MALLA REDDY ENGINEERING COLLEGE

(Autonomous) - Main Campus

(A UGC Autonomous Institution, Approved by AICTE, New Delhi & Affiliated to JNTUH, Hyderabad).
Accredited by NAAC with 'A++' Grade (Cycle- III),
NIRF Rank Band 201-250, ARIIA Band Performer,
NBA Tier-I Accredited (B.Tech- CE, EEE, ME, ECE & CSE, M.Tech - CSE, EPS, TE)
Maisammaguda, Medchal-Malkajgiri District, Secunderabad– 500100, Telangana State. www.mrec.ac.in

CERTIFICATE

Department of : _____

This is to certify that it is a bonafide record of Practical Work done by

Mr./Miss. _____ *bearing Regd. No. of*

_____ *B.Tech, M.Tech, MBA* _____ *Year,* _____ *Semester,* _____

Branch, in the _____ *Laboratory during the*

academic year _____ *has satisfactorily completed the course*

of, program prescribed by the Malla Reddy Engineering College under my Suprvision.

Lab Incharge **Head of the Dept.**

Internal Examiner **External Examiner**

Department of Computer Science and Engineering (IoT)

Course File

III B. Tech II Semester
Subject: IoT Cloud and Data Analytics Lab

Code: A6910

Academic Year 2022-23
Regulations: MR 20

Malla Reddy Engineering College
(Autonomous)
Department of Computer Science and Engineering-IoT
(Approved by AICTE & Affiliated to JNTUH, Hyderabad)
Maisammaguda, Dhulapally(Post via Kompally) Secunderabad - 500100

Contents

INDEX

Lab In charge **HOD**

Malla Reddy Engineering College
(Autonomous)
Department of Computer Science and Engineering-IoT

Institute Vision

To be a premier centre of professional education and research, offering quality programs in a socio-economic and ethical ambience.

Institute Mission

1. **To impart knowledge of advanced technologies using state-of-the-art infrastructural facilities.**
2. **To inculcate innovation and best practices in education, training and research.**
3. **To meet changing socio-economic needs in an ethical ambience.**

Vision

To attain global standards in Computer Science and Engineering education, training, and research to meet the growing needs of the industry with socio-economic and ethical considerations.

Mission

To impart quality education and research to undergraduate and postgraduate students in Computer Science and Engineering.

To encourage innovation and best practices in Computer Science and Engineering utilizing state-of-the-art facilities.

To develop entrepreneurial spirit and knowledge of emerging technologies based on ethical values and social relevance.

Programme Educational Objectives (PEOs)

PEO I:

Graduates to promote collaboration with the reputed industries to have the best careers.

PEO II:

Graduates will demonstrate technical skills, competency in IOT and exhibit team management capability with proper communication in a job environment.

PEO III:

To nurture the sense of creativity and innovation to adopt the socio-economic related activities Graduates will interact with their peers in other disciplines in industry and society and contribute to the economic growth of the country.

PEO IV:

To enable graduates to emerge as independent entrepreneurs and future leaders.

PEO V:

To provide the sate- of- the art facilities to forge the students in industry-ready IoT system development.

Programme Outcomes

PO	Programme Outcomes
PO 1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO 2	Problem analysis: Identify, formulate, review research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO 3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO 4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO 5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO 6	The engineer and society: Apply reasoning informed by the contextual knowledge to societal, health, safety, legal and cultural issues and the consequent responsibilities relevant professional engineering practice.
PO 7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO 8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO 9	Individual and team work: Function effectively as an individual and as a member or leader in diverse teams, and in multidisciplinary settings.
PO 10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as,being able to comprehend and write effective reports and design
PO 11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO 12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

