**ROBAN SYSTEMS**

**AN HELPING HAND FOR DISABLED PEOPLE**

**A PROJECT REPORT**

submitted by

**Ajay Das K, Fathima Irfana T P, Muhammed Fayis M T, Rabeeh C**

**Register Number: CCV19CS002, CCV19CS007, CCV19CS015, CCV19CS019**

**TO**

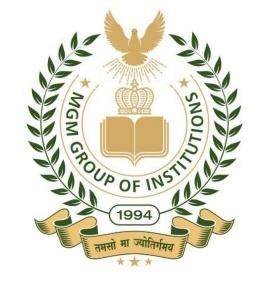
The APJ Abdul Kalam Technological University

in partial fulfilment of the requirements for the award of the Degree of

Bachelor Of Technology

In

Computer Science and Engineering

****

**Department Of Computer Science and Engineering**

**MGM COLLEGE OF ENGINEERING ANDPHARAMACEUTICAL**

**SCIENCES**

VALANCHERY

JANUARY 2023

**DECLARATION**

I undersigned hereby declare that the main project report " **Roban systems - An helping hand for disabled people**", submitted for partial fulfilment of the requirements for the award of the degree of Bachelor of Technology of the APJ Abdul Kalam Technological University, Kerala is a Bonafede work done by me under supervision of **MS. ANJANA**, Asst. Professor, Department of Computer Science and Engineering. This submission represents my ideas in my own words and where ideas or words of others have been included, we have adequately and accurately cited and referenced the sources. We also declare that we have adhered to academic honesty and integrity ethics and have not misrepresented or fabricated any data, idea, fact, or source in our submission. We understand that any violation of the above will causes disciplinary action by the institute and/or the University and can also evoke penal action from the sources that have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed as the basis for the award of any degree, diploma or similar title of any other University.

Place: Valanchery Signature

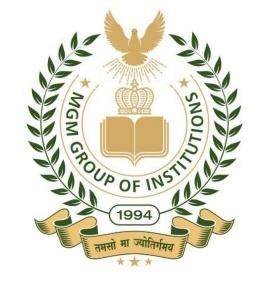
Date: 06 June 2023 Ajay Das K (CCV19CS002)

Fathima Irfana T P(CCV19CS007)

Muhammed Fayis M T (CCV19CS015)

Rabeeh C (CCV19CS019)

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING MGM COLLEGE OF ENGINEERING AND PHARMACEUTICAL SCIENCES, VALANCHERY**

****

**CERTIFICATE**

This is to certify that the project report entitled " **Roban Systems - An helping Hand for disabled people**" submitted by Ajay Das K, Fathima Irfana T P, Muhammed Fayis M T, Rabeeh C( CCV19CS002, CCV19CS007, CCV19CS015, CCV19CS019 )to the APJ Abdul Kalam Technological University in partial fulfilment of the requirements for the award of the Degree of Master of Technology in Computer Science and Engineering is a bonafide record of the project work carried out by her under my guidance and supervision during the year 2022- 2023. This report in any form has not been submitted to any other University or Institute for any purpose

Internal Supervisor External Supervisor Head of the Department

**ACKNOWLEDGEMENT**

Alone we can do so little; together we can do so much. Likewise, the present project work has been undertaken and completed with many people's direct and indirect help, and I would like to acknowledge the same.

First and foremost, I take immense pleasure in thanking the **Management** and respected principal **DR. PROF BABU JOHN,** for providing me with the wider facilities.

We sincerely thank **MS. SHABNA M**, Head of the Department of Computer Science and Engineering for allowing me to present this project and for timely suggestions.

We wish to express my deep sense of gratitude to the project coordinator **MS. SHABNA M**, Head of the Department of Computer Science and Engineering, who coordinated on the right path. Words are inadequate in offering my thanks to **MS. ANJANA T**, Asst Professor Department of Computer Science and Engineering, for her encouragement and guidance in carrying out the project.

Needless to mention that the teaching and the non-teaching faculty members had been the source of inspiration and timely support in the conduct of our project. We would like to express our heartfelt thanks to our beloved parents for their blessings, and our classmates for their help and wishes for the successful completion of this project.

Above all, we would like to thank the Almighty God for the blessings that helped us complete the venture smoothly. Without their support, it would be impossible for us to finish our work. That is why we wish to dedicate this section to recognize their support.

AJAY DAS K

FATHIMA IRFANA TP

MUHAMMED FAYIS M T

RABEEH C

**ABSTRACT**

Smart home applications are pervasive and have gained popularity due to the overwhelming use of the Internet of Things (IoT). The revolution in IoT technologies made homes more convenient, efficient and perhaps more secure. The need to advance smart home technology is necessary at this stage as IoT is abundantly used in the automation industry. However, most of the proposed solutions are lacking in certain key areas of the system i.e., high interoperability, data independence, privacy, and optimization in general. The use of machine learning algorithms requires high-end hardware and is usually deployed on servers, where computation is convenient, but at the cost of bandwidth. However, more recently edge AI-enabled systems are being proposed to shift the computation burden from the server side to the client side enabling smart devices. In this paper, we take advantage of edge AI-enabled technology to propose a fully featured cohesive system for smart homes based on IoT and edge computing. The proposed system makes use of industry standards adopted for fog computing as well as providing robust responses from connected IoT sensors in a typical smart home. The proposed system employs edge devices as a computational platform in terms of reducing energy costs and providing security, while remotely controlling all appliances behind a secure gateway. A case study of human fall detection is evaluated by a custom lightweight deep neural network architecture implemented over the edge device of the proposed framework. The case study was validated using the Le2i dataset. During the training, the early stopping threshold was achieved with 98% accuracy for the training set and 94% for the validation set. The model size of the network was 6.4 MB which is significantly lower than other networks with similar performance.

Keywords:- Artificial intelligence, Edge intelligence, IoT, Smart home, Deep learning

**TABLE OF CONTENTS**

**Contents Page No**

**ACKNOWLEDGEMENT**

**ABSTRACT**

**LIST OF FIGURES**

**CHAPTER 1: INTRODUCTION**

**CHAPTER 2: LITERATURE SURVEY**

2.1 Enabling Automation And-Edge Intelligence Over Resource Const 2.2

2.2 Interacting with a Digital Twin using Amazon Alexa

2.3 Amazon Alexa traffic traces

2.4 Iot-Based Electro Synthesis Ecosystem

2.5 Access Control And Surveillance In A Smart Home

2.6 Smart Iot Surveillance Multi-Camera Monitoring System

2.7 Trustbuilder: A Non-Repudiation Scheme For IoT Cloud Applications

2.8 Accuracy Determination Using Deep Learning Technique In the Cloud-

Based Iot Sensor Environment

2.9 Self-Secured Model For Cloud-Based IOT Systems

2.10 Taking MQTT And Nodemcu To IOT: Communication In the Internet Of Things

**CHAPTER 3: MODULES**

3.1 Bed and Ventilation Module

3.2 Basic Needs Module

3.3 Patient Monitor Module

3.4 Medical Aid Module

3.5 security and Entertainment Module

**CHAPTER 1**

**INTRODUCTION**

Over the last few decades, many researchers have focused on connecting everyday objects and weaving a web of interconnected devices. These can be physical devices, automobiles, embedded systems or everyday home appliances. The connectivity of these devices is enabled by the underlying technology known as the Internet of Things (IoT) [1]. These devices can communicate with one another and seamlessly collect and transfer data in between to adapt and reciprocate to dynamic situations. The IoT is gaining tremendous traction from the IT industry as well as individual enthusiasts alike. This is because microcontroller technologies have evolved that enable these complex infrastructures and are more accessible to average developers. The use of single board computers also refers to as System on Chip (SoC) in IoT-based systems, enables rapid interconnection of environmental sensors and decent computational capabilities along with vision sensors that make it easy to develop fairly in large applications. Advancements in machine learning algorithms for mobile and low-power devices also pave the way for intelligent devices in IoT systems. A typical home automation system connects heterogeneous devices, collects data, and decides based on the observed data from these sensors. The means of communication between these devices is typically Wi-Fi or Bluetooth and in some cases a cellular network (3G/4G/5G) [2]. We were focusing on how IoT can drastically improve the lives of disabled people by addressing limited access in many ways. We also spoke about how smart homes and home automation systems are rising in popularity, allowing people to control various aspects of their homes from a single place without getting up. However, for disabled people, this technology holds much more significance. It is not adding additional value but solving some of their biggest hurdles like being unable to access specific items or devices. Smart homes single-handedly allow disabled people to live more independently with an easy-access lifestyle. With the sophistication of IoT technology, it has become a possibility for disabled people to rely lesser on their caregivers

**CHAPTER 2**

**LITERATURE SURVEY**

**2.1 Enabling Automation And-Edge Intelligence Over Resource Const**

Smart home applications are pervasive and have gained popularity due to the overwhelming use of the Internet of Things (IoT). The revolution in IoT technologies made homes more convenient, efficient and perhaps more secure. The need to advance smart home technology is necessary at this stage as IoT is abundantly used in the automation industry. However, most of the proposed solutions are lacking in certain key areas of the system i.e., high interoperability, data independence, privacy, and optimization in general. The use of machine learning algorithms requires high-end hardware and is usually deployed on servers, where computation is convenient, but at the cost of bandwidth. However, more recently edge AI-enabled systems are being proposed to shift the computation burden from the server side to the client side enabling smart devices. In this paper, we take advantage of edge AI-enabled technology to propose a fully featured cohesive system for smart homes based on IoT and edge computing. The proposed system makes use of industry standards adopted for fog computing as well as providing robust responses from connected IoT sensors in a typical smart home. The proposed system employs edge devices as a computational platform in terms of reducing energy costs and providing security, while remotely controlling all appliances behind a secure gateway. A case study of human fall detection is evaluated by a custom lightweight deep neural network architecture implemented over the edge device of the proposed framework. The case study was validated using the Le2i dataset. During the training, the early stopping threshold was achieved with 98% accuracy for the training set and 94% for the validation set. The model size of the network was 6.4 MB which is significantly lower than other networks with similar performance.

**2.2 Interacting with a Digital Twin using Amazon Alexa**

The Digital Twin is an evolving concept with many facets and applications, for instance, engineering simulation, system control, and product-centric information management. This article focuses on the latter where literature uses the Product Avatar concept to refer to a product's digital counterpart. Such an avatar used to have one or more graphical interfaces to support user interactions with information about a product item. Over the last few years, voice user interfaces became more mature, and companies, such as Amazon and Google, used them to create digital assistants that support their users during tasks or by taking them over directly. This paper focuses on the hypothesis that a company could use a voice-enabled digital assistant to interact with item-level information. Our study used product tracking and tracing, and quality control in the production as a realistic application case. The design of the assistant is based on the information needs outlined in the Electronic Product Code Information Services (EPCIS) standard. We implemented this design in a small-scale demonstrator on an Echo Show 5 smart speaker with an integrated touch display and an embedded Amazon Alexa assistant. This paper concludes that significant technological barriers, such as low transcription accuracy for object identifier information and the handling of factory noise, remain. A significant non-technological barrier is the mistrust regarding the closed voice assistant technologies from companies, such as Amazon and Google. An approach to address the latter barrier is to use open technologies, such as the privacy-focused assistant Mycroft or Mozilla's transcription solution DeepSpeech.

**2.3 Amazon Alexa traffic traces**

The number of devices that make up the Internet of Things (IoT) has been increasing every year, including smart speakers such as Amazon Echo devices. These devices have become very popular around the world where users with a smart speaker are estimated to be about 83 million in 2020. However, there has also been great concern about how they can affect the privacy and security of their users [1]. Responding to voice commands requires devices to continuously listen for the corresponding wake word, with the privacy implications that this entails. Additionally, the interactions that users may have with the virtual assistant can reveal private information about the user. In this document, we publicly share two datasets that can help conduct privacy and security studies on the Amazon Echo Dot smart speaker. The included data contains 300.000 raw PCAP traces containing all the communications between the device and Amazon servers from 100 different voice commands in two different languages. The data can be used to train machine learning algorithms to find patterns that can characterize both, the voice commands and people using the device as well as Alexa as the device generating the traffic.

**2.4 Iot-Based Electro Synthesis Ecosystem**

In this paper, we presented the electro-synthesis ecosystem using esp8266 with OTA com- pilation integrated, the MQTT Broker for the control system that would be installed on a Raspberry Pi and MQTT protocol for the wireless data transfer layer which provides the electrosynthesis machine to work 24/7 and with the AI that can be implemented, the whole ecosystem would be smart and the data-driven can be used in QC verification. This system is designed and simulated and the source codes are available via GitHub (https://github.com/iraniothome/ChemIoT).

**2.5 Access Control And Surveillance In A Smart Home**

In the past years, smart home solutions have become more and more popular with the introduction of a high number of both Internet of Things (IoT) applications and smart devices. The home automation and security systems market is in continuous growth, traditional security systems are evolving fast, and more and more people choose Smart home solutions. In this paper, we propose two IoT-based systems in the context of Smart homes: qToggle for multiple home automation, and MotionEyeOS, a video surveillance OS for single-board computers. Most goggle devices are based on ESP8266/ESP8285 chips or Raspberry Pi boards and smart sensors, while MotionEye uses Raspberry Pi boards.

**2.6 Smart Iot Surveillance Multi-Camera Monitoring System**

The surveillance digital monitoring process market has changed aggressively over the last 15 years. Where the process was needed for people to secure their businesses by a simple surveillance camera system to give them that much-needed peace of mind. However, the existing monitoring systems are very expensive due to internal specific SDK configuration which was embedded in the camera itself. Some of them provide a solution such as a high-priced customized command centre that has several screens view which communicate with several cameras to monitor the cameras with a specific detection analysis module. In this paper, a cheaper solution to the monitoring system for existing surveillance cameras is introduced to overcome the solution. This system only implements open-source image processing methods to produce a monitoring system with customizable modules for video analytics (video content analysis) with the input of live video sources. This is to allow a wider range of camera models that can be used for this system. A real-time analysis module will assist in the use of the system to ease the user. The combined video stream and the module will help the user immensely for surveillance propose. Based on the result, the proposed system achieved a higher affordability level of up to 95% with 90% usability compared to existing products.

**2.7 Trustbuilder: A Non-Repudiation Scheme For IoT Cloud Applications**

The IoT cloud computing paradigm is emerging for IoT applications. In such a paradigm, how to guarantee non-repudiation service provisioning has attracted research efforts recently. While existing solutions could work for distributed IoT cloud applications, storage, computation, and economic cost are still a practical challenging concern. In this paper, we propose a new, cost-lower non-repudiation scheme for IoT cloud applications. The proposed scheme guarantees that neither the IoT client nor the cloud could re-repudiate a service enjoyment and provisioning. Specifically, the proposed scheme employs a blockchain to achieve non-repudiation. First, when the cloud provides a service, it encrypts the service, stores a cryptographic hash digest of the encrypted service on the blockchain, and then sends the encrypted service to the IoT client. Second, the IoT client needs to acknowledge the hash digest on the blockchain to ob- in the service. Third, the cloud sends the decryption key to the blockchain under the IoT client’s public key to finish service provisioning. We show that the proposed scheme achieves non-repudiation fairly. We prototyped, evaluated, and open-sourced the proposed scheme. Experimental results confirmed the efficiency of the proposed scheme and the speedup compared with the state-of-the-art solution.

**2.8 Accuracy Determination Using Deep Learning Technique In Cloud-Based Iot Sensor Environment**

devices. However, when connected to wireless connections, unlimited access to IoT gadgets poses potential risks. As it eases cost constraints on sensor nodes, the cloud service with IoT networks has received greater attention. In addition, the high complexity of the distribution and networking of IoT makes them vulnerable to attacks. Intrusion detection systems (IDSs) are selected to ensure the security of reliable information and operations. IDS successfully detects anomalies in complex network situations and guarantees network security. Deep Convolution Network (DCN) IDS have a slow learning curve and poor categorization precision. Deep Learning (DL) methods are often used in a wide range of safety data processing, imaging, and signal processing like Poor transfer learning ability, reusability of modules, and integration. To overcome the constraints of Machine Learning (ML) IDS is intended to provide a comprehensive mechanism to learn the detection mechanism for multi-cloud IoT environments. The proposed IDS approach increases training efficiencies while increasing detection accuracy. Experimental investigations of the proposed system using the considered database confirms that the performance of the proposed system is capable and in the range of acceptance relative to existing methods. Further, achieving detection capability, reliability, and accuracy of 97.51, 96.28, and 94.41% respectively are achieved.

**2.9 Self-Secured Model For Cloud-Based IOT Systems**

A difficult problem to solve concerns the secure installation and startup of devices connected to the Internet of Things (IoT) via the Internet. To provide additional value-added services, this article deals with the verified configuration of IoT devices in a secure manner using the Internet. Following a review of the safe self-configuration limitations imposed on IoT and Cloud technologies; offer a Cloud-based architecture that enables the communication between IoT devices and several federated Cloud services. Specifically discuss two situations, one cloud environment and federated cloud infrastructure interact with IoT devices, and handle unique issues. In addition, it provides many operational design features that take into account the truly open hardware and software products already on the market.

