**INSTALLATION GUIDE**

**SOFTWARE USED:**

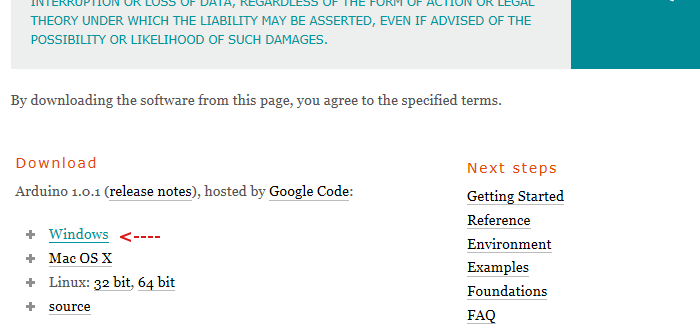
1. **Arduino IDE**
2. **Python**
3. **Sheetsu**
4. **Google sheet**
5. **ARDUINO-:**

We are using Arduino/nodemcu as microcontroller for running our main code, That fetched the data from the gps module and send it to the google spreadsheet.For uploading the code in the microcontroller we have used an **integrated development environment** for this.

**STEPS TO INSTALL ARDUINO IDE:**

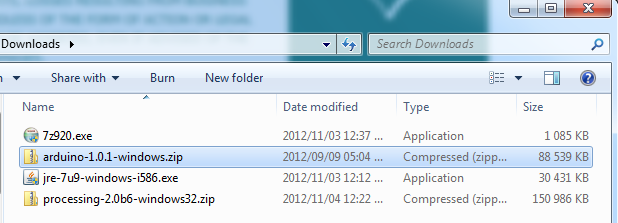
**1.DOWNLOAD**

Go to the [Arduino website](http://arduino.cc/) and click on the download link to go to the download page and then click on windows to download the zip file of windows and then download the arduino software.



**2.INSTALL**

After downloading, locate the downloaded file on the computer and extract the folder from the downloaded zipped file. Copy the folder to a suitable place such as your desktop.



**Install the Arduino Windows Drivers**

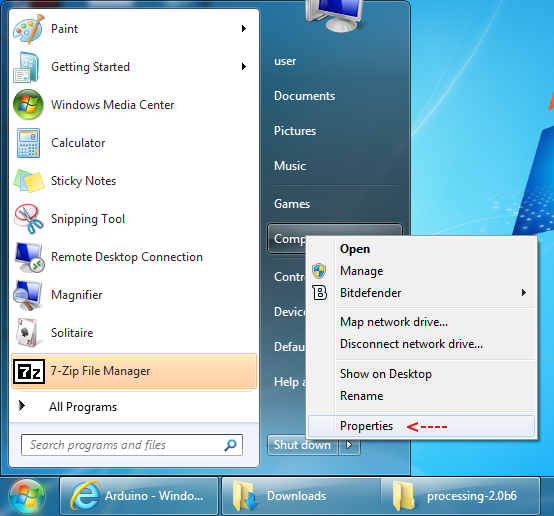
**a.Plug the Arduino Into the PC**

Plug the Arduino board into the PC. Windows will try to install drivers, but will fail.

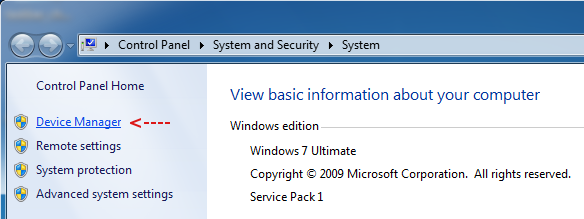
**b.Start the Windows Device Manager**

Click the Windows Start menu button.

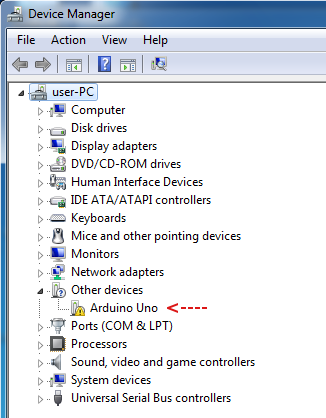
Right-click **Computer** on the menu and then click **Properties** from the pop-up menu:



Click the **Device Manager** link to start the device manager

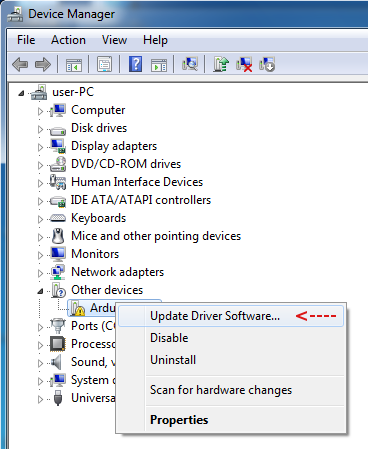


The Device Manager will open and display the Arduino Uno:

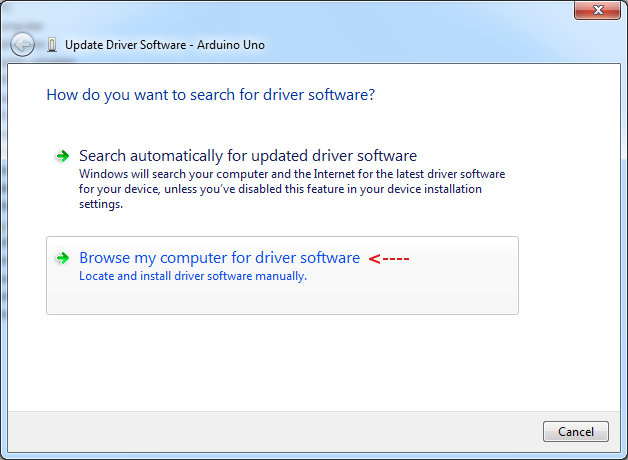


**3.Install the Device Driver**

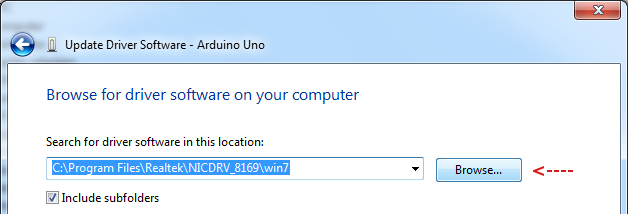
In the Device Manager Window, right-click the Arduino board and then click **Update Driver Software...** on the pop-up menu:



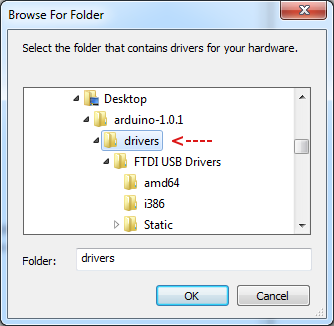
The **Update Driver Software** dialog box will pop up. Click **Browse my computer for driver software**:



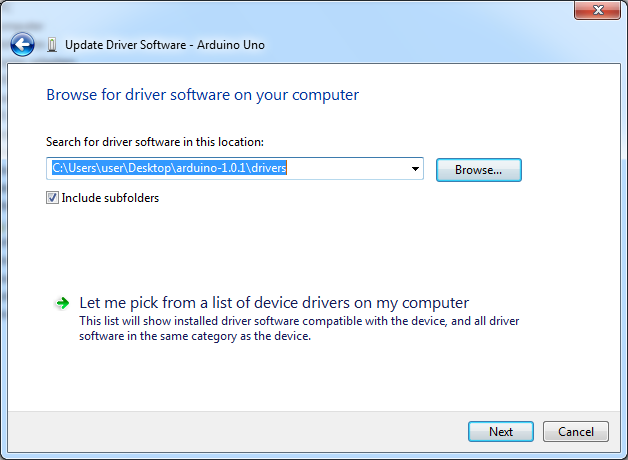
Next, click the **Browse...** button:



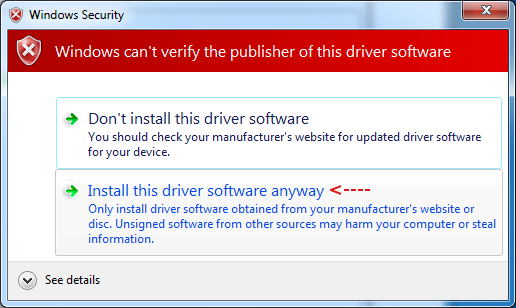
Navigate to the **drivers** folder in the Arduino folder that you downloaded:



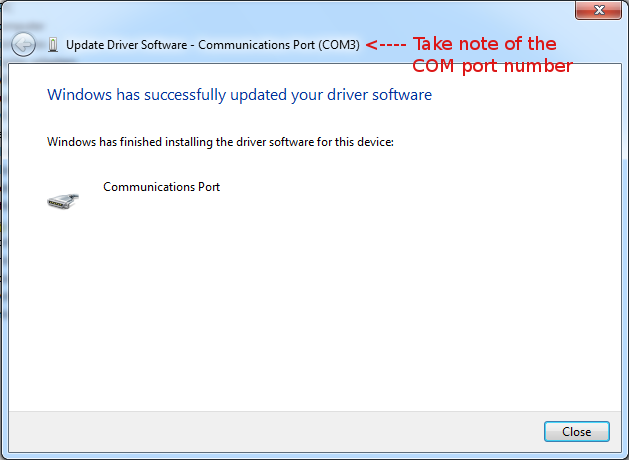
After selecting the driver folder, click the **Next** button:



In the dialog box that pops up, click **Install this driver software anyway**:



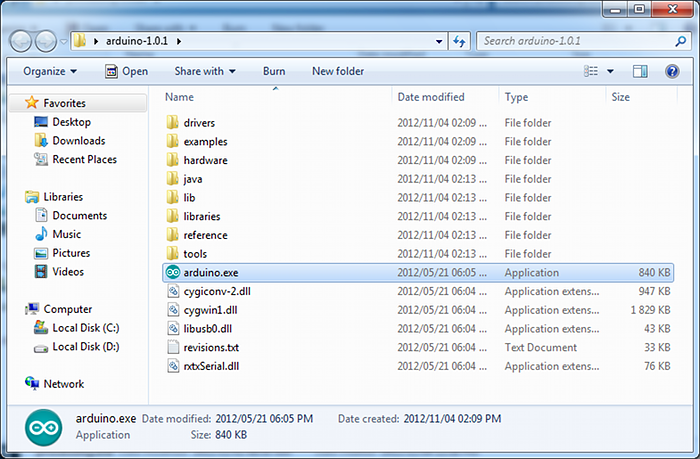
After some time, the driver installation will finish and you will see the following dialog box. Take note of the port that the Arduino was configured as. In this case it was **COM3**.