Bloom's Taxonomy Triangle

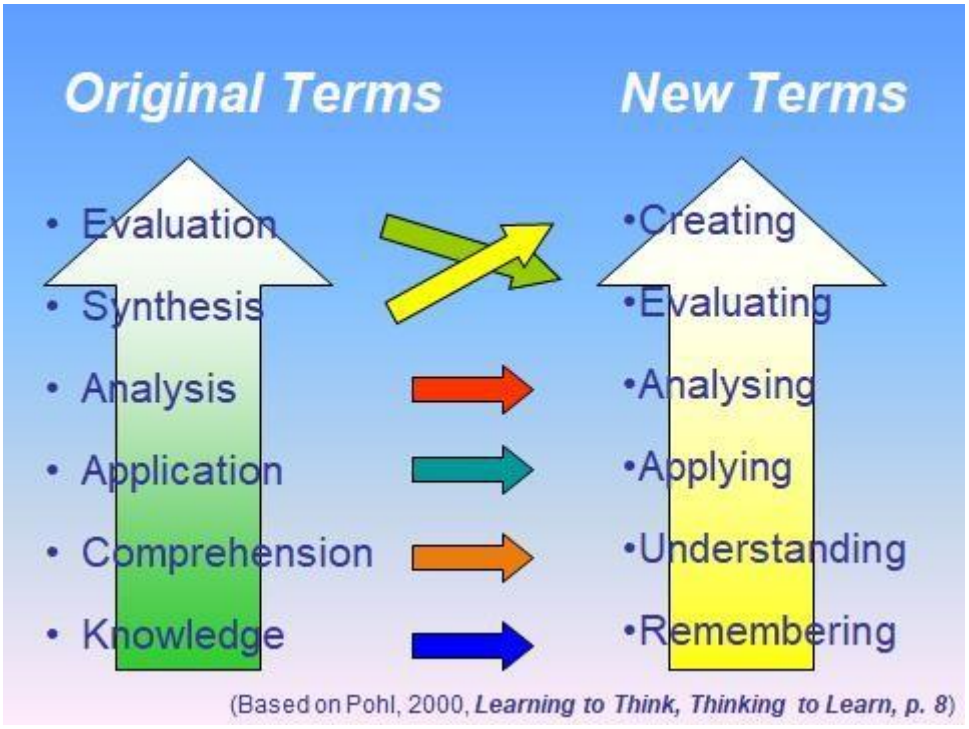
Overview

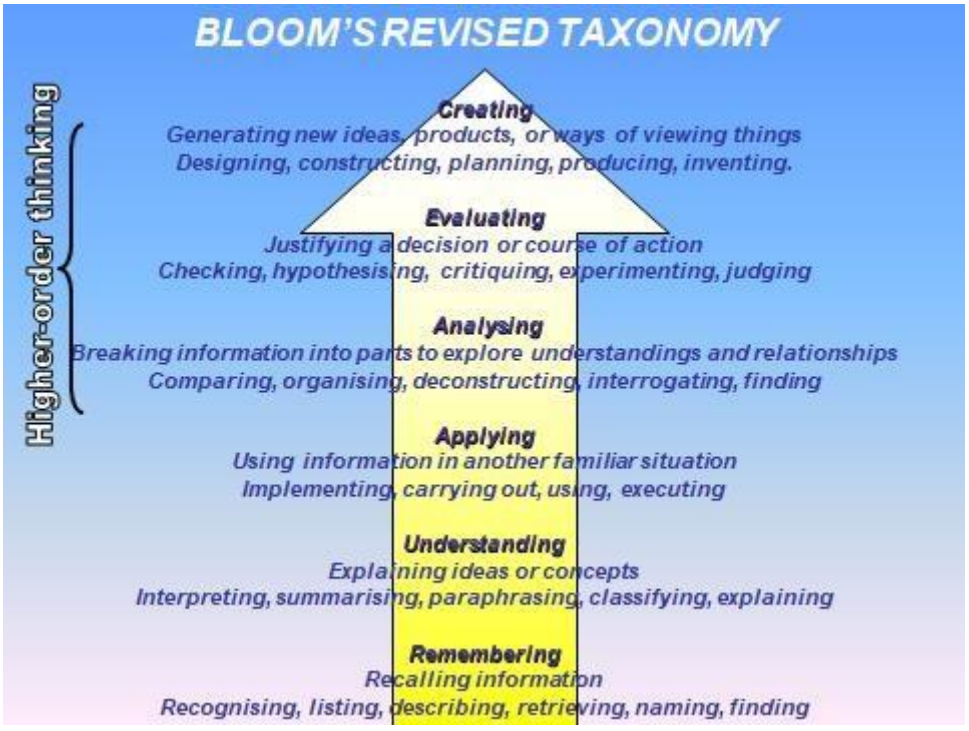
- Bloom's Taxonomy and higher-order thinking
- Take a walk down memory lane
- Investigate the Revised Taxonomy
  - New terms
  - New emphasis
- Explore each of the six levels
- See how questioning plays an important role within the framework (oral language)
- Use the taxonomy to plan a unit
- Look at an integrated approach
- Begin planning a unit with a *SMART Blooms* Planning Matrix

Bloom's Revised Taxonomy

- Taxonomy of Cognitive Objectives
- 1950s- developed by Benjamin Bloom
- Means of expressing qualitatively different kinds of thinking
- Adapted for classroom use as a planning tool
- Continues to be one of the most universally applied models
- Provides a way to organise thinking skills into six levels, from the most basic to the higher order levels of thinking
- 1990s- Lorin Anderson (former student of Bloom) revisited the taxonomy
- As a result, a number of changes were made

(Pohl, 2000, *Learning to Think, Thinking to Learn, pp. 7-8)*


Original Terms
New Terms
• Evaluation
• Synthesis
• Analysis
• Application
• Comprehension
• Knowledge
•Creating
•Evaluating
•Analysing
•Applying
•Understanding
•Remembering
(Based on Pohl, 2000, Learning to Think, Thinking to Learn, p. 8)


BLOOM'S REVISED TAXONOMY
Higher-order thinking
Creating
Generating new ideas, products, or ways of viewing things
Designing, constructing, planning, producing, inventing.
Evaluating
Justifying a decision or course of action
Checking, hypothesising, critiquing, experimenting, judging
Analysing
Breaking information into parts to explore understandings and relationships
Comparing, organising, deconstructing, interrogating, finding
Applying
Using information in another familiar situation
Implementing, carrying out, using, executing
Understanding
Explaining ideas or concepts
Interpreting, summarising, paraphrasing, classifying, explaining
Remembering
Recalling information
Recognising, listing, describing, retrieving, naming, finding

