ECOBINS IN SMART CITIES

Final Project Report

Paulami Ray
MSIM, School of Information Sciences
University of Illinois Urbana Champaign
paulami2@illinois.edu

Garima Garg
MSIM, School of Information Sciences
University of Illinois Urbana Champaign
garimag2@illinois.edu

Milind Mathur
MSIM, School of Information Sciences
University of Illinois Urbana Champaign
milindm2@illinois.edu

Kumkum Yadav
MSIM, School of Information Sciences
University of Illinois Urbana Champaign
kumkumy2@illinois.edu

Abstract – Eco bins in smart cities would help in better waste management with reduced time and labor. This could be implemented with the application of Internet of Things where the garbage levels would be measured through sensors and tracked through mobile app.

Index Terms – garbage bins, IoT, IR sensors, mobile app, google maps

I. INTRODUCTION

Waste generation has been improvised over the years now and effective methods to collect the garbage is the utmost need of the hour in any city across the world. Due to population explosion the garbage is produced in humongous amounts these days. According to the Los Angeles Times, nations generate 1.3 billion tons of organic, inorganic and electronic wastes per annum with an anticipated value of 4 billion tons by the 2100.

As a stepping stone for the effective waste management, we need to ensure that all the waste bins in cities are properly monitored with no overflows. In this paper we have tried to curb this issue through the applications of IoT device which would efficiently notify about the fill level of the bins. This would help to save fuel, time and human labor in checking the each bin for the fill levels.

II. IDEA PROPOSAL - ECOBINS

Eco bins could be a smart solution for effective waste management where sensors could be fitted in the garbage bin to check the level of the garbage. Each garbage can would be provided with an id code that would uniquely identify the bin. A warning notification would be fired to the garbage collection department as soon as the garbage reaches a certain limit in the bin. Based on the garbage level, the department officials would come to collect the waste from the specific bins which has attained the optimum level.

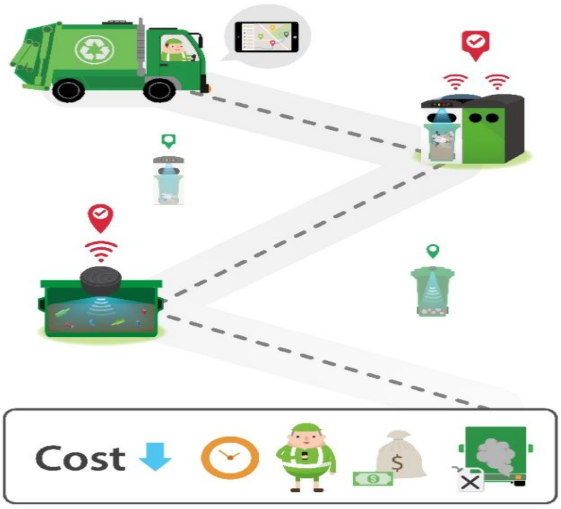


Fig 1: Garbage bin use case diagram

III.COMPONENTS USED

The following components has been used for the implementation of the demo for this project.

- InfraRed Sensors (IR)
- Raspberry Pi3 kit
- WiFi module
- Google APIs like Maps Direction, Maps Direction matrix, Maps Geocoding, Maps JavaScript, Places API Web service.

IV. DATA SET

To analyse the data we have used the dataset of UK cities where the bins are distributed across places with their latitudes and longitude.

V. USE CASE

The garbage bin would be projected on the map. The bins which would be completely filled will be marked with Red, the bins which would be filled to the medium will be marked with Yellow, the bins which would be least filled will be marked with Blue. The path would be mentioned in the map for the garbage bins with immediate and medium attention.

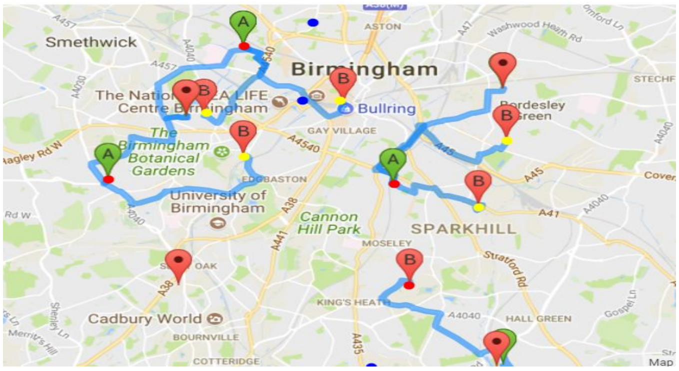


Fig 2: Google map setup

VI. MOBILE APP

A mobile app - "Kachra App" would track the route and location of the garbage bins. There would be provision to scan the QR Code to access the application.