**2.10 Taking MQTT And Nodemcu To IOT: Communication In the Internet Of Things**

The Internet of Things (IoT) allows connection among devices using the Internet with the ability to gather and exchange data. These devices are usually attached with micro-controllers like Arduino, sensors, actuators and internet connectivity. In this context, Message Queuing Telemetry Transport protocol (MQTT) plays an important role to exchange the data or information between the devices in IoT without knowing the identities of each other. This paper presents different service models for communication in the Internet of Things(IoT). Model A presents the use of serial USB as a transmission medium while Model B uses the Message Queuing Telemetry Transport protocol (MQTT) which deploys a Wi-Fi module (ESP8266-12) to connect the system to the internet. For communication, the concept of publisher and subscriber is used. Messages are published or subscribed with the help of a broker or server. This agent is in charge of dispersing messages to intent clients depending on the choice of the topic of a message. The broker in MQTT is also called the server. Some brokers used in MQTT are: -Mosquitto, Adafruit, hiveMQ

**CHAPTER 3**

**MODULES AND REQUIREMENTS**

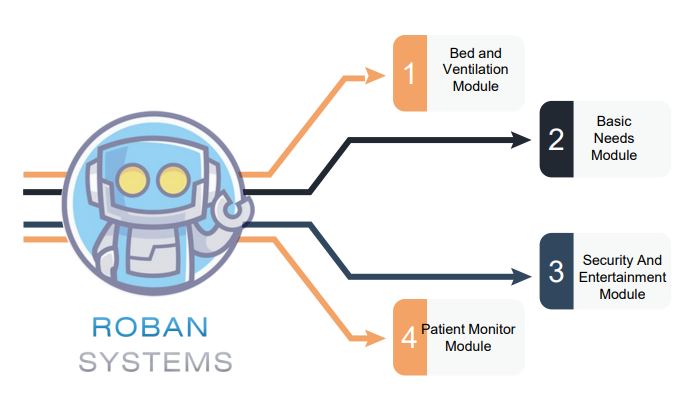
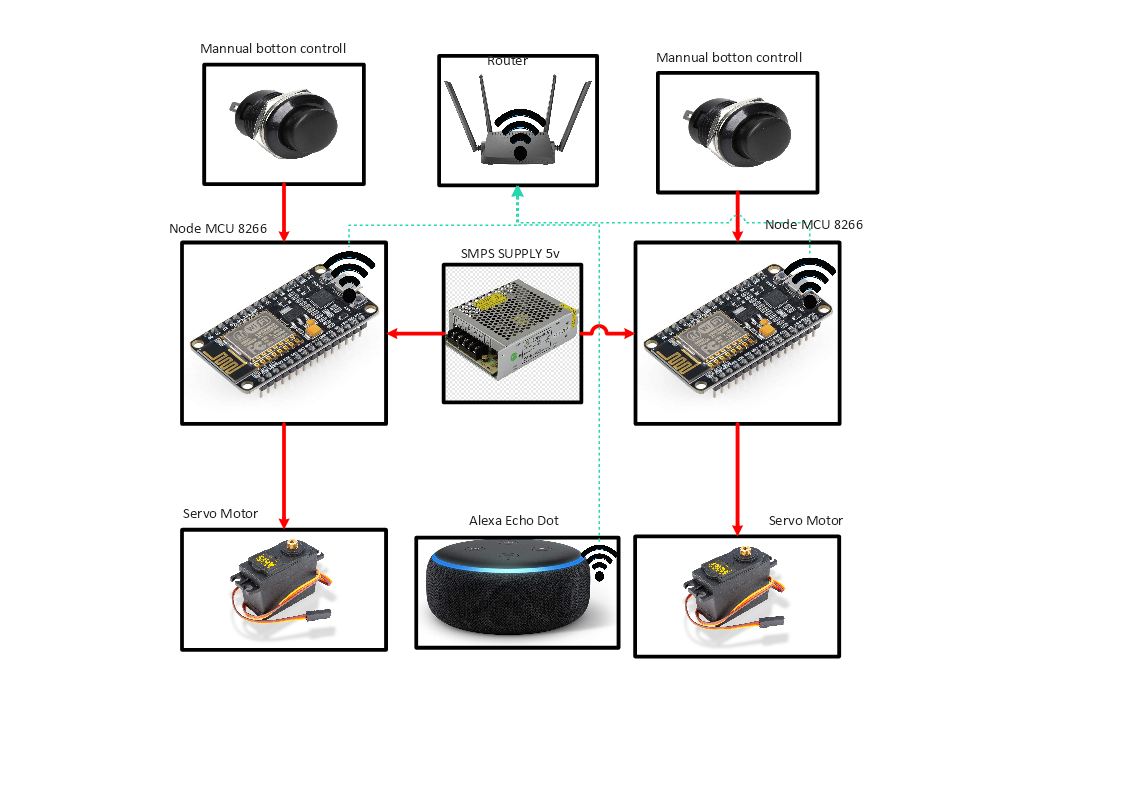


Fig 3 Modules of Roban System

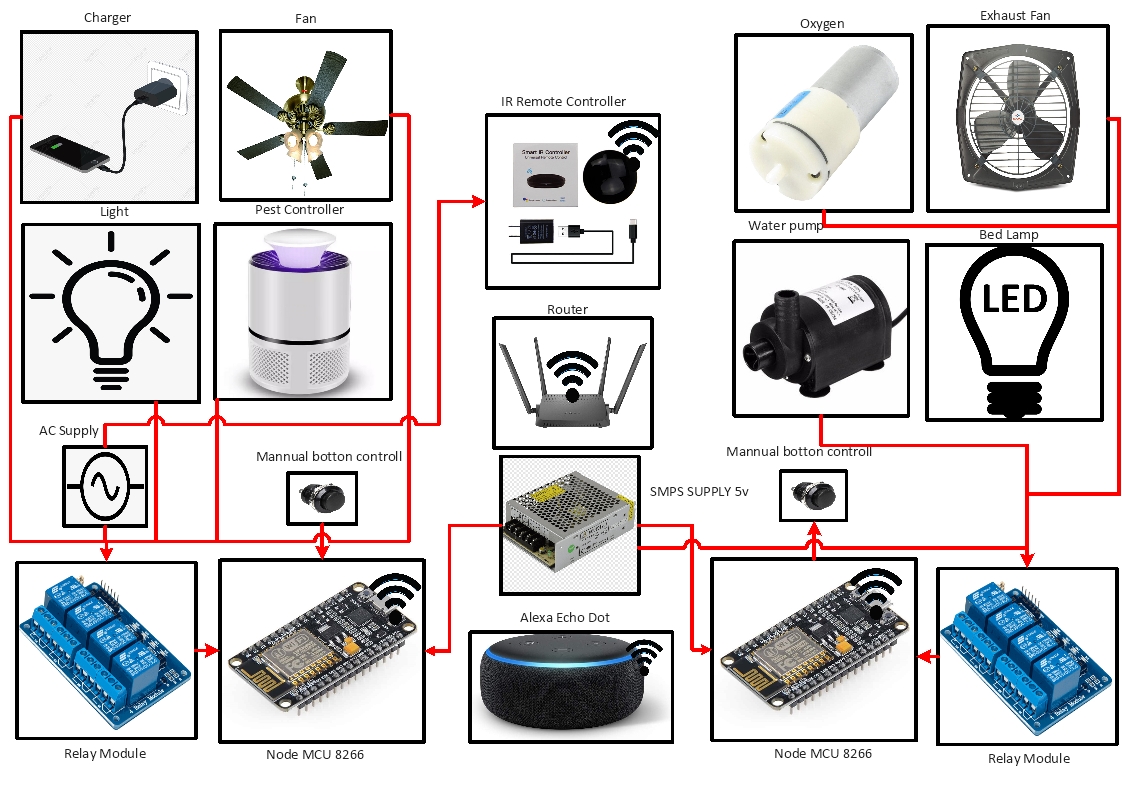
#### 3.1 Bed and Ventilation Module

#### The bed and ventilation module is used to adjust the bed position and adjust the ventilation by sliding the curtains. Disabled people can adjust the bed positions by themselves through voice commands, so they can acquire their comfortable positions. By controlling the ventilation system by voice command they can adjust the natural lighting and ventilation.



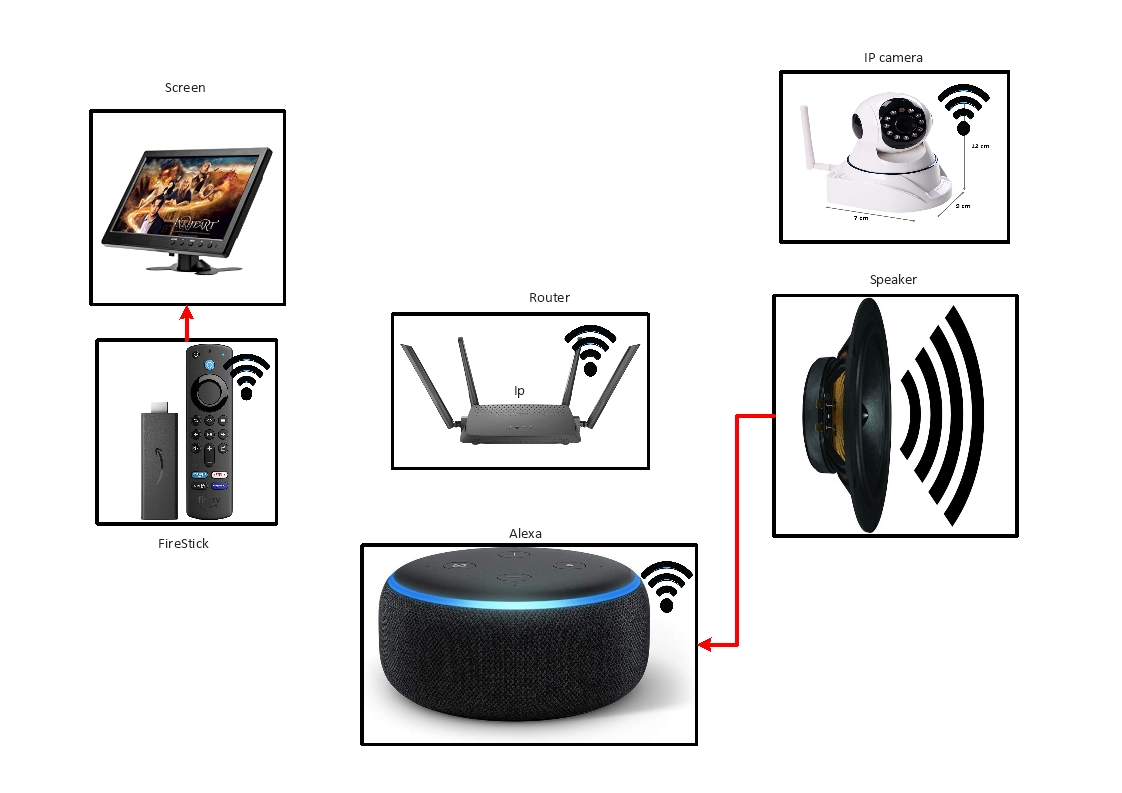
#### 3.2 Basic Needs Module

#### The basic need module consists of an Artificial ventilation system like fans, or exhaust fans, a lighting system like AC and DC lights, warm lights and a pest control system. Disabled people can adjust the bed positions by themselves through voice commands so they can full fill their needs using the IoT



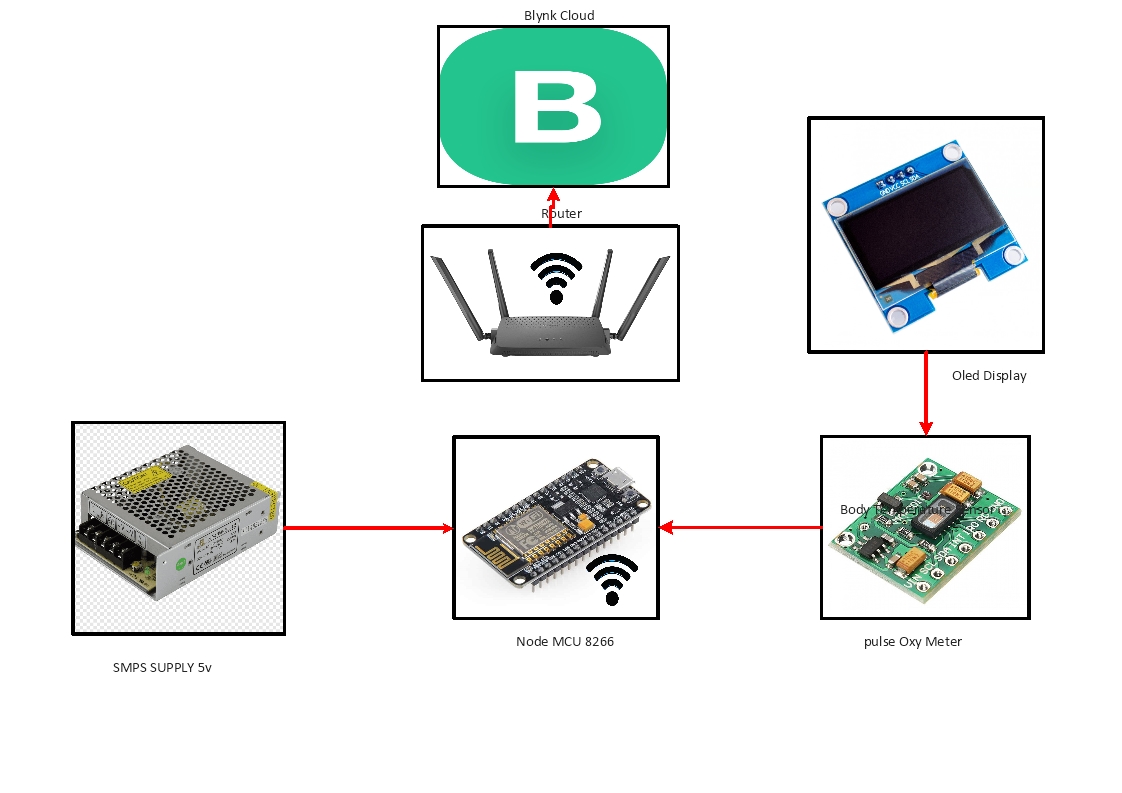
#### 3.3 security and Entertainment Module

Disabled people are seen in a closed environment so there is a chance of mental depression. So we connect entertainment like TV, social media, and streaming platforms in this module Disabled people can use them through voice commands. And security is also essential for them so we connected a security camera we can use this camera to monitor the patient’s external and internal environment.



#### 3.4 Patient Monitor Module

#### This module consists of several sensors that are connected over IoT, and by that sensor, we can monitor the patient’s health and make health plans and it can also alert the authorities when there are any emergencies happen. We can implement patient monitoring like temp, oxygen, pulse, etc. and monitor the environment of the patient.



#### 3.6 Requirements

**3.6.1 system configuration**

The system must be full fill the configuration of 4GB RAM and a minimum operating system of Windows 7 or more.

**Arduino – Installation**

In this section, we will learn in easy steps, how to set up the Arduino IDE on our computer and prepare the board to receive the program via USB cable.

Step 1 − First you must have your Arduino board (you can choose your favorite board) and a USB cable. In case you use Arduino UNO, Arduino Duemilanove, Nano, Arduino Mega 2560, or Diecimila, you will need a standard USB cable (A plug to B plug), the kind you would connect to a USB printer as shown in the following image.

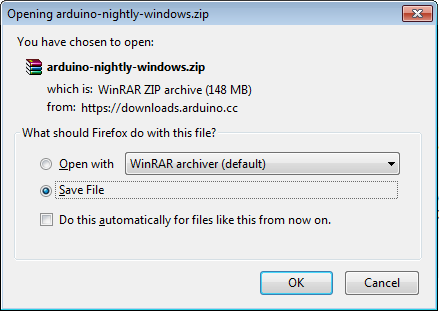


In case you use Arduino Nano, you will need an A to Mini-B cable instead as shown in the following image.



Step 2 − Download Arduino IDE Software.

You can get different versions of Arduino IDE from the [Download page](https://www.arduino.cc/en/Main/Software) on the Arduino Official website. You must select your software, which is compatible with your operating system (Windows, IOS, or Linux). After your file download is complete, unzip the file.



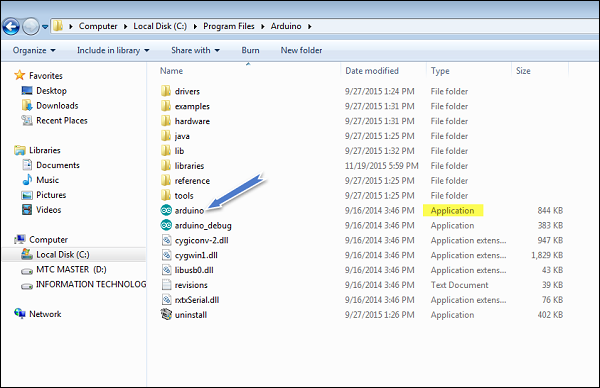
Step 3 − Power up your board.

The Arduino Uno, Mega, Duemilanove and Arduino Nano automatically draw power from either, the USB connection to the computer or an external power supply. If you are using an Arduino Diecimila, you have to make sure that the board is configured to draw power from the USB connection. The power source is selected with a jumper, a small piece of plastic that fits onto two of the three pins between the USB and power jacks. Check that it is on the two pins closest to the USB port.

Connect the Arduino board to your computer using the USB cable. The green power LED (labeled PWR) should glow.

Step 4 − Launch Arduino IDE.

After your Arduino IDE software is downloaded, you need to unzip the folder. Inside the folder, you can find the application icon with an infinity label (application.exe). Double-click the icon to start the IDE.

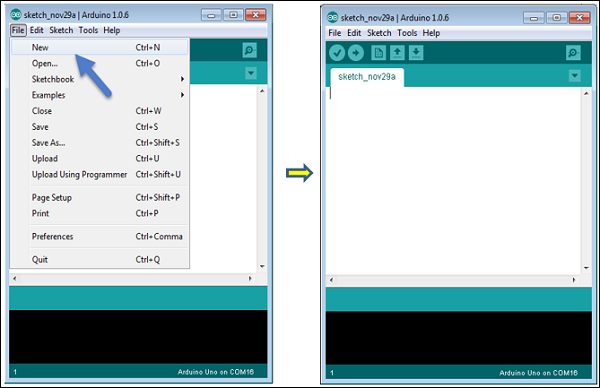


Step 5 − Open your first project.

Once the software starts, you have two options −

* Create a new project.
* Open an existing project example.

To create a new project, select File → New.