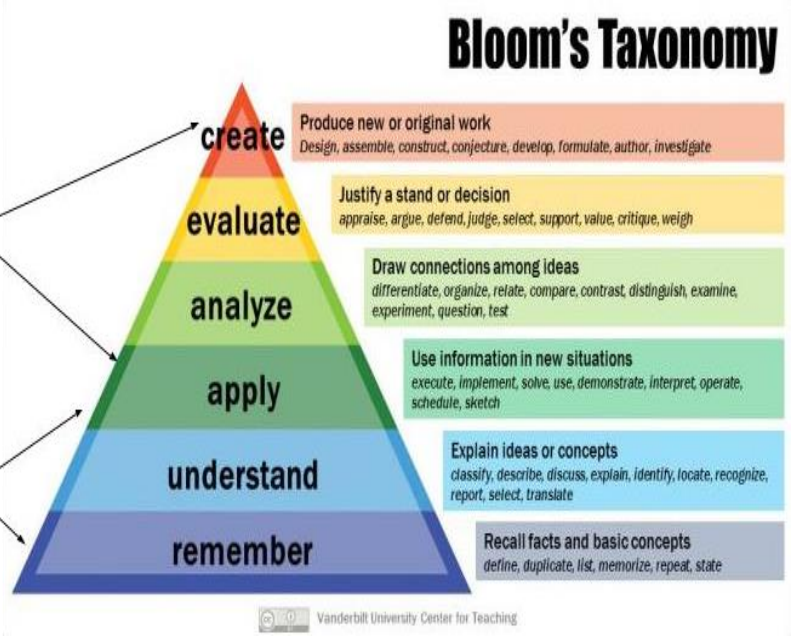
REVISED Bloom's Taxonomy Action Verbs

Definitions	I. Remembering	II. Understanding	III. Applying	IV. Analyzing	V. Evaluating	VI. Creating
Bloom's Definition	Exhibit memory of previously learned material by recalling facts, terms, basic concepts, and answers.	Demonstrate understanding of facts and ideas by organizing, comparing, translating, interpreting, giving descriptions, and stating main ideas.	Solve problems to new situations by applying acquired knowledge, facts, techniques and rules in a different way.	Examine and break information into parts by identifying motives or causes. Make inferences and find evidence to support generalizations.	Present and defend opinions by making judgments about information, validity of ideas, or quality of work based on a set of criteria.	Compile information together in a different way by combining elements in a new pattern or proposing alternative solutions.
Verbs	• Choose • Define • Find • How • Label • List • Match • Name • Omit • Recall • Relate • Select • Show • Spell • Tell • What • When • Where • Which • Who • Why	• Classify • Compare • Contrast • Demonstrate • Explain • Extend • Illustrate • Infer • Interpret • Outline • Relate • Rephrase • Show • Summarize • Translate	• Apply • Build • Choose • Construct • Develop • Experiment with • Identify • Interview • Make use of • Model • Organize • Plan • Select • Solve • Utilize	• Analyze • Assume • Categorize • Classify • Compare • Conclusion • Contrast • Discover • Dissect • Distinguish • Divide • Examine • Function • Inference • Inspect • List • Motive • Relationships • Simplify • Survey • Take part in • Test for • Theme	• Agree • Appraise • Assess • Award • Choose • Compare • Conclude • Criteria • Criticize • Decide • Deduct • Defend • Determine • Disprove • Estimate • Evaluate • Explain • Importance • Influence • Interpret • Judge • Justify • Mark • Measure • Opinion • Perceive • Prioritize • Prove • Rate • Recommend • Rule on • Select • Support • Value	• Adapt • Build • Change • Choose • Combine • Compile • Compose • Construct • Create • Delete • Design • Develop • Discuss • Elaborate • Estimate • Formulate • Happen • Imagine • Improve • Invent • Make up • Maximize • Minimize • Modify • Original • Originate • Plan • Predict • Propose • Solution • Solve • Suppose • Test • Theory

Anderson, L. W., & Krathwohl, D. R. (2001). A taxonomy for learning, teaching, and assessing, Abridged Edition. Boston, MA: Allyn and Bacon.

Focus of individual time in traditional model

Focus of class time in traditional model



The Graduate Attributes for UG Engineering Student

1. **Engineering Knowledge:** apply Knowledge of mathematics, science, engineering fundamentals and an engineering specialization to the solution of complex engineering problems.

2. **Problem Analysis:** Identify, formulate, research literature and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences and engineering sciences.

3. **Design / Development of Solutions:** Design solutions for complex engineering problems and design system components or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal and environmental considerations.

4. **Conduct investigations of complex problems** using research-based knowledge and research methods including design of experiments, analysis and interpretation of data and synthesis of information to provide valid conclusions.

5. **Modern Tool Usage:** Select and apply appropriate techniques, resources, and modern engineering and IT tools, including prediction and modeling, to broadly-defined engineering activities, with an understanding of the limitations.

6. **The Engineer and society:** Demonstrate understanding of the societal, health, safety, legal and cultural issues, and the consequent responsibilities relevant to engineering technology practice.

7. **Environment and sustainability:** Understand the impact of engineering/technology solutions in societal and environmental context, and demonstrate knowledge of, and need for sustainable development.

8. **Ethics:** Understand and commit to professional ethics and responsibilities and norms of engineering technology practice.

9. **Individual and Team work:** Function effectively as an individual, and as a member or leader in diverse technical teams.

10. **Communication:** Communicate effectively on Broadly-defined engineering activities with the engineering community and with society at large, by being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project Management and Finance:** Demonstrate knowledge and understanding of engineering management principles and apply the same to one's own work, as a member and leader in a team and to manage projects in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the ability to engage in independent and life-long learning in specialized technologies.

Course Objectives:

Develop ability to

- Demonstrate the working of IoT
- Identify the need of cloud computing for IoT
- Apply Machine Learning Algorithms for IoT data
- Predict and visualize output using Data Analytic tools

Course Outcomes (COs):

After completion of the course, student would be able to

- Demonstrate the working of IoT
- Identify the need of cloud computing for IoT
- Apply Machine Learning Algorithms for IoT data
- Predict and visualize output using Data Analytic tools

CO-PO,PSOMapping

(3/2/1indicatesstrengthofcorrelation)3-Strong,2-Medium,1-Weak

COs	ProgrammeOutcomes(POs)												PSOs		
	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12	PSO1	PSO2	PSO3
CO1	1	1	1	1								2	2		
CO2	1	2	2	2	2							1	1		
CO3	1	2	3	2	1							2	2		

2020-21 Onwards (MR-20)	MALLA REDDY ENGINEERING COLLEGE (Autonomous)	B.Tech. IV Semester		
Code: A6903	Internet of Things Fundamentals Lab	L	T	P
Credits: 1.5		-	-	3