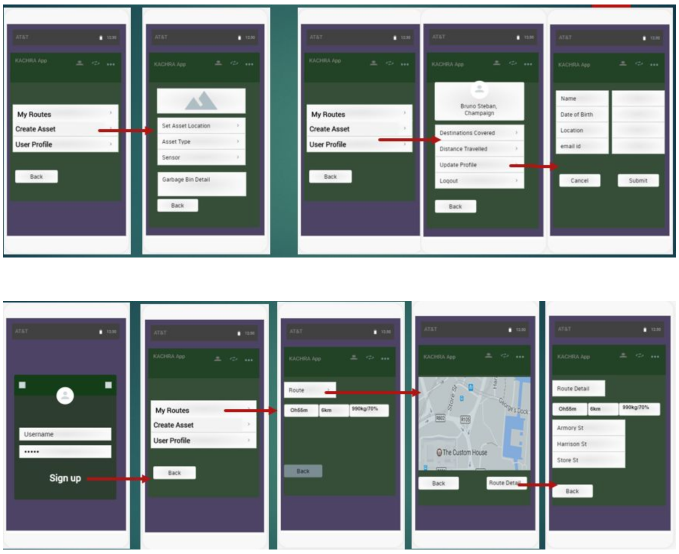


Fig 3: Mobile app layout

VII. CIRCUIT SETUP

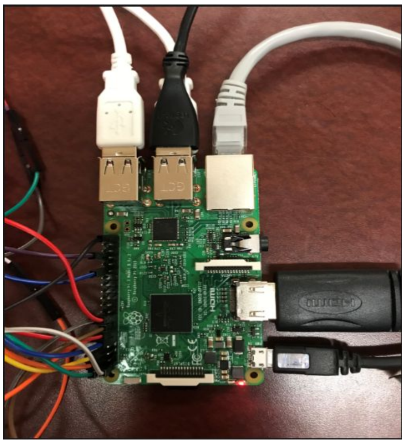


Fig 4 : Raspberry Pi 3 setup

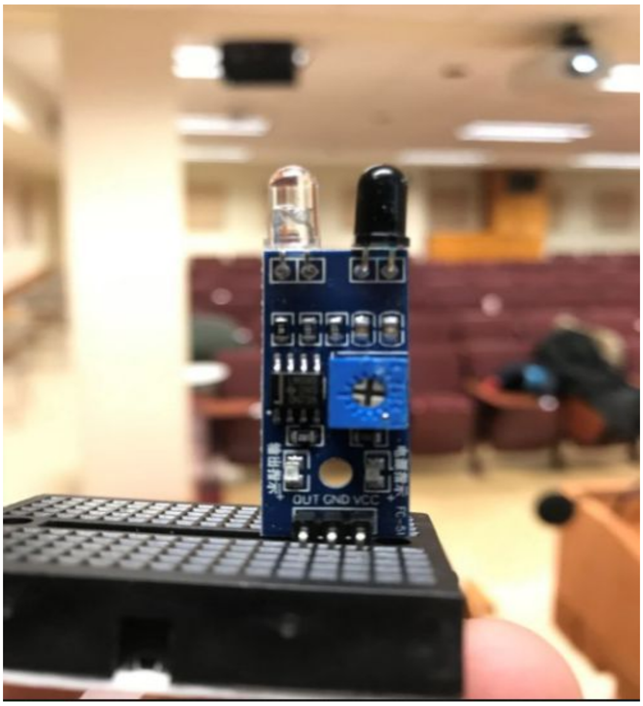


Fig 5 : IR Sensors Module

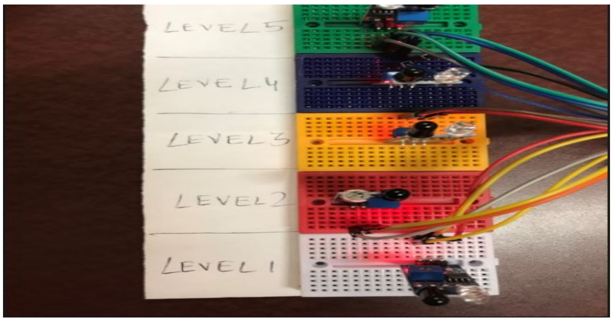


Fig 6 : Circuit setups

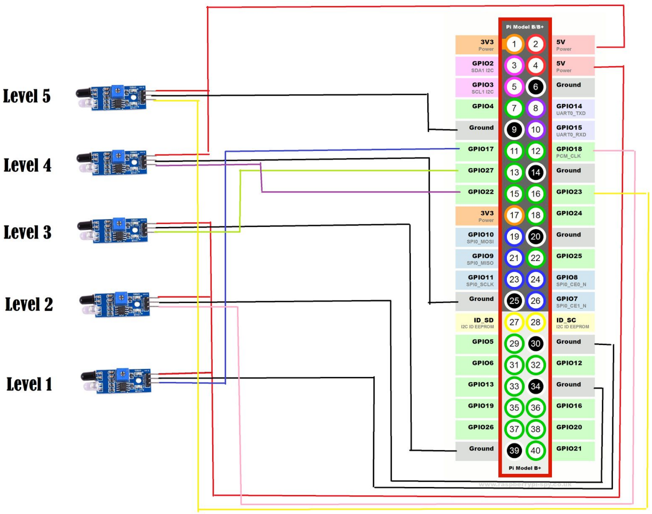
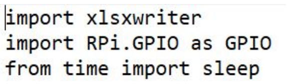


Fig 8: Sensor circuit setup [1]

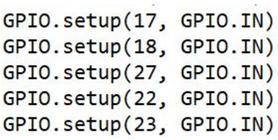
The Diagram above shows the 5 different IR sensor and receiver modules depicting 5 levels of fullness of the trashcan. Each of the sensor modules have a variable resistance to vary the sensing capability and all of the sensors have been given 5V as Vcc and have been grounded on different ports and each module have their output pins connected to the GPIO pins of the Raspberry pi as shown in the above diagram. Each of the sensors are read through different readable