To open an existing project example, select File → Example → Basics → Blink.

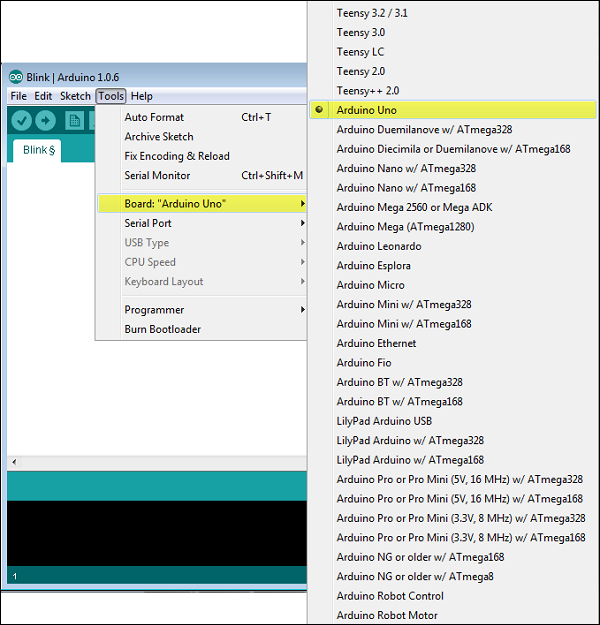
Open Project

Here, we are selecting just one of the examples with the name Blink. It turns the LED on and off with some time delay. You can select any other example from the list.

Step 6 − Select your Arduino board.

To avoid any error while uploading your program to the board, you must select the correct Arduino board name, which matches with the board connected to your computer.

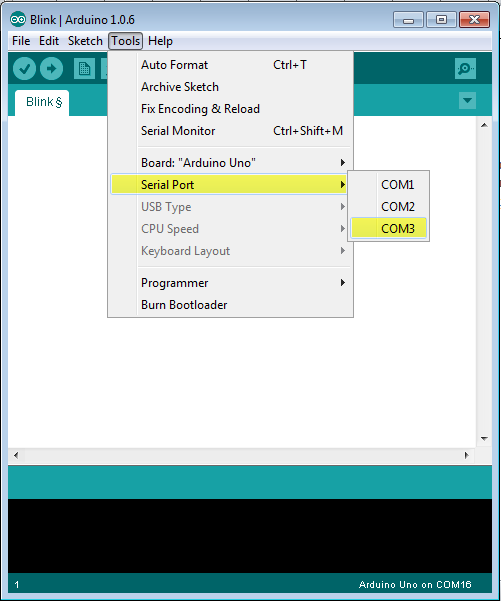
Go to Tools → Board and select your board.



Here, we have selected Arduino Uno board according to our tutorial, but you must select the name matching the board that you are using.

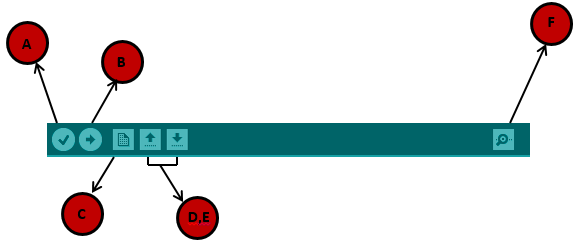
Step 7 − Select your serial port.

Select the serial device of the Arduino board. Go to Tools → Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your Arduino board and re-open the menu, the entry that disappears should be of the Arduino board. Reconnect the board and select that serial port.



Step 8 − Upload the program to your board.

Before explaining how we can upload our program to the board, we must demonstrate the function of each symbol appearing in the Arduino IDE toolbar.



A − Used to check if there is any compilation error.

B − Used to upload a program to the Arduino board.

C − Shortcut used to create a new sketch.

D − Used to directly open one of the example sketch.

E − Used to save your sketch.

F − Serial monitor used to receive serial data from the board and send the serial data to the board.

Now, simply click the "Upload" button in the environment. Wait a few seconds; you will see the RX and TX LEDs on the board, flashing. If the upload is successful, the message "Done uploading" will appear in the status bar.

Note − If you have an Arduino Mini, NG, or other board, you need to press the reset button physically on the board, immediately before clicking the upload button on the Arduino Software.

**3.6.2 C++ language**

As Arduino. began developing new MCU boards based on non-AVR processors like the ARM/SAM MCU used in the Arduino Due, they needed to modify the Arduino IDE so it would be relatively easy to change the IDE to support alternate toolchains to allow Arduino C/C++ to be compiled for these new processors. They did this with the introduction of the Board Manager and the SAM Core. A "core" is the collection of software components required by the Board Manager and the Arduino IDE to compile an Arduino C/C++ source file for the target MCU's machine language**.**

**Program Structure**

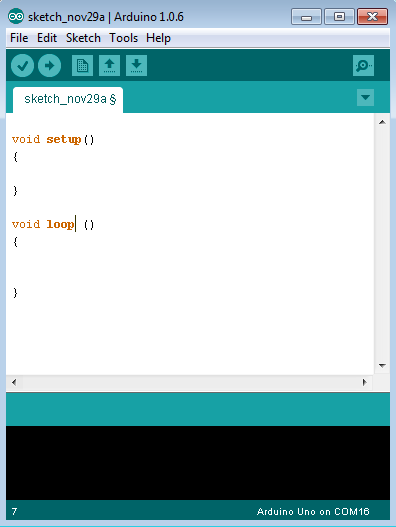
The Arduino software is open-source. The source code for the Java environment is released under the GPL and the C/C++ microcontroller libraries are under the LGPL.

Sketch − The first new terminology is the Arduino program called “sketch”.

Structure-Arduino programs can be divided in three main parts: Structure, Values (variables and constants), and Functions. In this tutorial, we will learn about the Arduino software program, step by step, and how we can write the program without any syntax or compilation error.

Let us start with the Structure. Software structure consist of two main functions −

* Setup( ) function
* Loop( ) function



Void setup ( ) {

}

* PURPOSE − The setup() function is called when a sketch starts. Use it to initialize the variables, pin modes, start using libraries, etc. The setup function will only run once, after each power up or reset of the Arduino board.
* INPUT − -
* OUTPUT − -
* RETURN − -

Void Loop ( ) {

}

* PURPOSE − After creating a setup() function, which initializes and sets the initial values, the loop() function does precisely what its name suggests, and loops consecutively, allowing your program to change and respond. Use it to actively control the Arduino board.
* INPUT − -
* OUTPUT − -
* RETURN − -

**3.6.3** **Alexa**

Alexa Voice Service (AVS) Integration is a new feature of AWS IoT Core that enables device makers to make any connected device an Alexa Built-in device. AVS for AWS IoT reduces both the cost and complexity of producing Alexa Built-in devices by offloading compute and memory-intensive tasks from physical devices to the cloud. With the reduction in the engineering bill of materials (BOM) cost, device makers can now cost-effectively build new categories of differentiated voice-enabled products such as light switches, thermostats, small appliances and more. This allows end-consumers to talk directly to Alexa in new parts of their home, office, or hotel rooms for a truly ambient experience.

**Alexa Voice Service (AVS) Integration for AWS IoT Core**

The Alexa Voice Service (AVS) Integration is a new feature of AWS IoT Core that enables device makers to make any connected device an Alexa Built-in device. AVS for AWS IoT reduces both the cost and complexity of producing Alexa Built-in devices by offloading compute and memory-intensive tasks from physical devices to the cloud. With the reduction in the engineering bill of materials (eBoM) cost, device makers can now cost-effectively build new categories of differentiated voice-enabled products such as light switches, thermostats, small appliances and more. This allows end-consumers to talk directly to Alexa in new parts of their home, office, or hotel rooms for a truly ambient experience.

**Combining Amazon Alexa with AWS IoT**

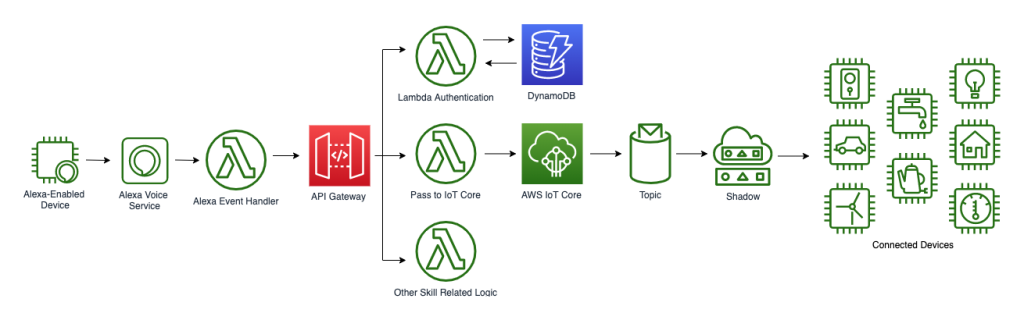
When creating a smart building environment, utilizing both Amazon Alexa and AWS IoT enables builders to create an end-to-end voice-controlled building experience. Unlike traditional architectures, Amazon Alexa and AWS IoT allow for cost-optimized serverless workloads. This means that you don’t have to provision long-running servers. Instead, your functions run only when users invoke Alexa with voice commands.

**Prerequisites**

* Alexa Skills Kit account
* Amazon Web Services account
* Alexa device (Amazon Echo, Alexa mobile application, third-party hardware with Alexa, and others)

**Architecture**

This post walks you through the details of building this architecture on AWS.



**Deploy architecture in AWS account**

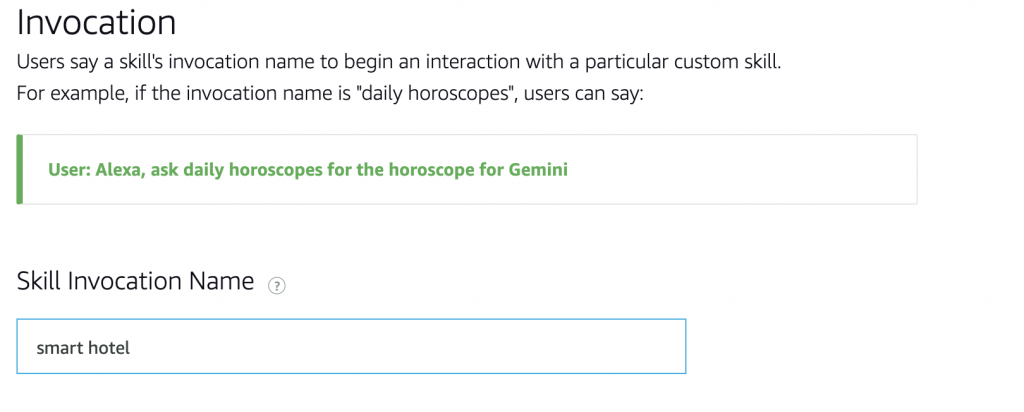
1. [Download](https://github.com/aws-samples/smart-home-iot-alexa-workshop/blob/master/code/SmartHotelBackend.zip) the HotelBackEnd Lambda function zip file.
2. Upload it to an S3 bucket in the same region you’re working in. Note the name of your S3 bucket.
3. Download and deploy this AWS CloudFormation template
   * Click “Raw” button and copy/paste into text editor. Save as IoTBlog.yaml.
4. During CloudFormation deployment, enter the S3 bucket name and file name (not including .zip) as parameters.

**Alexa skill creation**

This post doesn’t focus on the development of an Alexa skill. However, we cover some basics. For a more detailed introduction to developing Alexa skills, see the following resources:

* Alexa Skills Kit
* Resources for Alexa Smart Home Developers

First, navigate to the Alexa Console Create an invocation name that your hotel customers and staff will use to invoke the skill. I’ve set the invocation name of my skill to smart hotel. The invocation name should be two or more words and follow the Alexa Skills Kit guidelines



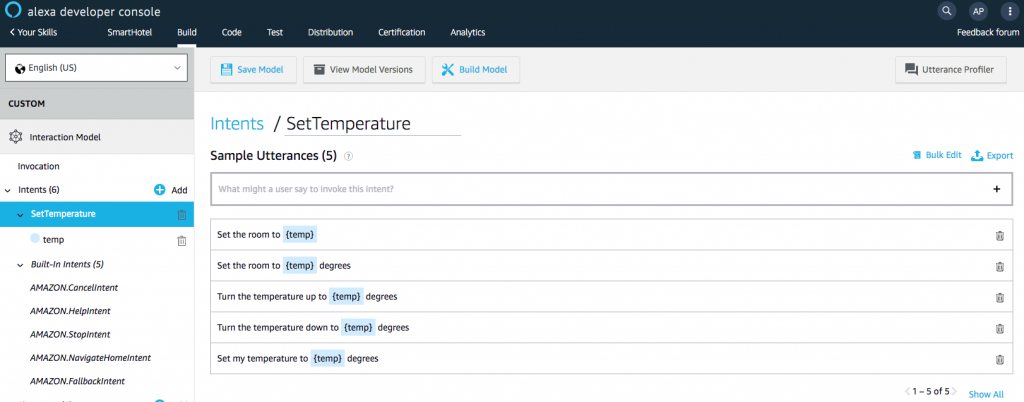
Next, we add our first intent. An *intent* represents an action that fulfills a user’s spoken request. Intents are specified in a JSON structure called the *intent schema*. Each intent is mapped to several utterances. The *sample utterances* specify the words and phrases users can say to invoke your intents.

We create an intent called *SetTemperature* with many utterances that a user can say to map to this intent. Intents can optionally have arguments called *slots*. Slots allow for variables to be sent within the intent.

We set the slot by double clicking *temp* and selecting *Add* to create new slot. Scroll down and set the slot type to *AMAZON.NUMBER*. Verify all of the *temp*slots are highlighted as seen below. The intent is then passed to the backend for processing, which we cover shortly. See the following example:

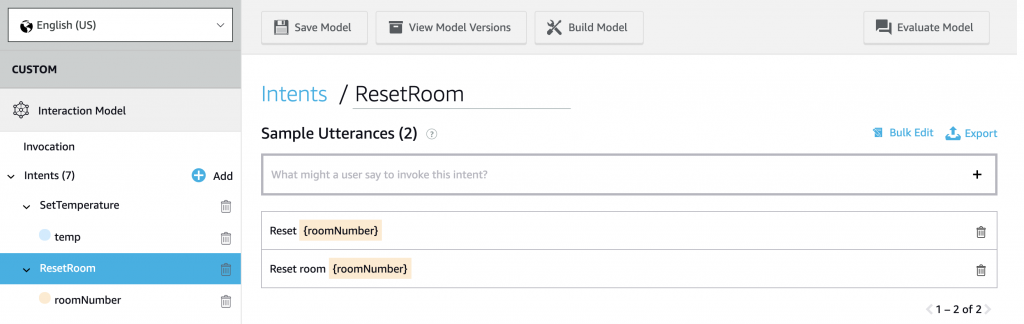
 User: Alexa, ask *Smart Hotel* to set my temperature to 72 degrees.

* Invocation Name: Smart Hotel
* Intent: SetTemperature
* Slot: 72

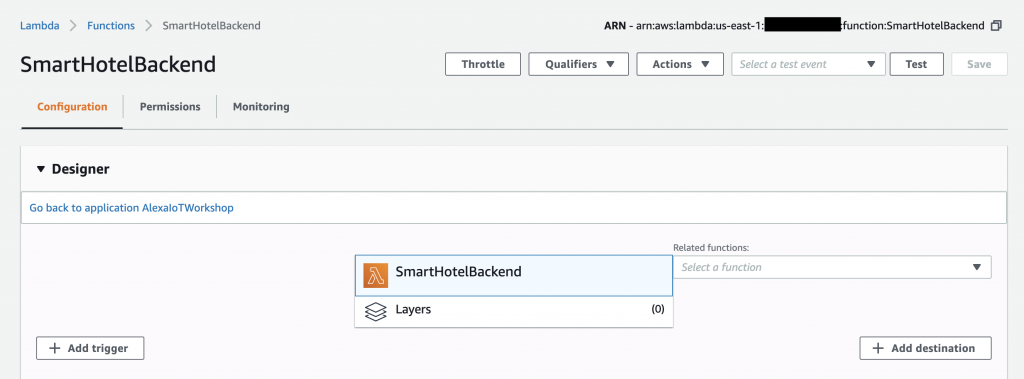


The second intent is for the use case where a hotel employee wants to reset all the smart devices in a room to prepare for a new customer. Follow the same steps as before for the *ResetRoom* intent creation.

* Intent: ResetRoom
* Slot: roomNumber



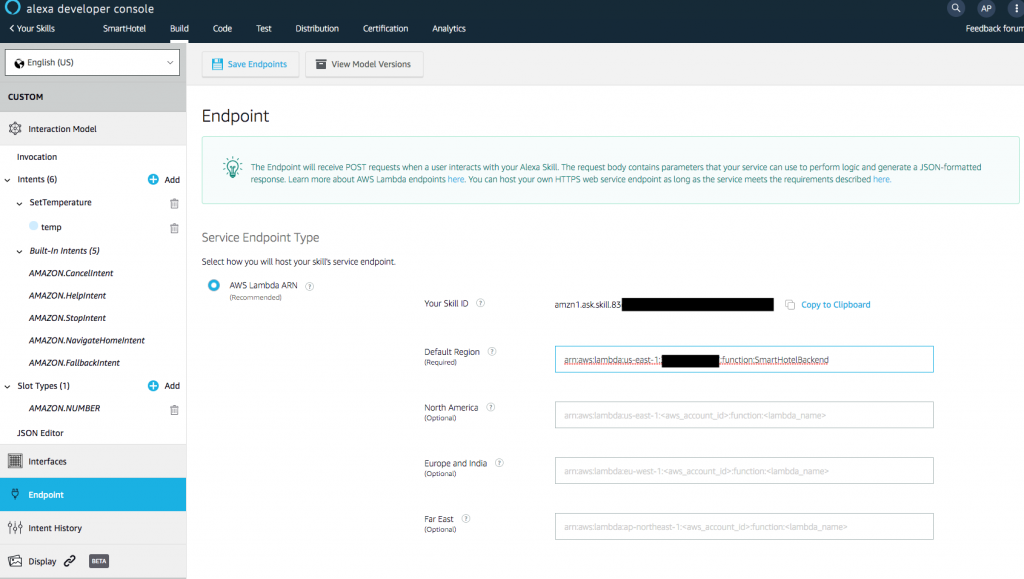
We stop the Alexa Skill building here, but keep in mind that the skill would typically have many intents (SetLights, SetShades, StartVacuum, OrderRoomService, and more) with each intent holding a robust set of utterances. When done, choose Save, and then choose Build Model.