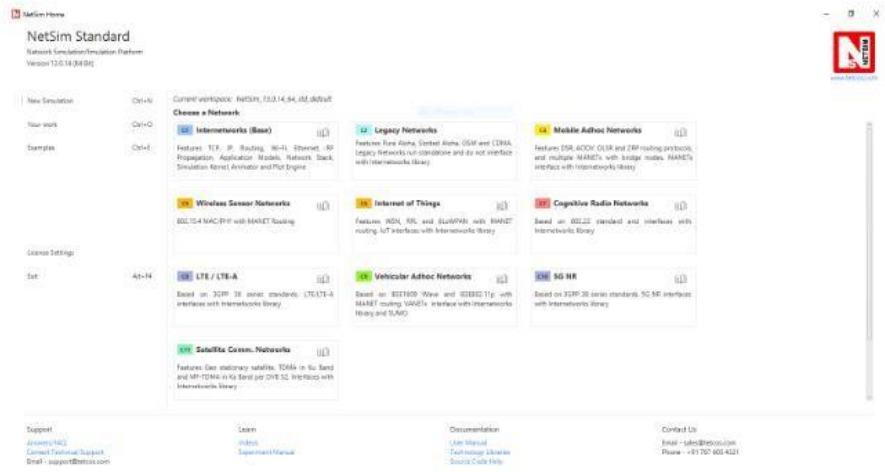
List of Programs:

- Study of IoT simulators.
- Simulate data collection using IoT simulators(IOTIFY/NETSIM)
- Study of Hardware platforms Arduino/Raspberrypi/NodeMCU
- Implement sensor data collection using IoT gateways (Arduino/Raspberrypi/NodeMCU)
- Develop your own Application that stores IoT data in open source IoT cloud platform analytic tools.
- Study of Streaming IoT data into Google cloud platform using Qwik lab environment.
- Write a program to implement the Linear regression for a sample training dataset stored as a .CSV file. Compute the accuracy of the classifier, considering few test datasets.
- Build a decision tree classifier for weather prediction dataset. Compute the accuracy of the classifier, considering few test datasets.
- Develop application for Smart Traffic that analyze the IoT data and predict the Traffic Jam.
- Visualize the predicted output using Data Analytics tool.

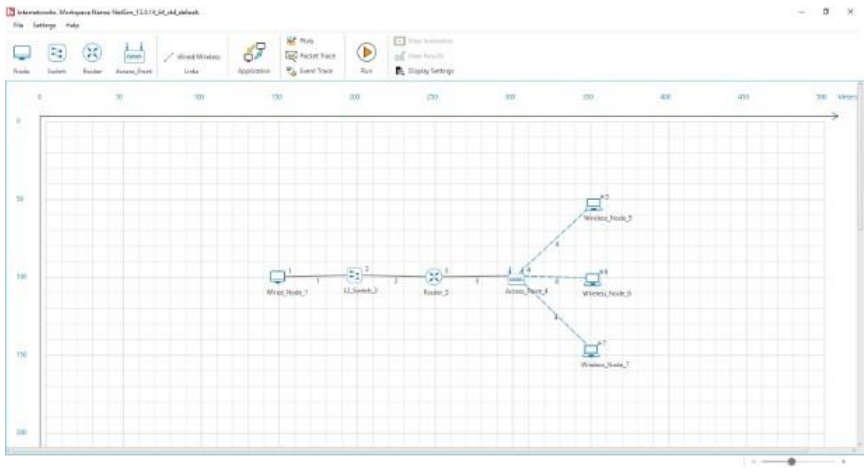
1.Study of IoT simulators.

Introduction to NetSim

NetSim is a network simulation tool that allows you to create network scenarios, model traffic, design protocols and analyze network performance. Users can study the behavior of a network by test combinations of network parameters. NetSim home screen will appear as shown below



Network Design Window: NetSim design window or the GUI see Figure below enables users to model a network comprising of network devices like switches, routers, nodes, etc., connect them through links, and model application traffic to flow through the network. The network devices shown are specific to the network technologies chosen by the user.



Description:

1. File - In order to save the network scenario before or after running the simulation into the current workspace,

Click on File à Save to save the simulation inside the current workspace. Users can specify their own Experiment Name and Description (Optional).

Click on File à Save As to save an already saved simulation in a different name after performing required modifications to it.

Click on Close, to close the design window or GUI. It will take you to the home screen of NetSim.

2. Settings - Go to Settings à Grid/Map Settings and choose the type of environment. Here we have chosen the Grid/Map in the form of a Grid. Map option can be used for specific cases like while designing VANET scenarios.

3. Help - Help option allows the users to access all the help features.

About NetSim – Assists the users with basic information like,

Which version of NetSim is used and whether it is a 32-bit build or 64-bit build?

What kind of License is being used? Whether Floating or Node Locked?

Video Tutorials – Assists the users by directing them to our dedicated YouTube Channel **"TETCOS"**, where we have lots of video presentations ranging from short to long, covering different versions of NetSim up to the latest release.

Answers/FAQ – Assists the user by directing them to our **"NetSim Support Portal"**, where one can find a well-structured **"Knowledge Base"**, consisting of answers or solutions to all the commonest queries which a new user can go through.

Raise a Support Ticket – Assists the user by directing them to our **"NetSim Support Portal"**, where one can **"Submit a ticket"** or in other words raise his/her query, which reaches our dedicated Helpdesk and due support will be provided to the user.

User Manual – Assists the user with the usability of the entire tool and its features. It highly facilitates a new user with lots of key information about NetSim.