GPIO pins and the Raspberry pi then reads the level of fullness of the trash can and then creates an excel file containing information about the Trash can like the Trash can ID, longitude, latitude, level of fullness, city, state and country. This excel file is then mailed to the desired email id as an attachment.

Code Explanation:



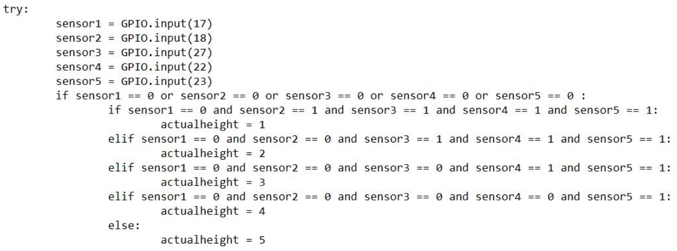
```
import xlsxwriter
import RPi.GPIO as GPIO
from time import sleep
```

Following libraries were imported for the creation of Excel file and for sending out an email with that file as an attachment.



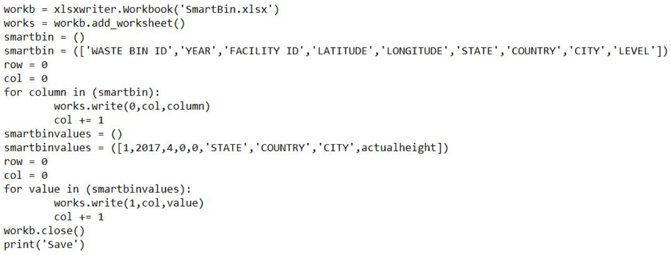
```
GPIO.setup(17, GPIO.IN)
GPIO.setup(18, GPIO.IN)
GPIO.setup(27, GPIO.IN)
GPIO.setup(22, GPIO.IN)
GPIO.setup(23, GPIO.IN)
```

The following code shows the setup of the input output pins of the Raspberry Pi what were used to read the outputs from 5 IR sensor modules defined as Level 1 till Level 5.