Now that you have built your Alexa skill, you must link it with a Lambda function in order to pass the intent schema to be processed. In our case, we will be using Lambda and API Gateway to authenticate the user and send the command to IoT Core, which will handle the interaction with the hardware. To link the skill with Lambda, go to your AWS account and navigate to the *SmartHotelBackend*Lambda function. 

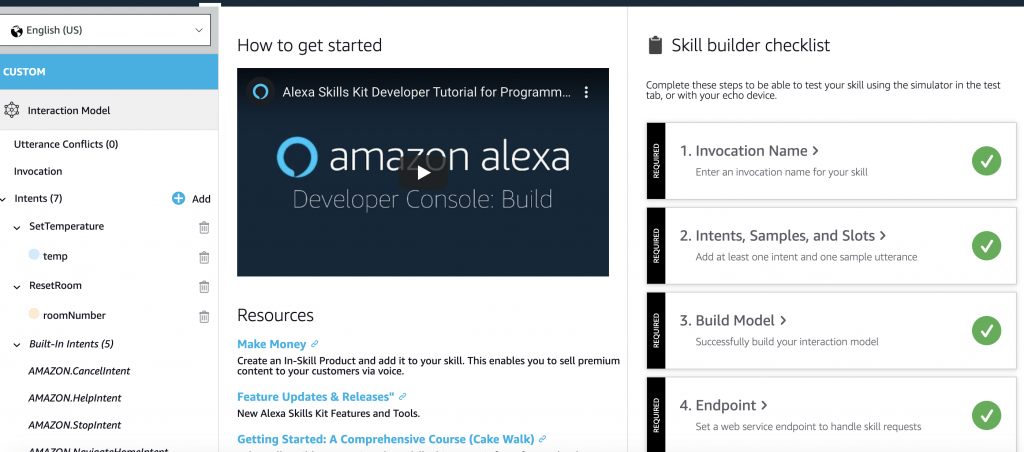
Copy the ARN of the Lambda function.

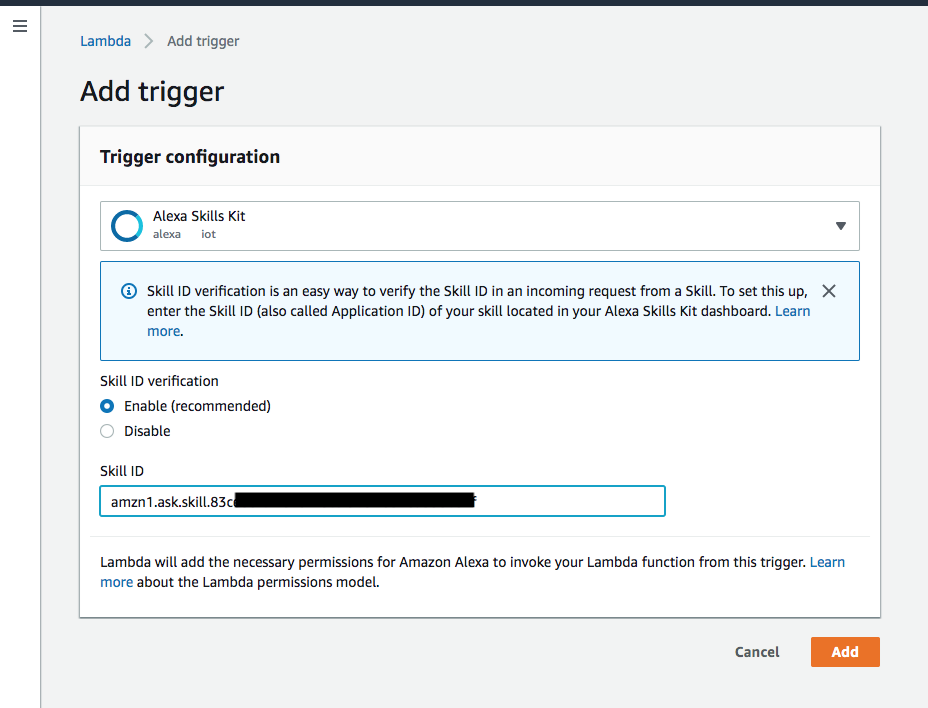
The ARN of the Lambda function being copied

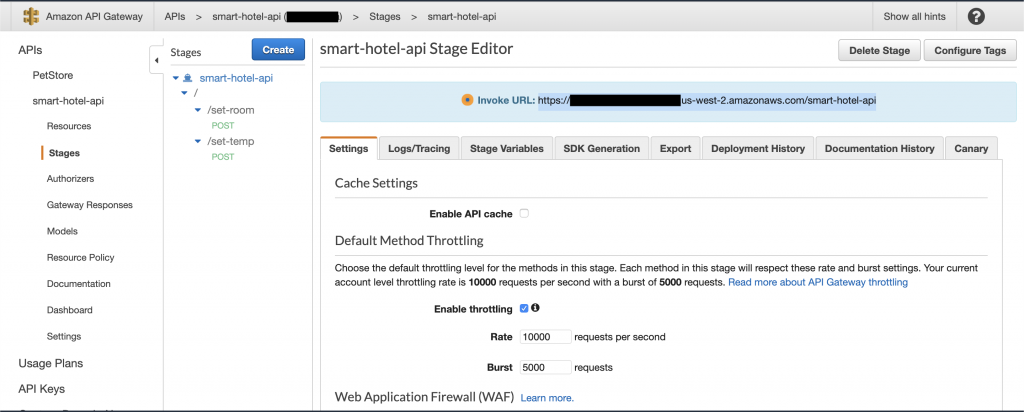
Paste it in the Endpoint tab of the Alexa Developer Console. Make note of Your Skill ID. Choose Save Endpoints.

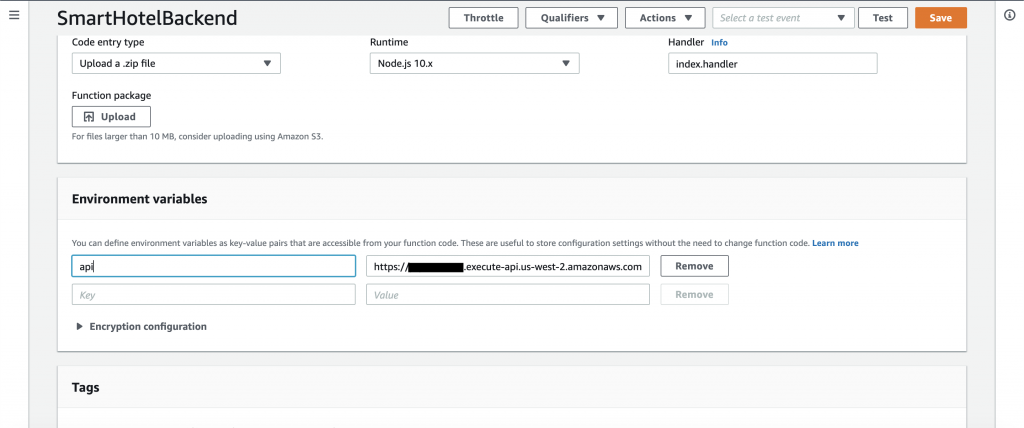


Choose the Build tab, and then choose Build Model.

  
Now copy the Skill ID and paste it into the Add trigger menu within your Lambda function.

  
Navigate to Amazon API Gateway, and copy your API Invoke URL from Stages > smart-hotel-api.

  
Paste the API Invoke URL into the environment variable section of your *SmartHotelBackend*Lambda function with api as the key.



**Lambda and API Gateway**

The Alexa Event Handler Lambda function receives the JSON payload from the Alexa skill and sends it to API Gateway for authentication and routing to the relevant Lambda function. Then, API Gateway authorizes the request using a [Lambda authorizer](https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-use-lambda-authorizer.html). This function verifies the access token from the request and optionally checks it against an internal Amazon DynamoDB database. For example, the authorizer might look up the hotel customer’s credentials to determine whether the user is authorized to receive the requested content.

After the request has been authorized, API Gateway sends the request on to the specified API URI, which invokes the Lambda function associated with that specified logic. For this example, we send the request on to the Lambda function that triggers AWS IoT Core functionality.

**Lambda authorizer template**

import json

#Format return policy that API Gateway expects

def generatePolicy(principalId, effect, resource):

return {

'principalId': principalId,

'policyDocument': {

'Version': '2012-10-17',

'Statement': [{

'Action': 'execute-api:Invoke',

'Effect': effect,

'Resource': resource

}]

}

};

def customLogicFunction(token):

#Run your custom authorization here

#i.e. Check your DynamoDB table for token associated with user

#Return true or false

def lambda\_handler(event, context):

#if(customLogicFunction(event['authorizationToken']) == true)

return generatePolicy('user', 'Allow', event['methodArn'])

#else

#return generatePolicy('user', 'Deny', event['methodArn'])

Python

This is the Lambda authorizer that does your custom authorization logic. Notice the format of response API Gateway is expecting. API Gateway passes the source token to this Lambda authorizer function in the event.authorizationToken attribute. The Lambda authorizer function reads the token and acts as follows:  
If the token value is Allow, the authorizer function returns a 200 OK HTTP response and an IAM policy that looks like the following, and the method request succeeds:

{

"Version": "2012-10-17",

"Statement": [

{

"Action": "execute-api:Invoke",

"Effect": "Allow",

"Resource": "arn:aws:execute-api:us-east-1:xxxxxxxxxxxx:m88ssxznb7/ESTestInvoke-stage/GET/"

}

]

}

JSON

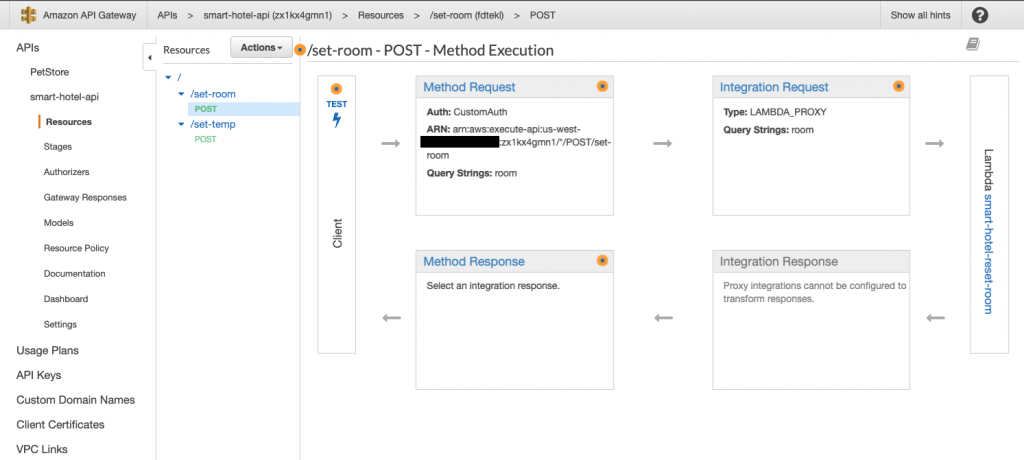
Finally, be sure to set the Lambda function trigger to API Gateway, associate a role based on least privilege, and set the timeout time to 10 seconds.

**API Gateway**

The CloudFormation template creates two URIs for our API. One is for authorized hotel staff to reset the room (“Alexa, tell smart hotel skill to reset room 211”), which will use the authorizer function. The second is for hotel guests to set the temperature (“Alexa, tell smart hotel skill to set room temperature to 72 degrees”), which will not need authorization. Each URI will have a method with a Lambda function that is called.

The first lambda function is for the room reset logic. We called this “smart-hotel-reset-room”. This function sends messages to the IoT Core topic associated with the specific room. Each device within the room is subscribed to it. Examples of this are turning off the TV, resetting room temperature, turning lights off, and closing the blinds.

Notice the method request requires authorization using our custom authorizer function. This is to ensure only authorized hotel employees can reset the room.

  
Here is the code that backs the /set-room resource:

import json

import boto3

import datetime

timestamp = datetime.datetime.utcnow().strftime('%Y-%m-%dT%H:%M:%S.%f')

client = boto3.client('iot-data')

def lambda\_handler(event, context):

roomId = event['multiValueQueryStringParameters']['room'][0]

# TODO implement

temp = 72

response = client.publish(

topic = 'resetRoom',

qos = 0,

payload = json.dumps({"roomid": roomId, "timestamp": timestamp, "shades": "up", "theater": "stopped", "thermostat": temp})

)

return {

"statusCode": 200,

}

Python

The second Lambda function is for customers to set the temperature. We called it “smart-hotel-set-temp”, and the code simply writes to the temperature topic for the specific room. The thermostat is subscribed to this topic. The roomId and temperature are passed in from the API gateway request. Note that we are not utilizing [Device Shadow](https://docs.aws.amazon.com/iot/latest/developerguide/iot-device-shadows.html) here. Part 2 of this post series will cover the hardware side of this solution and will utilize Device Shadow for devices with sporadic internet connection.

import json

import boto3

import datetime

timestamp = datetime.datetime.utcnow().strftime('%Y-%m-%dT%H:%M:%S.%f')

client = boto3.client('iot-data')

def lambda\_handler(event, context):

temp = event['multiValueQueryStringParameters']['temp'][0]

print(event)

roomId = 101

response = client.publish(

topic = 'setTemp',

qos = 0,

payload = json.dumps({"roomid": roomId,"timestamp": timestamp, "thermostat": temp })

)

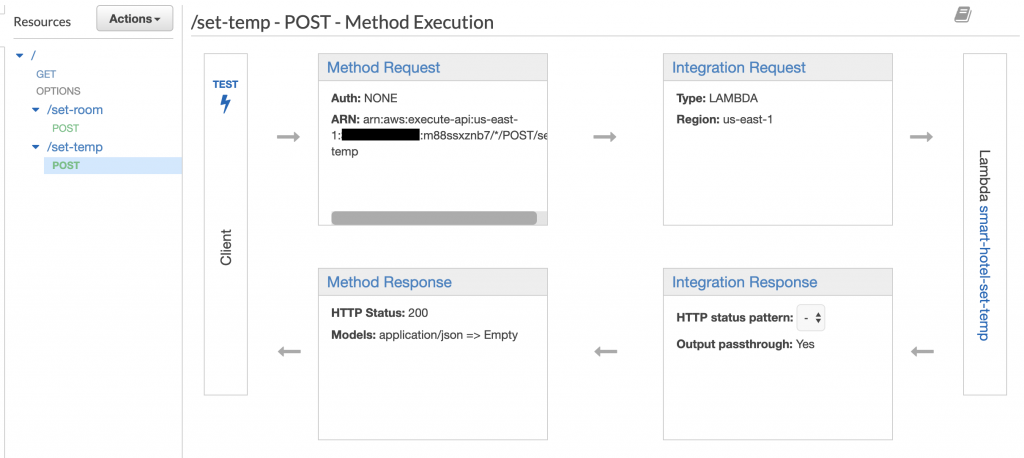
return {

"statusCode": 200,

}

Python

We do not authorize calls to this URI because the customers do not need an account to change their temperature.



**Alexa skill event handler**

This function is named *Alexa Event Handler* in the architecture diagram above. This function is called whenever a user invokes the Smart Hotel Alexa Skill. It routes the command to the appropriate API method (either /set-temp or /set-room). As your skill becomes more complex, you add more resources or methods to your API to accommodate additional functionality.

Creating an event handler for an Alexa skill can be the subject of a post in itself. However, we supply the following sample code that routes to either method based on specific intents. We created a SetTemp and ResetRoom intent in Alexa Skills Kit.

