Technical Report

for

Trippie

Version 1.0 approved

Prepared by Ravneet Singh, Gaganpreet Singh Grewal and Bojan Lazic

February 9, 2018

Declaration of Joint Authorship

We, Ravneet Singh, Gaganpreet Grewal and Bojan Lazic, hereby declare that all the content in this technical report titled “Software Requirements Specification for Trippie” is our own joint work.

* All content from external sources are clearly stated and cited.
* The following report is a joint work done by the team members.
* All contributions made by others are acknowledged. It is clearly shown what has been done by the team, and anything used externally is quoted and sourced.
* This report is intended for academic purposes for Humber college.

Approved Proposal

2018-02-05

***Proposal for the development of Trippie***

Prepared by Gaganpreet Singh Grewal, Ravneet Singh, and Bojan Lazic  
*Computer Engineering Technology Students*

https://github.com/princess97/TripPlanner\_V1

**Executive Summary**

As a student in the Computer Engineering Technology program, I will be integrating the knowledge and skills I have learned from our program into this Internet of Things themed capstone project. This proposal requests the approval to build the hardware portion that will connect to a database as well as to a mobile device application. The internet connected hardware will include a custom PCB with the following sensors and actuators Temperature Sensor, Gps Sensor and Stepper Motor. The database will store The database will store the different temperature readings, locations already visited by the user and locations which user wants to visit in future.. The mobile device functionality will include Mobile application will easily analyze the data collected by the sensor and it will also show the temperature of the particular location which user wants to visit. For example, If the temperature will be above or below certain range mobile application will alert the user, not to visit that place. and will be further detailed in the mobile application proposal. I will be collaborating with the following company/department Humber Tech Group, Humber Parts Crib, Prototype Lab. In the winter semester I plan to form a group with the following students, who are also building similar hardware this term and working on the mobile application with me Gaganpreet Grewal(N01139945), Ravneet Singh(N01148757) and Bojan Lezic(N01109108). The hardware will be completed in CENG 317 Hardware Production Techniques independently and the application will be completed in CENG 319 Software Project. These will be integrated together in the subsequent term in CENG 355 Computer Systems Project as a member of a 2 or 3 student group.

**Background**

The problem solved by this project is Most of the people in this world like to travel different places. Sometimes they visit a place which is totally unknown to them, so they need a guide who helps them to explore that place. In return, they have to pay the guide. This increases the buget of the person.. A bit of background about this topic is In order to save money we are working on a project which will help people to explore different places in the world easily and free of cost. Gaganpreet Singh is working on a temperature sensor which will give the reading of temperature of the environment. Bojan Lezic is working on a stepper motor which will be attached with a plastic shelter to protect the delicate hardware of the project. Since this machine is going to be at many different places, the GPS sensor will come in handy to distinguish where each raspberry pi will be located. Ravneet Singh is working on the GPS sensor..

Existing products on the market include [1]. I have searched for prior art via Humber’s IEEE subscription selecting “My Subscribed Content”[2] and have found and read [3] which provides insight into similar efforts.

In the Computer Engineering Technology program we have learned about the following topics from the respective relevant courses:

* Java Docs from CENG 212 Programming Techniques In Java,
* Construction of circuits from CENG 215 Digital And Interfacing Systems,
* Rapid application development and Gantt charts from CENG 216 Intro to Software Engineering,
* Micro computing from CENG 252 Embedded Systems,
* SQL from CENG 254 Database With Java,
* Web access of databases from CENG 256 Internet Scripting; and,
* Wireless protocols such as 802.11 from TECH152 Telecom Networks.

This knowledge and skill set will enable me to build the subsystems and integrate them together as my capstone project.

**Methodology**

This proposal is assigned in the first week of class and is due at the beginning of class in the second week of the fall semester. My coursework will focus on the first two of the 3 phases of this project:  
 Phase 1 Hardware build.  
 Phase 2 System integration.  
 Phase 3 Demonstration to future employers.

