INTERNSHIP REPORT AT INTERNET OF THINGS AND MACHINE LEARNING BOLT IoT

Attended by

Sri Sai Teja. Palisetti (18121004)

in partial fulfilment for the award of the degree of

Bachelor of Technology

In

ELECTRONICS AND COMMUNICATION

from 25.06.2021 to 26.08.2021

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING

SCHOOL OF ELECTRICAL SCIENCES

HINDUSTAN INSTITUTE OF TECHNOLOGY AND SCIENCE

PADUR 603 103

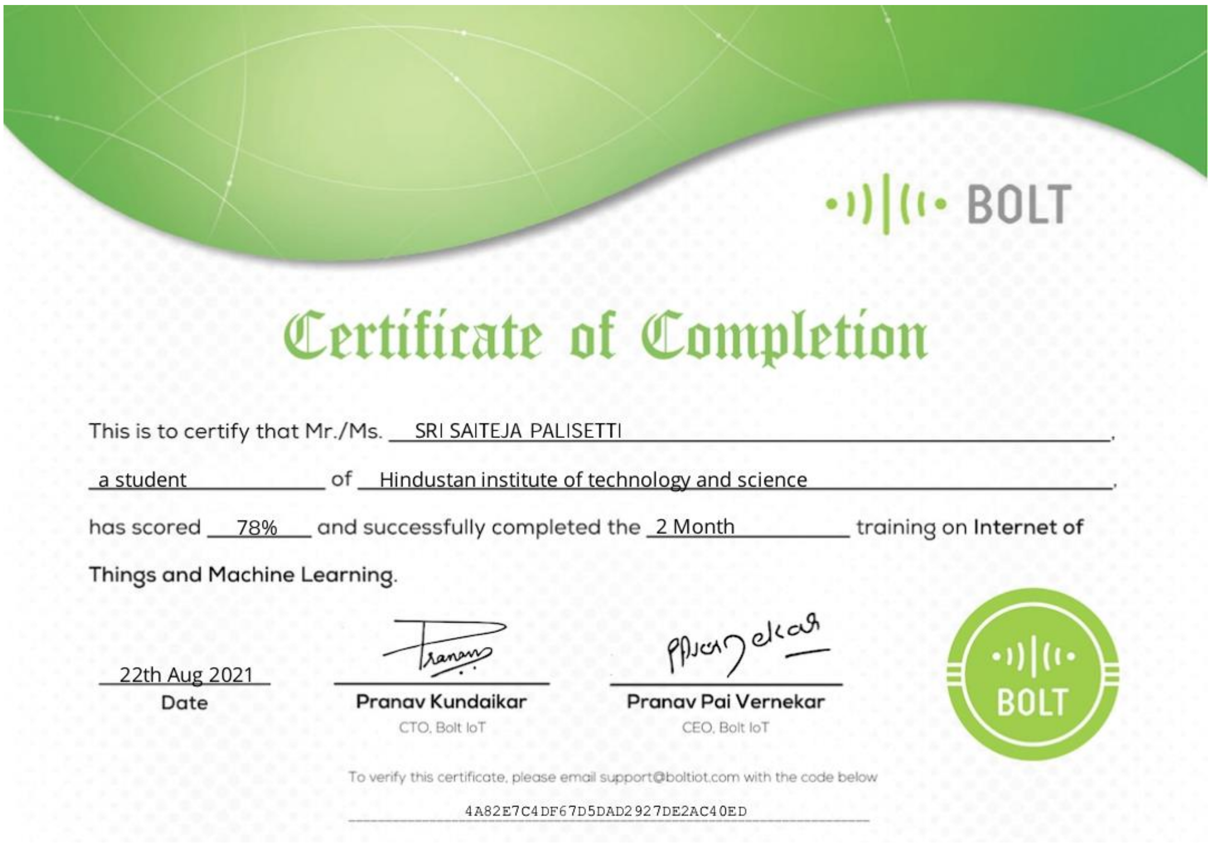
HINDUSTAN
INSTITUTE OF TECHNOLOGY & SCIENCE
(DEEMED TO BE UNIVERSITY)
CHENNAI

BONAFIDE CERTIFICATE

This is to certify that the “Internship report” submitted by Sri Sai Teja Palisetti is the work done by him and submitted during 2020–2021 academic year, in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in ELECTRONICS AND COMMUNICATION ENGINEERING , at BOLT IoT INTERNSHIP.

HEAD OF THE DEPARTMENT	**INTERNSHIP COORDINATOR**
Dr.A.L.Vallikannu	Ms.K.Thenkumari

CERTIFICATION



•))|((• BOLT

Certificate of Completion

This is to certify that Mr./Ms. SRI SAITEJA PALISETTI,

a student of Hindustan institute of technology and science,

has scored 78% and successfully completed the 2 Month training on **Internet of Things and Machine Learning**.

22th Aug 2021
Date

[illegible]
Pranav Kundaikar
CTO, Bolt IoT

[illegible]
Pranav Pai Vernekar
CEO, Bolt IoT

BOLT

To verify this certificate, please email support@boltiot.com with the code below

4A82E7C4DF67D5DAD2927DE2AC40ED

TABLE OF CONTENT

1.Introduction

Bolt is an IoT platform to easily and quickly build products and services. Bolt comes with a Wi-Fi/GSM chip and a cloud platform which helps you connect your devices and sensors to the Internet. With Bolt Cloud you can control and monitor them over the internet, create personalized dashboards to visualize the data, monitor the device health, run machine learning algorithms and lot more. Build scalable IoT systems in just a days' time.

With Bolt IoT's flexibility of implementations, be it using a custom website or app, or a virtual private server, or even the editor in bolt cloud, I can link any device to the internet and control it or analyse the data.

As I had taken the course for IoT and ML offered by Bolt, which was a really great one. The team was very interactive and helpful. Since it was the first time, I used an IoT platform, I had almost no issues in using the module.

The course helped me understand the wide range of applications of the IoT, and now I have many ideas I can apply this too. Using Bolt IoT, finally I have done a mini project what I have learn in this internship and submitted in the Bolt IoT flat form.

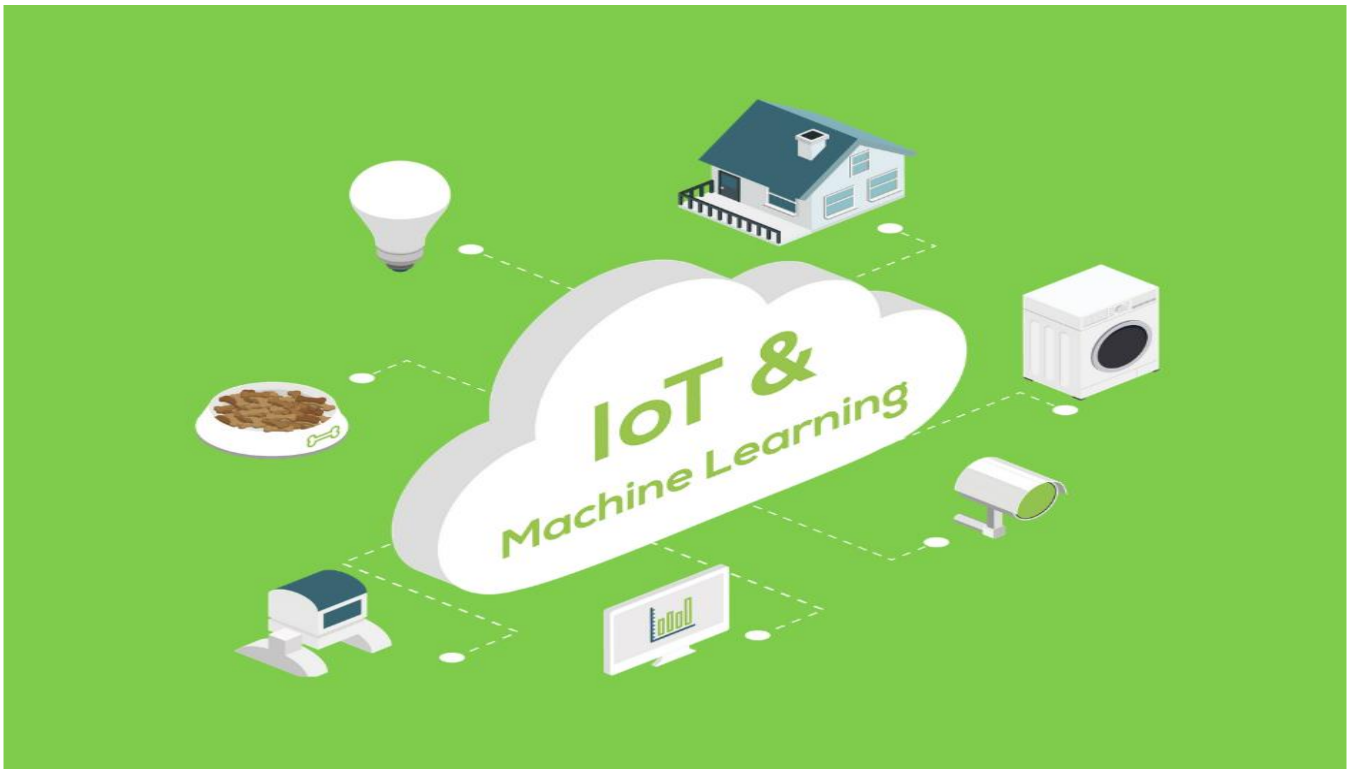
Services: Internship matching, online training

Founder: Mr.Pranav Pai Vernekar

Industry: Education, Employment

Headquarters: Bengaluru, Karnataka, India

Bolt IoT is India's no.1 internship and training platform with 30000+ paid internships in Engineering, MBA, media, law, arts, and other streams. The Top Bolt IoT Program.



I started an internship program in Bolt IoT at 25/06/2021-26/08/2021 (2 months training).

- **Sensor Applications**
- **Data Analytics**
- **Connected to the Bolt wi-fi module**
- **Different programming languages**
- **Mini Projects**

2. Agenda

- **WEEK 1**:-Introduction to the IoT Platform & Getting started with the IoT Cloud
- **WEEK 2**:- Building your first IoT sensor project
- **WEEK 3**:- HTML Programming Language and Web Development
- **WEEK 4**:- Java Script Programming Language
- **WEEK 5**:- Data Visualization and Analytics & Anomaly Detection
- **WEEK 6**:- APIs Key's & Home Automation
- **WEEK 7**:- VPS (Virtual Private Server) & Introduction to Linux OS
- **WEEK 8**:- Python Programming Language & Machine Learning

3.1 Introduction to the IoT Platform

Bolt IoT Platform offers you vast array of features and multiple ways to work with it. When you get started, you can start by learning any of the features.