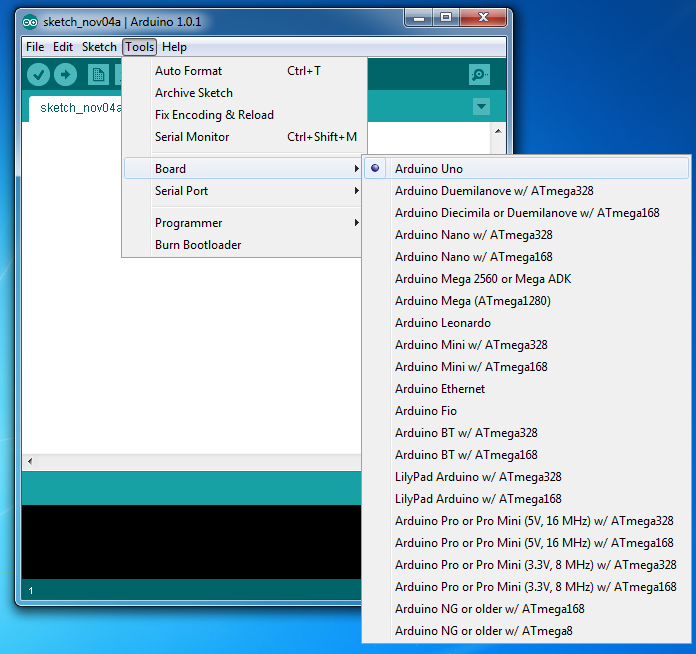
### **4. Setting up the Arduino Software**

The setup will only need to be done once, unless you change the board type or port that the Arduino is connected to.

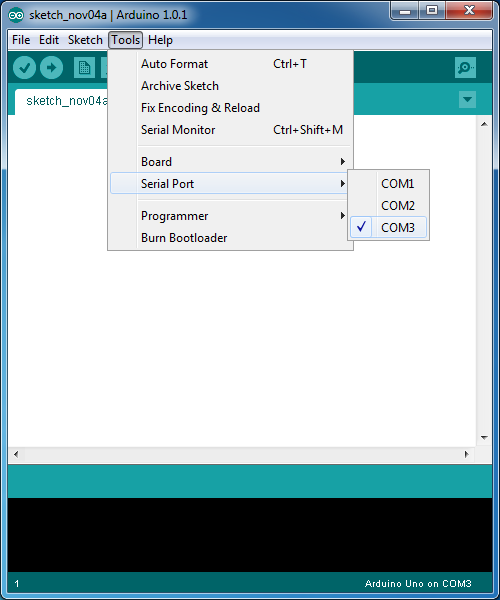
Navigate to the folder that you downloaded and start the Arduino software IDE by double-clicking the Arduino application:



Check that the correct Arduino board is selected. Change if necessary:

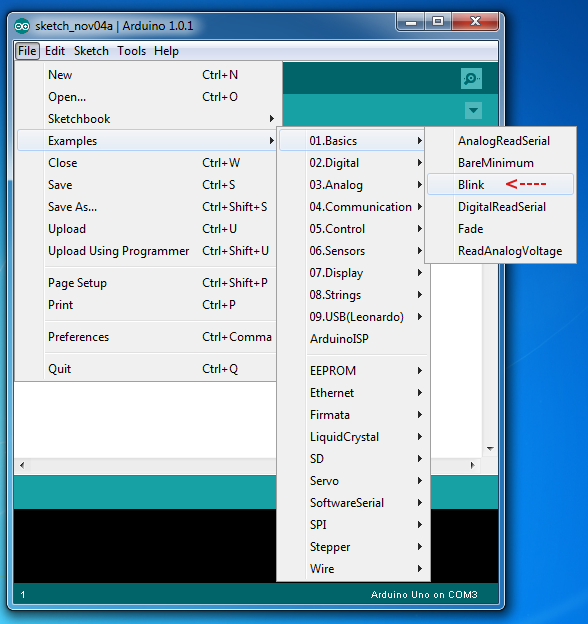


Now check that the correct serial port is selected and change if necessary. This is the serial port that you took note of after installing the Arduino driver.

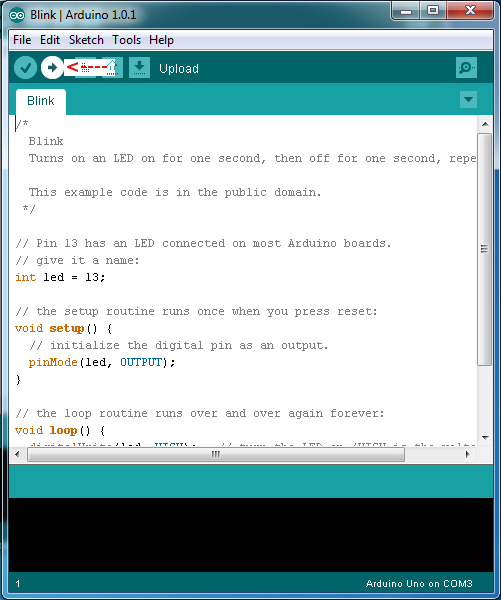


### **5. Testing the Installation**

Open the **Blink** sketch in the Arduino IDE:



Click the Arduino Upload button to load the sketch to the Arduino. This sketch will flash the on-board LED on the Arduino. If the sketch runs then you know that you have successfully installed the Arduino software and driver.



