# PillBOX

**A smart pill box for patients**

נעה יפה, אמיר ג'ונפור, פלג נויפלד

## 1. Introduction

PillBOX is a smart pill box for patients. After the pharmacist fills it with pills and configures pill taking hours, the patient can take the device home and configure the Wi-Fi credentials and a contact list on the PillBOX website. The PillBOX can then be placed in a visible location and begin its work. When it is time to take a pill, PillBOX lights up the exact cell in which the pills are located using a red LED. The PillBOX uses a magnetic hall effect sensor to detect if the cell has been opened after the LED has been lit, thus determining if the pill was taken or not. If after 30 minutes the pill was not taken, an SMS alert is sent to the pre-configured contact list, letting them know that the pills weren't taken, and that they should contact the patient (See appendix A for a flow chart describing this procedure). The motivation for such a product is that most of the people who use such pill boxes are elderly people, who tend to forget taking their pills or which pills they are supposed to take. Our system alerts them via LED and alerts family members of friends if pills were not taken, allowing the contacts to take action to make sure the pills are taken on time, each time.

## 2. Components (See Appendix B for a diagram of the system, and Appendix C for sample screenshots)

I. Texas Instruments CC1350 LaunchPad – runs the BLE app and the main PillBOX application logic, including sensor logic

II. Adafruit feather HUZZAH ESP8266 – connected to the CC1350 for Wi-Fi functionality.

III. PillBOX Android App – allows the pharmacist to connect to the CC1350 using BLE and configure current time and pill taking hours.

IV. PillBOX Website – written in python, allows clients to configure their list of contacts, and an admin user to supervise over all clients. The website was uploaded to Amazon EC2 cloud computing service onboard an ubuntu 16.04 virtual machine. The website also exposed a REST API for the feather huzzah to trigger alerts from the board itself.

V. Amazon SNS – the Simple Notification Service by Amazon cloud services was used to manage clients contact lists and send them SMS notifications when triggered by the board. The services' API was accessed from the PillBOX website.

3. **Texas Instruments CC1350 LaunchPad**

The CC1350 LaunchPad runs 2 tasks (threads):

I. BLE task – allows connection to the board from the PillBOX app, used by the pharmacist. The BLE task exposes 5 variables to the Android app user: current hour and current minutes, and morning, noon and evening pill taking time. The app user can view and change those values using the Android app. The task also defines the unique name of the board: "Board_XXX" with XXX being the serial number of the specific board.

II. PillBOX logic – the first thing this task has to do is calculate the different time intervals and get our internal clock started. The task uses the variables from the BLE task to determine the length of the intervals.
The next thing this task does is manage the clock. Because we chose not to use an RTC component, the task “sleeps” for the interval lengths between pill taking times. When it is time to take a pill, the task awakens, turns on the LED attached to the relevant cell of the pillBOX and checks (at 1 second intervals for 30 minutes) if the cell has been opened. If the cell has been opened, we assume the pills have been taken and the LED is turned off. If at the end of 30 minutes the cell has not been opened, we assume the pills have not been taken and signal the ESP board using a GPIO pin that an alert SMS has to be sent to the contact list defined on the PillBOX website.

4. **Adafruit feather HUZZAH ESP8266**

the feather HUZZAH was added to the project to add Wi-Fi functionality, which the CC1350 lacks. The CC1350 and the feather HUZZAH are connected between them with a GPIO jumper wire, with the CC1350 pin in output mode, and the feather pin in input mode. The default value of the CC1350 output pin is 1, and when the CC1350 determines that an alert should be sent via SMS, it lowers the output pin to 0. This is detected by the feather HUZZAH, which in turns sends an HTTP POST request to the PillBOX website. The POST request contains the board id, so the PillBOX website knows which board sent it.

The feather HUZZAH firmware performs the following steps:

I. On device boot, it reads the first 96 bytes from its nonvolatile memory, parses them into username and password, and tries to connect to Wi-Fi with them.
II. If it fails, the board launches an access point. The user can then connect to the board with any Wi-Fi enabled device (smartphone, tablet, PC, etc) and browse the ip address 192.168.4.1, Where he can set the SSID and password for his home Wi-Fi.
III. After setting the SSID and password, the board will write those values into its nonvolatile memory and reset. When it boots, it tries to connect to Wi-Fi using the new credentials. From that moment, assuming it was successful, the board listens constantly on pin #14, defined in input mode. if the CC1350 drops the value on this pin from 1 to 0, the board sends a POST request to the address "/api/autopublish" in the PillBOX website, containing the board id.
IV. The board id and PillBOX website address are hard coded into every board and are not configurable by the user.
V. An orange LED on the board is lit to indicate a successful connection to Wi-Fi.