Here I have created a few paths that you can take to get started with Bolt IoT:

Path 1: Interfacing a sensor to collect data

IoT is often associated with collecting sensor data. This is more popular with industrial IoT. If that is your aim then follow the steps given below:

Step 1: Getting started: Installing the mobile app
Step 2: Video tutorial: Setup and build first project

Path 2: Blinking an LED

Many makers prefer to start learning any hardware project by glowing an LED. If that is the path you wish to take then follow the steps below:

Step 1: Getting started: Installing the mobile app
Step 2: Text tutorial: Setup and build first project
Step 3: Controlling output devices (Glowing an LED)
Step 4: Controlling output devices (with APIs)

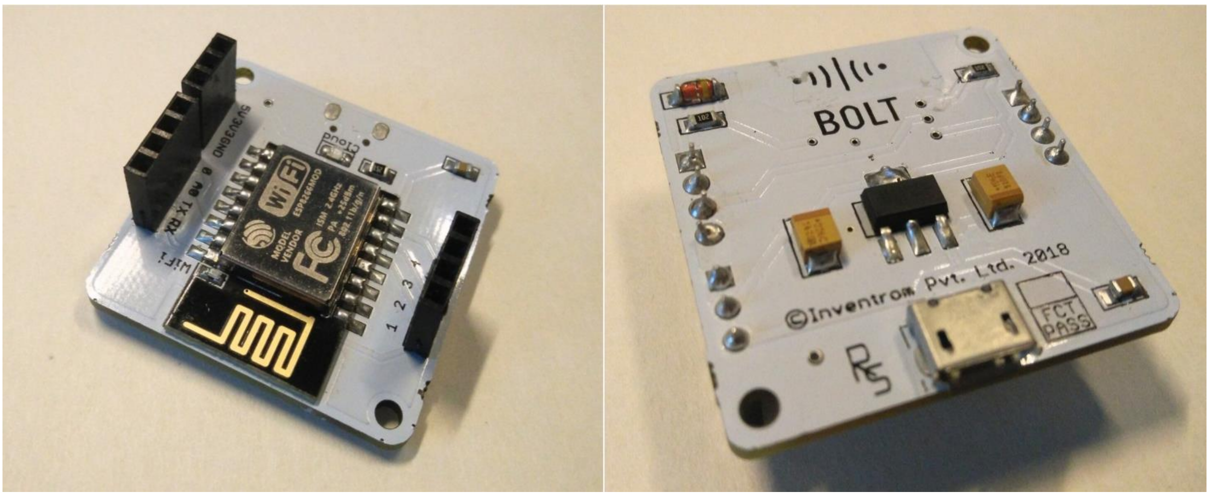
The Bolt IoT Platform consists of three major components:

1. Bolt Wi-Fi module
2. Bolt Cloud
3. Bolt Mobile App

In this article, I will share with you the specifications of the Wi-Fi Modules and how to use them.

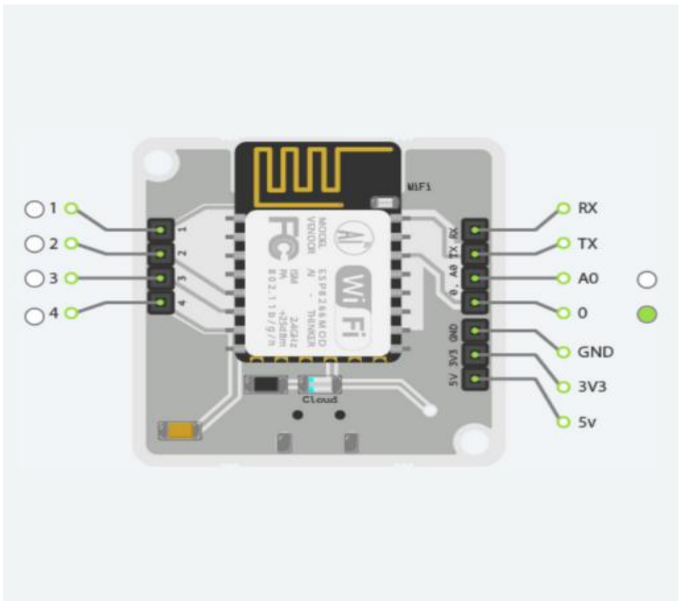
3.2 Getting started with the IoT Cloud

Here are the photos of the Wi-Fi module:


Bolt WiFi Module. Front and backside.

Bolt WiFi Module Pinouts

WiFi Module Pinouts



When you design any circuit, choose the correct pin based on your requirement. Here are few examples:

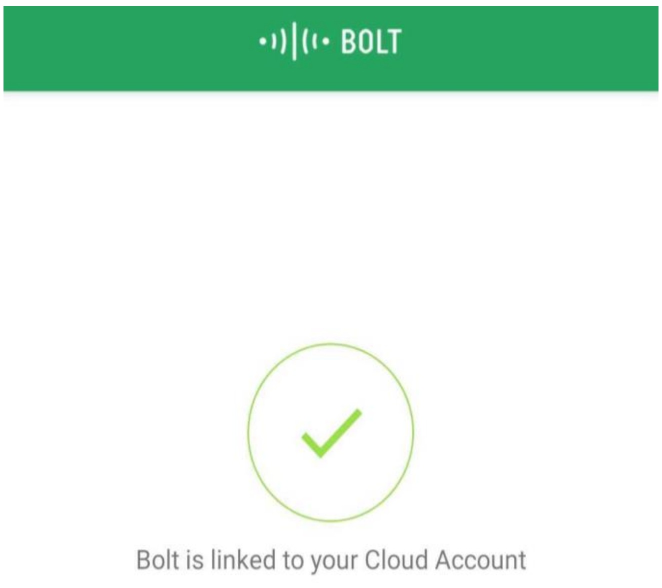
1. If you are connecting an LDR (Light Dependent Resistor) then use the pin A0. This is because LDR is an analogue sensor and A0 is the only pin with an ADC on it.

2. If you are connecting an LED then use any pin from 0 to 4. All these five pins are Digital I/O (Input and Output) pins. Since LED is an output device you may connect to any of these pins. Keep in mind that you cannot connect it to Pin A0 as it is an analog only pin.

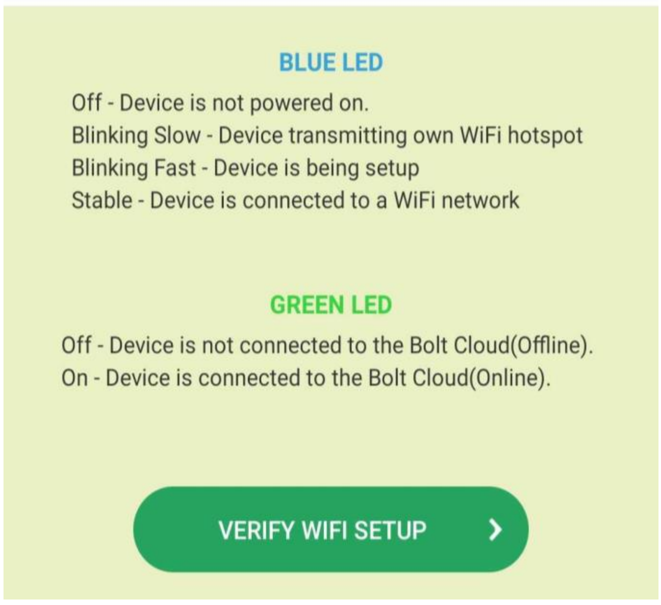
Pin	**Type**	**Description**
A0	Analog Input	This pin comes with an ADC (Analouge to Digital Converter). This is the only pin to which we can connect an Analouge sensor. This is an input only pin i.e. it can only collect input. It does not give any output.
0 , 1, 2, 3, 4	Digital I/O (Input and Output)	You can only connect a digital sensor to this pin.
TX	UART Pin	
RX	UART Pin	
3.3V	Power pin which gives a value of 3.3 volts.	
5V	Power pin which gives a value of 5 volts.	
GND	Ground pin	Used to ground and complete any circuit connection.

3.3 Building your first IoT sensor project

Step 1: Take out your WiFi Module
Step 2: Power it on
Step 3: Check your WiFi network frequency
Step 4: Login into the Bolt Cloud
Step 5: Connecting Bolt WiFi Module to the Internet



Bolt is linked to your Cloud Account



BLUE LED

Off - Device is not powered on.
Blinking Slow - Device transmitting own WiFi hotspot
Blinking Fast - Device is being setup
Stable - Device is connected to a WiFi network

GREEN LED

Off - Device is not connected to the Bolt Cloud(Offline).
On - Device is connected to the Bolt Cloud(Online).

Light Monitoring for Plants using LDR (Light dependent resistor)

Hardware components

- Bolt IoT Bolt WiFi Module
- LDR (Light Dependent Resistor)
- Resistor 10k ohm
- Jumper wires

Software apps and online services

- Bolt Cloud
- Bolt IoT Android App or Bolt IoT iOS App

Story

We all know that plants require sunlight for their healthy growth. However, at times we may not be able to keep a track of it or maybe we are not sure if our plants are getting enough sunlight.

In this project, we will build a system so that we could monitor the light our plants get and send the data to Bolt Cloud. In fact, this product is commercially available by Xiaomi_. But as makers, we shall build this product on our own.

What will you build as part of this project?

- Using this project, you will be able to build a light monitoring system to collect the data and send it to the cloud.
- You will also learn to visualise the data in form of graphs.
- This project can then be extended to Predict the future sensor values via machine learning over the Bolt Cloud.
- You may also send alerts over SMS and email using the features of Bolt Cloud Pro.

Gathering all required components

These are the components required.

1. Bolt WiFi Module

2. LDR (Light Dependent Resistor)

3. Resistor 10K

Let's get started and build our system

Before you move to Step 1, make sure that your Bolt WiFi Module is connected to Bolt Cloud and the green LED on Bolt Module is Glowing. If not then follow the steps in this project to set up the device: Setting Up the Bolt WiFi Module

Part A: Building the circuit

Switch off before you get started

Make sure you have not powered on your Bolt Module while connecting the circuit. This will ensure that in case we make any mistake, it will not short circuit your device. Switch off the power if it is connected.
We need to do two steps to build our circuit:

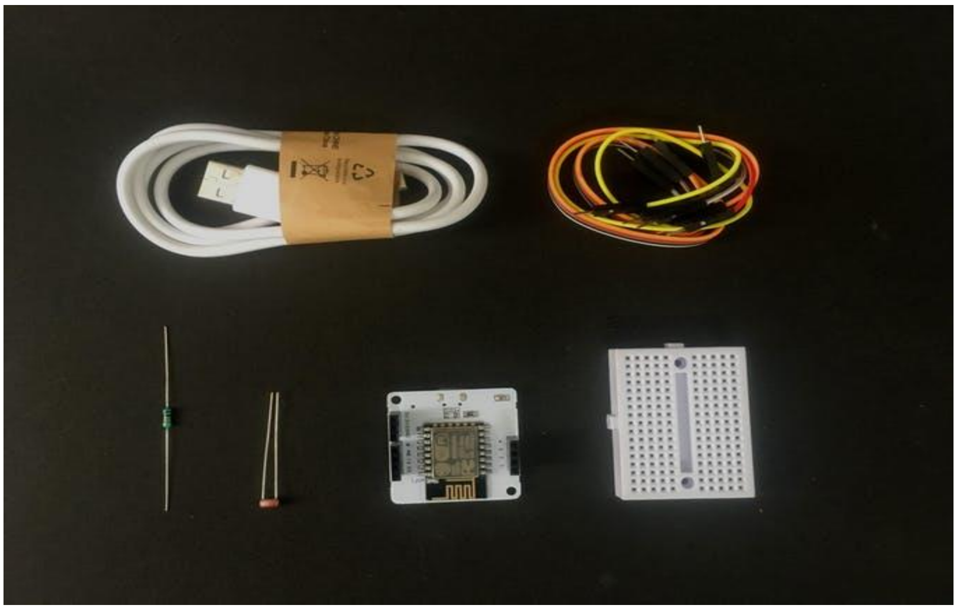
Step 1: Connect one end of the LDR to the A0 (analog) pin of the Bolt device and other ends of the LDR to the 5V pin of the Bolt as shown in the image below.