'use strict';

const Alexa = require('alexa-sdk');

const req = require('request');

const APP\_ID = undefined;

const SKILL\_NAME = 'SmartHotel';

const GET\_FACT\_MESSAGE = "";

const HELP\_MESSAGE = 'Please repeat.';

const HELP\_REPROMPT = 'What can I help you with?';

const STOP\_MESSAGE = 'Goodbye!';

var AWS = require('aws-sdk');

const apiLink = process.env.api;

const handlers = {

'LaunchRequest': function () {

this.emit('SetTemperature');

},

'SetTemperature': function () {

var temp = this.event.request.intent.slots.temp.value;

var url = apiLink + '/set-temp?temp=' + temp

req.post(url, function(err, res, body) {

if(err){

console.log('error', err);

} else{

console.log('success', body);

}

});

const speechOutput = 'Sounds good! I\'ve set your temperature to ' + temp;

this.response.cardRenderer(SKILL\_NAME, speechOutput);

this.response.speak(speechOutput);

this.emit(':responseReady');

},

'ResetRoom': function () {

var room = this.event.request.intent.slots.roomNumber.value;

var postBody = {url: apiLink + '/set-room?room=' + room,

headers: {Authorization: this.event.context.System.apiAccessToken}}

req.post(postBody, function(err, res, body) {

if(err){

console.log('error', err);

} else{

console.log('success', body);

}

});

const speechOutput = 'Okay! Resetting room ' + room + ' now.';

this.response.cardRenderer(SKILL\_NAME, speechOutput);

this.response.speak(speechOutput);

this.emit(':responseReady');

},

'AMAZON.HelpIntent': function () {

const speechOutput = HELP\_MESSAGE;

const reprompt = HELP\_REPROMPT;

this.response.speak(speechOutput).listen(reprompt);

this.emit(':responseReady');

},

'AMAZON.CancelIntent': function () {

this.response.speak(STOP\_MESSAGE);

this.emit(':responseReady');

},

'AMAZON.StopIntent': function () {

this.response.speak(STOP\_MESSAGE);

this.emit(':responseReady');

},

};

exports.handler = function (event, context, callback) {

const alexa = Alexa.handler(event, context, callback);

alexa.appId = APP\_ID;

alexa.registerHandlers(handlers);

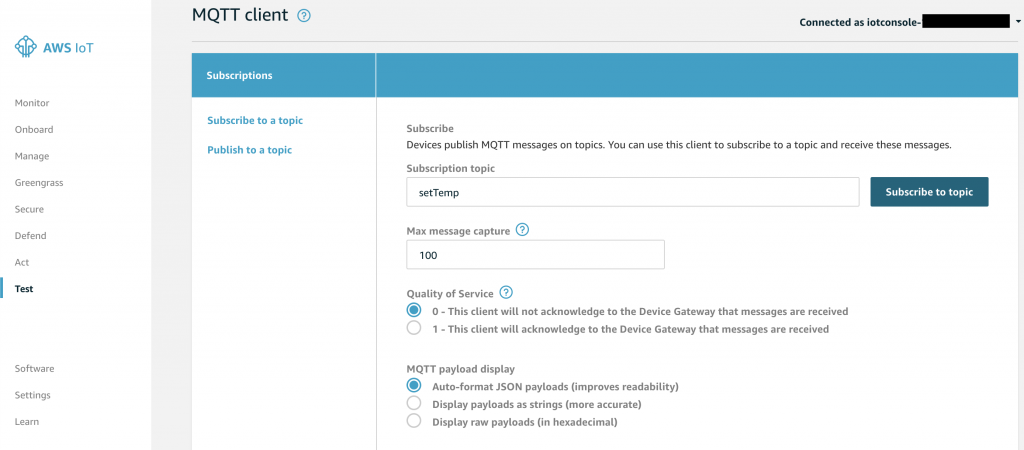
alexa.execute();

};

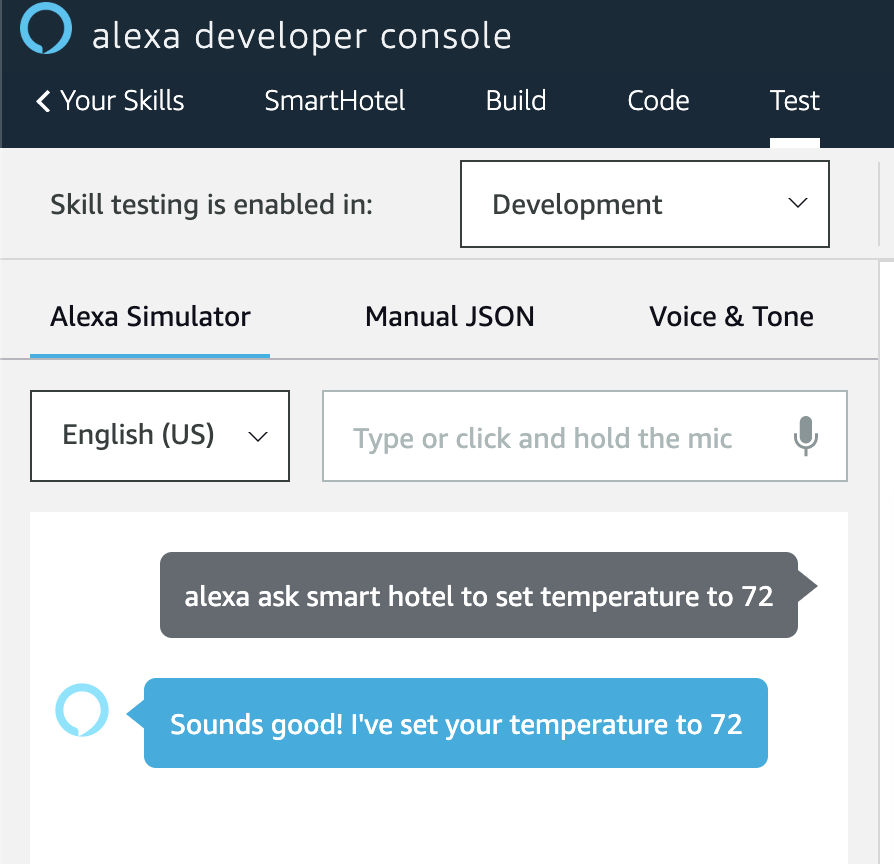
Python

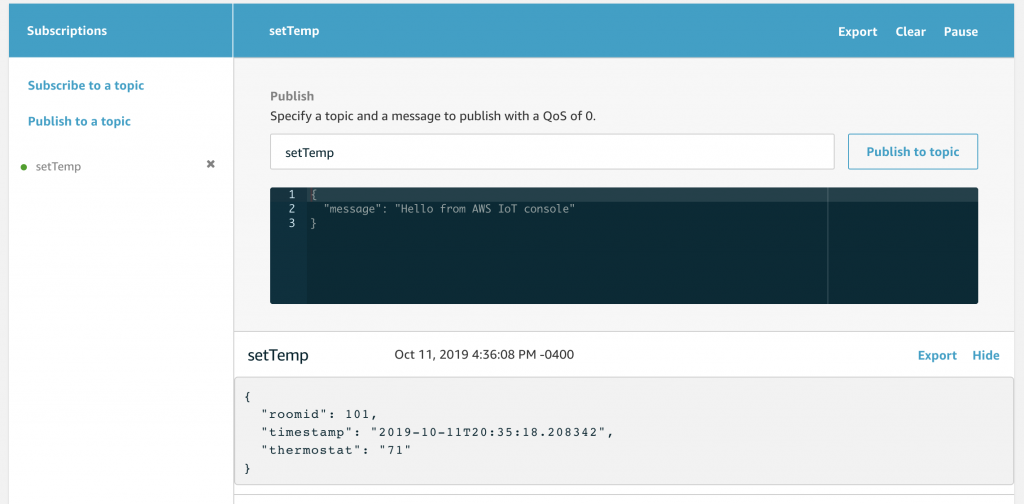
**Testing**

Now that we’ve set up our Alexa to AWS IoT architecture, let us see the topics where the messages are going. Go to AWS IoT Core, and choose the Test tab. Subscribe to the *setTemp* topic.



In a new tab, go to Alexa Skills Kit and choose the Test tab. Either speak through the microphone or type in the box to interact with your skill.

  
Navigate back to the AWS IoT Core test console to see the message.



You can do the same for resetRoom topic.

This shows you the topic’s incoming messages and simulates what the hardware device will subscribe to. When the device receives the setTemp message, it will grab the temperature value and change the temperature value of the device. The resetRoom topic will be subscribed to by multiple devices within the hotel room such as the shades, the thermostat, and the television or smart plug

**3.6.4** **Blynk platform**

Everything you need to build and manage connected hardware: device provisioning, sensor data visualization, remote control with mobile and web applications, Over-The-Air firmware updates, secure cloud, data analytics, user and access management, alerts, automation and much more.Blynk platform powers low-batch manufacturers of smart home products, complex HVAC systems, agricultural equipment, and everyone in between. These companies build branded apps with no code and get the full back-end IoT infrastructure through one web app.

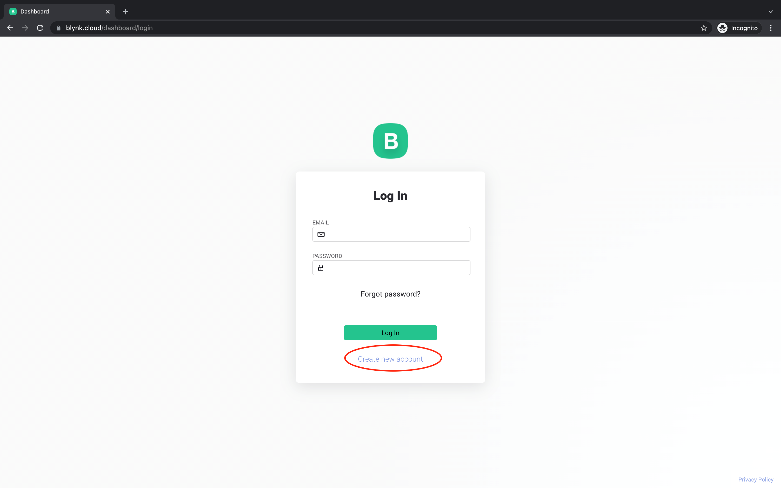
**connect Blynk IoT mobile app**

Simply connect your TOWER kit to your mobile phone. Once paired, you can send notifications or view graphs of the temperature in your home. You need the Blynk IoT app to do this and in this article we'll show you how.

**Start by registering on Blynk.cloud**

The first step is to create an account on the Blynk app. You need a new account even if you have used a previous version of the app.

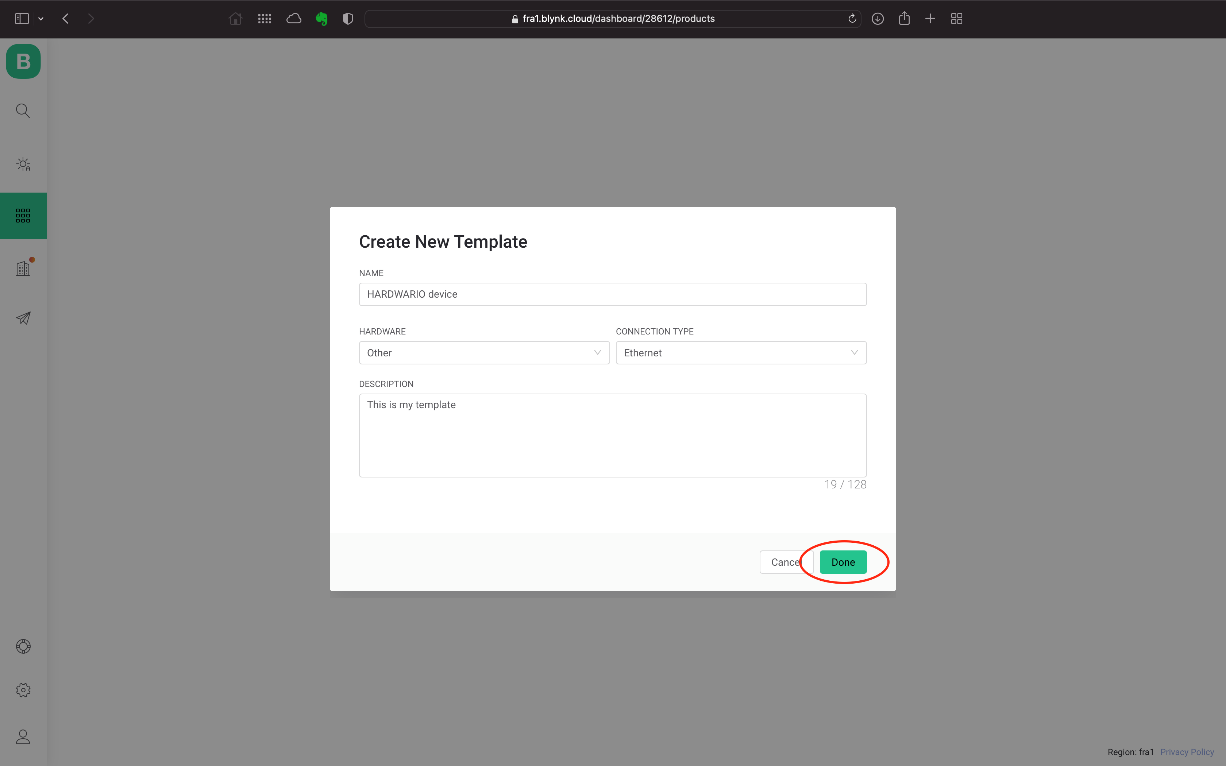
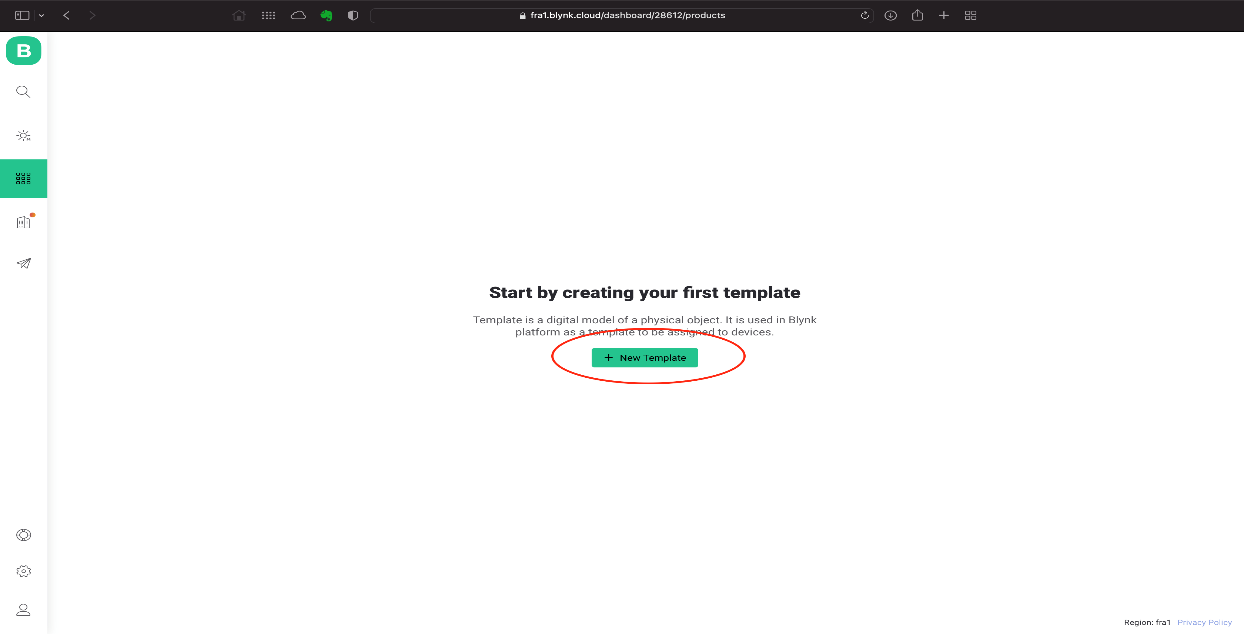
Open [blynk.cloud](https://blynk.cloud/) in your browser and click **Create new account**. Then fill in your email and a message will be sent to you with a link to create a password.



**Create a template - template**

You can skip the introductory tutorial and save it for later.

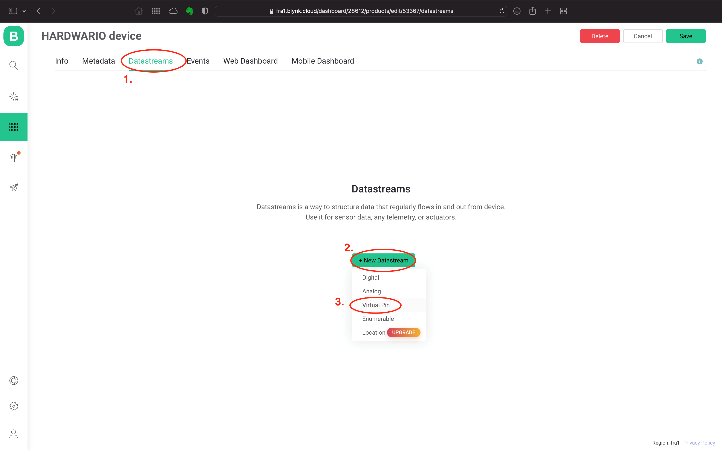
Select **Templates** in the left menu and click **+ Create template** on the right. Fill in the name of the template in the form, the hardware and connection type you choose does not matter now. Click the **Done** button to finish creating the template.



**Add data streams**

For each value you want to write to Blynk, you need to create a data stream. In this manual we will use the same principle as in the previous version of Blynk - **virtual pins**.

On the open page of the created template, click on **Datastreams** in the menu. After clicking on the **+ New Datastream** button, select the **Virtual Pin** option.

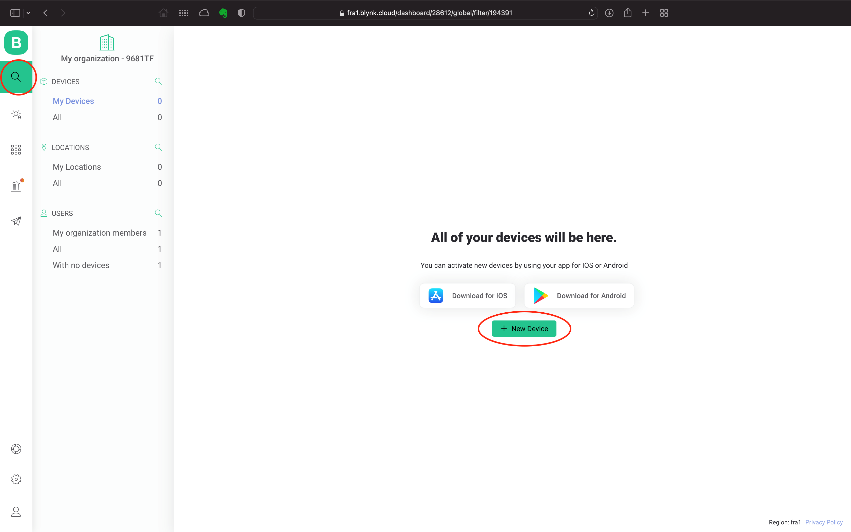


Fill in the datastream name, color, assign it a number (each datastream will have a unique number), select the datatype, and if necessary, the unit and other parameters. Click **Create** to finish adding.