Source Code Help – Assists the user with a structured documentation for **"NetSim Source Code Help"**, which helps the users who are doing their R&D using NetSim with a structured code documentation consisting of more than 5000 pages with very much ease of navigation from one part of the document to another.

Open Source Code – Assists the user to open the entire source codes of NetSim protocol libraries in Visual Studio, where one can start initiating the debugging process or performing modifications to existing code or adding new lines of code. Visual Studio Community Edition is a highly recommended IDE to our users who are using the R&D Version of NetSim.

Experiments – Assists the user with separate links provided for 30+ different experiments covering almost all the network technologies present in NetSim.

Technology Libraries – Assists the user by directing them to a folder comprising of individual technology library files comprising all the components present in NetSim.

Below the menu options, the entire region constitutes the Ribbon/Toolbar using which the following actions can be performed:

Click and drop network devices and right click to edit properties

Click on Wired/Wireless links to connect the devices to one another. It automatically detects whether to use a Wired/Wireless link based on the devices we are trying to connect

Click on Application to configure different types of applications and generate traffic

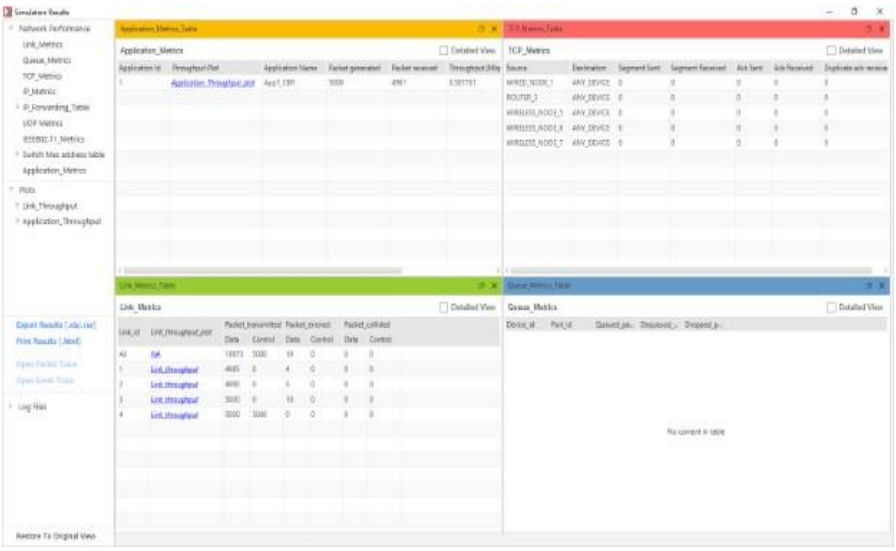
Click on Plots, Packet Trace, and Event Trace and click on the enable check box option which appears in their respective windows to generate additional metrics to further analyze the network performance.

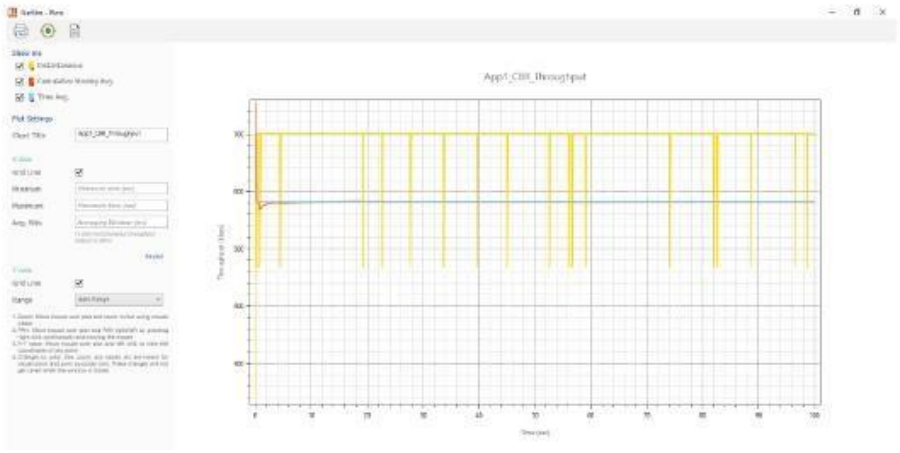
Click on Run to perform the simulation and specify the simulation time in seconds.

Next to Run, we have View Animation and View Results options. Both the options remain hidden before we run the simulation or if the respective windows are already open.

Display Settings option is mainly used to display various parameters like Device Name, IP, etc., to provide a better understanding especially during the design and animation.

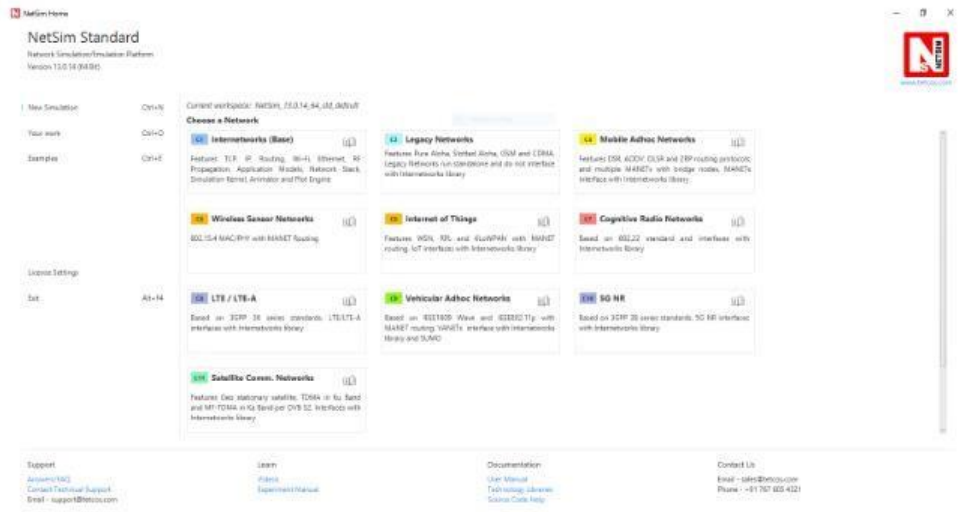
Results Window: Upon completion of simulation, Network statistics or network performance metrics reported in the form of graphs and tables. The report includes metrics like throughput, simulation time, packets generated, packets dropped, collision counts etc.