**2. PYTHON:-**

To get started working with Python 3, you’ll need to have access to the Python interpreter. There are several common ways to accomplish this:

* Python can be obtained from the **Python Software Foundation** website at [python.org](https://www.python.org). Typically, that involves downloading the appropriate **installer** for your operating system and running it on your machine.
* Some operating systems, notably Linux, provide a **package manager** that can be run to install Python.
* On macOS, the best way to install Python 3 involves installing a package manager called **Homebrew**. You’ll see how to do this in the relevant section in the tutorial.
* On mobile operating systems like Android and iOS, you can install apps that provide a Python programming environment. This can be a great way to practice your coding skills on the go.

Alternatively, there are several websites that allow you to access a Python interpreter online without installing anything on your computer at all.

**Note:** There is a chance that Python may have been shipped with your operating system and is already installed. Even if that is the case, it may be that the installed version is outdated, in which case you will want to obtain the latest version anyhow.

In this Python installation guide, you’ll see step by step how to set up a working Python 3 distribution on Windows, macOS, Linux, iOS, and Android. So let’s get started!

## 

## Windows

It is highly unlikely that your Windows system shipped with Python already installed. Windows systems typically do not. Fortunately, installing does not involve much more than downloading the Python installer from the [python.org website](https://www.python.org) and running it. Let’s take a look at how to install Python 3 on Windows:

### Step 1: Download the Python 3 Installer

1. Open a browser window and navigate to the [Download page for Windows](https://www.python.org/downloads/windows/) at [python.org](https://www.python.org/).
2. Underneath the heading at the top that says **Python Releases for Windows**, click on the link for the **Latest Python 3 Release - Python 3.x.x**. (As of this writing, the latest is Python 3.6.5.)
3. Scroll to the bottom and select either **Windows x86-64 executable installer** for 64-bit or **Windows x86 executable installer** for 32-bit. (See below.)

#### Sidebar: 32-bit or 64-bit Python?

For Windows, you can choose either the 32-bit or 64-bit installer. Here’s what the difference between the two comes down to:

* If your system has a 32-bit processor, then you should choose the 32-bit installer.
* On a 64-bit system, either installer will actually work for most purposes. The 32-bit version will generally use less memory, but the 64-bit version performs better for applications with intensive computation.
* If you’re unsure which version to pick, go with the 64-bit version.

**Note:** Remember that if you get this choice “wrong” and would like to switch to another version of Python, you can just uninstall Python and then re-install it by downloading another installer from [python.org](https://python.org).

### 

### 

### 

### 

### 

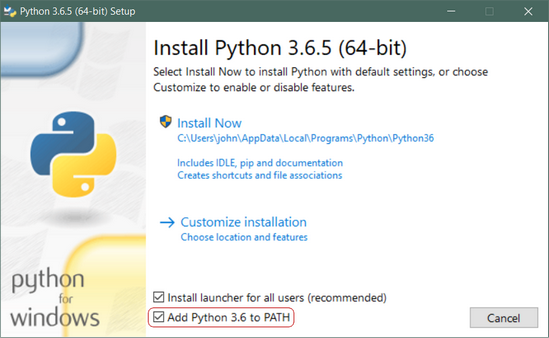
### 

### 

### 

### Step 2: Run the Installer

Once you have chosen and downloaded an installer, simply run it by double-clicking on the downloaded file. A dialog should appear that looks something like this:



**Important:** You want to be sure to check the box that says **Add Python 3.x to PATH** as shown to ensure that the interpreter will be placed in your execution path.

Then just click **Install Now**. That should be all there is to it. A few minutes later you should have a working Python 3 installation on your system.

## Linux

There is a very good chance your Linux distribution has Python installed already, but it probably won’t be the latest version, and it may be Python 2 instead of Python 3.

To find out what version(s) you have, open a terminal window and try the following commands:

* python --version
* python2 --version
* python3 --version

One or more of these commands should respond with a version, as below:

$ python3 --version

Python 3.6.5

If the version shown is Python 2.x.x or a version of Python 3 that is not the latest (3.6.5 as of this writing), then you will want to install the latest version. The procedure for doing this will depend on the Linux distribution you are running.

Note that it is frequently easier to use a tool called pyenv to manage multiple Python versions on Linux. To learn more about it, see our article [here](https://realpython.com/python-virtual-environments-a-primer/#using-different-versions-of-python).

### Ubuntu

Depending on the version of the Ubuntu distribution you run, the Python install instructions vary. You can determine your local Ubuntu version by running the following command:

$ lsb\_release -a

No LSB modules are available.