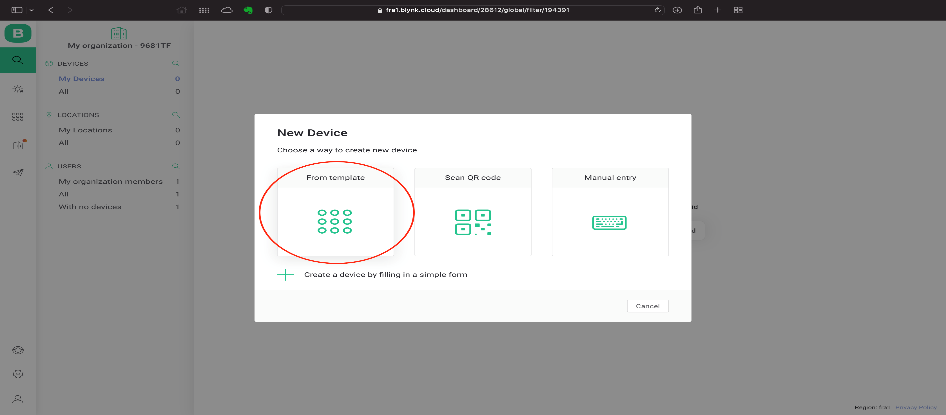
If you want to add more datastreams, repeat the same procedure.

**Create a device from the template**

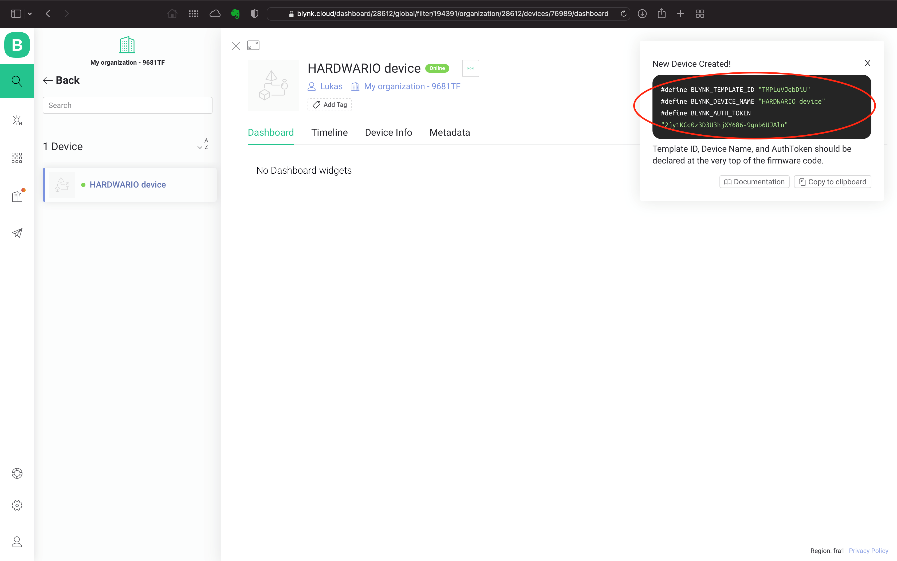
Navigate to the home page of Blynk by clicking on the magnifying glass on the left. Click the **+ New Device** button to open the window to create a new device.



In the creation method selection, choose **From template**. In the next form, select your template and assign a name to the device. Click **Create** to complete the process.



Notice that you have a notification on the right with information about the new device. They contain information about **TEMPLATE ID** and **AUTH TOKEN**, both of which you will need to load in Playground in a moment.

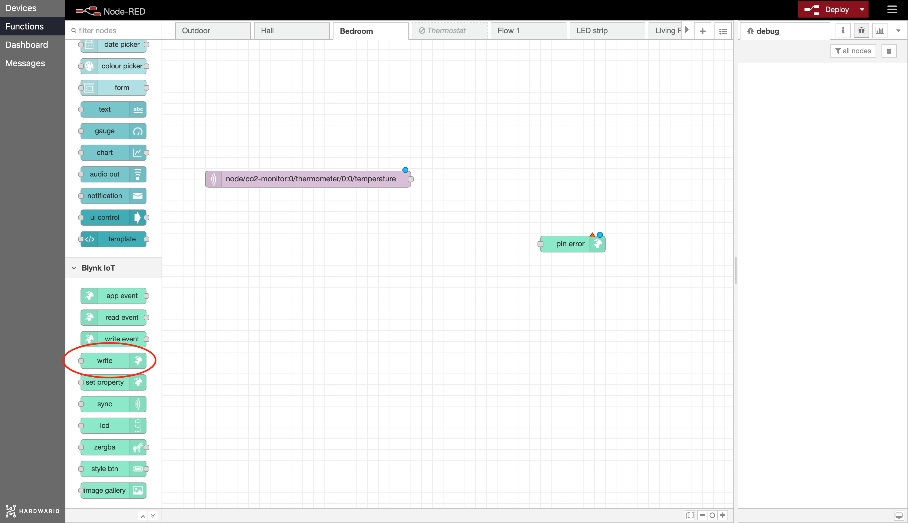


**Move to Playground**

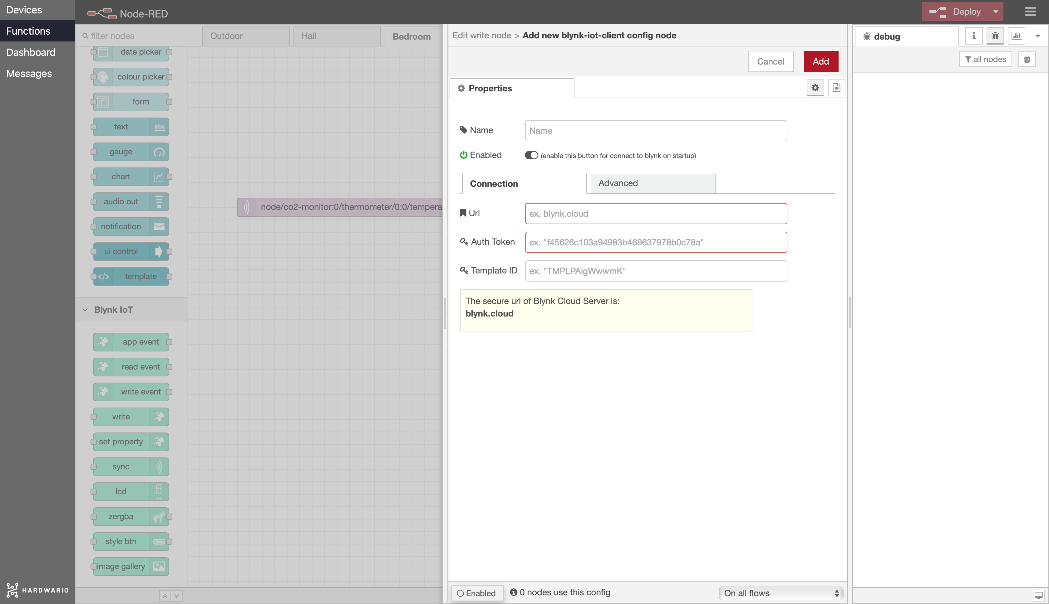
For a functional connection, we recommend updating Playground to the latest version, it includes the pre-installed Blynk IoT plugin (node-red-contrib-blynk-iot). Alternatively, install this plugin.

In the **Functions** tab, find the nodes that work with the new version of Blynk - they are marked with the **Blynk IoT** section.

1. Select the **write** node to write values to the datastream you created.

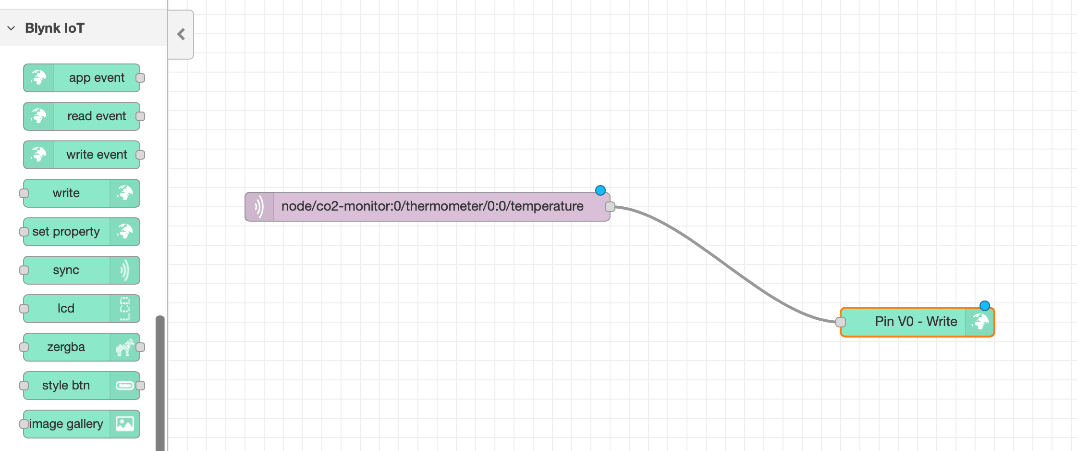


1. Double click to get to its settings.
2. Right click on the small pencil, click to open a new window. In the **Url** field, enter blynk.cloud, in the **Auth Token** and **Template ID** fields, copy the values from the device details in the web application on your computer.



1. Confirm the settings by clicking **Add**.
2. Fill in the Virtual Pin number of your datastream and click **Done** to save everything.

Now you can set up your flow in Node-RED. For example, it may look as simple as this.

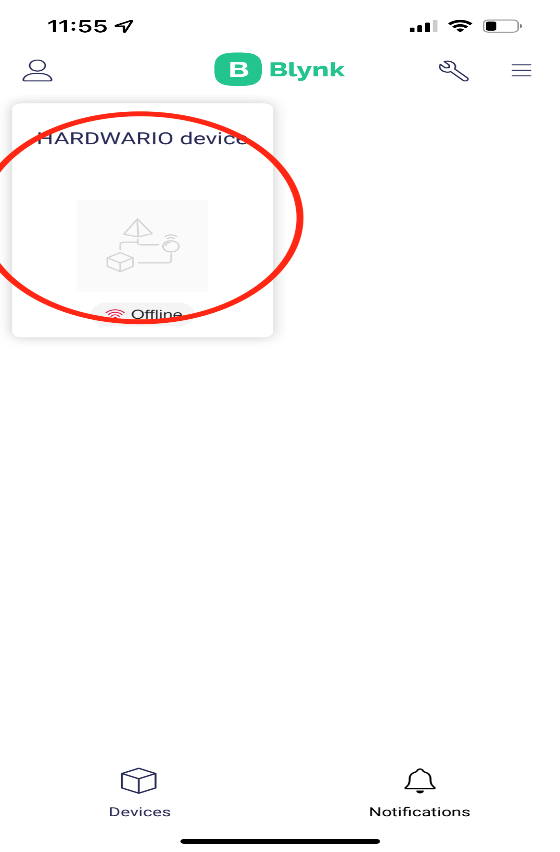


**Set up your mobile app**

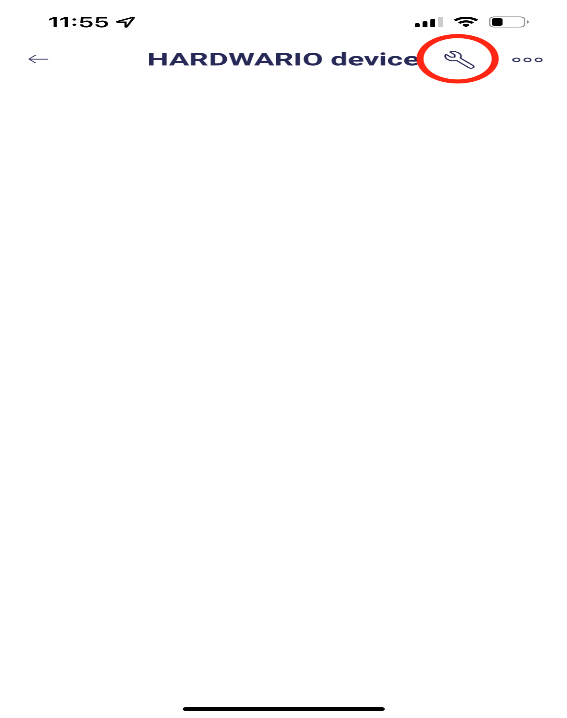
**You can download the Blynk IoT** app from the [App store](https://apps.apple.com/us/app/blynk-iot/id1559317868) or [Google Play](https://play.google.com/store/apps/details?id=cloud.blynk). Log in with your details.

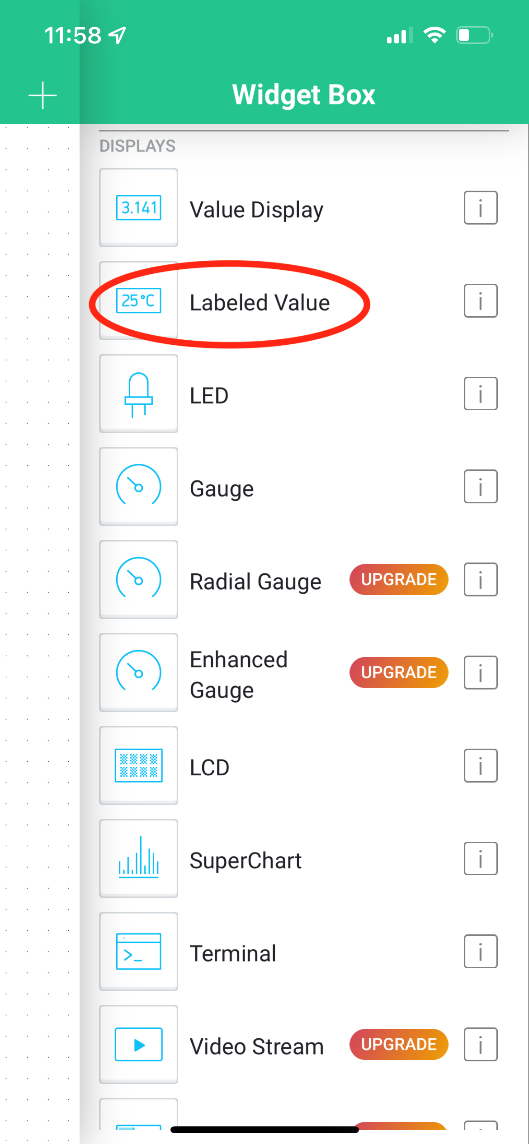


Right on the home page you will see the device you created.

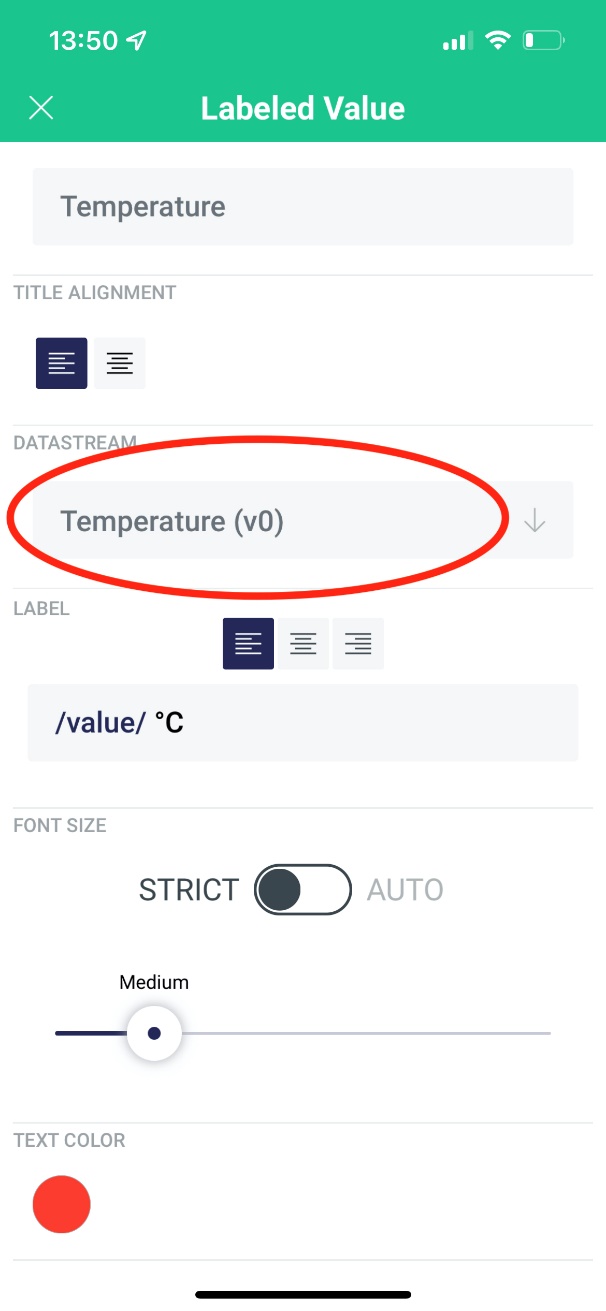


Open it by clicking and set up your dashboard:

1. Under the **key** on the top right you will find the dashboard setup page.}
2. Use the **+** button , or click somewhere on the desktop to add a new widget. In the demo we use **Labeled Value**.



1. Press the added widget to display its settings window. The most important thing is to add the ***Datastream*** from your template for the selected Virtual Pin.



1. You're done. Your widget in the Blynk IoT app will now start saving the data sent from Playground.

**3.6.5** **The Arduino Integrated Development Environment**

The Arduino Integrated Development Environment - or Arduino Software (IDE) - contains a text editor for writing code, a message area, a text console, a toolbar with buttons for common functions and a series of menus. It connects to the Arduino hardware to upload programs and communicate with them.

**3.6.6 The Proteus Design Suite**

The Proteus Design Suite is a proprietary software tool suite used primarily for electronic design automation. The software is used mainly by electronic design engineers and technicians to create schematics and electronic prints for manufacturing printed circuit boards.