*Phase 1 Hardware build*

The hardware build will be completed in the fall term. It will fit within the CENG Project maximum dimensions of 12 13/16" x 6" x 2 7/8" (32.5cm x 15.25cm x 7.25cm) which represents the space below the tray in the parts kit. The highest AC voltage that will be used is 16Vrms from a wall adaptor from which +/- 15V or as high as 45 VDC can be obtained. Maximum power consumption will be 20 Watts.

*Phase 2 System integration*

The system integration will be completed in the fall term.

*Phase 3 Demonstration to future employers*

This project will showcase the knowledge and skills that I have learned to potential employers.

The brief description below provides rough effort and non-labour estimates respectively for each phase. A Gantt chart will be added by week 3 to provide more project schedule details and a more complete budget will be added by week 4. It is important to start tasks as soon as possible to be able to meet deadlines.

Raspberry Pi 3 starter kit, Temprature Sensor(DS18B20), GPS sensor, Stepper Motor, Jumper wires(Male - Female, Female - Female, Male - Male)

**Concluding remarks**

This proposal presents a plan for providing an IoT solution for This is an opportunity for us to integrate many things to create a collaborative project which will help users to plan and visit the different places in the world without the help of any guide and in bonus it will tell the temperature of that place which user wants to visit. This will reduce the budget of the user and save lots of time.. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative IoT capstone project demonstrating my ability to learn how to support projects such as the initiative described by [3]. I request approval of this project.

**References**

[1] N/A

[2] Institute of Electrical and Electronics Engineers. (2015, August 28). IEEE Xplore Digital Library [Online]. Available: https://ieeexplore.ieee.org/search/advsearch.jsp

[3] N/A

Executive summary

The following technical report will be on the software requirements for the *TripPi*. This device is a multi-sensor which is stationary at a set location, which works with our app TripPlanners. The report will cover the requirements for the device as well as the app, and will go over the specifications while providing diagrams for the application. This report is significant as it provides insight on all the needs for the project.

Table of Contents

Executive summary v

Table of Contents vii

Revision History vii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions\* 1

1.3 Intended Audience and Reading Suggestions 1

1.4 System Overview\* 1

2. Overall Description 3

2.1 Product Perspective 3

2.2 Product Functions 3

2.3 Operating Environment 3

2.4 User Documentation 3

2.5 Mobile Application Requirements 3

2.6 Database Requirements 5

2.7 Hardware Requirements 5

3. Conclusion and Recommendations 7

References 8

Appendix A: Glossary 9

Appendix B: Analysis Models 9

Appendix C: To Be Determined List 9

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
|  |  |  |  |
|  |  |  |  |

# Introduction

## Purpose

The purpose of our project is to work as a personal travel guide. Our android app will work anywhere in the world. We have used google places APIs to accomplish this. We will upload co-ordinates and temperature of a particular location and user can decide whether or not to visit that place based on the temperature. Our app will alert the user if temperature is below or above certain degree Celsius.

## Document Conventions\*

This document uses generic technical document conventions such as numbered sections. In terms of font sizes. The section headings are larger as they indicate the start of every new section making them easier to find. All the headings are bolded to indicate each separate section branch, and each new section is shown on an odd page.

## Intended Audience and Reading Suggestions

The intended audience for this SRS document is our mentor Professor Kristian Medri and our team of three would like to improve this project in the future. This document can also be used as a guide by future users of the app and even other developers that might want to contribute. Since, this software is open source.

## System Overview\*

This document will go all of the information regarding the TripPlanner project. The main idea is that our device will be able to send accurate information regarding the temperature and coordinates depending on where the device is.

# Overall Description

## Product Perspective

The purpose of our device is to provide a GPS coordinates, as well as a temperature in an area where the device is set to read from. The device has the ability to protect itself from harsh environments using a timed stepper motor connected to a small trapdoor. This device is meant to provide a reliable, native reading on the characteristics for the location which the user chooses.