Step 2: Connect the 10K ohm resistor between the GND and A0 pin of the Bolt so that LDR and the resistor form a series connection.

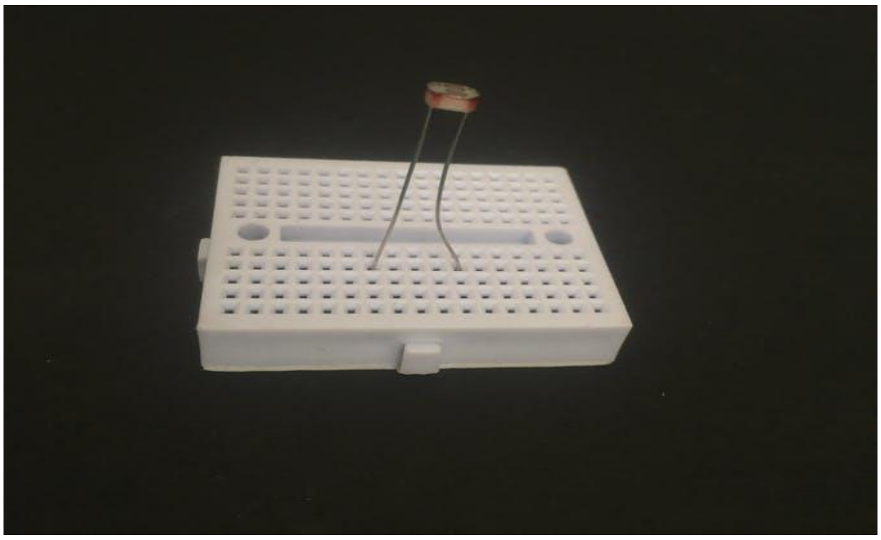
Confused about the two steps above? Don't worry. I will explain the above two steps in detail with help of images. Connect your circuit as shown in the images.

Before you connect, if you do not have experience with breadboard then I recommend watching this video: How to Use a Breadboard.

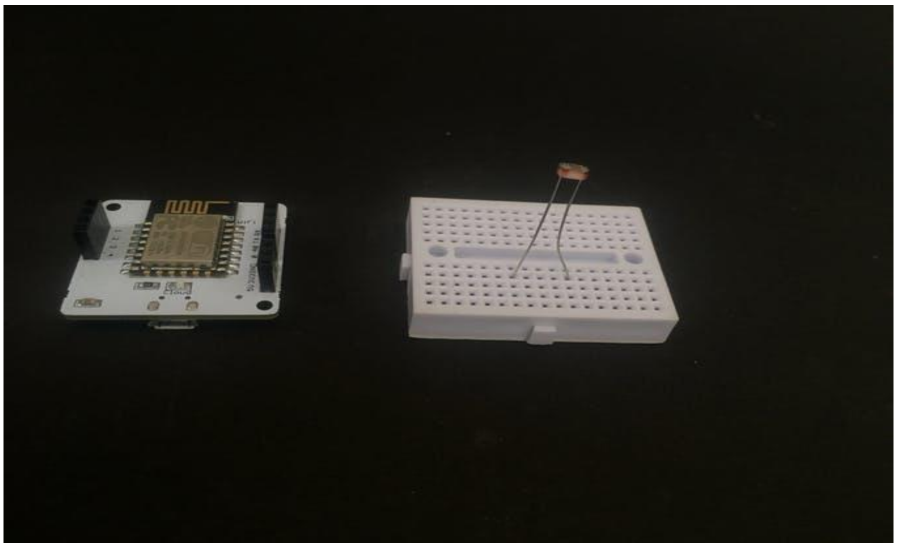
Step 1: Here are all the components that you will require



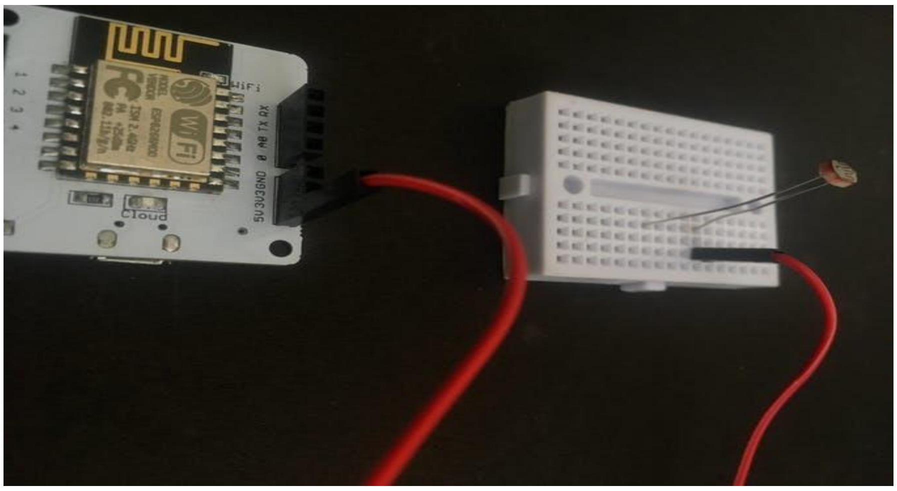
Step 2: Connect the LDR to the breadboard in the holes as shown below



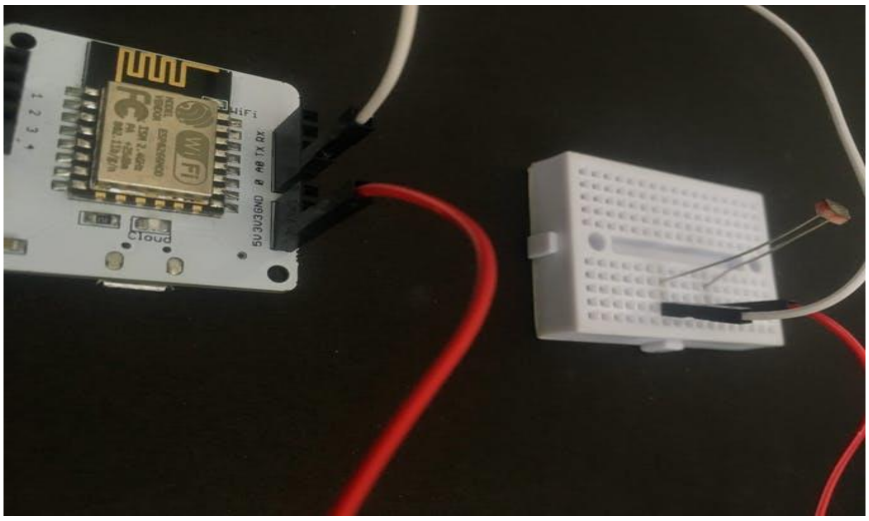
Step 3: Place the bolt WiFi module next to it



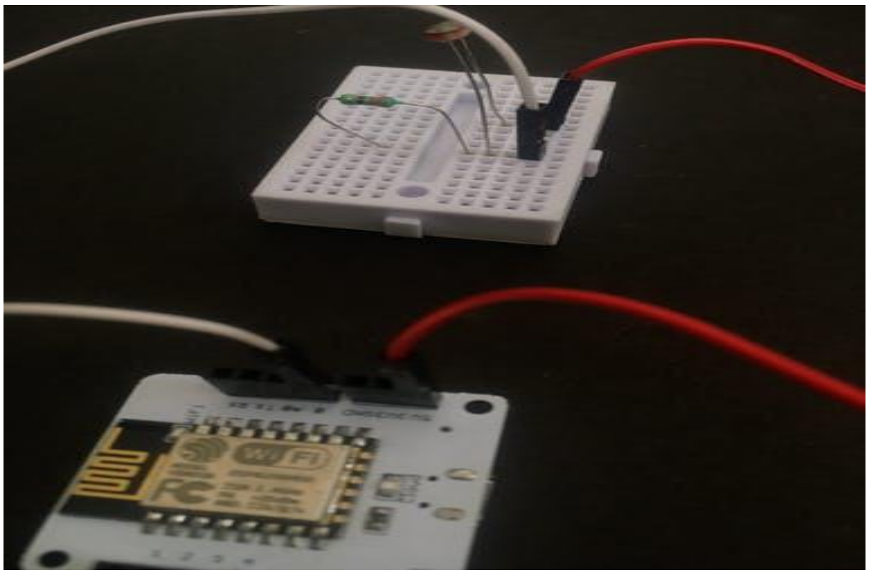
Step 4: Connecting one end of LDR to 5V of Bolt



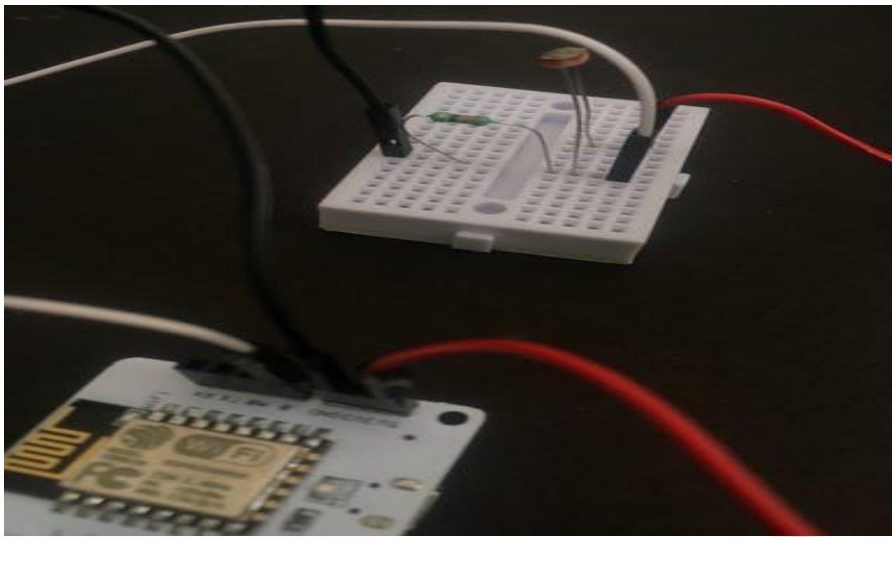
Step 5: Connecting the other end of LDR to A0 of Bolt



Step 7: Add the resistor in breadboard as shown



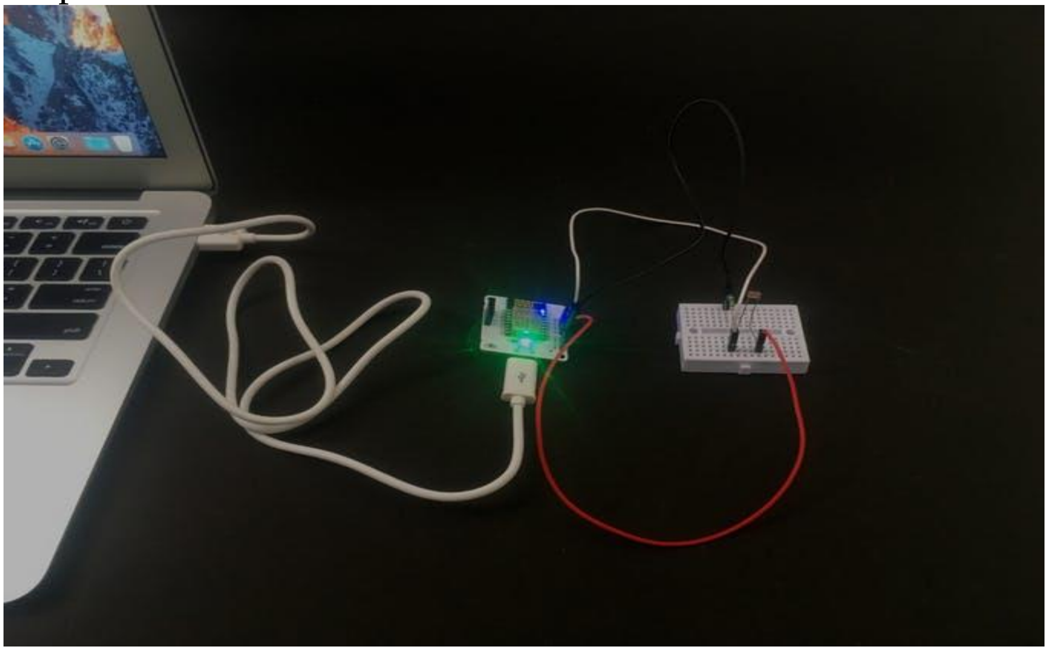
Step 8: Connecting resistor with the GND pin of Bolt WiFi Module



Check before you power on

Make sure connections are made accurately. Ideally, review them again. Wrong connections can lead to a short circuit which can further lead to the device getting damaged permanently.