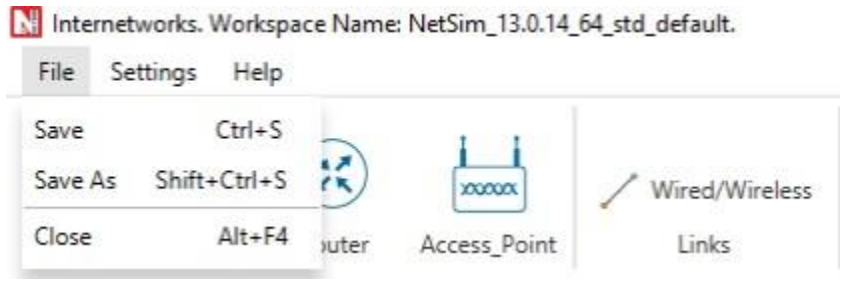


How does a user create and save an experiment in workspace?

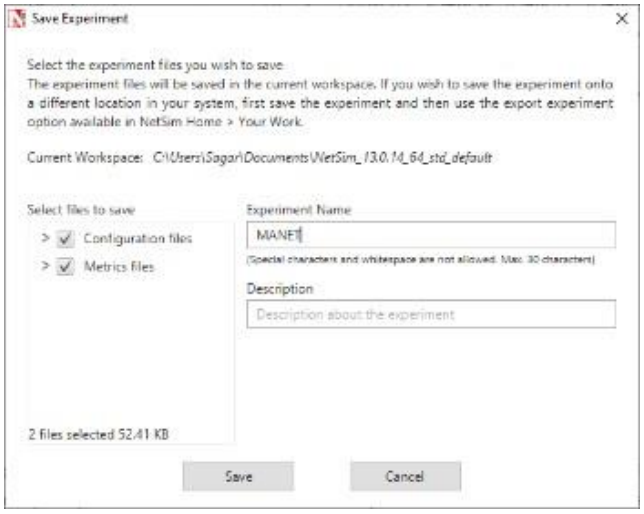
To create an experiment, select New Simulation-> <Any Network> in the NetSim Home.



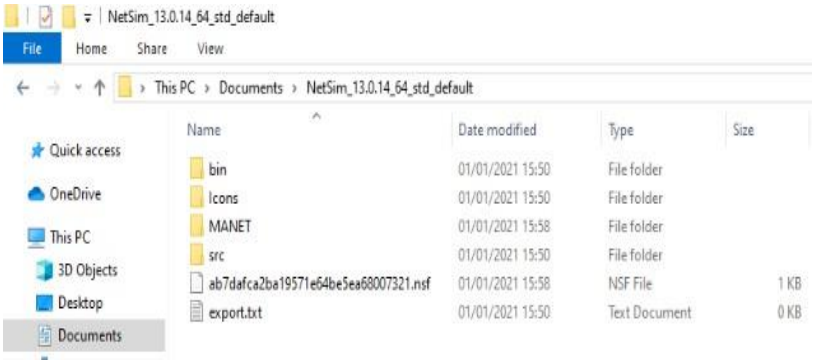
Create a network and save the experiment by clicking on File->Save button on the top left.



A save popup window appears which contains Experiment Name, Folder Name, Workspace path and Description.



Specify the Experiment Name and Description (Optional) and then click on Save. The workspace path is non-editable. Hence all the experiments will be saved in the default workspace path. After specifying the Experiment Name click on Save. In our example we saved with the name MANET and this experiment can be found in the default workspace path.



Users can also see the saved experiments in Your work menu.

Typical sequence of steps to do experiments in this manual:

The typical steps involved in doing experiments in NetSim are

Network Set up: Drag and drop devices, and connect them using wired or wireless links

Configure Properties: Configure device, protocol or link properties by right clicking on the device or link and modifying parameters in the properties window.

Model Traffic: Click on the Application icon present on the ribbon and set traffic flows.

Enable Trace/Plots (optional): Click on packet trace, event trace and Plots to enable. Packet trace logs packet flow, event trace logs each event (NetSim is a discrete event simulator) and the Plots button enables charting of various throughputs over time.

Save/Save As/Open/Edit: Click on File à Save / File à Save As to save the experiments in the current workspace. Saved experiments can then opened from NetSim home screen to run the simulation or to modify the parameters and again run the simulation.

View Animation/View Results: Visualize through the animator to understand working and to analyze results and draw inferences.

NOTE: Example Configuration files for all experiments would available where NetSim has been installed.
This directory is (<NetSim_Install_Directory>\Docs\Sample_Configuration\NetSim_Experiment_Manual)

3.Introduction Getting Started with IoT (Arduino).