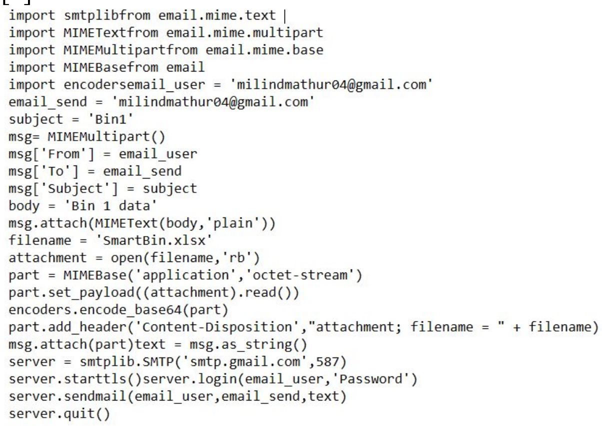
```
try:
        sensor1 = GPIO.input(17)
        sensor2 = GPIO.input(18)
        sensor3 = GPIO.input(27)
        sensor4 = GPIO.input(22)
        sensor5 = GPIO.input(23)
        if sensor1 == 0 or sensor2 == 0 or sensor3 == 0 or sensor4 == 0 or sensor5 == 0 :
                if sensor1 == 0 and sensor2 == 1 and sensor3 == 1 and sensor4 == 1 and sensor5 == 1:
                        actualheight = 1
                elif sensor1 == 0 and sensor2 == 0 and sensor3 == 1 and sensor4 == 1 and sensor5 == 1:
                        actualheight = 2
                elif sensor1 == 0 and sensor2 == 0 and sensor3 == 0 and sensor4 == 1 and sensor5 == 1:
                        actualheight = 3
                elif sensor1 == 0 and sensor2 == 0 and sensor3 == 0 and sensor4 == 0 and sensor5 == 1:
                        actualheight = 4
                else:
                        actualheight = 5
```

Here the code is trying to read if any of the sensors detect any obstacle thereby reading the level of fullness of the trash can the level can be detected with the help of the series by which the sensors detect an obstacle.



```
workb = xlsxwriter.Workbook('SmartBin.xlsx')
works = workb.add_worksheet()
smartbin = ()
smartbin = (['WASTE BIN ID','YEAR','FACILITY ID','LATITUDE','LONGITUDE','STATE','COUNTRY','CITY','LEVEL'])
row = 0
col = 0
for column in (smartbin):
        works.write(0,col,column)
        col += 1
smartbinvalues = ()
smartbinvalues = ([1,2017,4,0,0,'STATE','COUNTRY','CITY',actualheight])
row = 0
col = 0
for value in (smartbinvalues):
        works.write(1,col,value)
        col += 1
workb.close()
print('Save')
```

The following code calls the library for creating the excel file and defining the columns and values to be filled within the worksheet and save it in the local memory of the Raspberry pi. [2]



```
import smtplibfrom email.mime.text |
import MIMETextfrom email.mime.multipart
import MIMEMultipartfrom email.mime.base
import MIMEBasefrom email
import encodersemail_user = 'milindmathur04@gmail.com'
email_send = 'milindmathur04@gmail.com'
subject = 'Bin1'
msg= MIMEMultipart()
msg['From'] = email_user
msg['To'] = email_send
msg['Subject'] = subject
body = 'Bin 1 data'
msg.attach(MIMEText(body,'plain'))
filename = 'SmartBin.xlsx'
attachment = open(filename,'rb')
part = MIMEBase('application','octet-stream')
part.set_payload((attachment).read())
encoders.encode_base64(part)
part.add_header('Content-Disposition',"attachment; filename = " + filename)
msg.attach(part)text = msg.as_string()
server = smtplib.SMTP('smtp.gmail.com',587)
server.starttls()server.login(email_user,'Password')
server.sendmail(email_user,email_send,text)
server.quit()
```

The following code states the import of libraries and functions that are used to send an email from the Raspberry Pi using the local WiFi connection with an attachment of the Excel file that was created earlier [3][4].