Distributor ID: Ubuntu

Description: Ubuntu 16.04.4 LTS

Release: 16.04

Codename: xenial

Depending on the version number you see under Release in the console output, follow the instructions below:

* **Ubuntu 17.10, Ubuntu 18.04** (and above) come with Python 3.6 by default. You should be able to invoke it with the command python3.
* **Ubuntu 16.10 and 17.04** do not come with Python 3.6 by default, but it is in the Universe repository. You should be able to install it with the following commands:

$ sudo apt-get update

$ sudo apt-get install python3.6

* You can then invoke it with the command python3.6.
* If you are using **Ubuntu 14.04 or 16.04**, Python 3.6 is not in the Universe repository, and you need to get it from a Personal Package Archive (PPA). For example, to install Python from the [“deadsnakes” PPA](https://launchpad.net/~deadsnakes/+archive/ubuntu/ppa), do the following:

$ sudo add-apt-repository ppa:deadsnakes/ppa

$ sudo apt-get update

$ sudo apt-get install python3.6

* As above, invoke with the command python3.6.

**3. SHEETSU**

[Sheetsu](http://sheetsu.com) allows you to create a REST API from a Google Spreadsheet. Let’s say it in a different way: Sheetsu turns your Google Spreadsheet into a REST API. Ok, let’s try one more time: Sheetsu connects whatever data you need to a simple Google Spreadsheet in a few steps:

1 . Open a Google Spreadsheet and copy the link

2 . Paste it into [Sheetsu](https://sheetsu.com/?utm_source=blog&utm_content=rest_api_for_dummies)

3 . Get your REST API link!

#### Let me give you an example:

If you use this [Sheetsu](https://sheetsu.com/?utm_source=blog&utm_content=rest_api_for_dummies) REST API link to build a website, every time you change something in your spreadsheet, it will automatically change the content on your website too, in real-time! We have “caching” which is a fancy way of saying we will make sure it will all happen fast.

### Sheetsu helps to connect Google Sheets to anything - Web, Mobile, IoT or any service's API.

**4. GOOGLE SHEET:**

Google sheet is being used to store the data.

The data is added to the sheet using Google sheets **script editor.**

The data is sent to the google sheet using a GET request over an URL which has a pattern of https://script.google.com/….. When you enter this URL in a web browser, the Google’s server responds back asking the browser to redirect to another URL with domain script.googleusercontent.com with a new GET request. For a web browser, the URL redirection is a very common thing and it works without any problem. However, for ESP8266, this is not straight forward. The ESP8266 needs to correctly decode the header information received from the first server to extract the redirect URL and make a second GET request to the new server. To make this redirection simpler, [Sujay Phadke](https://github.com/electronicsguy), aka *electronicsguy*, has shared on GitHub a beautiful piece of code in the form of Arduino Library named [HTTPSRedirect](https://github.com/electronicsguy/ESP8266). There are other [examples](http://www.instructables.com/id/Post-to-Google-Docs-with-Arduino/) of Arduino posting data to Google spreadsheets using a third party API service, like [pushingbox](http://pushingbox.com), for handling Google’s https requirement and URL redirection. The use of [HTTPSRedirect](https://github.com/electronicsguy/ESP8266) library makes the task much simpler by avoiding the need of any third party service. So, the first thing you need to do is to copy the HTTPSRedirect library files from GitHub and install into your Arduino libraries folder. I have also posted the zipped library files at the following link for convenience.

To create a Google App Scripts, go to **Tools > Script Editor** from the Google Sheets, add the javascript code and publish the webapp.

**NOTE:** The webapp controls how and where (in which cell) the data is entered into google sheet.

**FUNCTIONING OF MODULE FITTED ON THE BUSES.**

**1. MODULE EQUIPMENTS**

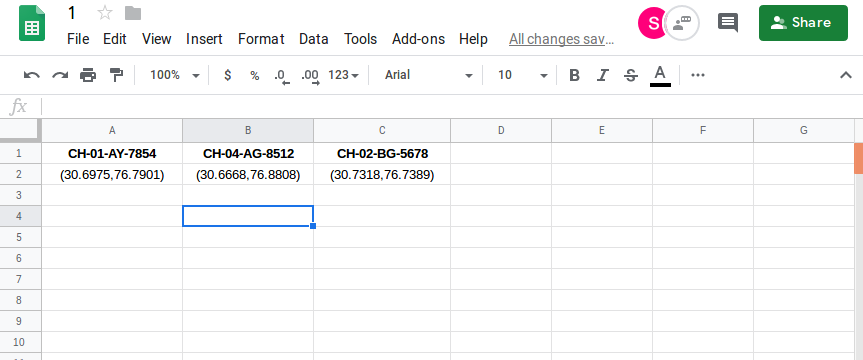
The module which is fitted on the buses consists of GSM module, Arduino, GPS module and DC jack(providing the power to the Arduino). GSM module is having a networking sim which is used for provideing internet services for interfacing the module with the internet.