**3.6.7 Node MCU**

NodeMCU is a low-cost open-source IoT platform. It initially included firmware which runs on the ESP8266 Wi-Fi SoC from Espressif Systems, and hardware which was based on the ESP-12 module. Later, support for the ESP32 32-bit MCU was added.

**Installing Arduino IDE Software**

**Step 1: ​**

Install Arduino IDE software from the link <http://www.arduino.cc/en/main/software>

Add TipAsk QuestionCommentDownload

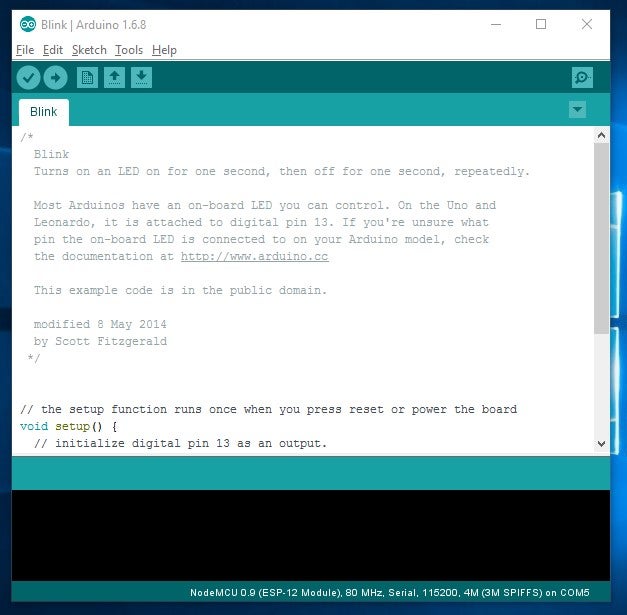
**Step 2: Arduino IDE Icon**

[](https://content.instructables.com/FCF/56HA/IMF3TY91/FCF56HAIMF3TY91.jpg?auto=webp&frame=1&fit=bounds&md=47b36aa7b211b6c8f74569ac11dc2e38)

After installing Arduino IDE icon is created on the Desktop as show in the figure

Add TipAsk QuestionCommentDownload

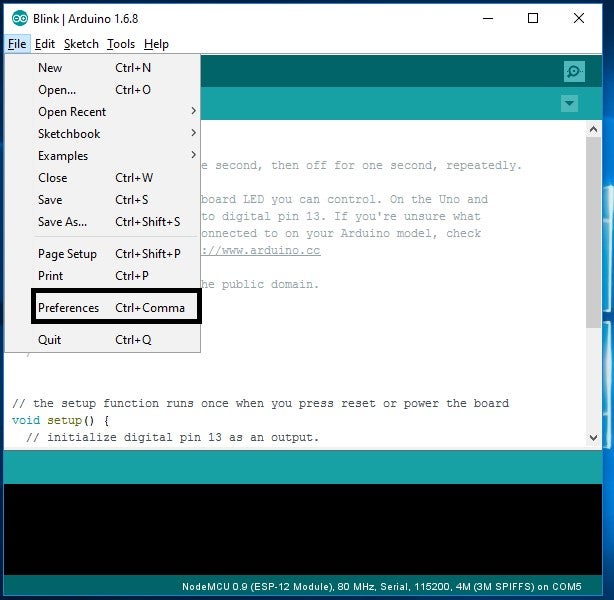
**Step 3: Opening Arduino IDE**

[](https://content.instructables.com/F4S/4P2Y/IMF3TYCB/F4S4P2YIMF3TYCB.jpg?auto=webp&frame=1&fit=bounds&md=02a48a2d4ddbc6219c53b04ff4478d48)

Click on the Icon to open the Arduino window as shown in the figure

Add TipAsk QuestionCommentDownload

**Step 4: Preferences**

[](https://content.instructables.com/FHP/AKIV/IMF3TYG4/FHPAKIVIMF3TYG4.jpg?auto=webp&frame=1&fit=bounds&md=1a77c2aef5525bc8bccd3d683088fe31)

Open the File and click on the Preferences as shown in the figure

Add TipAsk QuestionCommentDownload

**Step 5: Adding ESP8266 Board Manager**

[](https://content.instructables.com/FDV/1WRF/IMF3TZO9/FDV1WRFIMF3TZO9.jpg?auto=webp&frame=1&fit=bounds&md=2dfe6fde6714f26be7f06e42bdcdfbd9)

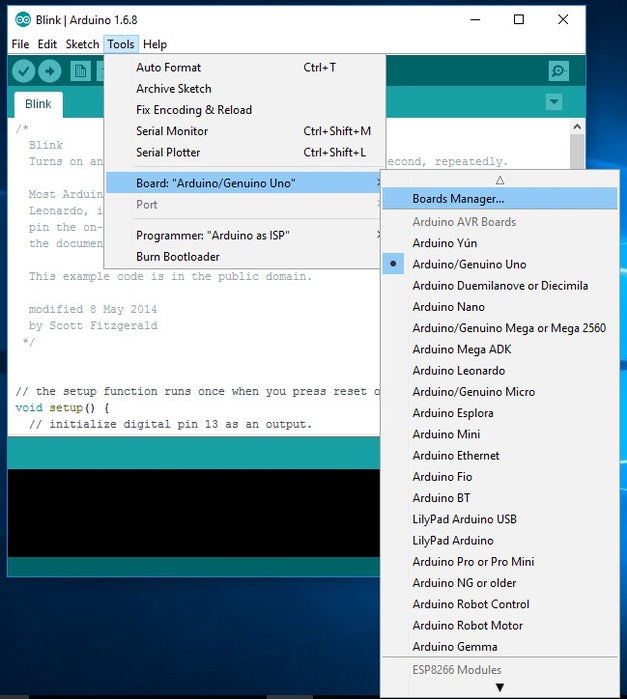
In the Additional Boards Manager enter below URL.

<http://arduino.esp8266.com/stable/package_esp8266com_index.json>

As highlighted in the figure and enter OK.

Add TipAsk QuestionCommentDownload

**Step 6: Selecting Board**

[](https://content.instructables.com/FFS/FWV0/IMF3TZQ3/FFSFWV0IMF3TZQ3.jpg?auto=webp&frame=1&fit=bounds&md=aba8ceec6a280ebf81fdc64c147bb8fe)

Now open the tools in that select **Board: “Arduino/Genuino Uno”**and click on the **Boards Manager** as shown in the figure

Add TipAsk QuestionCommentDownload

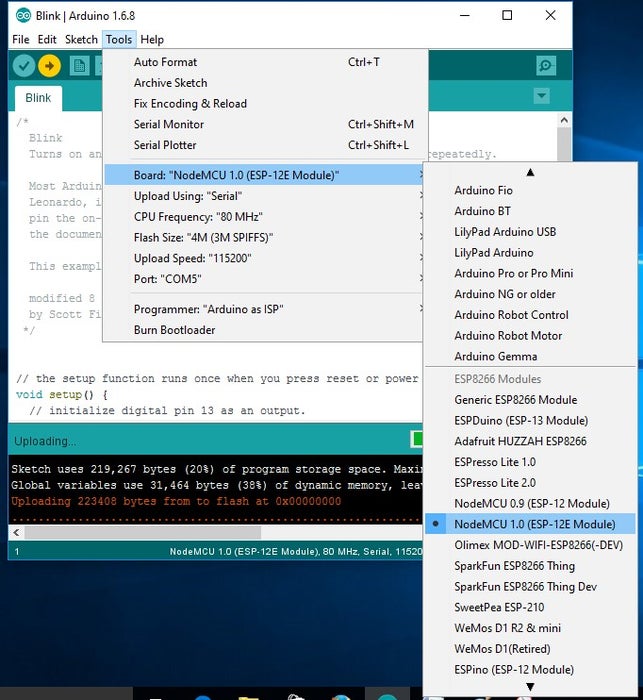
**Step 7: ESP8266 Board Package**

[](https://content.instructables.com/FKP/T36T/IMF3TZVF/FKPT36TIMF3TZVF.jpg?auto=webp&frame=1&fit=bounds&md=abc63cb338450347c3a8679cb2485ecf)

The Boards Manager window opens, scroll the window page to bottom till you see the module with the name ESP8266. Once we get it, select that module and select version and click on the Install button. When it is installed it shows Installed in the module as shown in the figure and then close the window.

Add TipAsk QuestionCommentDownload

**Step 8: Selecting ESP8266 Arduino Board**

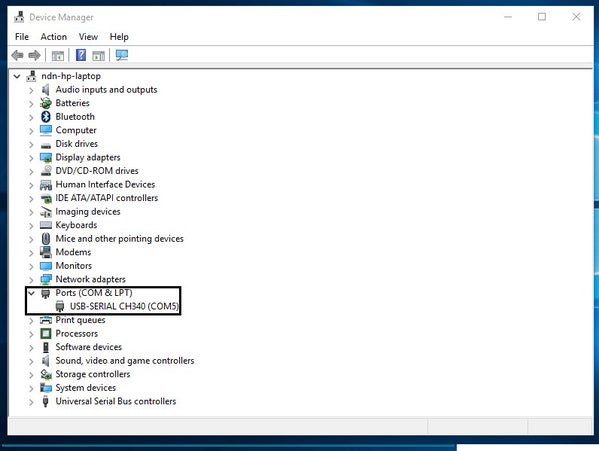
[](https://content.instructables.com/F1F/8PAV/IMF3U001/F1F8PAVIMF3U001.jpg?auto=webp&frame=1&fit=bounds&md=660255e814d48e2b8ed02b635821ca70)

To run the esp8266 with Arduino we have to select the **Board: “Arduino/Genuino Uno”** and then change it to **NodeMCU 1.0 (ESP-12E Module)** or other esp8266 modules depending on what you have .This can be done by scrolling down, as shown in the figure

Add TipAsk QuestionCommentDownload

**Step 9: Connecting ESP8266 to the PC**

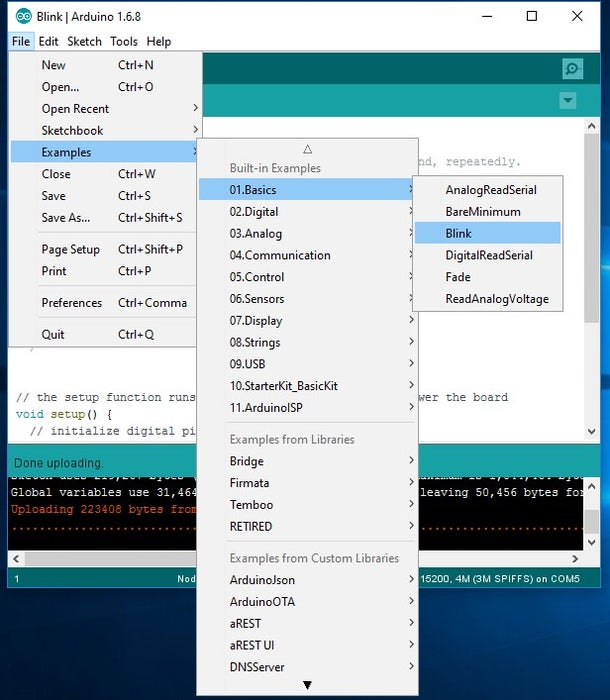
[](https://content.instructables.com/F41/QW7M/IMF3U080/F41QW7MIMF3U080.jpg?auto=webp&frame=1&width=1024&fit=bounds&md=167d277c58cec1b5edccd4332c1575ee)

[](https://content.instructables.com/F2D/0LHE/IMF3U07X/F2D0LHEIMF3U07X.jpg?auto=webp&frame=1&fit=bounds&md=1bdc865f5d568f92a433759f23a18870)

Now Let’s connect the ESP8266 module to your computer through USB cable as shown in the figure. When module is connected to the USB, COM port is detected eg: here COM5 is shown in the figure.

Add TipAsk QuestionCommentDownload

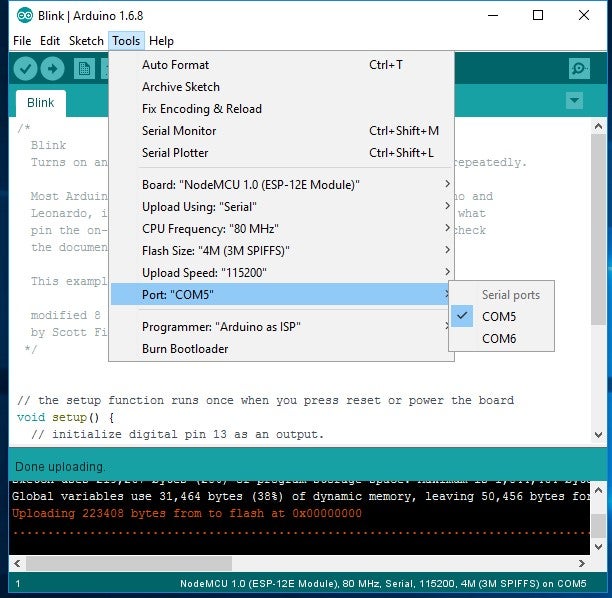
**Step 10: Selecting Example Program in Arduino IDE**

[](https://content.instructables.com/FUU/2F5X/IMF3U0BV/FUU2F5XIMF3U0BV.jpg?auto=webp&frame=1&fit=bounds&md=681a06776c86dc629f203ee94e528dc3)

Now open the File tab in that go to the Examples in that enter into Built-in example, go to 01.Basics and click on Blink to open the window

Add TipAsk QuestionCommentDownload

**Step 11: Selecting COM Port**

[](https://content.instructables.com/FE5/8PWW/IMF3U0L0/FE58PWWIMF3U0L0.jpg?auto=webp&frame=1&fit=bounds&md=7cd2f7d9794ffca45daa9f34a9517b71)

The Blink example will open on a new window , click on tools to select the port: “COM” based on which esp8266 module is connected to your respected COM port of the computer. To select COM port refer previous steps.

Add TipAsk QuestionCommentDownload

**Step 12: Uploading the Program to ESP8266 Module**

[](https://content.instructables.com/FTZ/VFL3/IMF3U0TL/FTZVFL3IMF3U0TL.jpg?auto=webp&frame=1&fit=bounds&md=73178573b9cde9b7df9194e8b1c5446a)

On the blink example code change all number 13 to number 16 and then click on the right arrow shown in the figure to upload the program to the module. This will start blinking the on board led on the nodemcu module

void setup()

{  
// initialize digital pin 16 as an output.

pinMode(16, OUTPUT);

}

// the loop function runs over and over again forever

void loop()

{

delay(10);

digitalWrite(16, HIGH); // turn the LED on (HIGH is the voltage level)

delay(1000); // wait for a second

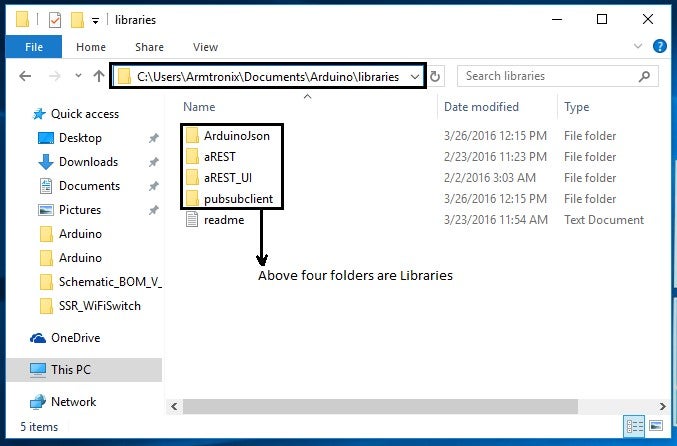
digitalWrite(16, LOW); // turn the LED off by making the voltage LOW

delay(1000); // wait for a second

}

Add TipAsk QuestionCommentDownload

**Step 13: Adding Libraries**

[](https://content.instructables.com/FLY/LSBD/IMF3U11U/FLYLSBDIMF3U11U.jpg?auto=webp&frame=1&fit=bounds&md=a438eb2a9c9a30e85790234f7aaf2308)

In case you need to add the libraries to the Arduino follow the example path is shown in the figure i.e C:\Users\Armtronix\Documents\Arduino\libraries. Enter into the libraries folder then paste the file in that as shown below.

**3.6.8 Sensors**

A sensor is a device that detects and responds to some type of input from the physical environment. The input can be light, heat, motion, moisture, pressure or any other environmental pheno MAX30100 Pulse Oximeter Heart Rate Sensor Module is an integrated pulse oximeter and heart-rate monitor sensor solution. It’s an optical sensor that derives its readings from emitting two wavelengths of light from two LEDs – a red and an infrared one – then measuring the absorbance of pulsing blood through a photodetector. This particular LED colour combination is optimized for reading the data through the tip of one’s finger. It is fully configurable through software registers and the digital output data is stored in a 16-deep FIFO within the device. It has an I2C digital interface to communicate with a host microcontroller.

The pulse oximetry subsystem in MAX30100 consists of ambient light cancellation (ALC), 16-bit sigma-delta ADC, and a proprietary discrete time filter. It has an ultra-low-power operation which makes it ideal for battery-operated systems. MAX30100 operates on a supply in the range of 1.8 to 3.3V. It can be used in wearable devices, fitness assistant devices, medical monitoring devices, etc. The MAX30100 operates from 1.8V and 3.3V power supplies and can be powered down through software with negligible standby current, permitting the power supply to remain connected at all times.

**Features :**

* It is an integrated pulse oximetry and heart rate monitor sensor solution.
* Integrated LEDs, Photo Sensor, and High-Performance Analog Front -End
* Complete Pulse Oximeter and Heart-Rate Sensor Solution Simplifies Design
* Measures absorbance of pulsing blood
* I2C interface plus INT pin
* Ultra-Low-Power Operation Increases Battery Life for Wearable Devices
* Programmable Sample Rate and LED Current for Power Savings
* Ultra-Low Shutdown Current (0.7µA, typ)
* Advanced Functionality Improves Measurement Performance
* High SNR Provides Robust Motion Artifact Resilience
* Integrated Ambient Light Cancellation
* High Sample Rate Capability
* Fast Data Output Capability

**3.6.9 Relay module**

A power relay module is an electrical switch that is operated by an electromagnet. The electromagnet is activated by a separate low-power signal from a microcontroller. When activated, the electromagnet pulls to either open or close an electrical circuit.