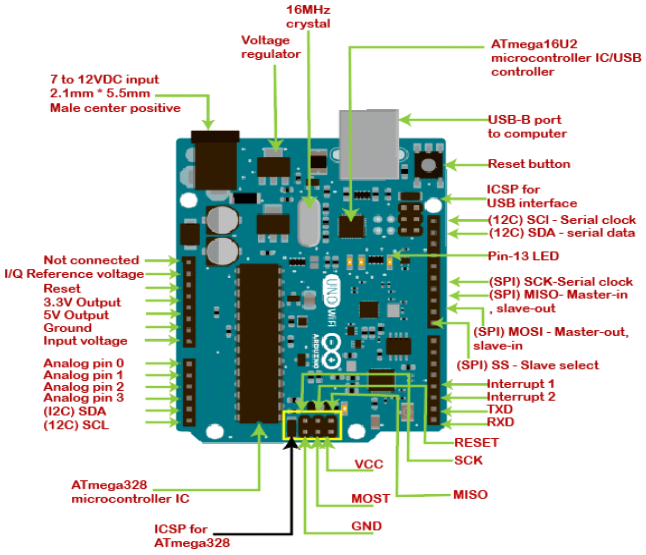
Arduino is a platform that makes it easy for you to build projects using electronics. Arduino can also help you easily build IoT projects in two ways: Using traditional Arduino boards and attaching communication breakout modules (like Bluetooth, WiFi, LoRA, GSM, etc) to them. Arduino is a micro controller that can be connected to one or more sensors and help you capture the data or information and then pass it on to processor.

Features of Aurdino:

- Open source based electronic programmable board(micro controller) and software(IDE).
- Accepts analog and digital signals as input and gives desired output.
- No extra hardware required to load a program into the controller board

Arduino UNO:

Feature	Value
Clock speed	16MHz
Operating Voltage	5V
Digital I/O	14
Analog Input	6
PWM	6
UART	1
interface	USB via ATMega16U2



Arduino IDE:

Arduino IDE is an open source that is used to program the arduino controller board. It is based on variations of the C and C++ programming language. It can be downloaded from arduino's official website and installed into PC.

Download and install the Arduino software (Arduino IDE 1.8.5).

Go to the Arduino website and click the download link to go to the download page. After downloading, locate the downloaded file on the computer and extract the folder from the downloaded file. Copy the folder to a suitable place such as your desktop.

Set Up:

- Power the board by connecting it to a PC via USB cable
- Launch the Arduino IDE
- Set the board type and the port for the board
- Tools-->Board-->select your board
- Tools-->Port-->select your port

Arduino IDE Overview:

- Program coded in Arduino IDE is called a SKETCH
- To create a new sketch
- File-->New
- To open an existing sketch
- File-->open
- There are some basic ready-to-use sketches available in the EXAMPLES section
  File-->Examples-->select any program
- Verify: Checks the code for compilation errors
- Upload: Uploads the final code to the controller board
- New: Creates a new blank sketch with basic structure
- Open: Opens an existing sketch
- Save: Saves the current sketch
- Serial Monitor: Opens the serial console
  All the data printed to the console are displayed here

Sketch Structure:

A sketch can be divided into two parts:

Setup()

Loop()

The function setup() is the point where the code starts, just like the main() function in c and c++ .

I/O variables, pin modes are initialized in the Setup() function. Loop() function, as the name suggests, iterates the specified task in the program.

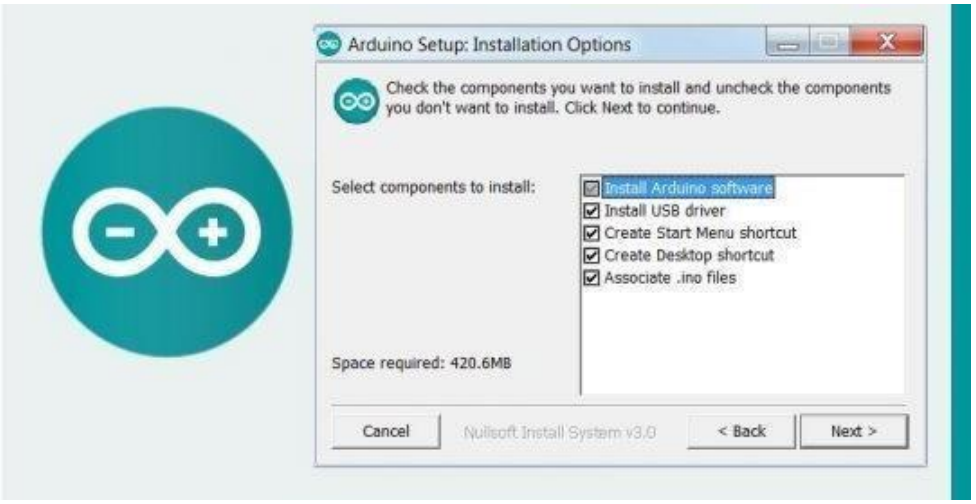
Aim: Familiarization with Arduino/Raspberry Pi and perform necessary software installation