## 5. **PillBOX Android App**

Based on the functionality of the Bluetooth Low Energy sample app, we built an android app capable of:

- Turning on the Android’s Bluetooth connection after requesting the user’s permission
- Scanning for BLE devices for a limited amount of time (the user can then tap the “Scan” button for another scan)
- Displaying only CC boards in the results list (by setting a filter to show devices whose MAC address are within a specific predefined range)

- Connecting to a BLE device and displaying its GATT services and characteristics
- Reviewing and changing the current values of characteristics
- Writing the values to the BLE device

The app is used by the pharmacist to configure the board’s current time and pill taking hours. After a value is set in the app and written to the BLE device, its field is colored for feedback and confirmation.

## 6. PillBOX Website

**The website was written in Python, and has two main functionalities:**

I. Provide a GUI for patients to configure their contact list for SMS alerts. The configuration is saved in Amazon SNS – every board has its own topic in SNS, and each contact is a subscriber in the topic.

II. Provide a REST API for the CC1350 and feather HUZZAH, so they can send the website a POST request, and the website then approaches Amazon SNS via the AWS API and publishes the clients' topic.

**The website was written using the following libraries:**

I. Boto3 – a python library for Amazon Web Services API. This package links the website to Amazon SNS and allows us to configure clients (add or delete boards, add or delete contacts for a specific board, send an SMS to all subscribers of a board, etc.).
This library requires credentials to access the Amazon SNS account. In our system, the credentials are in the file "cred" in the main website library. The first line is the user access key, and the second line is the secret key.

II. Flask – a python framework for web development. We also use the extension flask-restful, which provides an easy interface to incorporate REST API within a Flask app.

**The website contains the following pages or URLs:**

I. Login page ("/login"): this is also the index page. This page allows user / admin login.

The login system is based on cookies, a built-in function in Flask. Each user has a file in the directory "users", within the file there is a single line for the password.

II. Client display page ("/client/<board_id>/"): the main page for a client. This page shows the clients contacts list (with a button to delete each contact), and three additional links:

   a) Add a new contact ("/client/<board_id>/add") – a sub-page which allows users to add new contacts to their contact list. This action adds a subscriber to the client's topic in the Amazon SNS.

   b) Send a test SMS ("/client/<board_id>/testpublish") – a page which allows users to send a test SMS message to their contact list, to check that alerts are functioning properly. The user provides the message and the sender name. This action uses the Amazon API to publish to the client's topic.

   c) Logout.

III. Admin display page ("/admin/"): displayed only when the admin logs in, and an admin cookie is set. The admin sees all the current boards (clients), can enter their pages, and create new clients ("/register") – creating a new client opens a new file for the user in the "users" directory, and opens a new topic in the Amazon SNS.

IV. Logout page ("/logout"): deletes the cookies so the user is logged out of the system.

**The REST API between the feather HUZZAH and the PillBOX website:**

I. The website contains the following REST url: "/api/autopublish". This is not a real page, but a REST URL which waits for POST requests.

II. Whenever the CC1350 lowers its output pin from 1 to 0, the feather HUZZAH is triggered. It then sends a POST request via Wi-Fi to the PillBOX "/api/autopublish" URL.