Step 9: Power on the Bolt device



Now your circuit is ready.

In the next step, we will be to work on the software and cloud configuration.

Part B: Connect your Bolt device to the Bolt Cloud

Skip this step, if you already know how to connect your Bolt device to the Bolt Cloud. To check if it is connected to Cloud, have a look at the green coloured Cloud LED on the Bolt WiFi module. It should be glowing.

Follow the steps in this project to set up the device and to connect your Bolt device to the Bolt Cloud.:

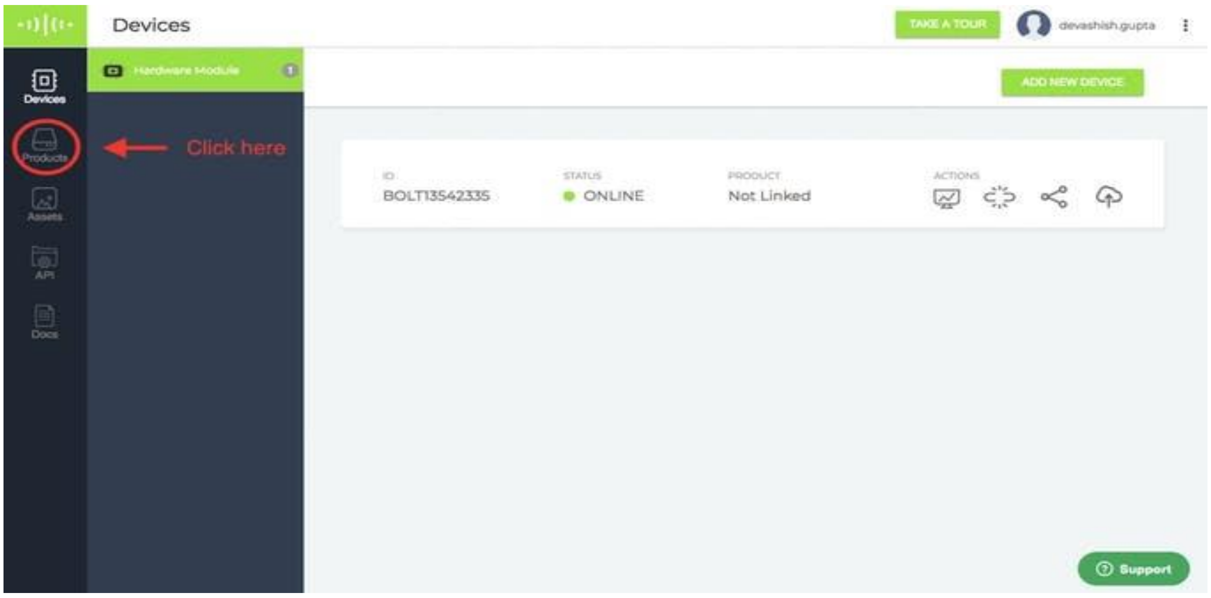
Setting Up the Bolt WiFi Module

Yes. I am aware that we have repeated this step. But we realised that many users tend to miss out on this step so wanted to be double sure about it. Now that our Bolt is active, we are just a step away from completing the system.

Part C: Collecting and visualising the data (Plotting Graph) on the Bolt Cloud

We will now work on the Bolt Cloud to build the software part of the system. Just follow these simple steps below:

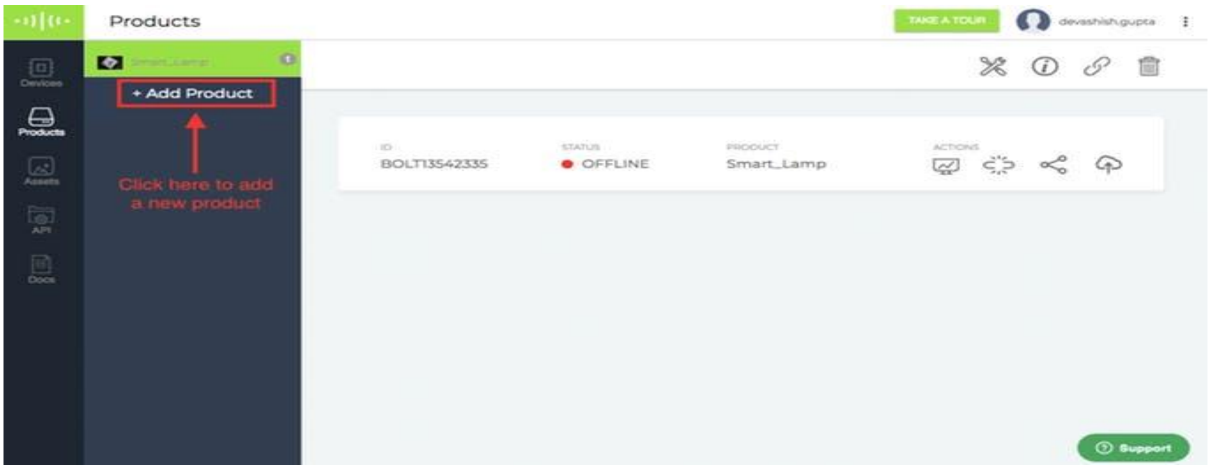
Step 1: Login into **cloud.boltiot.com** and click on the 'Product' tab.



ID	STATUS	PRODUCT	ACTIONS
BOLT13542335	● ONLINE	Not Linked	

Step 2:. Add a new product for your light monitoring system. Products are created once and can be used for multiple Bolt devices. This ensures scalability for the IoT products you build on Bolt.

Product names can only have alphabets, numbers, and underscore ( ) as a special character. Spaces are not allowed.



ID	STATUS	PRODUCT	ACTIONS
BOLT13542335	● OFFLINE	Smart_Lamp	

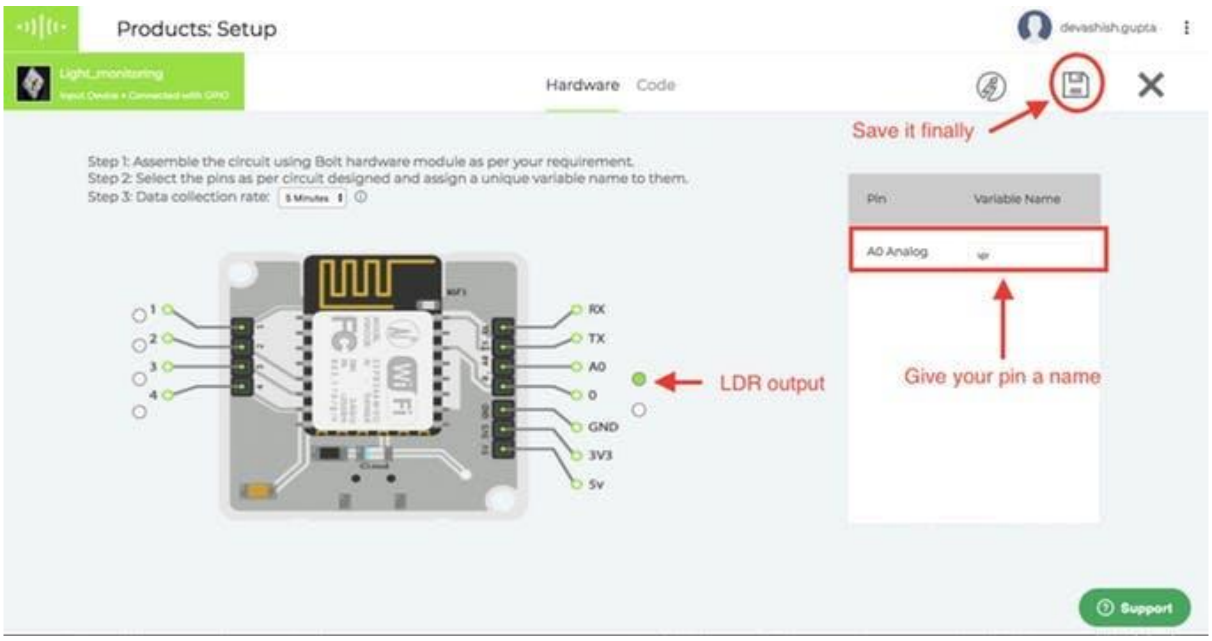
Step 3: We need to 'Configure your product'. For this click on the Configure Product icon as shown below in the image.

Select Configure tab

Step 4: Now choose the hardware pins which you plan to use for this project. Since LDR is an Analog Sensor and Bolt WiFi Module has only one Analogue pin, we have to choose only the A0 pin.

Step 5: Now give a suitable variable name to the pin value. By this, you will create a variable that will store the sensor value as it is received. We can then use this value in our project.

Step 6: With this, you have completed your hardware configuration. **Save your configuration** by clicking on the floppy disk icon.