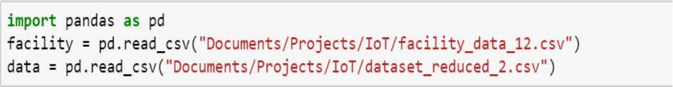
VIII. MAP REPRESENTATION AT FACILITY

Representation of a map at the side of the facility should be able to show the optimized path to be taken by the drivers at a particular facility to access garbage that requires immediate attention, medium attention and the least attention.

These three parameters of immediate attention, medium attention and the least attention can be ranked on the basis of a 'levels' parameter that has values from 1-5. Here the values of levels between 4 and 5 point to garbage cans needing immediate attention, the values corresponding to the range between 2 and 4 point towards garbage cans needing medium attention and the values less than 2 require the least of attention.

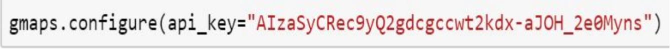
So in a way values 1,2,3,4,5 are in percentages 20% full, 40% full, 60% full, 80% full and 100% full respectively.

This python code uses pandas library loads the file sent from the Raspberry Pi and also the file with the facility which has the data from each facility with its location in latitude and longitude as shown in the figure.



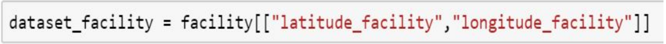
```
import pandas as pd
facility = pd.read_csv("Documents/Projects/IoT/facility_data_12.csv")
data = pd.read_csv("Documents/Projects/IoT/dataset_reduced_2.csv")
```

Here we would need to access the Google APIs for which we need to create a new project in Google Console API and add the required APIs needed for this code. The APIs that were required by us in mapping this code were Maps Direction, Maps Direction matrix, Maps Geocoding, Maps JavaScript, Places API Web service [8].

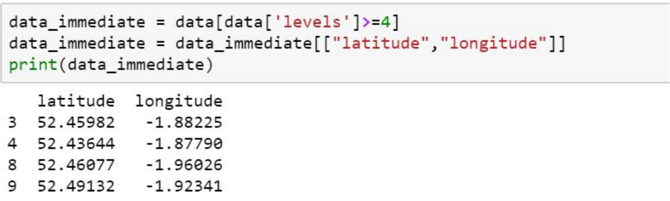


```
gmaps.configure(api_key="AIzaSyCRec9yQ2gdcgccwt2kdx-aJOH_2e0Myns")
```

Also many other libraries were used in this process like gmaps and gmaps.datasets. The ‘longitude’ and ‘latitude’ are then added into separate variables for facilities and for garbage cans location as per the levels variable.

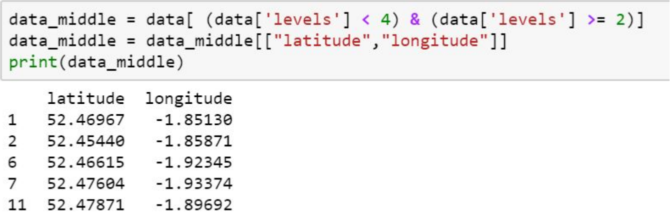


```
dataset_facility = facility[["latitude_facility","longitude_facility"]]
```



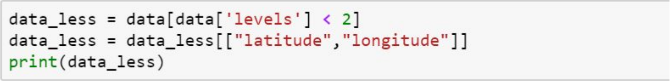
```
data_immediate = data[data['levels']>=4]
data_immediate = data_immediate[["latitude","longitude"]]
print(data_immediate)

   latitude  longitude
3  52.45982   -1.88225
4  52.43644   -1.87790
8  52.46077   -1.96026
9  52.49132   -1.92341
```



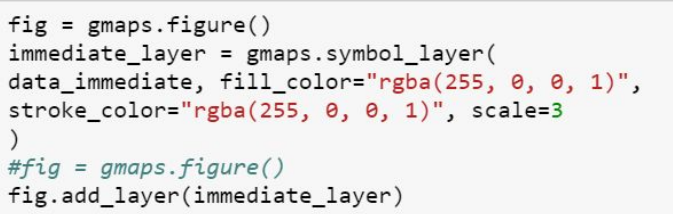
```
data_middle = data[ (data['levels'] < 4) & (data['levels'] >= 2)]
data_middle = data_middle[["latitude","longitude"]]
print(data_middle)

    latitude  longitude
1   52.46967   -1.85130
2   52.45440   -1.85871
6   52.46615   -1.92345
7   52.47604   -1.93374
11  52.47871   -1.89692
```