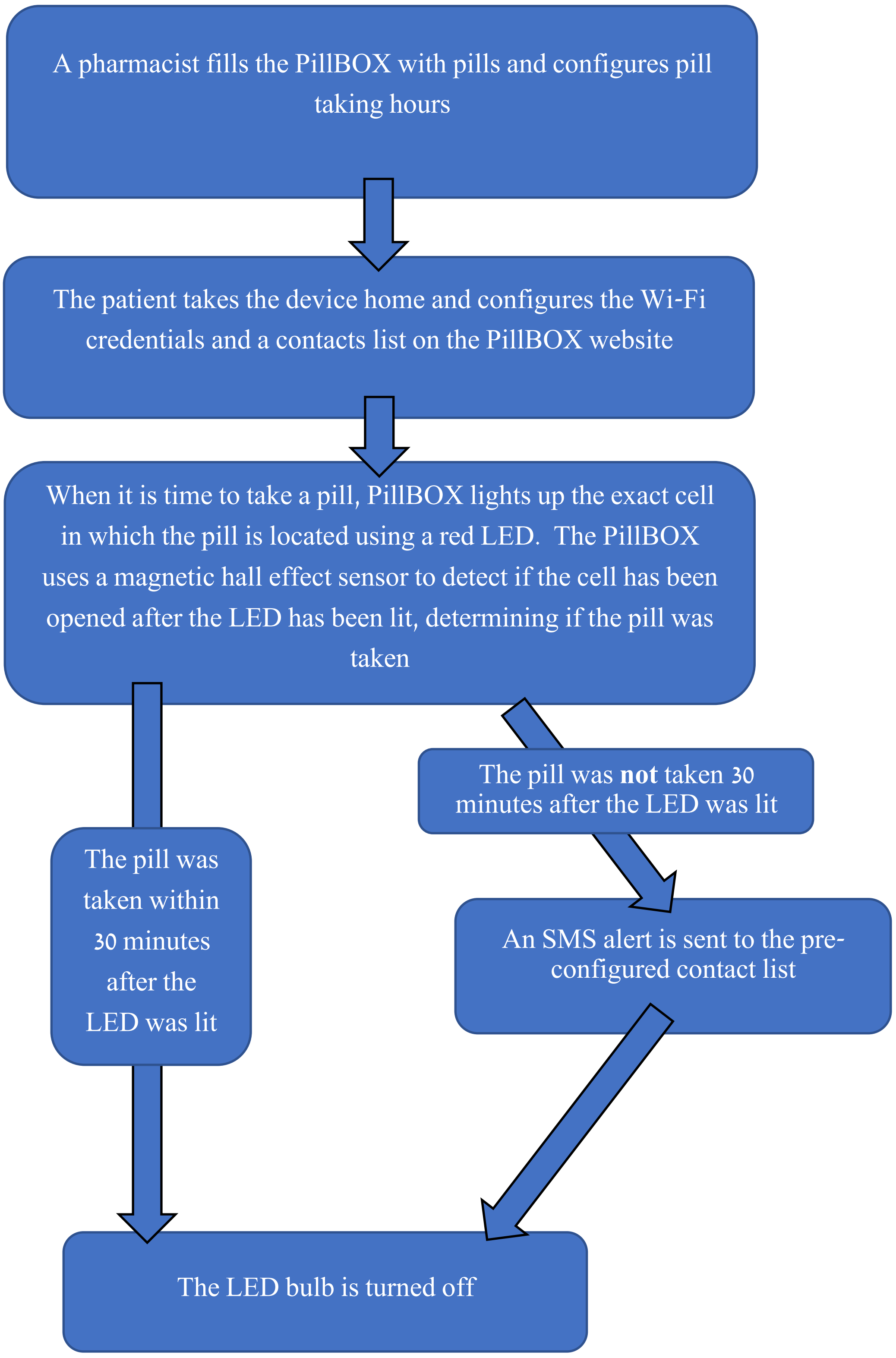
III. The POST request contains only the serial number of the board – which corresponds to a client id. The website then uses the Amazon API to publish an alert message to this client's subscribers, telling them that the pills were not taken. The exact message is not configurable by the user and is set in the file "rest.py".