## Product Functions

Both the device and application need an internet connection to function, and will connect to an SQL database hosted on hostmonster. While our device is connected to a network, it will be able to do the following:

* Read the temperature at its location, and send it to the server so the TripPlanner app can read it
* Read the coordinates at its location, and send it to the server so the TripPlanner app can read it
* Close and open its own trapdoor based on a 5 minute timer to read the temperature and protect itself from harsh temperatures

## Operating Environment

The device will use a 5V bipolar stepper motor, Raspberry Pi 3, active internet connection to send the location and temperature readings to the server, temperature sensor and a GPS sensor.

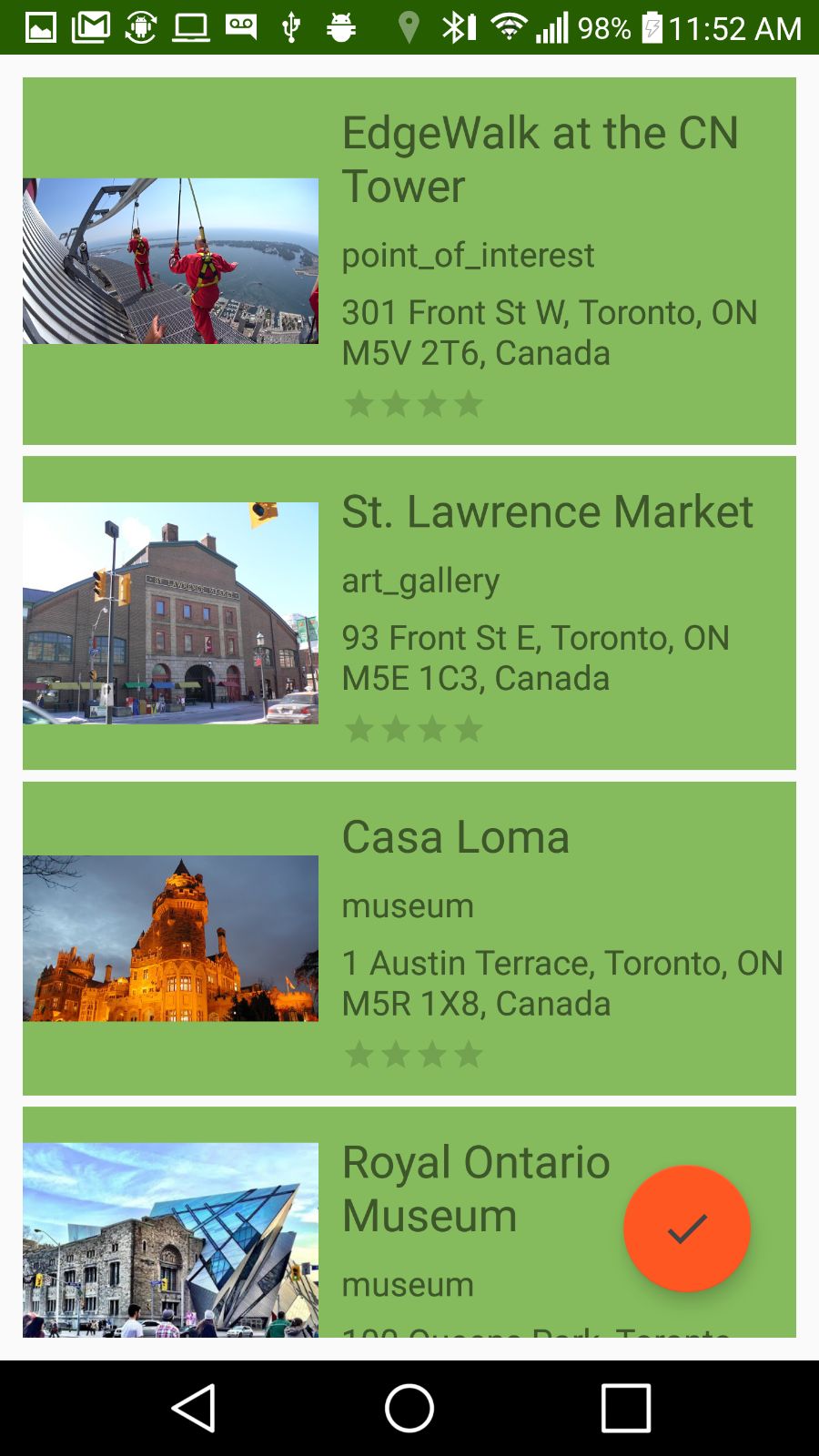
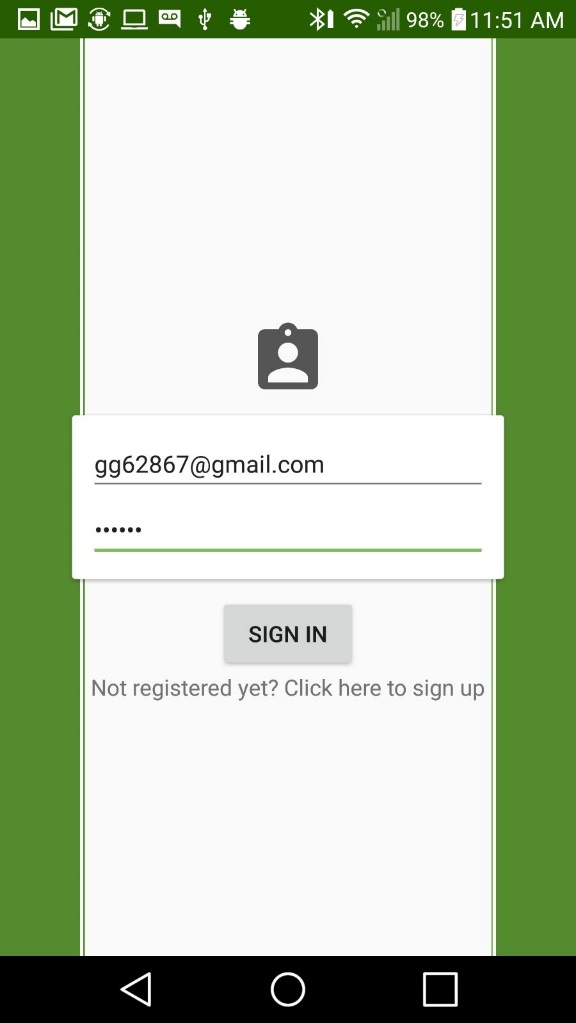
For the mobile app, the user needs to have a minimum SDK version 21 and an active internet connection.