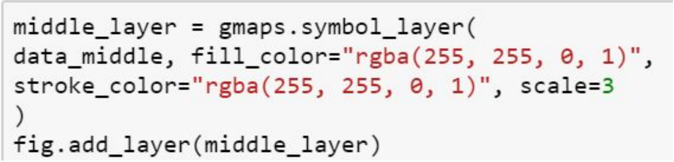


```
data_less = data[data['levels'] < 2]
data_less = data_less[["latitude","longitude"]]
print(data_less)
```

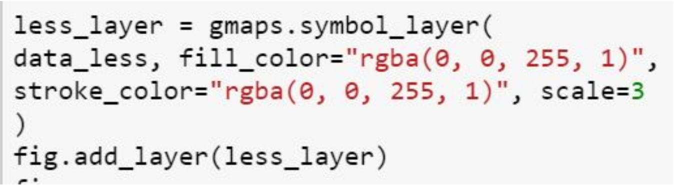
Once these different variables are assigned with these values we need to map these values on the graph using different markers. For facility we use markers, for garbage cans requiring immediate attention red dots are used, for garbage cans requiring middle attention have been marked with yellow dots and the garbage cans requiring less attention have been marked with yellow dots.



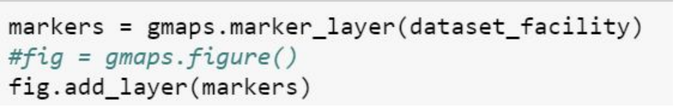
```
fig = gmaps.figure()
immediate_layer = gmaps.symbol_layer(
data_immediate, fill_color="rgba(255, 0, 0, 1)",
stroke_color="rgba(255, 0, 0, 1)", scale=3
)
#fig = gmaps.figure()
fig.add_layer(immediate_layer)
```



```
middle_layer = gmaps.symbol_layer(
data_middle, fill_color="rgba(255, 255, 0, 1)",
stroke_color="rgba(255, 255, 0, 1)", scale=3
)
fig.add_layer(middle_layer)
```



```
less_layer = gmaps.symbol_layer(
data_less, fill_color="rgba(0, 0, 255, 1)",
stroke_color="rgba(0, 0, 255, 1)", scale=3
)
fig.add_layer(less_layer)
```



```
markers = gmaps.marker_layer(dataset_facility)
#fig = gmaps.figure()
fig.add_layer(markers)
```

For this the graph can be shown as below :

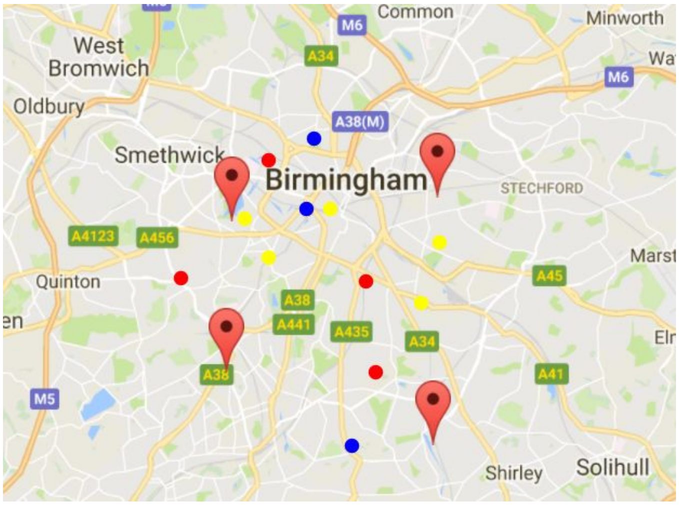
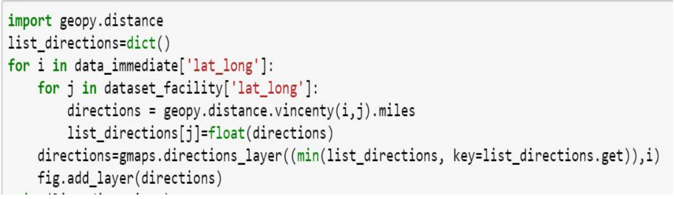