7. **Amazon SNS:**

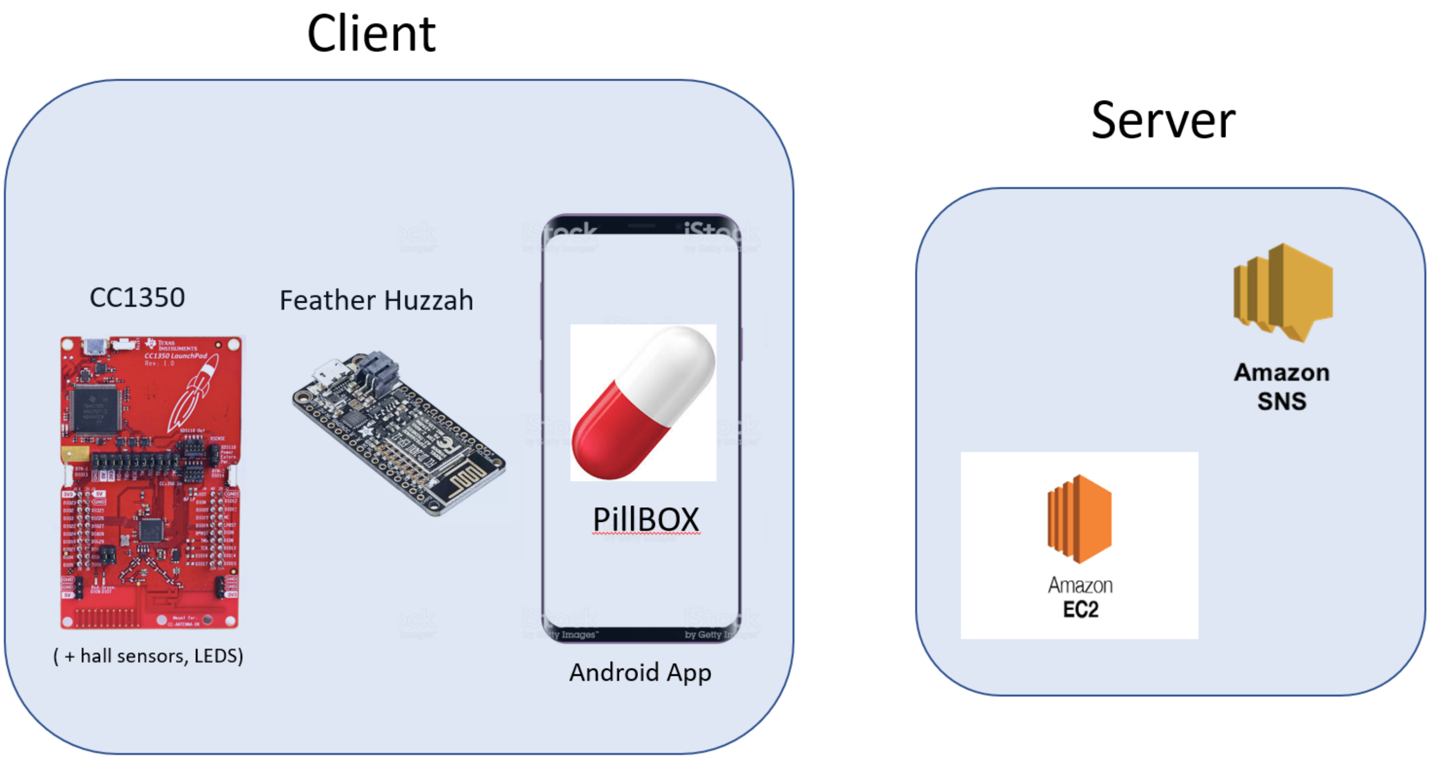
   We used the Simple Notification Service by amazon to manage clients contact lists and send the actual SMS alerts. This allows our system to be standalone and not be dependent on a permanent connection to a smartphone. It only requires a working Wi-Fi connection.

   The Amazon SNS is a **publish-subscribe** mechanism. It has topics (for us, each board, or client, is a topic) and each topic has subscribers (the subscribers is the clients contact list). After the topic and the subscribers list is set, SNS allows to publish to a specific topic – which is what we do when we send an alert to a client contact list. The SNS functionality is managed from out website using Boto3 library.

## Appendix A: A flow Chart of the System


A pharmacist fills the PillBOX with pills and configures pill taking hours
The patient takes the device home and configures the Wi-Fi credentials and a contacts list on the PillBOX website
When it is time to take a pill, PillBOX lights up the exact cell in which the pill is located using a red LED. The PillBOX uses a magnetic hall effect sensor to detect if the cell has been opened after the LED has been lit, determining if the pill was taken
The pill was taken within 30 minutes after the LED was lit
The pill was **not** taken 30 minutes after the LED was lit
An SMS alert is sent to the pre-configured contact list
The LED bulb is turned off