Step 1: Assemble the circuit using Bolt hardware module as per your requirement.
Step 2: Select the pins as per circuit designed and assign a unique variable name to them.
Step 3: Data collection rate: 5 Minutes

Pin	Variable Name
A0 Analog	lpr

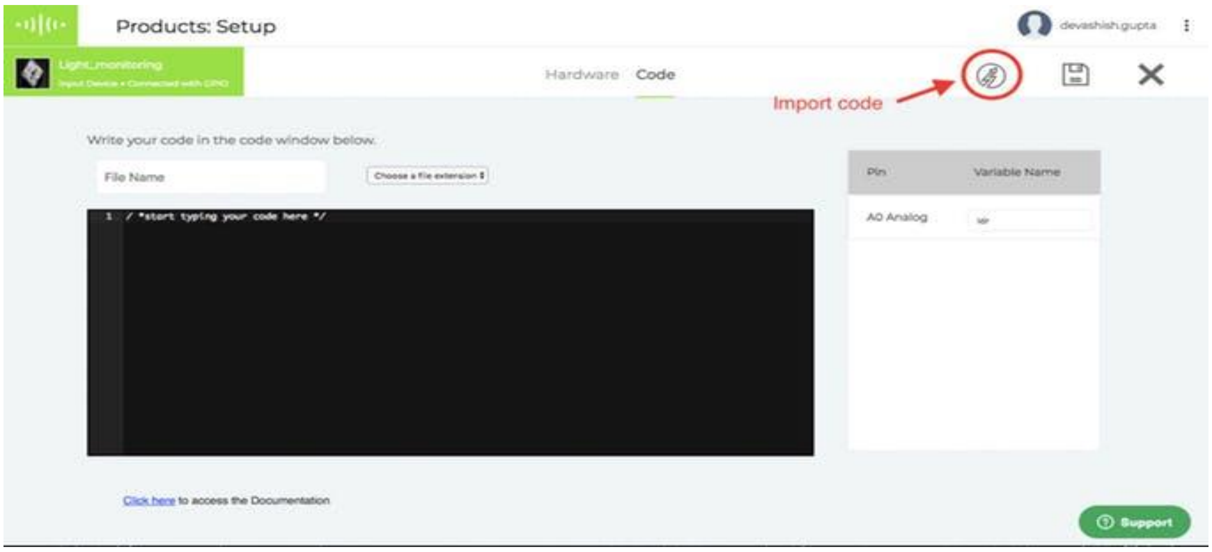
Hardware configuration

Step 7: Click on the 'Code' tab, and then click on the 'Import code' button to generate and import a new code.

While you can write your own code in HTML and Javascript for the Bolt cloud dashboard, in this tutorial of getting started, we shall import a template.

Here you will choose the option of Line Graph since we want to plot the line graph for our sensor value. However, in case you want to choose any other type of graph then you could do so.

The code tells the Bolt Cloud, how to plot the sensor data for visual representation.



```
/ *start typing your code here */
```

Understanding the code we imported
Now let me explain each line of the code so that you could make suitable changes as you wish.

setChartLibrary function sets the data visualisation library you would use. The most commonly used one on Bolt Cloud is the Google Library. However, you could use any other JavaScript or HTML code here to visualise the data.

setChartTitle function sets the title of the chart/graph. Give a suitable name for your graph here which will be shown in the heading of the page. This is different from the name of the code file.

setChartType function is where you choose which type of chart you want i.e. Line Graph, Bar Graph etc.

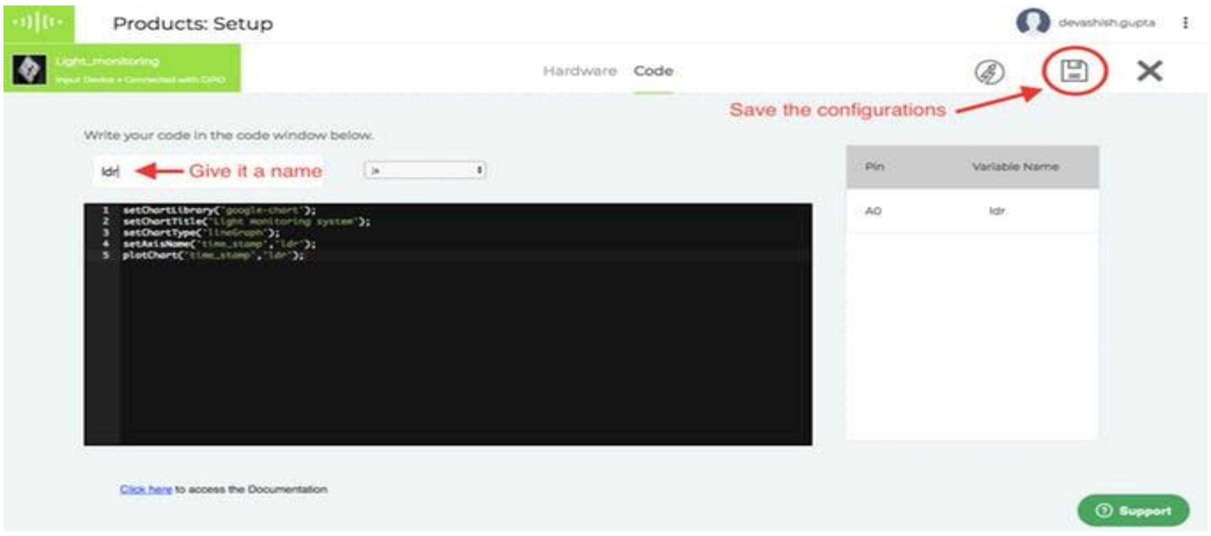
setAxisName will set the name for the X Axis and Y Axis

plotChart is where you choose which variable you want to choose in your chart.

If you want to know more about Data Visualisation and more features available in bolt cloud for it then you could click here. I would recomend that you visit this section after you have completed build this first project.

Step 8:

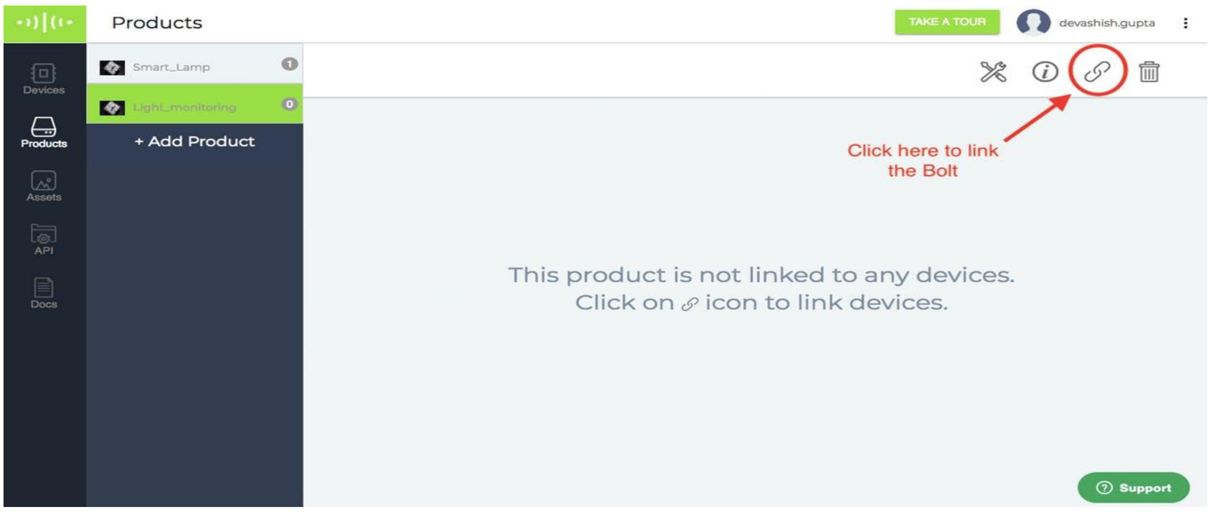
- Give a name to your code. Code names can only have an underscore (_) as a special character. Spaces are not allowed.
- Choose the extension as **js** since this is a Javascript code.
- Save the configurations



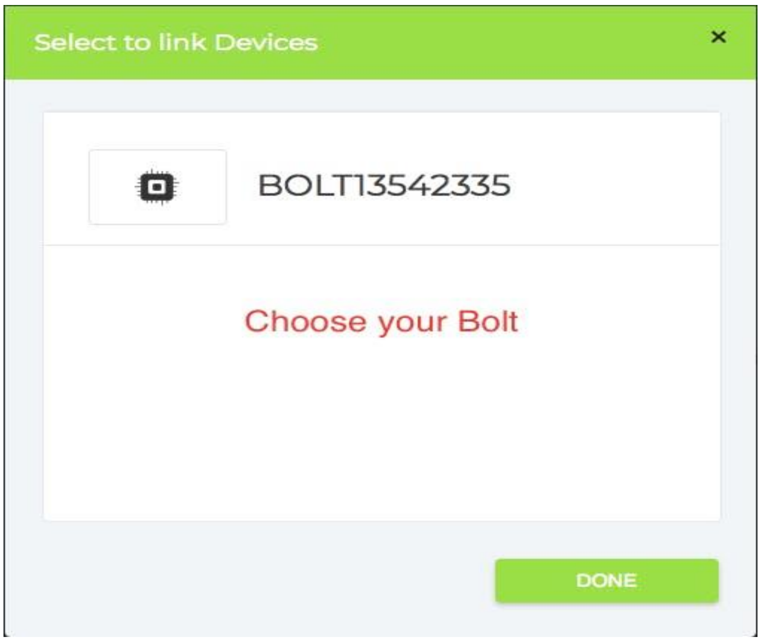
```
setChartLibrary('google-chart');
setChartTitle('Light monitoring system');
setChartType('lineGraph');
setAxisName('time_stamp','ldr');
plotChart('time_stamp','ldr');
```

Give it a name

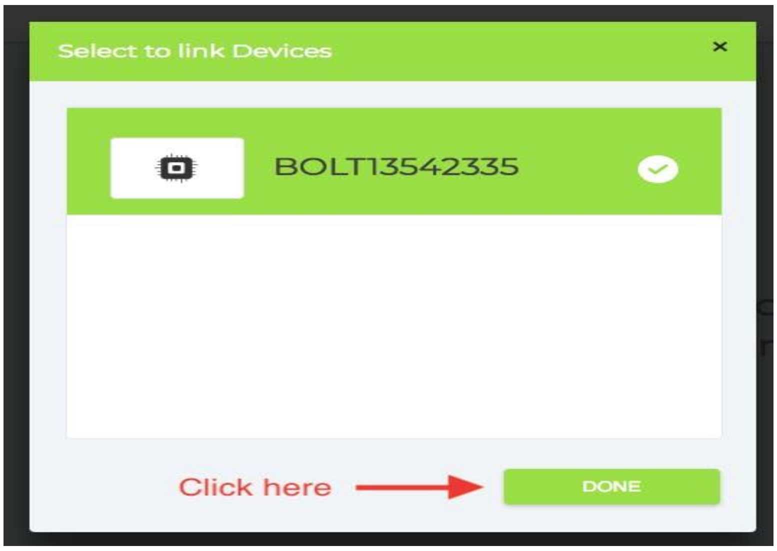
Step 9: Link Bolt with the 'Light_monitoring' product that you created. Linking allows your Bolt to access the configurations defined for a product.