Fig 9 Markings of facility and garbage cans

After this we need to map the shortest path from the facility to the garbage cans requiring immediate attention. This can be achieved by the code below.



```
import geopy.distance
list_directions=dict()
for i in data_immediate['lat_long']:
    for j in dataset_facility['lat_long']:
        directions = geopy.distance.vincenty(i,j).miles
        list_directions[j]=float(directions)
    directions=gmaps.directions_layer((min(list_directions, key=list_directions.get)),i)
    fig.add_layer(directions)
```

This required zipping the coordinated into a tuple format which was first done to convert it into tuple format and then get the code above.[9]

The graph from using the above code provides the shortest path from multiple facilities (origins) to multiple garbage cans requiring immediate attention(destinations). After this another graph wa constructed to plot the shortest distance between point of garbage cans requiring immediate attention to the cans requiring middle attention. The final graph can be shown below :

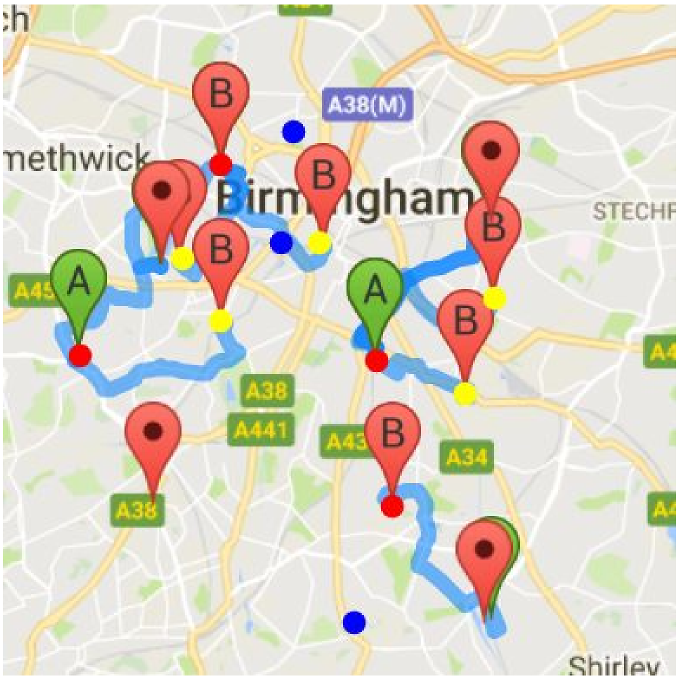


Fig 10 Final Route

This hence gives the idea of the shortest distance between facility and the garbage cans.

As we can observe there is one facility which has no route at all at A38. This shows that the facility does not provide much benefits and hence can be either repositioned to a closer to the much cluttered facility or be made non operational. This hence saves cost, manpower, energy and provides an optimum path.

IX. FUTURE SCOPE

- Smart Bins would be able to detect differences between recyclable and non-recyclable wastes with the help of sensors that can successfully detect the difference between them at right rate.
- Smart citizen points would be awarded to the people who would drop the wastes in appropriate garbage bins such as dry wastes and wet wastes.
- Add solar panels on the lid of the garbage bins so that it would help to power the Wi-Fi module and store power for the use at night. Thus the bins would act as the Wi-Fi hotspots without nonrenewable energy consumption.

X. KACHRA Application

We have created a KACHRA mobile Application which can be accessed by the users. The application have three features-

a) The application will show the optimized routes from the facility which users need to follow in order to collect the garbage based on the fill level and the nearness of the filled bin. Once the user collects the garbage from one location then they can check the next possible nearest filled garbage bin and so on.

b) The application also provides users the facility to check and update their profile which will help them to check how much distance they have covered, and distance travelled.

c) One of the most striking feature of the Application is that it provides users with the facility to add any new asset (Garbage bin) location with details such as location, asset type (biodegradable or non-biodegradable waste) and sensors type present in the Garbage bin. Thus, it can help us to add any new garbage bin location in the database in the future.