**2. FUNCTIONING**

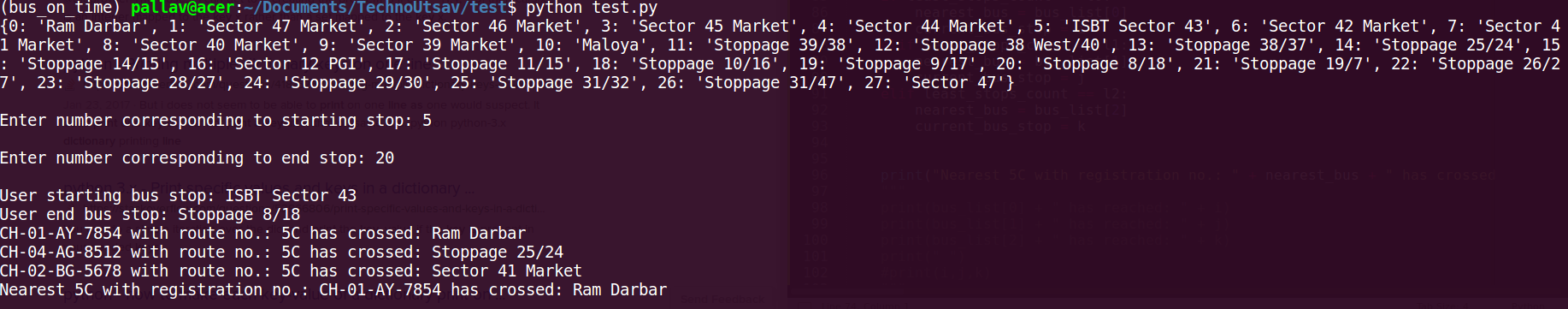
The bus is travelling and continously the latitude and longitude of the present location will be fetched by the arduino with the use of GPS module. Now that data will be in the arduino, now the ardunio is connected with the GSM module with the help of TX and RX pin on the microcontroller and the GSMS module. Now the data from the arduino is send to the GSM module and then as the module is having internet services so the data of latitude and longitude of the live location will be send to the google spreadsheet which is acting as a server for us.After that data will be fetched by the python script and then all the other things will be done on the python script which is running at backhand .

**BACKEND FUNCTIONING**

**1:**

The data sent from the device on the bus to the server will look like this. Here we have the data of three buses of same route number ie., they have the same path but different registration number. At this instant we have the data of the busses of lattitude and longitude.

**2:**

At the backend we have the python script running. As we can see above :

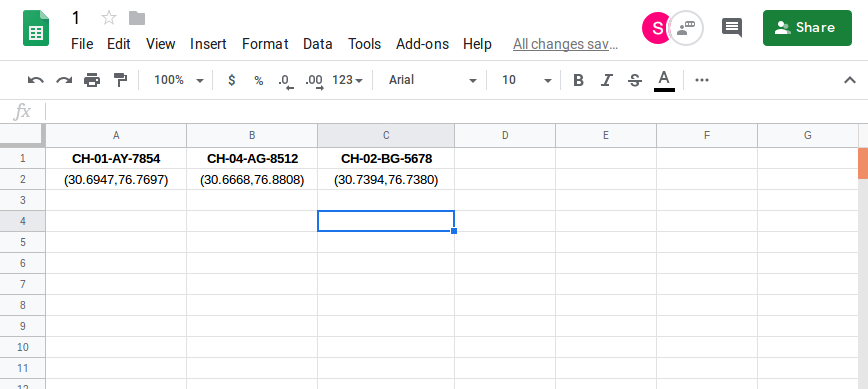
CH-01-AY-7854 – crossed Ram Darbar

CH-04-AG-8512 – crossed Sector 25/24

CH-02-BG-5678 – crossed Sector 47 market

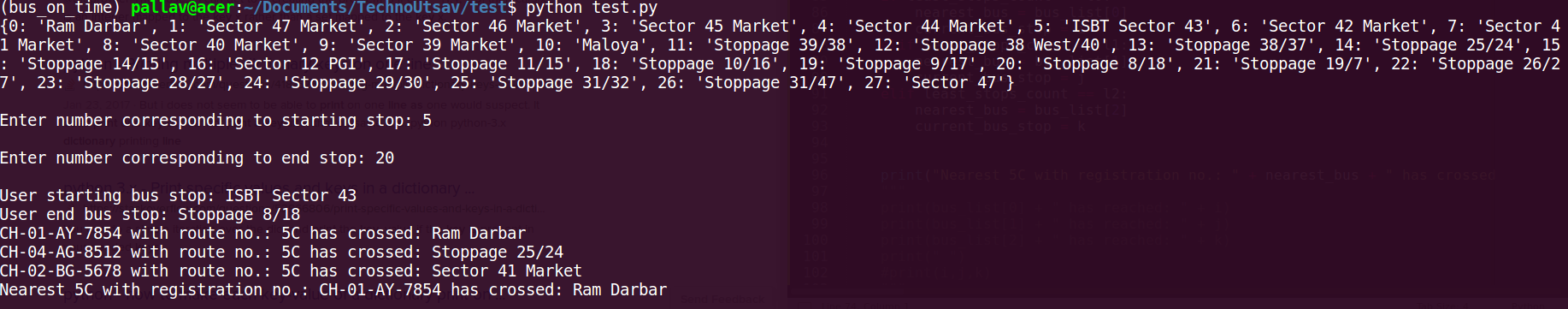
This is the current locations of the bus. This location keeps on updating when the data matches with the data on the server, which ensures bus reaches next bus stop.