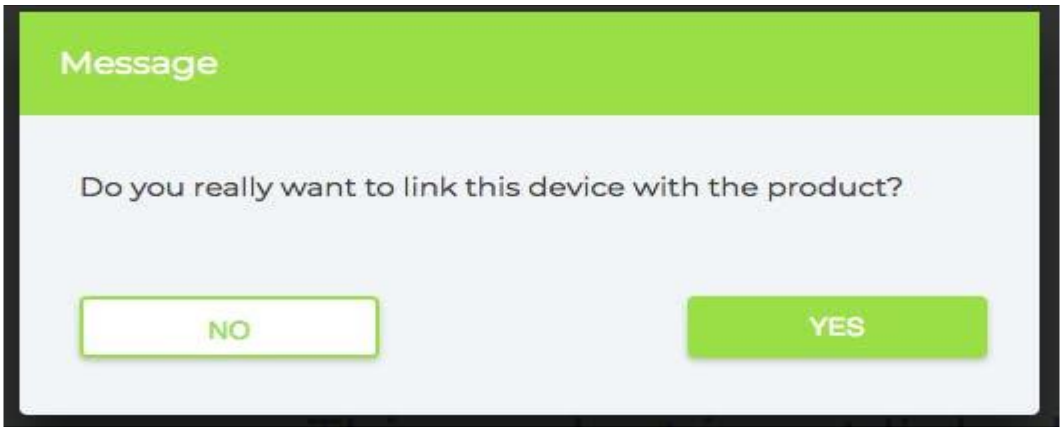
Link the Bolt



Choose your Bolt



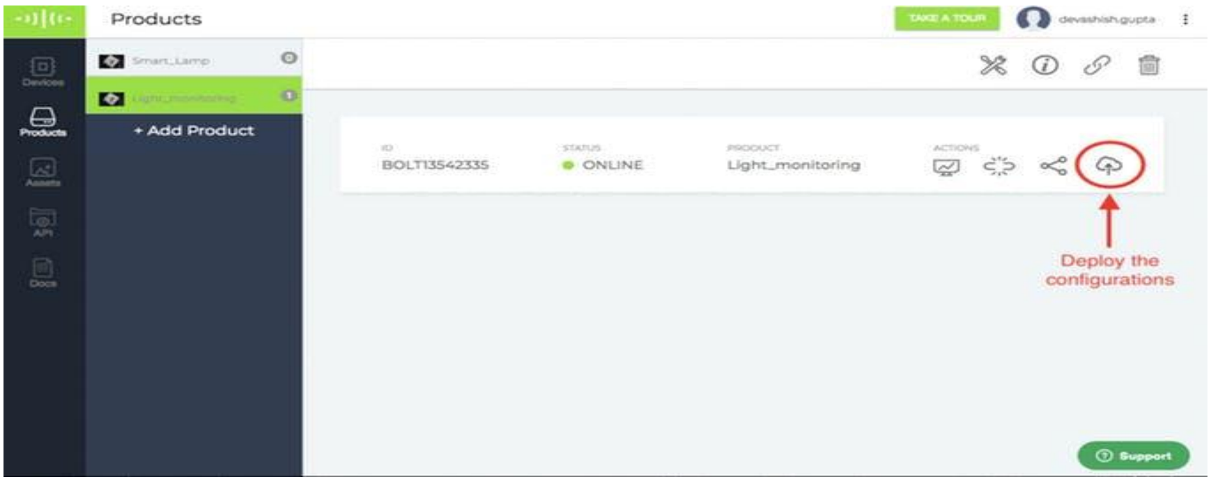
Save it



Do you really want to link this device with the product?

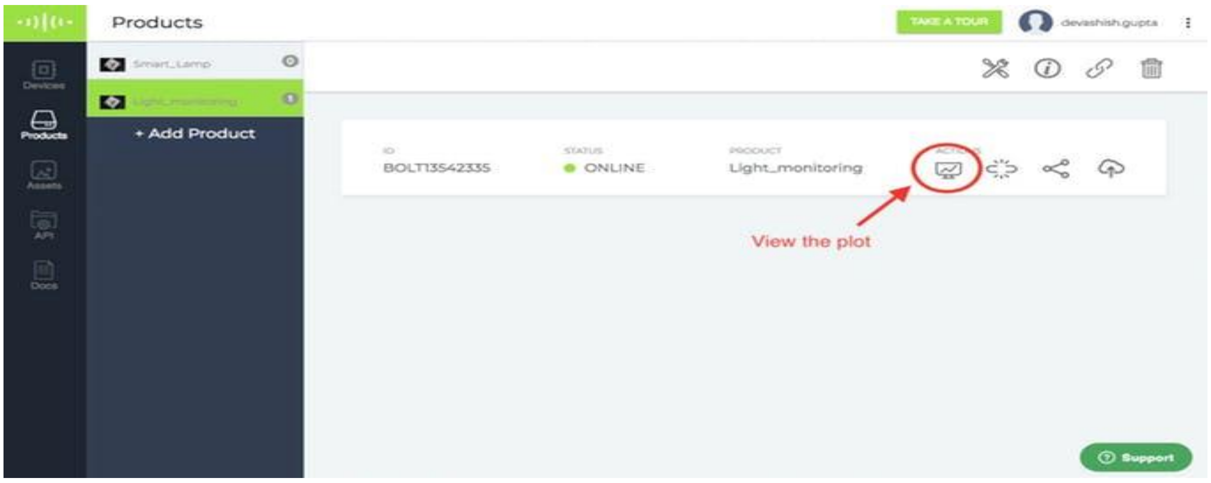
Click yes

Step 10: Deploy the code to your Bolt. Deploying means transferring the instructions and configuration to the Bolt device.



Deploy code

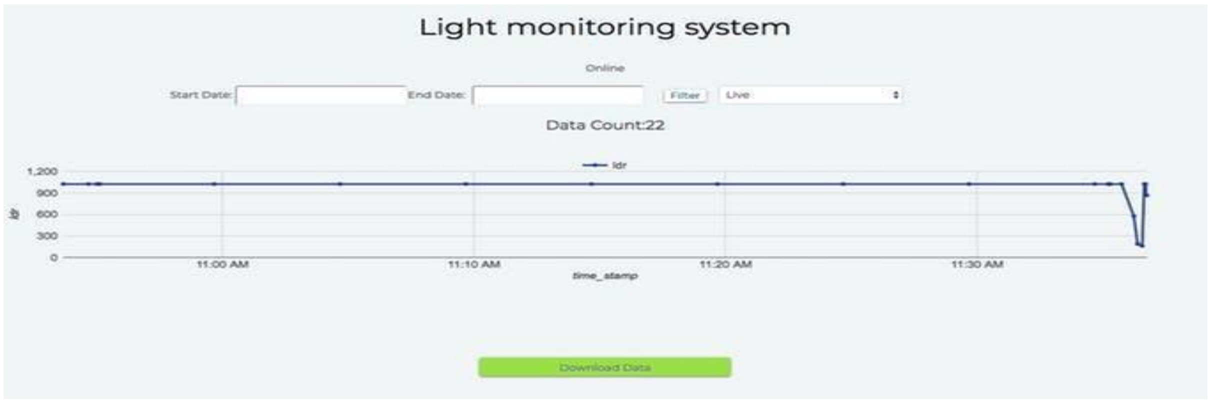
Step 11: We are now all set to see the output of our system. Click on the view this device button as shown with a red circle in the image below.



See output

Step 12: The data is sent to the cloud once every 5 minutes (this time could be different based on the time you choose in configuration). If you get a few sample data points at a faster rate, click on the **Push Data to Cloud** button. This button will send the latest data value to Bolt Cloud.

Output Plot



Working principle

This project is based on the principle that whenever the light falling on the sensor changes, the resistance of sensor changes which is then converted into a change in voltage. The ADC pin on Bolt WiFi Module converted this analog voltage level into digital values which are shown on the graphs.

We connect the LDR between 5v pin and the analog input pin (A0), so that when light intensity increases, the resistance of LDR decreases so the voltage across the LDR decreases and as a result, the voltage on the analog input pin increases.

This means that as the **light intensity increases, the voltage on the analog input pin also increases**. The Bolt then converts that the voltage a 10 bit (10 places in binary number system) digital value that varies from 0-1024 (0 to 2 raised to 10).

This digital data is then sent to the cloud where it is plotted for visual representation.

3.4 HTML Programming Language and Web Development

- HTML Tags
- Development Environment for Writing HTML Code
- Setting up Android phone for writing HTML code
- Building your first Webpage
- Paragraph tag in HTML
- Headings in HTML
- Text color in HTML
- Background color in HTML
- Embedding images in HTML
- Resizing images in HTML
- HTML Hyperlink
- Image as link in HTML
- span tag in HTML
- div tag in HTML
- Difference between div and span tags

3.5 Java Script Programming Language

- Introduction to JavaScript
- Variables in JavaScript
- Operators in JavaScript
- Functions in JavaScript
- getElementById in Javascript
- Comparison Operators in JavaScript
- Conditional statement in JavaScript
- JavaScript Events
- Ajax in JavaScript
- Ajax in JavaScript (Code)

Bolt IoT Javascript library

3rd Party JS library to control your Bolt devices
The JS library can be loaded in the head tag of your HTML document using the code,

HTML
```
<script src="https://unpkg.com/bolt-iot-wrapper/umd/boltIotWrapper.min.js"></script>
```

Once you have added the tag to include the script, you will need to first add the Bolt device using the device ID and API key. After this, you can control your Bolt device or get information about the device. Supported functions are,

1. Initialise using
   Devices.add("BOLTXXXX", "API_KEY");

2. Add Device(Optional)
   const instance = Devices.read("BOLTXXXX", "API_KEY");
   Use this function to add another device.

3. Analog Functions
   1. instance.Analog.read() // reads analog pin data return a promise
   2. instance.Analog.loopRead({milliseconds},{callback}) // reads analog pin continously in particular interval

4. Digital Functions
   1. instance.Digital.read({pin: "" | pins: []}) // read Digital signals of single or multiple pins. returns a promise
   2. instance.Digital.write({pin: "", state: ""}) // write digital signals
   3. instance.Digital.loopRead({pin | pins[]},{milliseconds},{callback}) // read digital singals in particular interval

5. UART Functions
   1. instance.UART.begin({baudRate}) // sets the baud rate
   2. instance.UART.read({till}) // reads till
   3. instance.UART.write({data}) // writes data
   4. instance.UART.readWrite({data},{till}) // reads and writes data

6. Utility Functions
   1. instance.Utility.isOnline() // returns a promise with resolved valueas true/false
   2. instance.Utility.restart() // restarts the Bolt device
   3. instance.Utility.version() // returns the device version details

3.6 Data Visualization and Analytics

- Line Graph
- Bar Graph
- Scatter Graph
- Area Graph
- Table Chart
- Stepped Graph
- Histogram Graph
- Gauge
- Multiple Graph
- Deploy Configuration Feature
- Arithmetic Operations (Beta)

3.7 APIs Key’s & Home Automation

APIs Key’s

The Bolt Cloud API provides an interface for communication between the Bolt devices and any 3rd party system e.g. mobile application, web server, python programs etc. The API contains very intuitive control, monitoring, communication and utility functions for the Bolt Devices connected to your account. The Bolt Cloud API uses HTTP protocol for the communication and uses the HTTP GET and HTTP POST methods. Hence users can execute actions and retrieve information from Bolt devices programmatically using conventional HTTP requests.

Here are a few use cases of the API:

- Use the API in native apps on iOS and Android to control and monitor Bolt devices over the Internet.
- Pull collect sensor data connect to Bolt device, to any other cloud to run your custom AI algorithms and analytics.
- Connect Bolt Cloud to any VPS (Virtual Private Server) and run your code in any language of your choice. Refer sample codes.
- Remote Operating System: Using the API, Bolt devices can work like a board with an OS i.e. similar to Raspberry Pi or Beagle Bone, with the exception of the OS, which in this case, will reside on a remote VPS (Virtual Private Server). The Bolt will receive data from the sensors and push to the VPS with a Linux OS. The processing will take place on the VPS and it will push the commands to control motors, LEDs, and actuators to the Bolt device. You can use all the features of a Linux OS in this kind of a system.