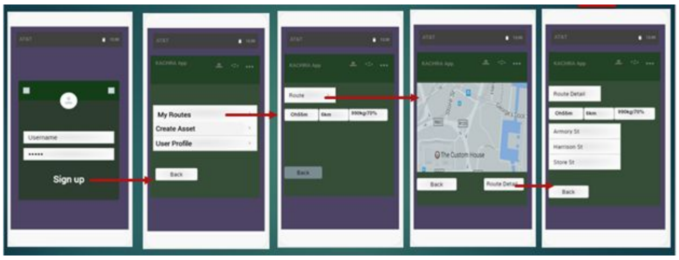
One can access the application by scanning the QR code or by accessing the link provided below:

Link:
https://www.fluidui.com/editor/live/preview/cF9aVW1YUXFBRzlveUt3UGd5Z3BNRnNmbnppMmYyODllYQ==

QR Code:

Figure 11 Kachra App QR code

Below is the application flow diagram:



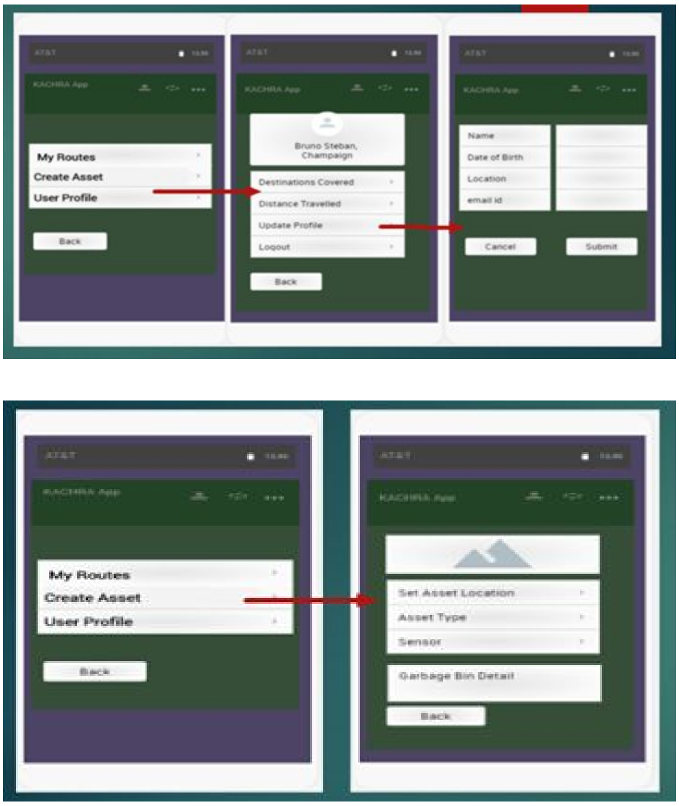


Fig 11 Flow diagram

XI. CONCLUSION

This demo project implementation would have several potential benefits like reduced fuel consumption as the garbage collection trucks would not need to stop at all the garbage bins to check if the bins need to be emptied, reduced time and labor and health benefits as a clean city would promote good health to all the citizens.

As can be seen in the graph in section VIII. Figure 10 we can conclude that not only do we devise a manner in which we can save costs, labour as well as fuel, we also can predict the possible locations of the facility which would help us to further analyze which areas require a facility close to them leading to higher profits and reduced costs.

References

[1] https://www.google.com/search?q=raspberry+pi+3+pin+diagram&safe=off&tbm=isch&source=iu&ictx=1&fir=Za2uXBmze-moqM%253A%252C9KfNNotuCqG5XM%252C_&usg=__3Npl3na799ej-SIx98wuTvUWaNk%3D&sa=X&ved=0ahUKEwjMvMCIi5TYAhWM6YMKHWVSC20Q9QEIKjAA#imgrc=Za2uXBmze-moqM:

[2] http://xlsxwriter.readthedocs.io/

[3] http://naelshiab.com/tutorial-send-email-python/

[4] https://docs.python.org/3/library/email.html

[5] https://www.smartbin.com/

[6] https://www.ijser.org/researchpaper/Smart-Bin-Implementation-for-Smart-Cities.pdf

[7] http://ieeexplore.ieee.org/document/7106974/?reload=true

[8] https://developers.google.com/maps/documentation/directions/

[9] https://github.com/googlemaps/google-maps-services-python