## User Documentation

The user documentation will be available on the TripPlanners official website. This documentation will come with instructions on how to use the app. The hardware will not have any documentation as the user will not be using the device.

## Mobile Application Requirements

If the user is already registered in our app then the first page which to be opened will be sign in page and it will allow our user to sign in into the app. The first time users of our app will have to sign up. When the user sign up then the app will redirect the user to a page which is called as a bucket list in which user add different places according to his plan to visit. After signing in to the app user is able to see their bucket list in which they can add different places which they want to visit in future. This page will be linked with the firebase database and it will show the readings of the temperature at different places which user wants to visit. Ravneet Singh is the lead developer for this.



## Database Requirements

Ravneet Singh is the lead developer for the database functionality of this system. The hosting space is controlled on Hostmonster and the database is powered by SQL. Following is the screenshot of the table:

## 

## Hardware Requirements

Bojan Lazic is working on the stepper motor and the contraption that has to be connected with it. This contraption will be used to protect the delicate temperature sensor.

Gaganpreet Singh Grewal is working on the temperature sensor and the design of the 3D contraption for the sensor. The tools used for the design is OpenSCAD and also windows 3D paint.

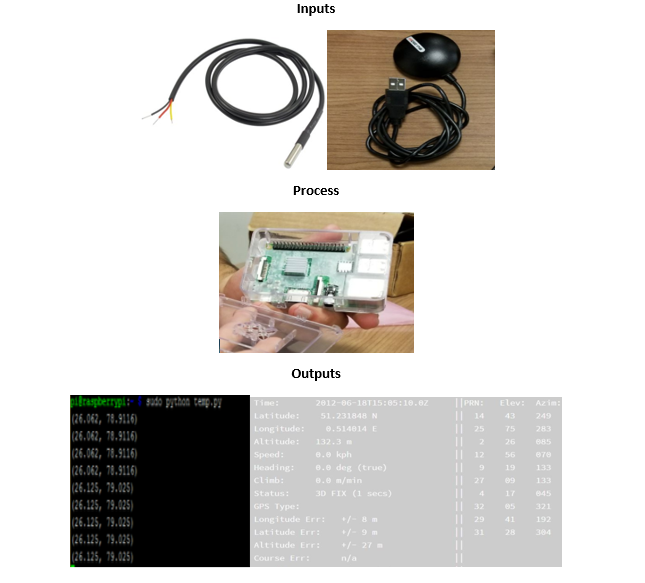
Ravneet Singh is working on getting the GPS coordinates and writing the python code to get the temperature sensor simultaneously. He is also working on the code to push that data to the table inside the database.

# Build Instructions

## Introduction

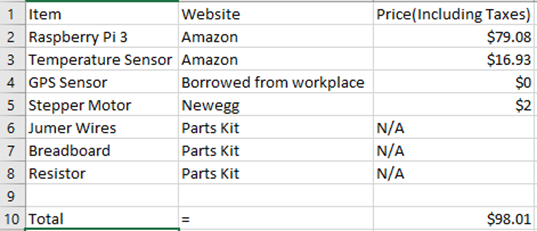
In our project we decided to use three sensors the Temperature Sensor, GPS sensor and a stepper motor. Our goal is make a hardware which will tell the location of the user and temperature at that particular location. The temperature sensor will detect the Temperature at various sites. GPS sensor will get the coordinates of the user’s particular location. We are connecting stepper motor to a contraption as a protection to our sensors. The database will store the different temperature readings, locations already visited by the user and locations which user wants to visit in future. The mobile device functionality will include Mobile application which will easily analyze the data collected by the sensor and it will also show the temperature of the particular location which user wants to visit.

## System Diagram



## Budget

The main components which are required for our project are a Raspberry pi, a temperature sensor, a GPS sensor and a stepper motor. These all components are available on Amazon, so you can easily get them. Some other components you will need to make this project are breadboard, jumper wires and a resistor. You can get these components from your parts kit.



## Time Commitment

The project began with the modification to previous projects which we made last semester. We all spend some time check our individual projects whether they are working or not. Then we started integrating all the three sensors together which took about 2.5 hours. Then we started implementing the code by which both the temperature and the GPS sensor will work together and we spend 2 hours to complete it. We also designed a 3d printed contraption for our project which is connected to a stepper motor as a protection to our sensors which took about 1 day to get printed and 1 hour to connect it to the stepper motor.

## Mechanical Assembly

The assembly of our project is very simple. Firstly, put the SD card into the pi and set it up. Connect the 3d printed contraption to stepper motor. Attach a GPS sensor directly the raspberry pi through USB port and for temperature sensor, connect it to the raspberry pi with the help of breadboard, 4.7k ohm resistor and jumper wires as shown in the pictures and follow the steps given below to connect temperature sensor to the raspberry pi.

#1 - Connect GPIO GND [Pin 6] on the Pi to the negative rail on the breadboard and connect GPIO 3.3V [Pin 1] on the Pi to the Positive rail on the breadboard.

#2 - Plug the DS18B20+ into your breadboard, ensuring that all three pins are in different rows. Familiarise yourself with the pin layout, as it’s quite easy to hook it up backwards!

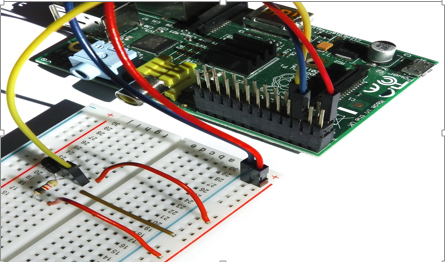
#3 - Connect DS18B20+ GND [Pin 1] to the negative rail of the breadboard.