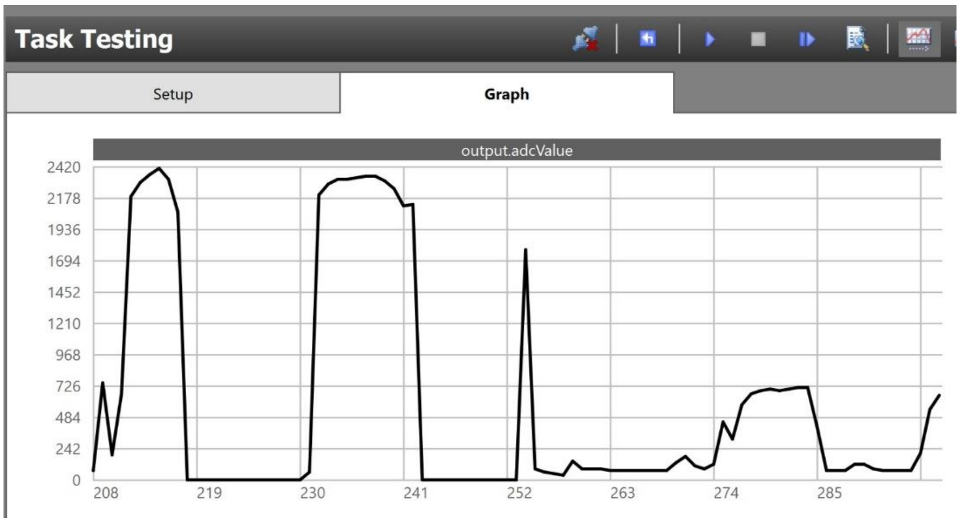
## Appendix B: A Diagram of the System



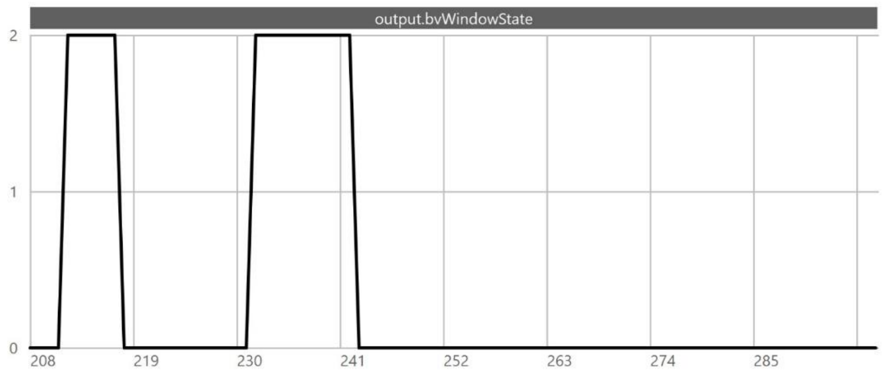
# Appendix C: Screenshots from the System Components

I. <u>Texas Instruments CC1350 LaunchPad</u> – Screenshots from Sensor Control Studio.

Determining threshold of Hall sensor using amplitude values of magnetic field:

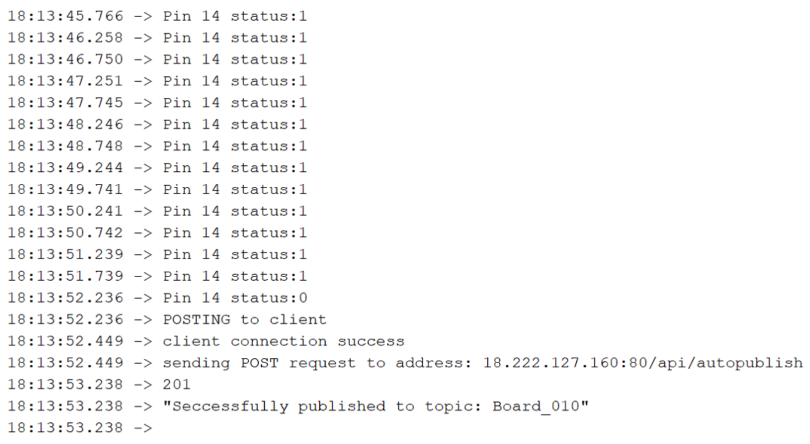


After determining the threshold, the graph shows whether the cell is open or not:



II. Adafruit feather HUZZAH ESP8266 – Screenshot from Arduino IDE.

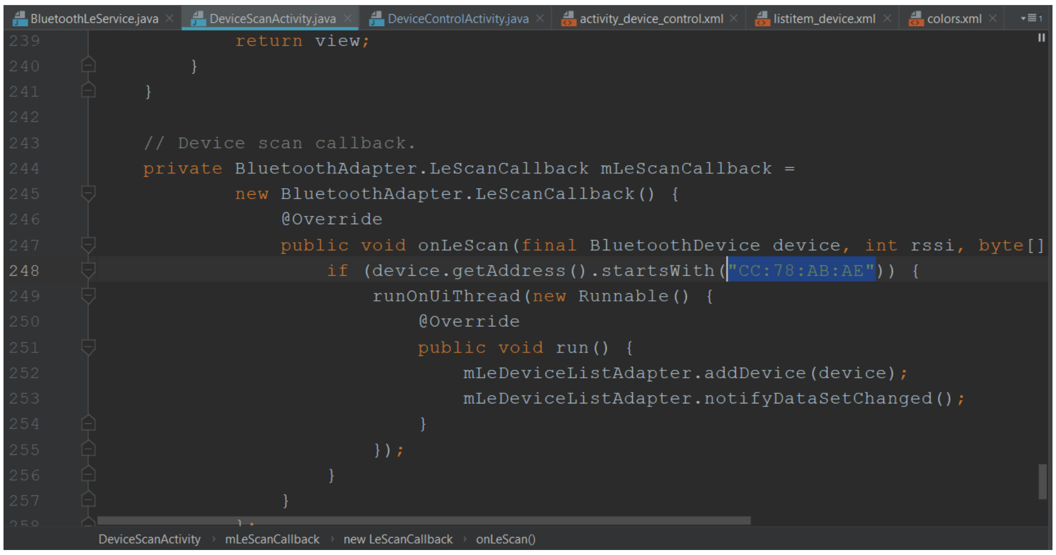
Monitoring one of ESP's pins to detect when a message is to be sent:



```
18:13:45.766 -> Pin 14 status:1
18:13:46.258 -> Pin 14 status:1
18:13:46.750 -> Pin 14 status:1
18:13:47.251 -> Pin 14 status:1
18:13:47.745 -> Pin 14 status:1
18:13:48.246 -> Pin 14 status:1
18:13:48.748 -> Pin 14 status:1
18:13:49.244 -> Pin 14 status:1
18:13:49.741 -> Pin 14 status:1
18:13:50.241 -> Pin 14 status:1
18:13:50.742 -> Pin 14 status:1
18:13:51.239 -> Pin 14 status:1
18:13:51.739 -> Pin 14 status:1
18:13:52.236 -> Pin 14 status:0
18:13:52.236 -> POSTING to client
18:13:52.449 -> client connection success
18:13:52.449 -> sending POST request to address: 18.222.127.160:80/api/autopublish
18:13:53.238 -> 201
18:13:53.238 -> "Seccessfully published to topic: Board_010"
18:13:53.238 ->
```

III. PillBOX Android App – Screenshot from Android Studio.

The function responsible for displaying BLE scan results. The highlighted line filters out any BLE device that is not a CC1350 board, using MAC address (to simulate filtering out any device that is not a PillBOX):