**3:**

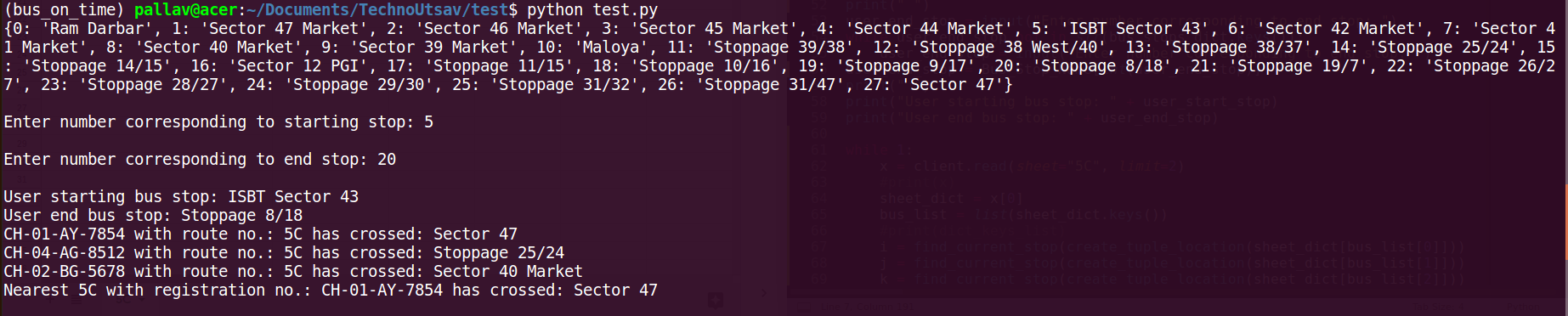
The data on the server gets updated as bus keep sending the GPS location. 

**4:**

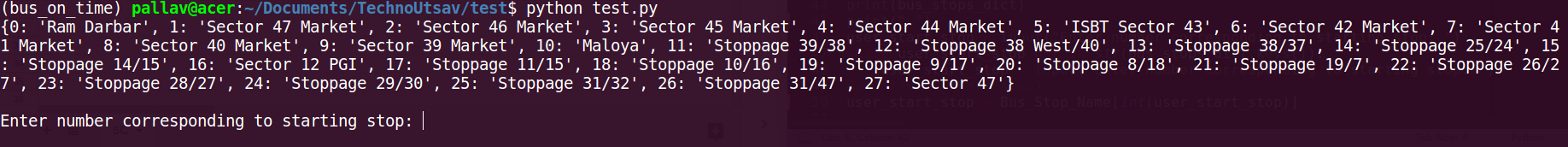
Previous:

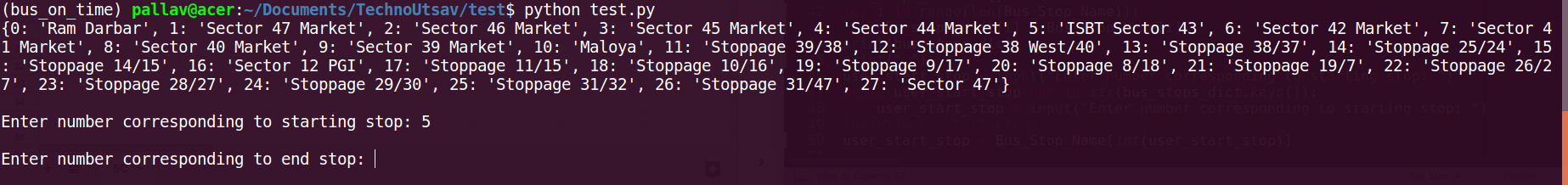


After a few second:

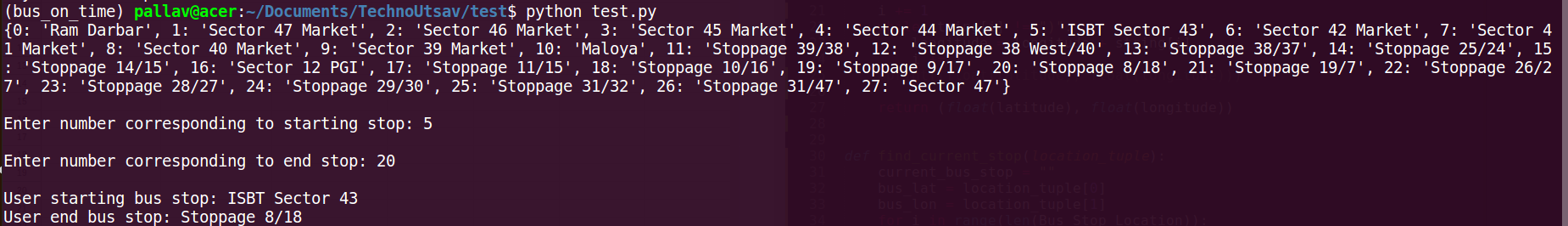
As we can see current locations of the two busses have changed. This has happened because the location on the server has been changing and at this point of time the location on the server has been matched to that in the list of bus stops.Hence it indicates that the bus has reached the next locations execpt the second bus whose location is still the previous one this indicates it is still not reached the next stoppage.