3.8 VPS (Virtual Private Server)

- Further understanding Cloud and Cloud Computing
- Before we create a Digital Ocean account
- Digital Ocean Droplet
- Troubleshooting issues with Digital Ocean
- Alternatives to Digital Ocean Droplet – Option1
- Alternatives to Digital Ocean Droplet - Option 2
- How to SSH into a linux based cloud server
- Accessing the Digital Ocean Droplet

3.9 Introduction to Linux OS

- Structure of an Operating System
- Interesting Facts about Linux
- Basic Linux Commands
- What is Sudo?
- Easter eggs in Linux

3.10 Python Programming Language

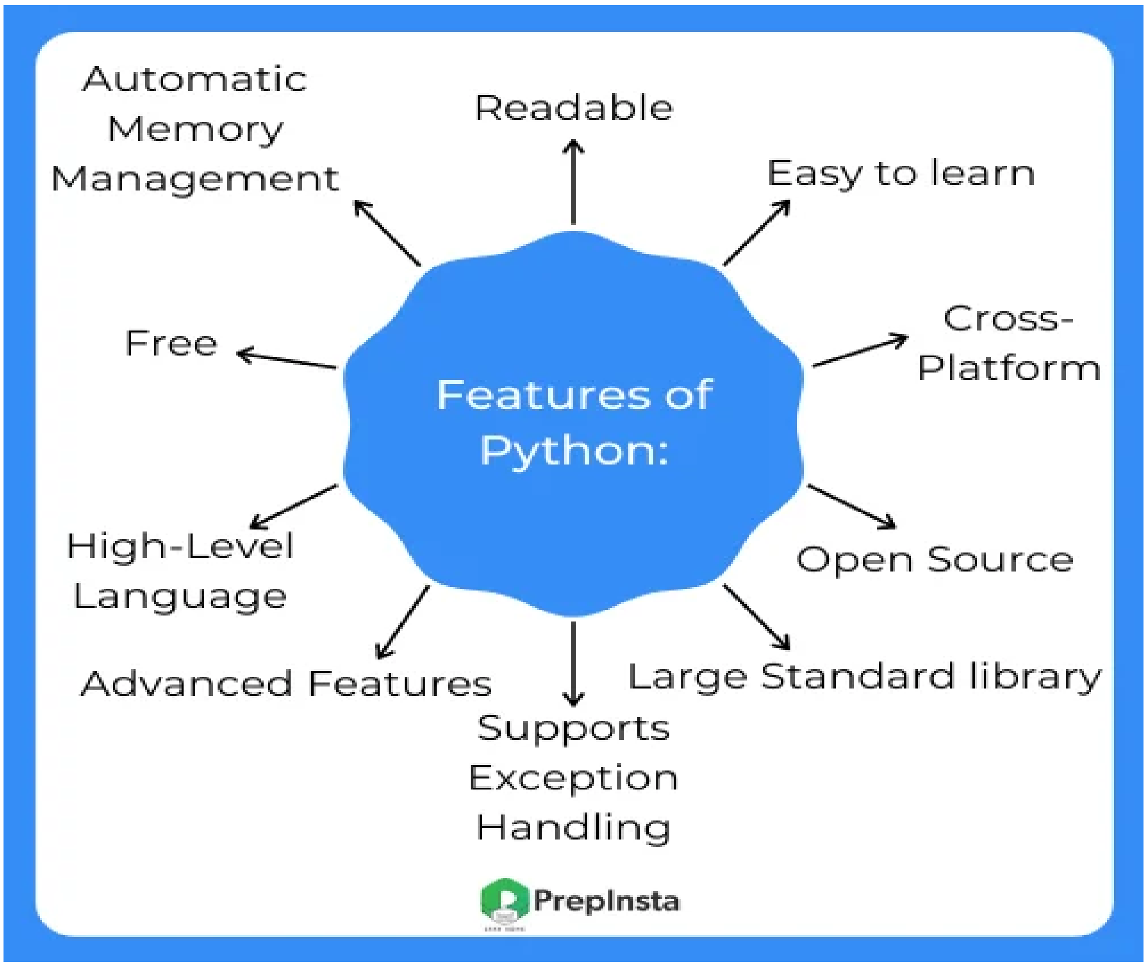
- Introduction to Python
- Hello World Program
- Indentation of Code
- Debugging Commons Problems
- Arithmetic’s
- Variables
- Conditional Statement in Python

- While Loop
- For Loop
- Random Function
- Setup the Bolt Python Library
- Checking Device Status and Restarting the Device
- Controlling LED using Bolt Python Library
- LED Intensity (Brightness) Control

Python History

Python is a widely used general-purpose, high-level programming language. It was initially designed by Guido van Rossum in 1991 and developed by Python Software Foundation. It was mainly developed for emphasis on code readability, and its syntax allows programmers to express concepts in fewer lines of code. The two of the most used versions have to do with Python 2.x & 3.x. There is a lot of competition between the two and both of them seem to have quite a number of different fanbase.

For various purposes such as developing, scripting, generation and software testing, this language is utilised. Due to its elegance and simplicity, top technology organisations like Dropbox, Google, Quora, Mozilla, Hewlett-Packard, Qualcomm, IBM, and Cisco have implemented Python.


Features of Python:
Automatic Memory Management
Readable
Easy to learn
Cross-Platform
Free
Open Source
High-Level Language
Large Standard library
Advanced Features
Supports Exception Handling
PrepInsta

Features in Python

There are many features in Python, some of which are discussed below

1. Easy to code:

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in python language and anybody can learn python basics in a few hours or days. It is also a developer-friendly language.

2. Free and Open Source:

Python language is freely available at the official website and you can download it from the given download link below click on the **Download Python** keyword.

Download Python

Since it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

3. Object-Oriented Language:

One of the key features of python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, object encapsulation, etc.

4. GUI Programming Support:

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in python.

PyQt5 is the most popular option for creating graphical apps with Python.

5. High-Level Language:

Python is a high-level language. When we write programs in python, we do not need to remember the system architecture, nor do we need to manage the memory.

6. Extensible feature:

Python is a Extensible language. We can write some Python code into C or C++ language and also we can compile that code in C/C++ language.

7. Python is Portable language:

Python language is also a portable language. For example, if we have python code for windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

8. Python is Integrated language:

Python is also an Integrated language because we can easily integrate python with other languages like c, c++, etc.

9. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. Unlike other languages C, C++, Java, etc. there is no need to compile python code; this makes it easier to debug our code. The source code of python is converted into an immediate form called **bytecode**.

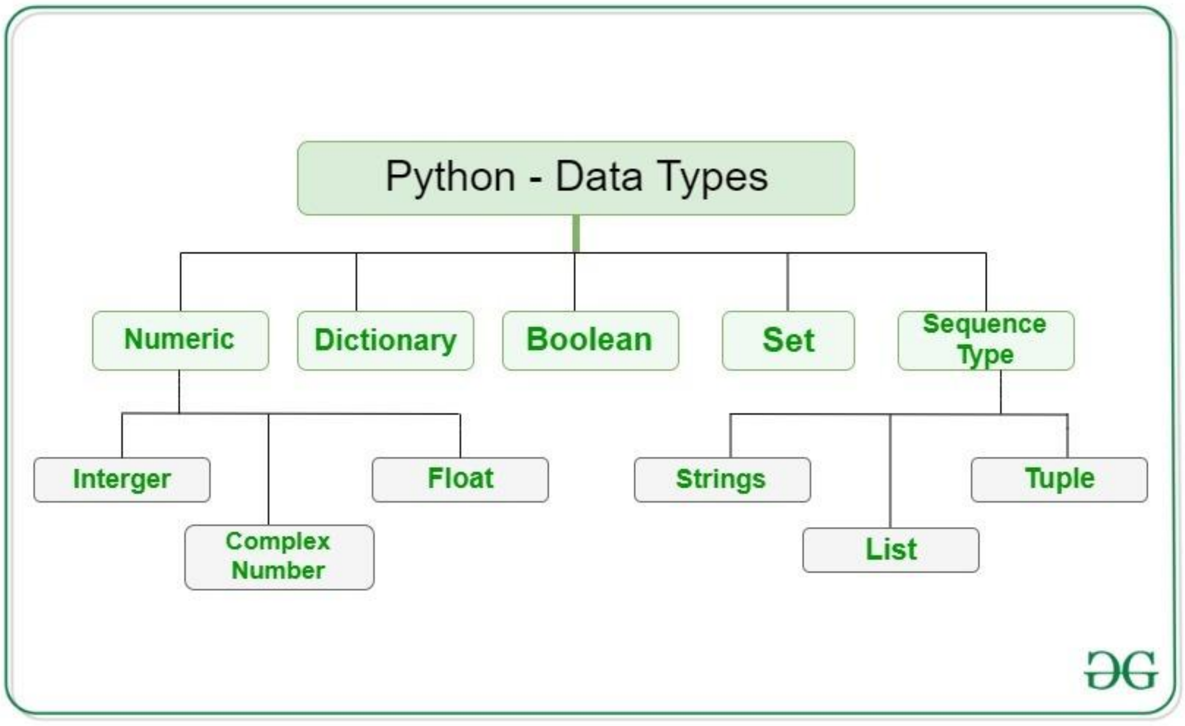
10. Large Standard Library

Python has a large standard library which provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in python such as regular expressions, unit-testing, web browsers, etc.

11. Dynamically Typed Language:

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don’t need to specify the type of variable.

Data Types in Python



Numeric

In Python, numeric data types represent the data which has numeric value. Numeric values can be integers, floating numbers or even complex numbers. These values are defined as int, float and complex classes in Python.

- **Integers** – This value is represented by int class. It contains positive or negative whole numbers (without fraction or decimal). In Python there is no limit to how long an integer value can be.
- **Float** – This value is represented by the float class. It is a real number with floating point representation. It is specified by a decimal point. Optionally, the character e or E followed by a positive or negative integer may be appended to specify scientific notation.
- **Complex Numbers** – Complex numbers are represented by complex classes. It is specified as (real part) + (imaginary part)j. For example – 2+3j.

Sequence Type

In Python, sequence is the ordered collection of similar or different data types. Sequences allow you to store multiple values in an organised and efficient fashion. There are several sequence types in Python –

- String
- List
- Tuple

1) String

In Python, Strings are arrays of bytes representing Unicode characters. A string is a collection of one or more characters put in a single quote, double-quote or triple quote. In python there is no character data type, a character is a string of length one. It is represented by the str class.

Creating String

Strings in Python can be created using single quotes or double quotes or even triple quotes.

2) List
Lists are just like the arrays, declared in other languages which is an ordered collection of data. It is very flexible as the items in a list do not need to be of the same type.

Creating List
Lists in Python can be created by just placing the sequence inside the square brackets[].

3) Tuple
Just like list, tuple is also an ordered collection of Python objects. The only difference between tuple and list is that tuples are immutable i.e. tuples cannot be modified after it is created. It is represented by tuple class.

Creating Tuple

In Python, tuples are created by placing a sequence of values separated by 'comma' with or without the use of parentheses for grouping of the data sequence. Tuples can contain any number of elements and of any datatype (like strings, integers, list, etc.).

Boolean

Data type with one of the two built-in values, True or False. Boolean objects that are equal to True are truthy (true), and those equal to False are falsy (false). But non-Boolean objects can be evaluated in Boolean context as well and determined to be true or false. It is denoted by the class bool.

