html, uelist.html and other common files for design

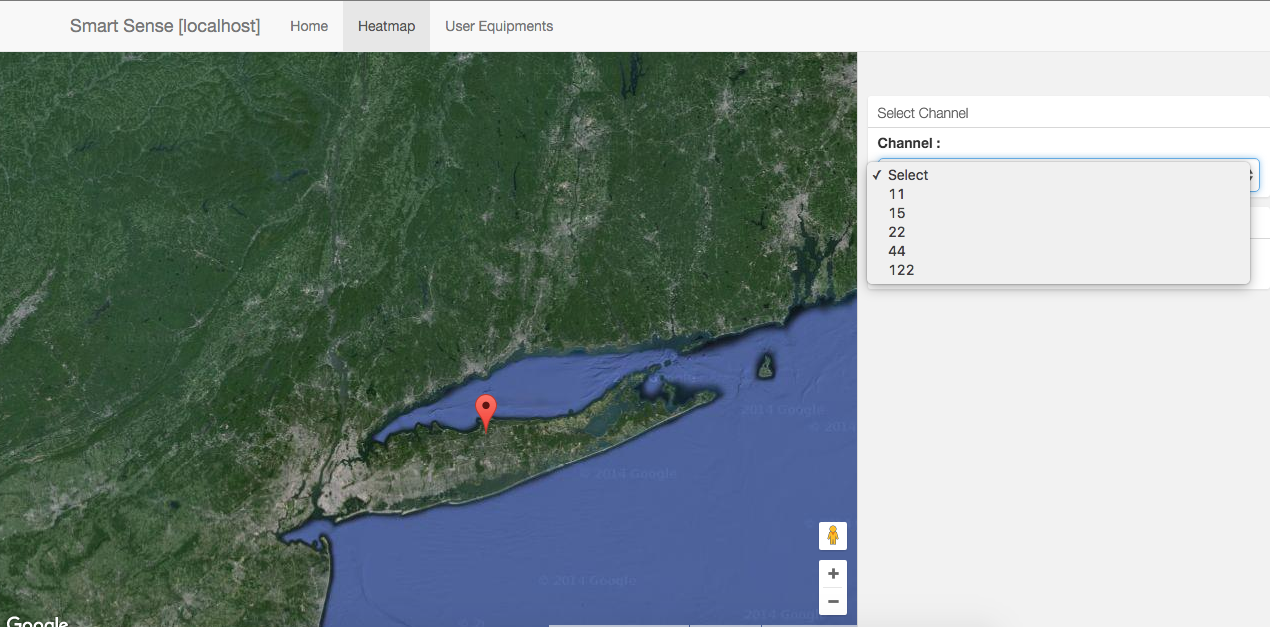
**IV. SCREENSHOTS**



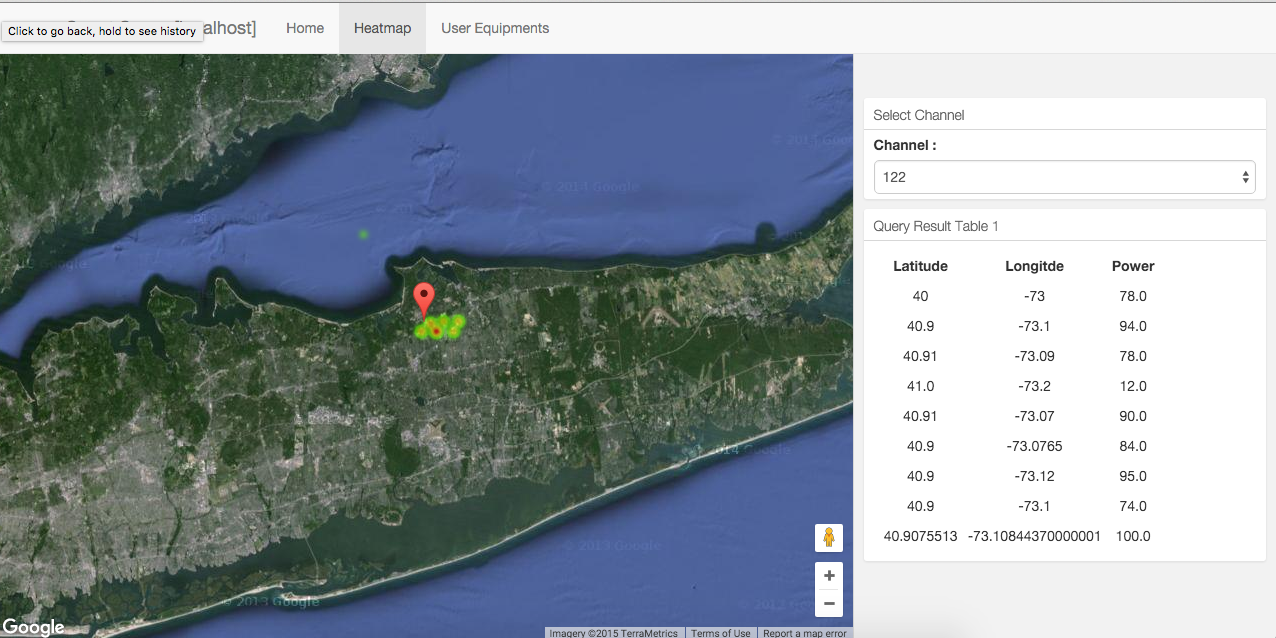
**Figure 1: Homepage: Get all measurements within the highlighted location**