#4 - Connect DS18B20+ VDD [Pin 3] to the positive rail of the breadboard.

#5 - Place your 4.7kΩ resistor between DS18B20+ DQ [Pin 2] and a free row on your breadboard.

#6 - Connect that free end of the 4.7kΩ resistor to the positive rail of the breadboard.

#7 - Finally, connect DS18B20+ DQ [Pin 2] to GPIO 4 [Pin 7] with a jumper wire.



Follow the steps given below to connect GPS sensor to the raspberry pi:

1. Place the Raspberry Pi on a non-conductive surface with GPIO pins facing up

2. Connect the raspberry-pi-3-model-b with the internet using a CAT-5 cable.

3. Connect the raspberry-pi-3-model-b with a mouse and a keyboard using USB.

4. Start terminal and input the following command to install the GPS drivers: "sudo apt-get install gpsd gpsd-clients python-gps"

5. Next we need to start the daemon. This is done using the following command: "sudo gpsd /dev/ttyUSB0 -F /var/run/gpsd.sock"

6. Ignore any messages from the console or in the log files, you may see it complaining about IPv6 but you can ignore that.

7. Connect the GPS receiver with the pi by USB.

8. Use the following command and the program will start running: cgps –s

Follow the steps given below to connect Stepper Motor to the raspberry pi:

1. Use the printed circuit board intended to drive the motor, rather than wiring the entire thing on the breadboard.

2. Attach all four wires (excluding the red common power wire, this will not be used) to the output pins of the L293D.

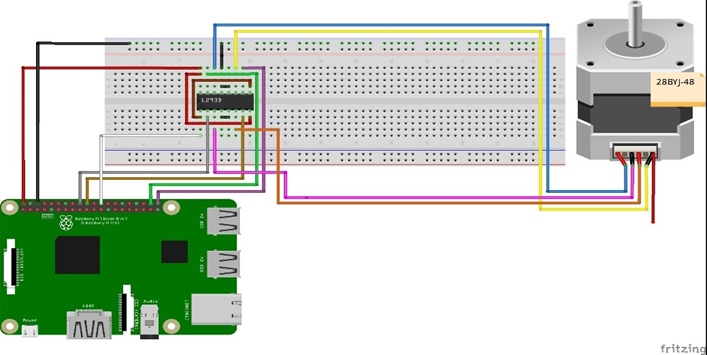
3. The input pins must be wired to the GPIO pins as per choice, but preferably pins 35 (GPIO19), 37 (GPIO26), 38 (GPIO20) and 40 (GPIO21).

4. Attach the enable pin from the chip (pin 1) to pin 33 of the raspberry pi (GPIO13) and attach the that enable pin from the chip to pin 9 of the chip, which is the enable pin for the other two input and output pins of the driver chip. Attach the ground pin of the driver chip to pin 39 of the raspberry pi as this is a ground pin.

5. The vss and vs pins of the chip must be wired to each other and wired to the 5V pin on the raspberry pi (pin 2).

6. Be sure that the modules are plugged into the sense hat. The BME280 will be sensing the temperature.

7. This will complete the wiring of the stepper motor. A diagram of the breadboarding can be seen below.



## Unit Testing

There are lot of build instructions in our project so unit testing should not be required because I did unit testing to get the project work. There is no Unit testing for the stepper motor because it is only used with the contraption that protect our sensors. For the GPS sensor the information of the user is received by connecting with multiple satellites orbiting the Earth. For this connection, the sensor must be under open sky. Unit testing is helpful if you face any problems in the mechanical assembly. Below is the unit testing which we did on the project. Testing our sensor.