**5:**

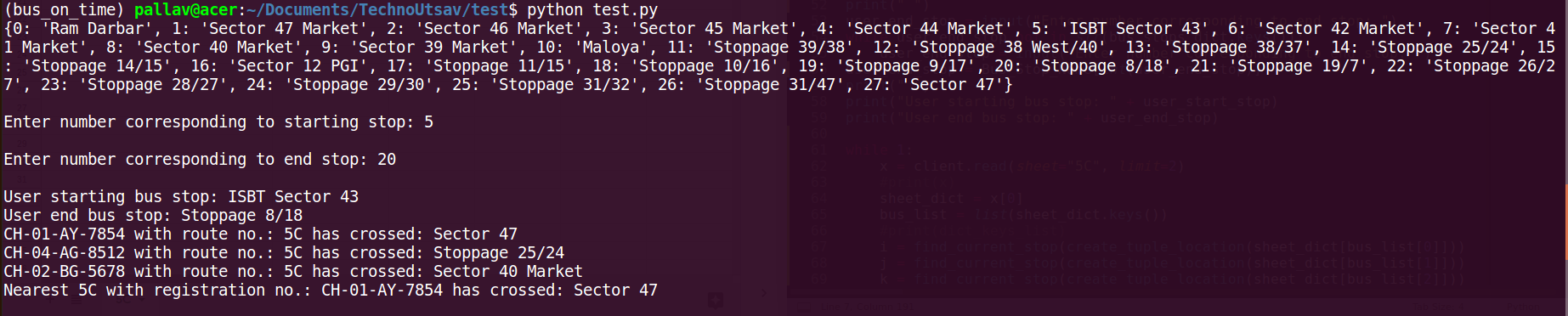
Now when user at any instant of time wants to find out about the bus from his route to destination he will enter the starting point.

After that he will enter the end stop. 

**6:**

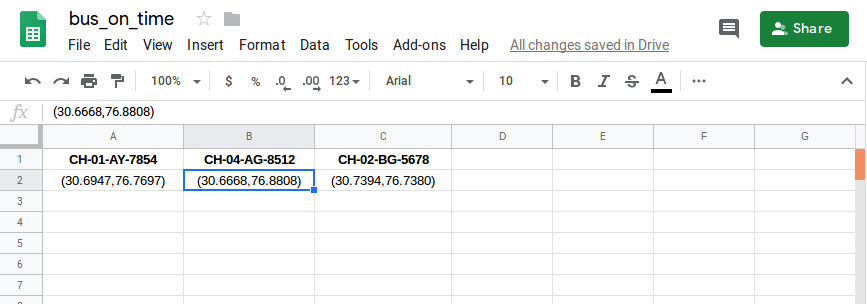
After entering both of the location the user will get to know the bus from his location coming the earliest and will also get that bus cuurent location and the bus nest stoppage. 

**7:**

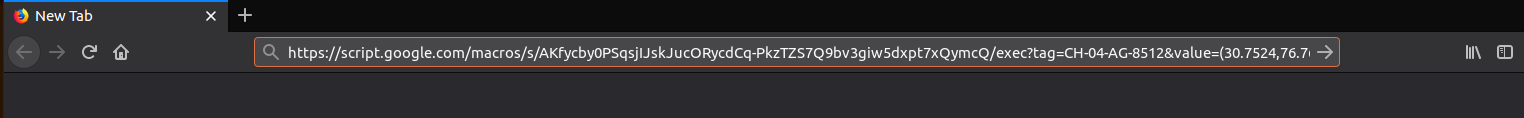
The user has entered the starting point as sector 43 and end point as sector 8/18. So the bus nearest to this location which has not crossed the user’s location and is yet to come to his location will be assigned to him, which in this case is bus with registration number CH-01-AY-7854.

**ADDING THE DATA TO GOOGLE SHEET:**

**1:**

Data already present on the google sheet is shown in the above figure.

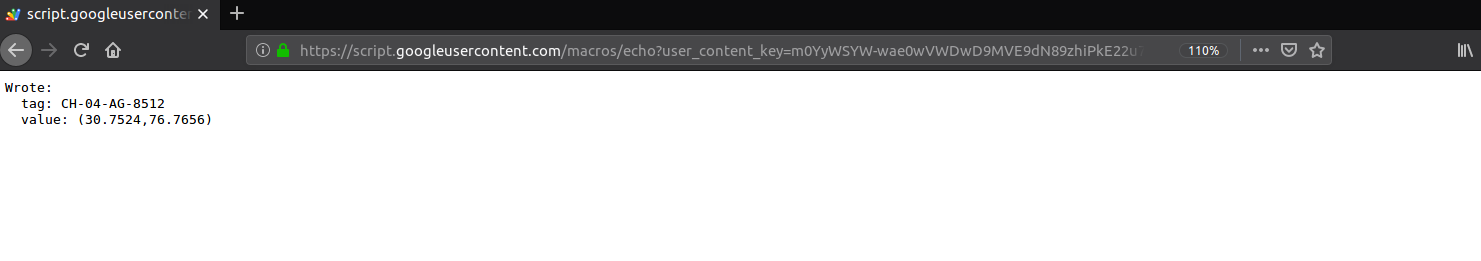
**2:**

Data containing the “Bus Registration No” and “Location” is posted to the published webapp (webapp is prepared by using javascript on the Script Editor)

Required URL is generated by the controller and sent using the wifi module(here ESP8266/NodeMCU) present on the module attached to the bus.

Then, as programmed on the webapp, the data is posted on google sheet at the specified cell.

**3:**



**4:**

Updated data on the Google sheet (highlighted column).