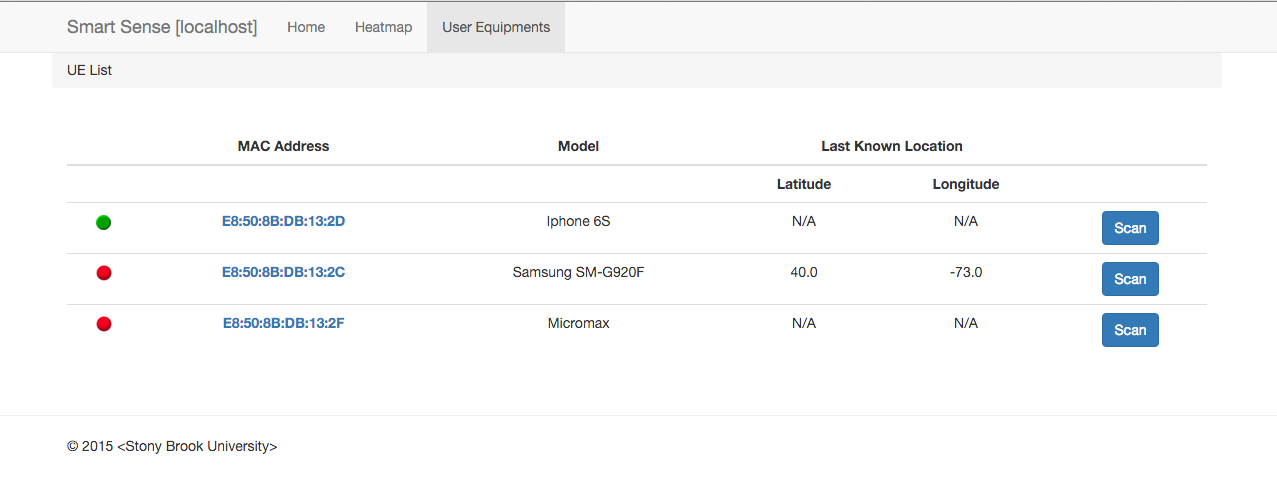
**Figure 2: Homepage: Results of the query for all reading around current location**



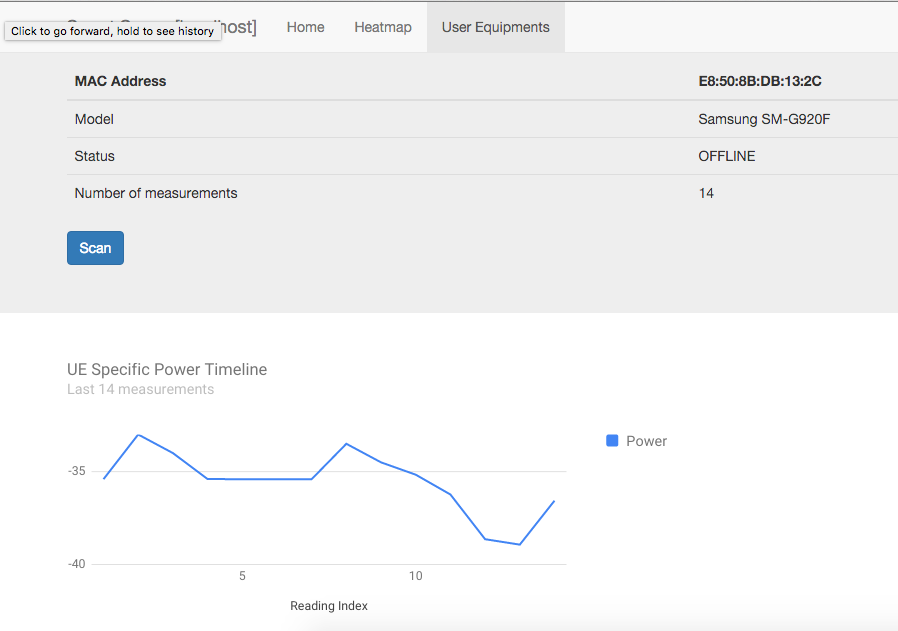
**Figure 3: Heatmap: Drop down to select all available channels.**



**Figure 4: Heatmap: Displays the heatmap of the selected channel (122 here)**



**Figure 5: UE List: List of all the Registered UEs present in the database. This page updates dynamically after regular intervals.**



**Figure 6: UE Specific: Details of a selected Registered UE with a line chart for power measured in last 20 readings. This page updates dynamically after regular intervals.**