#!/usr/bin/python

import os

#from gps import \*

import glob

import pygame, sys

import time

from time import \*

import serial

#os.system('modprobe w1-gpio')

#os.system('modprobe w1-therm')

base\_dir = '/sys/bus/w1/devices/'

device\_folder = glob.glob(base\_dir + '28\*')[0]

device\_file = device\_folder + '/w1\_slave'

def read\_temp\_raw():

f = open(device\_file, 'r')

lines = f.readlines()

f.close()

return lines

def read\_temp():

lines = read\_temp\_raw()

while lines[0].strip()[-3:] != 'YES':

time.sleep(0.2)

lines = read\_temp\_raw()

equals\_pos = lines[1].find('t=')

if equals\_pos != -1:

temp\_string = lines[1][equals\_pos+2:]

temp\_c = float(temp\_string) / 1000.0

temp\_f = temp\_c \* 9.0 / 5.0 + 32.0

return temp\_c, temp\_f

base\_dir = '/sys/bus/w1/devices'

#device\_folder = glob.glob(base\_dir + '28\*')[0]

device\_file = device\_folder + '/w1\_slave'

def read\_temp\_raw():

f = open(device\_file, 'r')

lines = f.readlines()

f.close

return lines

def read\_temp():

lines = read\_temp\_raw()

while lines[0].strip()[-3:] != 'YES':

time.sleep(0.2)

lines = read\_temp\_raw()

equals\_pos = lines[1].find('t=')

if equals\_pos != -1:

temp\_string = lines[1][equals\_pos+2:]

temp\_c = float(temp\_string) / 1000.0

return temp\_c

#initialise serial port on /ttyUSB0

ser = serial.Serial('/dev/ttyUSB0',4800,timeout = None)

fix = 1

x = 0

while x == 0:

gps = ser.readline()

# print all NMEA strings

# print gps

# lat = "cfhbdf"

# print lat

# check gps fix status

if gps[1:6] == "GPGSA":

fix = int(gps[9:10])

# print time, lat and long from #GPGGA string

if gps[1 : 6] == "GPGGA":

# get time

#time = gps[7:9] + ":" + gps[9:11] + ":" + gps[11:13]

# if 2 or 3D fix get lat and long

if fix > 1:

lat = " " + gps[18:20] + "." + gps[20:22] + "." + gps[23:27] + gps[28:29]

#lat = " " + gpsd.fix.latitude

lon = " " + gps[30:33] + "." + gps[33:35] + "." + gps[36:40] + gps[41:42]

print "Latitude: " + lat + " | Longitude: " + lon + " | Temperature: " + str(read\_temp())

# if no fix

else:

lat = " No Valid Data "

lon = " No Valid Data"

print "Latitude: " + lat + " | Longitude: " + lon + " | Temperature: " + str(read\_temp())

## Production Testing

The production testing is very easy. First, run the program and it will show the temperature readings on the screen. If the raspberry-pi-3-model-b is purchased in bulk, i.e. more than 100 raspberrypi’s then cost of each pi drops down to CA$35 per pi. No such discount is offered by the GPS receiver. Since the code running on the pi is free and same for every scenario the total cost of production would reach CA$9,375 which is 10.7% cheaper than the original cost per unit.

## References

<http://www.circuitbasics.com/raspberry-pi-ds18b20-temperature-sensor-tutorial/>

<https://www.modmypi.com/blog/ds18b20-one-wire-digital-temperature-sensor-and-the-raspberry-pi>

<http://usglobalsat.com/p-688-bu-353-s4.aspx#images/product/large/688.jpg>

<https://www.raspberrypi.org/forums/viewtopic.php?f=28&t=128028>

<https://github.com/Bojan-sensei/CENG317/blob/master/firmware/BME280.py>

# Conclusion and Recommendations

This Technical report presents a plan for providing an IoT solution for This is an opportunity for us to integrate many things to create a collaborative project which will help users to plan and visit the different places in the world without the help of any guide and in bonus it will tell the temperature of that place which user wants to visit. This will reduce the budget of the user and save lots of time. This is an opportunity to integrate the knowledge and skills developed in our program to create a collaborative IoT capstone project demonstrating my ability to learn how to support projects.

References

Wiegers, K. E. (1999). IEEE Software Requirements Specification Template. Retrieved February 12, 2018, from https://cs.gmu.edu/~dfleck/classes/cs421/spring08/srs\_template.doc

Appendix A: Glossary

<Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.>

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>