Note – True and False with capital 'T' and 'F' are valid booleans otherwise python will throw an error.

Set

In Python, Set is an unordered collection of data types that is iterable, mutable and has no duplicate elements. The order of elements in a set is undefined though it may consist of various elements.

Creating Sets

Sets can be created by using the built-in set() function with an iterable object or a sequence by placing the sequence inside curly braces, separated by ‘comma’. Type of elements in a set need not be the same, various mixed-up data type values can also be passed to the set.

Dictionary

Dictionary in Python is an unordered collection of data values, used to store data values like a map, which unlike other Data Types that hold only a single value as an element, Dictionary holds key:value pair. Key-value is provided in the dictionary to make it more optimised. Each key-value pair in a Dictionary is separated by a colon :, whereas each key is separated by a ‘comma’.

Creating Dictionary

In Python, a Dictionary can be created by placing a sequence of elements within curly {} braces, separated by ‘comma’. Values in a dictionary can be of any datatype and can be duplicated, whereas keys can’t be repeated and must be immutable. Dictionary can also be created by the built-in function dict(). An empty dictionary can be created by just placing it in curly braces{}.

3.11 Machine Learning

Whether or not you’re excited by the idea of artificial neural networks one day growing sophisticated enough to replicate human consciousness, there are undeniable practical advantages to machine learning, namely:

- **Intelligent big data management** – The sheer volume and variety of data being generated as humans and other environmental forces interact with technology would be impossible to process and draw insights from without the speed and sophistication of machine learning.
- **Smart devices** – From wearable devices that track health and fitness goals to self-driving cars to "smart cities" with infrastructure that can automatically reduce wasted time and energy, the Internet of Things (IoT) holds great promise, and machine learning can help make sense of this significant increase in data.
- **Rich consumer experiences** – Machine learning enables search engines, web apps, and other technology to customize results and recommendations to match user preferences, creating delightfully personalized experiences for consumers.

How does machine learning work?

Machine learning is incredibly complex and how it works varies depending on the task and the algorithm used to accomplish it. However, at its core, a machine learning model is a computer looking at data and identifying patterns, and then using those insights to better complete its assigned task. Any task that relies upon a set of data points or rules can be automated using machine learning, even those more complex tasks such as responding to customer service calls and reviewing resumes.

Depending on the situation, machine learning algorithm’s function using more or less human intervention/reinforcement. The four major machine learning models are supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning.

With **supervised learning**, the computer is provided with a labeled set of data that enables it to learn how to do a human task. This is the least complex model, as it attempts to replicate human learning.

With **unsupervised learning**, the computer is provided with unlabeled data and extracts previously unknown patterns/insights from it. There are many different ways machine learning algorithms do this, including:

- Clustering, in which the computer finds similar data points within a data set and groups them accordingly (creating "clusters").
- Density estimation, in which the computer discovers insights by looking at how a data set is distributed.
- Anomaly detection, in which the computer identifies data points within a data set that are significantly different from the rest of the data.
- Principal component analysis (PCA), in which the computer analyzes a data set and summarizes it so that it can be used to make accurate predictions.

With **semi-supervised learning**, the computer is provided with a set of partially labeled data and performs its task using the labeled data to understand the parameters for interpreting the unlabeled data.

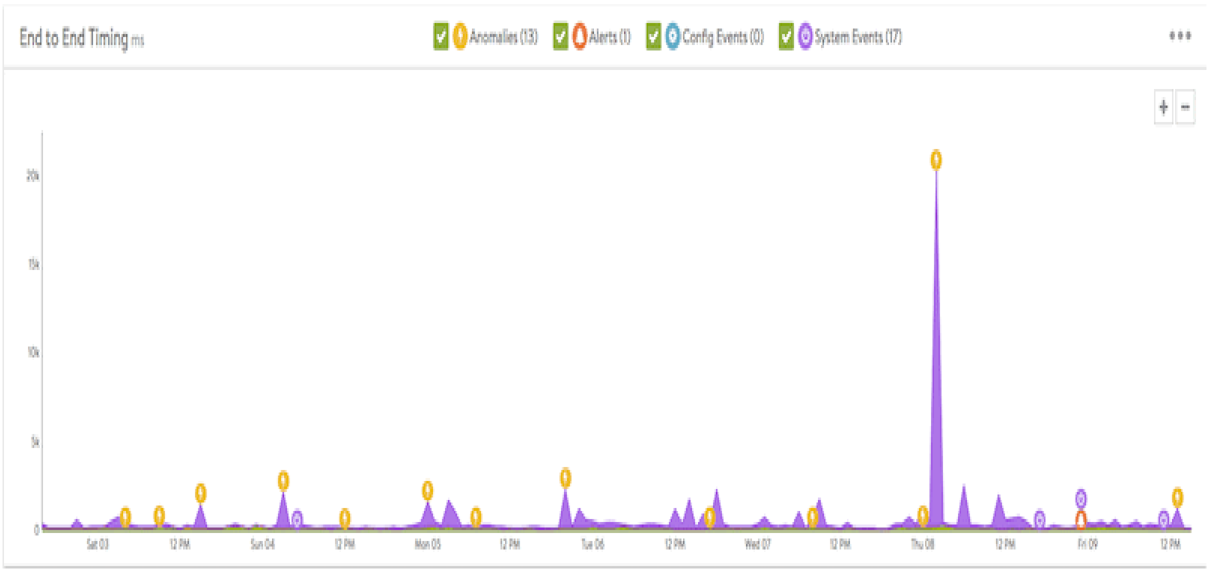
With **reinforcement learning**, the computer observes its environment and uses that data to identify the ideal behavior that will minimize risk and/or maximize reward. This is an iterative approach that requires some kind of reinforcement signal to help the computer better identify its best action.

3.12 Anomaly Detection

Anomaly detection, also called outlier detection, is the identification of unexpected events, observations, or items that differ significantly from the norm. Often applied to unlabeled data by data scientists in a process called unsupervised anomaly detection, any type of anomaly detection rests upon two basic assumptions:

- Anomalies in data occur only very rarely
- The features of data anomalies are significantly different from those of normal instances

Typically, anomalous data is linked to some sort of problem or rare event such as hacking, bank fraud, malfunctioning equipment, structural defects / infrastructure failures, or textual errors. For this reason, identifying actual anomalies rather than false positives or data noise is essential from a business perspective.



Anomaly Detection FAQs

What is Anomaly Detection?

Anomaly detection is the identification of rare events, items, or observations which are suspicious because they differ significantly from standard behaviors or patterns. Anomalies in data are also called standard deviations, outliers, noise, novelties, and exceptions.

In the network anomaly detection/network intrusion and abuse detection context, interesting events are often not rare—just unusual. For example, unexpected jumps in activity are typically notable, although such a spurt in activity may fall outside many traditional statistical anomaly detection techniques.

Many outlier detection methods, especially unsupervised techniques, do not detect this kind of sudden jump in activity as an outlier or rare object. However, these types of micro clusters can often be identified more readily by a cluster analysis algorithm.

There are three main classes of anomaly detection techniques: unsupervised, semi-supervised, and supervised. Essentially, the correct anomaly detection method depends on the available labels in the dataset.

Supervised anomaly detection techniques demand a data set with a complete set of "normal" and "abnormal" labels for a classification algorithm to work with. This kind of technique also involves training the classifier. This is similar to traditional pattern recognition, except that with outlier detection there is a naturally strong imbalance between the classes. Not all statistical classification algorithms are well-suited for the inherently unbalanced nature of anomaly detection.

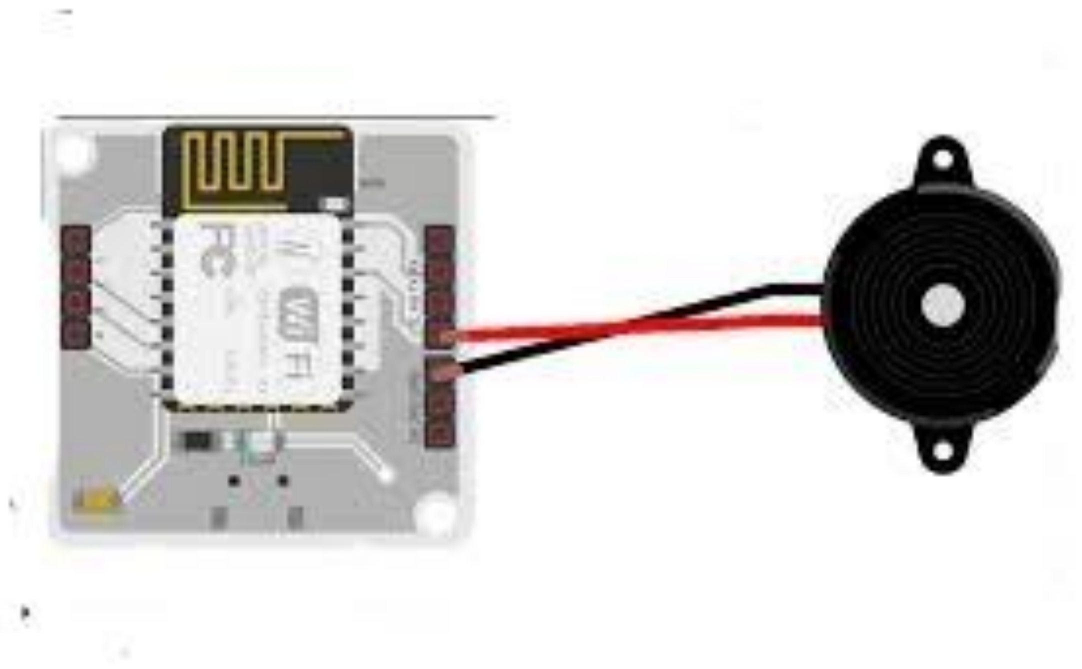
Semi-supervised anomaly detection techniques use a normal, labeled training data set to construct a model representing normal behavior. They then use that model to detect anomalies by testing how likely the model is to generate any one instance encountered.

Unsupervised methods of anomaly detection detect anomalies in an unlabeled test set of data based solely on the intrinsic properties of that data. The working assumption is that, as in most cases, the large majority of the instances in the data set will be normal. The anomaly detection algorithm will then detect instances that appear to fit with the rest of the data set least congruently.

4.Project Submission

Controlling Buzzer Using Bolt Iot

In this project ,i discussed about Controlling buzzer using bolt WIFI module with internet.



Components which I have used in this project are one Buzzer, one WIFI module, USB cable, then + terminal of the Buzzer is connected to the 0 & - terminal is connected to the ground then we have to give 5v power supply the we have to give the HTML source code then run the code then, when we click ON Buzzer will be ON then we click OFF Buzzer will be OFF.

SOURCE CODE

```html
<!DOCTYPE html>
<html>
    <head>
        <title>Bolt IoT Platform</title>
        <script type="text/javascript"
src="https://cloud.boltiot.com/static/js/boltCommands.js"></script>
        <script>
        setKey('{{ApiKey}}','{{Name}}');
        </script>
    </head>
    <body>
        <center>
        <button onclick="digitalWrite(0, 'HIGH');">ON</button>
        <button onclick="digitalWrite(0, 'LOW');">OFF</button>
        </center>
    </body>
</html>
```

OUT PUT :