* Relay works on the principle of electromagnetic induction.
* When the electromagnet is applied with some current, it induces a magnetic field around it.
* Above image shows working of the relay. A switch is used to apply DC current to the load.
* In the relay, Copper coil and the iron core acts as electromagnet.
* When the coil is applied with DC current, it starts attracting the contact as shown. This is called energizing of relay.
* When the supply is removed it retrieves back to the original position. This is called De energizing of relay.

**3.6.10 SMPS**

The full form of SMPS is Switched Mode Power Supply also known as Switching Mode Power Supply. SMPS is an electronic power supply system that makes use of a switching regulator to transfer electrical power effectively

**3.6.11 Pump/valve**

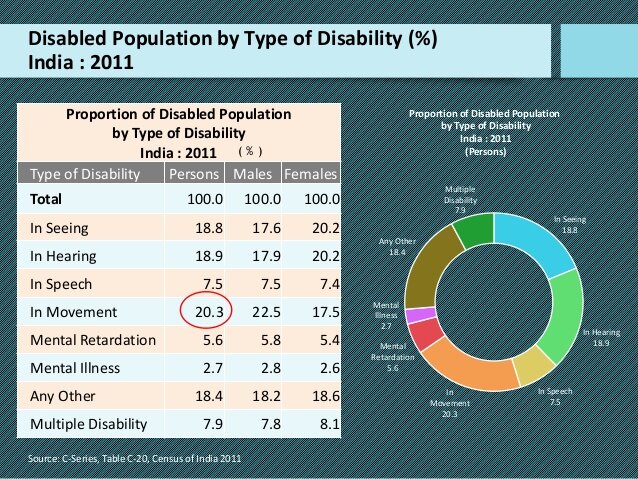
Reciprocating pumps are a type of positive displacement pump, consisting of a: Fluid-End (consisting of two chambers, suction and discharge, separated by spring-loaded valves) and a Suction Inlet (where fluid flows from piping through a valve into the first chamber, aka suction chamber). Valves regulate and control flow and pressure in pumping systems. They also play an important role in site safety. Understanding the types of valves and how they work can help end users select the right valves for their application’s use.

**CHAPTER 4**

**PROPOSED SYSTEM**

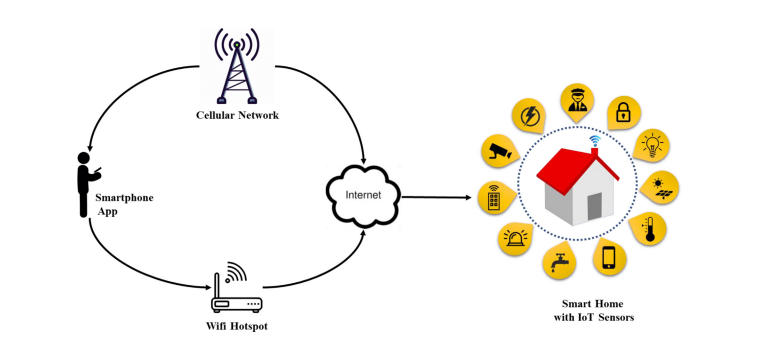
**3.1 PROBLEM STATEMENT**

Disabilities is an umbrella term, covering impairments, activity limitations, and participation restrictions. An impairment is a problem in body function or structure, An activity limitation is a difficulty encountered by an individual in executing a task or action, and A participation restriction is a problem experienced by an individual in involvement in life situations. As per Census 2011, in India, out of the total population of 121 crores, about 2.68 Cr persons are ‘Disabled’ (2.21%of the total population) Out of 2.68 crores, 1.5 crores are males and 1.18 crore are females Majority (69%) of the disabled population resided in rural areas.



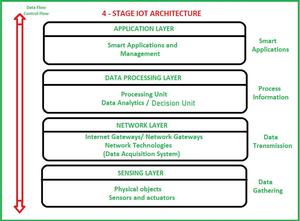
**3.2 PROPOSED METHODOLOGY**

Amazon Echo and Google Home have been one of the most publicized uses of IoT in people’s homes. To use utilities and basic services within a house, people have to no longer depend on switches and conventional physical methods of interaction. For example, IoT devices allow a visually impaired user to change the heat settings without needing to program a controller. These devices work by using a voice recognition system, without any kind of special setup. For people suffering from paralysis or those who are completely bedridden, such technologies are no less than a boon as they perform functions like unlocking a door without In most cases, people with disabilities require constant monitoring which can often be challenging and demanding for them. Michael J Fox Foundation uses IoT devices to monitor hundreds of people with disabilities and long-term illnesses. They gather millions of diverse data points that hold a clue to cure a disease. The data collected is so enormous that it can be used not only for those who are ill and challenged but also to evolve newer models for developing preventive measures for any new kind of disease as well. IoT will break the accessibility barriers. IoT-enabled smart environments will create an enabling system that embodies inclusiveness, a smart ecosystem that helps challenged people to live their lives freely.



**3.3 MODULE DESCRIPTION**

Internet of Things (IoT) technology has a wide variety of applications and the use of the Internet of Things is growing so faster. Depending upon different application areas of the Internet of Things, it works accordingly as per it has been designed/developed. But it has not a standard defined architecture of working which is strictly followed universally. The architecture of IoT depends upon its functionality and implementation in different sectors. Still, there is a basic process flow based on which IoT is built.



**Sensing Layer**

Sensors, actuators, and devices are present in this Sensing layer. These Sensors or Actuators accept data(physical/environmental parameters), process data, and emit data over the network.

**Network Layer**

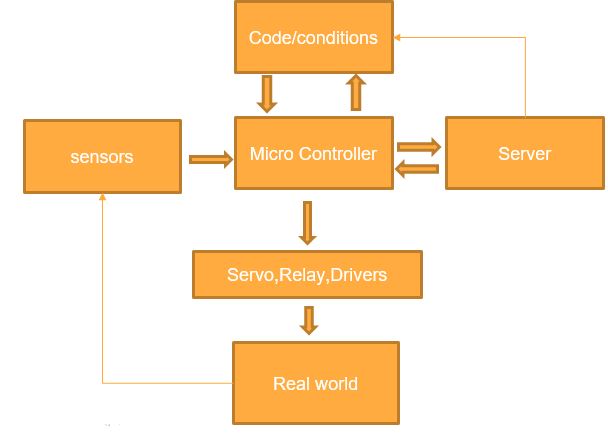
Internet/Network gateways and Data Acquisition Systems (DAS) are present in this layer. DAS performs data aggregation and conversion fufunctionsCollecting data and aggregating data then converting analogue data of sensors to digital data etc). Advanced gateways which mainly open up a connection between Sensor networks and the Internet also perform many basic gateway functionalities like malware protection and filtering sometimes decision-making based on inputted data and data management services, etc.

**Data processing Layer**

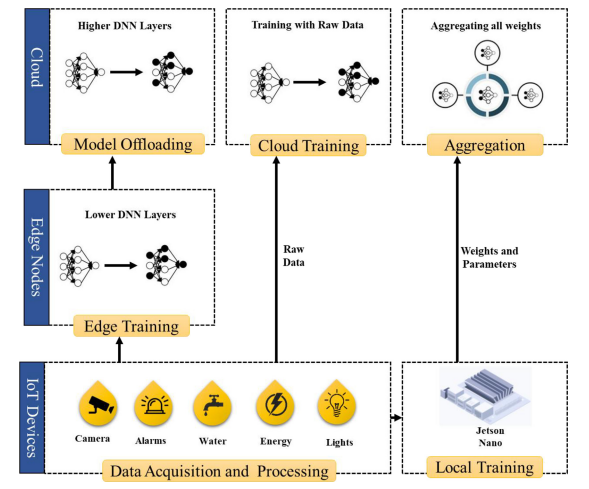
This is the processing unit of the IoT ecosystem. Here data is analyzed and pre-processed before sending it to the data centre from where data is accessed by software applications often termed business applications where data is monitored and managed and further actions are also prepared. So here Edge IT or edge analytics comes into the picture.

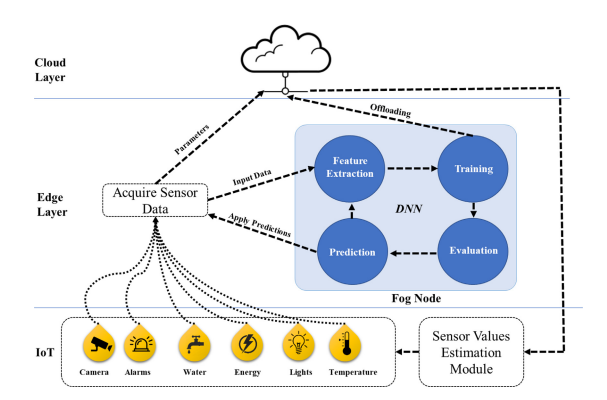
**Application Layer**

This is the last layer of the 4 stages of IoT architecture. Data centres or cloud is the management stage of data where data is managed and is used by end-user applications like agriculture, health care, aerospace, farming, defence, etc.



**3.4 Proposed framework**

****

****

Home automation, security, safety, privacy, and energy consumption is an emerging field of technology and can be implemented with different microcontrollers and processors i.e., Desktop PC, laptops, Raspberry Pi, Arduino, etc., All these microcontroller and processors have their pros and cons, but Raspberry Pi is more versatile and efficient option compared to other units. The smart home concepts can have many components including 1) Sensors, 2) Gateways, 3) Protocols, 4) Firmware and 5) Cloud-based databases. The main focus of bringing AI to the smart home is to automate certain processes that need more than thresholds and pre-defined configurations. For example, one such task is fall detection. The process of integrating AI in the smart home is more natural and requires a significant amount of computational overhead. Today, most of the data generated by IoT devices resides outside the scope of the cloud so it is more natural to bring the machine learning capabilities closer to the source of the data and reduce the amount of bandwidth it consumes by transferring the entire dataset for each training session. Therefore, we leverage an approach called mobile edge compcomputinere the collaborative nature of the architecture makes it easy to train data locally on the device, and then send data to the higher-level layers of the DNN to train it on the edge server. The remaining data is relayed to the cloud where the resource-intensive tasks are offloaded and the remaining mode is trained on the cloud d.the architecture derived from federated learning proposed by Konecˇny´ et al. The authors proposed a method called on deon-demand learning co-inference with edge synergy. The authors argue that using the right sizing approach for DNN layers, the early exit point of the layer must be maintained to offload computationally intense tasks to the cloud from the compacomparativeurce constraint devices. We make use of the same technique discussed here and build upon our framework to work with SqueezeNet for fall detection using Jetson Nano as an edgaedgerocessing node. In this architecture, the data acquisition and processing layer acquires data and is processed locally on the Raspberry Pi as it is the main processing device for IoT devices. The Raspberry Pi itself is not capable of training the model as it lacks GPUthe capabilities that Jetson Nano possesses. The Jetson Nano receives the data and trains a preliminary model for text and multimedia data acquired from the surveillance camera. The multimedia data from the surveillance cameras are much larger in size compared to other devices, therefore the data is trained on the edge node, where the deep learning model is trained using Jetson Nano capable of performing certain levels of computation. However, we assume that the number of multimedia devices may grow in the future and the edge device mentioned here may not respond to all data for training. For this purpose, we split the model and lower-level layers are placed on the server, while the rest of the higher-level layers of the DNN are placed on the cloud with presumably an infinite amount of resources. The cloud model can receive weights from the edge server, it can also update parameters sent from the Jetson Nano or it can also be trained by the data directly sent to the cloud by the IoT devices themselves however, the data sent by the devices directly can compromise the acy and may require additional security. The aggregation module of the cloud layer provides means of aggregation for all parameters sent to the cloud and it helps update all the weights of a single model that can send predictions and estimations along with trends to the IoT devices. To deploy AI services through the cloud, we make use of cloud services provided by Amazon, called Amazon Web Services. It provides off-the-shelf technologies to integrate and deploy AI in IoT applications without too many complications. Moreover, it has a built-in dashboard to support Jetson Nano and is compatible with most IoT applications in general. Details of the working of the edge device and device-based learning for the IoT devices are presented in Fig. 3. The edge layer is equipped with a slightly more powerful device Jetson Nano that acts as a Fog Node and performs machine learning tasks for text and multimedia data. The device receives data and sends it to a feature extraction model. Then the features are sent to train the model itself, where they can be assessed for offloading the cloud or evaluated for training accuracy. The final predictions are sent back to the device and thereafter to the cloud for storage and updating of the cloud-based learning model. The cloud sends updated parameters to the Raspberry Pi which controls and overlooks the connected IoT devices. In this case, the Raspberry Pi changes the values of certain IoT devices like temperature sensors, and surveillance cameras to change parameters. For instance, the temperature sensor can raise the threshold based on the values sent by the cloud model for the next day. In the case of water management, the estimation module can predict the water requirements for the day based on values estimated by the model on previous days of the week. The estimation module uses a REST API and MQTT protocol to connect to the devices and updates the relevant sensor with appropriate results received from the model.

**3.3 Hardware Implementation**

The hardware utilized in IoT systems includes devices for a remote dashboard, devices for control, servers, a routing or bridge device, and sensors. These devices manage key tasks and functions such as system activation, action specifications, security, communication, and detection to support-specific goals and actions.

**IoT − Sensors**

he most important hardware in IoT might be its sensors. These devices consist of energy modules, power management modules, RF modules, and sensing modules. RF modules manage communications through their signal processing, WiFi, ZigBee, Bluetooth, radio transceiver, duplexer, and BAW The sensing module manages sensing through assorted active and passive measurement devices. Here is a list of some of the measurement devices used in IoT –

|  |  |  |
| --- | --- | --- |
| **S.No** | **Devices** | |
| 1. | accelerometers | temperature sensors |
| 2. | magnetometers | proximity sensors |
| 3. | gyroscopes | image sensors |
| 4. | acoustic sensors | light sensors |
| 5. | pressure sensors | gas RFID sensors |
| 6. | humidity sensors | micro flow sensors |

**Standard Devices**

The desktop, tablet, and cellphone remain integral parts of IoT as the command centre and remotes.

The desktop provides the user with the highest level of control over the system and its settings.

The tablet provides access to the key features of the system in a way resembling the desktop and also acts as a remote.

The cellphone allows some essential settings modification and also provides remote functionality.

Other key connected devices include standard network devices like routers and switches.

**3.3 Internet of Things - Software**

IoT software addresses its key areas of networking and action through platforms, embedded systems, partner systems, and middleware. These individual and master applications are responsible for data collection, device integration, real-time analytics, and application and process extension within the IoT network. They exploit integration with critical business systems (e.g., ordering systems, robotics, scheduling, and more) in the execution of related tasks.

**Data Collection**

This software manages sensing, measurements, light data filtering, light data security, and aggregation of data. It uses certain protocols to aid sensors in connecting with real-time, machine-to-machine networks. Then it collects data from multiple devices and distributes it in accordance with settings. It also works in reverse by distributing data over devices. The system eventually transmits all collected data to a central server.

**Device Integration**

Software-supporting integration binds (dependent relationships) all system devices to create the body of the IoT system. It ensures the necessary cooperation and stable networking between devices. These applications are the defining software technology of the IoT network because, without them, it is not an IoT system. They manage the various applications, protocols, and limitations of each device to allow communication.

**Real-Time Analytics**

These applications take data or input from various devices and convert it into viable actions or clear patterns for human analysis. They analyze information based on various settings and designs in order to perform automation-related tasks or provide the data required by the industry.

**Application and Process Extension**

These applications extend the reach of existing systems and software to allow a wider, more effective system. They integrate predefined devices for specific purposes such as allowing certain mobile devices or engineering instruments access. It supports improved productivity and more accurate data collection.

**3.4 Internet of Things - Technology and Protocols**

IoT primarily exploits standard protocols and networking technologies. However, the major enabling technologies and protocols of IoT are RFID, NFC, low-energy Bluetooth, low-energy wireless, low-energy radio protocols, LTE-A, and WiFi-Direct. These technologies support the specific networking functionality needed in an IoT system in contrast to a standard uniform network of common systems.

**NFC and RFID**

RFID (radio-frequency identification) and NFC (near-field communication) provide simple, lowenergy, and versatile options for identity and access tokens, connection bootstrapping, and payments.RFID technology employs 2-way radio transmitter-receivers to identify and track tags associated with objects.NFC consists of communication protocols for electronic devices, typically a mobile device and a standard device.

**Low-Energy Bluetooth**

This technology supports the low-power, long-use need of IoT function while exploiting a standard technology with native support across systems.

**Low-Energy Wireless**

This technology replaces the most power hungry aspect of an IoT system. Though sensors and other elements can power down over long periods, communication links (i.e., wireless) must remain in listening mode. Low-energy wireless not only reduces consumption, but also extends the life of the device through less use.

**Radio Protocols**

ZigBee, Z-Wave, and Thread are radio protocols for creating low-rate private area networks. These technologies are low-power, but offer high throughput unlike many similar options. This increases the power of small local device networks without the typical costs.

**LTE-A**

LTE-A, or LTE Advanced, delivers an important upgrade to LTE technology by increasing not only its coverage, but also reducing its latency and raising its throughput. It gives IoT a tremendous power through expanding its range, with its most significant applications being vehicle, UAV, and similar communication.

**WiFi-Direct**

WiFi-Direct eliminates the need for an access point. It allows P2P (peer-to-peer) connections with the speed of WiFi, but with lower latency. WiFi-Direct eliminates an element of a network that often bogs it down, and it does not compromise on speed or throughput.

.

**CHAPTER 4**

**CONCLUSION AND FUTURE SCOPE**