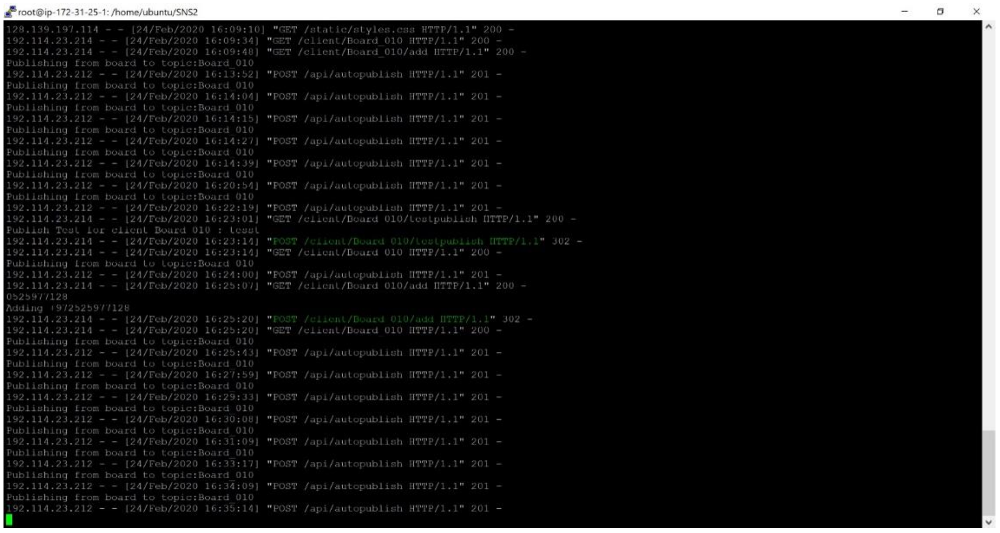

BluetoothLeService.java
DeviceScanActivity.java
DeviceControlActivity.java
activity_device_control.xml
listitem_device.xml
colors.xml
DeviceScanActivity › mLeScanCallback › new LeScanCallback › onLeScan()

```
                return view;
            }
        }

        // Device scan callback.
        private BluetoothAdapter.LeScanCallback mLeScanCallback =
                new BluetoothAdapter.LeScanCallback() {
                    @Override
                    public void onLeScan(final BluetoothDevice device, int rssi, byte[]
                        if (device.getAddress().startsWith("CC:78:AB:AE")) {
                            runOnUiThread(new Runnable() {
                                @Override
                                public void run() {
                                    mLeDeviceListAdapter.addDevice(device);
                                    mLeDeviceListAdapter.notifyDataSetChanged();
                                }
                            });
                        }
                    }
```

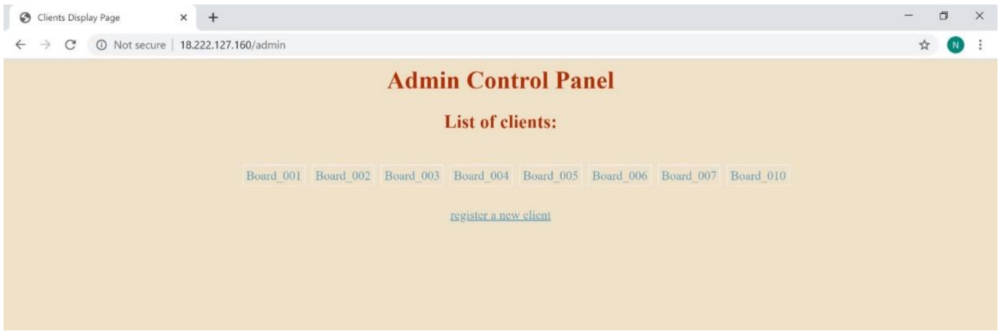
IV. <u>PillBOX Website</u> – Screenshots from Amazon’s EC2 service (provides web servers storing).

Monitoring Notifications server using HTTP packets. The lines starting with “Publishing from board to topic” stand for an SMS sent to a client’s contacts list.


root@ip-172-31-25-1: /home/ubuntu/SNS2

```
128.139.197.114 - - [24/Feb/2020 16:09:10] "GET /static/styles.css HTTP/1.1" 200 -
192.114.23.214 - - [24/Feb/2020 16:09:34] "GET /client/Board_010 HTTP/1.1" 200 -
192.114.23.214 - - [24/Feb/2020 16:09:48] "GET /client/Board_010/add HTTP/1.1" 200 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:13:52] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:14:04] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:14:15] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:14:27] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:14:39] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:20:54] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:22:19] "POST /api/autopublish HTTP/1.1" 201 -
192.114.23.214 - - [24/Feb/2020 16:23:01] "GET /client/Board_010/testpublish HTTP/1.1" 200 -
Publish Test for client Board_010 : testt
192.114.23.214 - - [24/Feb/2020 16:23:14] "POST /client/Board_010/testpublish HTTP/1.1" 302 -
192.114.23.214 - - [24/Feb/2020 16:23:14] "GET /client/Board_010 HTTP/1.1" 200 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:24:00] "POST /api/autopublish HTTP/1.1" 201 -
192.114.23.214 - - [24/Feb/2020 16:25:07] "GET /client/Board_010/add HTTP/1.1" 200 -
0525977128
Adding +972525977128
192.114.23.214 - - [24/Feb/2020 16:25:20] "POST /client/Board_010/add HTTP/1.1" 302 -
192.114.23.214 - - [24/Feb/2020 16:25:20] "GET /client/Board_010 HTTP/1.1" 200 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:25:43] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:27:59] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:29:33] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:30:08] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:31:09] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:33:17] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:34:09] "POST /api/autopublish HTTP/1.1" 201 -
Publishing from board to topic:Board_010
192.114.23.212 - - [24/Feb/2020 16:35:14] "POST /api/autopublish HTTP/1.1" 201 -
```

Controlling list of clients in Amazon web server:


Clients Display Page

Not secure | 18.222.127.160/admin

**Admin Control Panel**

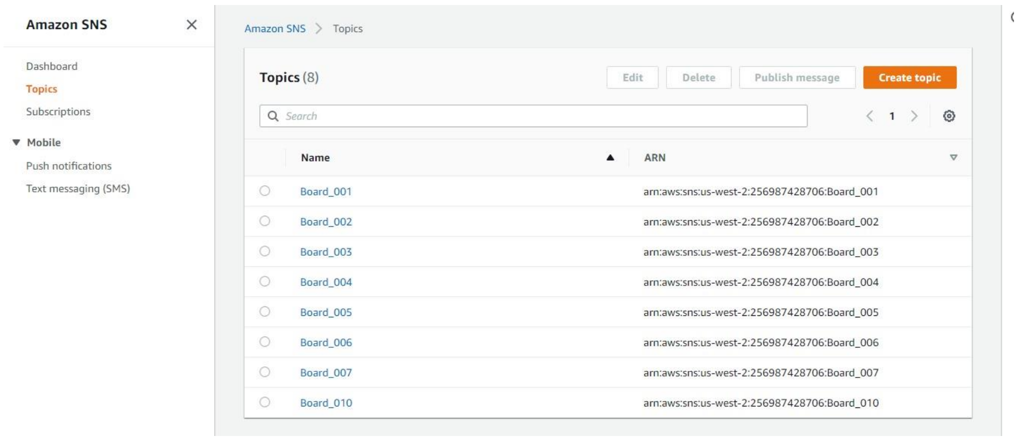
**List of clients:**

Board_001 Board_002 Board_003 Board_004 Board_005 Board_006 Board_007 Board_010

register a new client

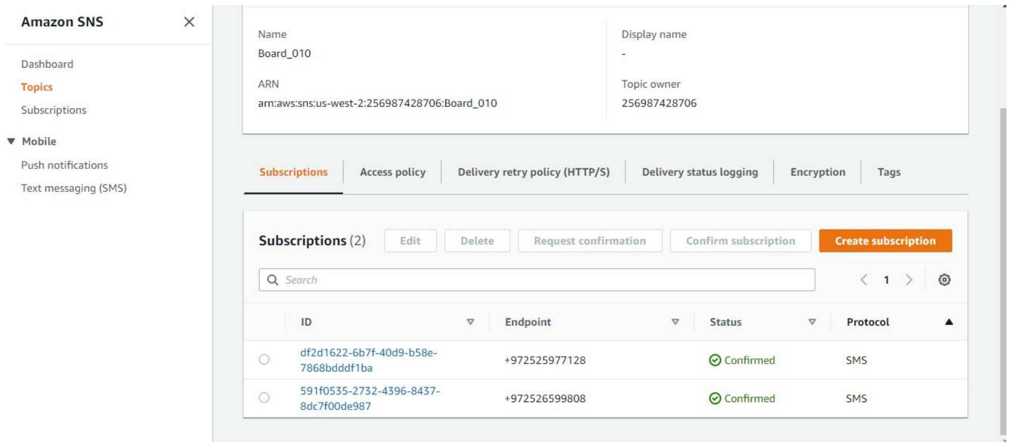
V. Amazon SNS – Screenshots from the Simple Notification Service by Amazon:

Controlling the list of clients:



| Name | ARN |
|---|---|
| Board_001 | arn:aws:sns:us-west-2:256987428706:Board_001 |
| Board_002 | arn:aws:sns:us-west-2:256987428706:Board_002 |
| Board_003 | arn:aws:sns:us-west-2:256987428706:Board_003 |
| Board_004 | arn:aws:sns:us-west-2:256987428706:Board_004 |
| Board_005 | arn:aws:sns:us-west-2:256987428706:Board_005 |
| Board_006 | arn:aws:sns:us-west-2:256987428706:Board_006 |
| Board_007 | arn:aws:sns:us-west-2:256987428706:Board_007 |
| Board_010 | arn:aws:sns:us-west-2:256987428706:Board_010 |

Controlling a list of a client's subscriptions:



Name
Board_010

Display name
-

ARN
arn:aws:sns:us-west-2:256987428706:Board_010

Topic owner
256987428706

| ID | Endpoint | Status | Protocol |
|---|---|---|---|
| df2d1622-6b7f-40d9-b58e-7868bdddf1ba | +972525977128 | Confirmed | SMS |
| 591f0535-2732-4396-8437-8dc7f00de987 | +972526599808 | Confirmed | SMS |