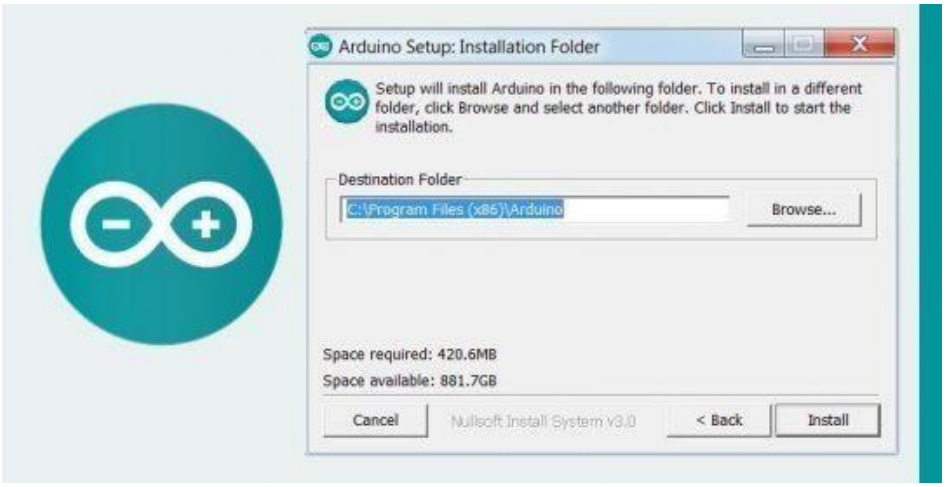
- Arduino is a platform that makes it easy for you to build projects using electronics.
- IoT is a way of using electronics - to make electronic modules talk to each other remotely and wirelessly (often using a Cloud) to solve problems.
- Now, Arduino can also help you easily build IoT projects in two ways: Using traditional Arduino boards and attaching communication breakout modules (like nRF Bluetooth, WiFi, LoRA, GSM, etc) to them.
- Arduino is a micro controller that can be connected to one or more sensors and help you capture the data or information and then pass it on to processor. If you know the full stack of IoT then you should also look at Raspberry.
- RaspPi is a microprocessor so the basic difference between Arduino and RasPi is that RaspPi is controller plus processor and Arduino is just a micro controller.
- They suit the need for different use cases. You can easily read online about this both.

Download and install the Arduino software (Arduino IDE 1.8.5)

- Go to the Arduino website and click the download link to go to the download page.
- After downloading, locate the downloaded file on the computer and extract the folder from the downloaded file. Copy the folder to a suitable place such as your desktop.

Read the Arduino License agreement and click the “I Agree” button.





Demonstrate NodeMCU and its Working

The NodeMCU ESP8266 development board comes with the ESP-12E module containing ESP8266 chip having Tensilica Xtensa 32-bit LX106 RISC microprocessor. This microprocessor supports RTOS and operates at 80MHz to 160 MHz adjustable clock frequency. NodeMCU has 128 KB RAM and 4MB of Flash memory to store data and programs. Its high processing power with in-built Wi-Fi / Bluetooth and Deep Sleep Operating features make it ideal for IoT projects. NodeMCU can be powered using Micro USB jack and VIN pin (External Supply Pin). It supports UART, SPI, and I2C interface. The NodeMCU Development Board can be easily programmed with Arduino IDE since it is easy to use. Programming NodeMCU with the Arduino IDE will hardly take 5-10 minutes. All you need is the Arduino IDE, a USB cable and the NodeMCU board itself.

NodeMCU ESP8266 Specifications & Features

- Microcontroller: Tensilica 32-bit RISC CPU Xtensa LX106
- Operating Voltage: 3.3V
- Input Voltage: 7-12V
- Digital I/O Pins (DIO): 16
- Analog Input Pins (ADC): 1
- UARTs: 1
- SPIs: 1
- I2Cs: 1
- Flash Memory: 4 MB
- SRAM: 64 KB
- Clock Speed: 80 MHz

- USB-TTL based on CP2102 is included onboard, Enabling Plug n Play
- PCB Antenna
- Small Sized module to fit smartly inside your IoT projects
- Power: Micro-USB, 3.3V, GND, Vin

  Micro-USB: NodeMCU can be powered through the USB port

  3.3V: Regulated 3.3V can be supplied to this pin to power the board

  GND: Ground pins

  Vin: External Power Supply
- Control Pins: EN, RST

  The pin and the button resets the microcontroller
- Analog Pin:A0

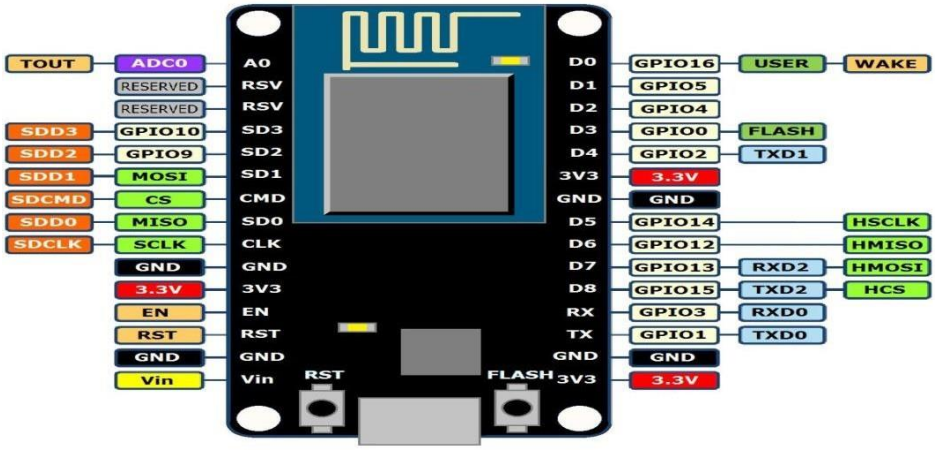
  Used to measure analog voltage in the range of 0-3.3V
- GPIO Pins: GPIO1 to GPIO16

  NodeMCU has 16 general purpose input-output pins on its board
- SPI Pins: SD1, CMD, SD0, CLK

  NodeMCU has four pins available for SPI communication
- UART Pins: TXD0, RXD0, TXD2, RXD2

  NodeMCU has two UART interfaces, UART0 (RXD0 & TXD0) and UART1 (RXD1 & TXD1). UART1 is used to upload the firmware/program
- I2C Pins:

  NodeMCU has I2C functionality support but due to the internal functionality of these pins, you have to find which pin is I2C.



Node MCU/ESP8266:

Step 1: Connect Your NodeMCU to the Computer

Step 2: Install the COM/Serial Port Driver

Step 3: Install the Arduino IDE 1.6.4 or Greater

Step 4: Install the ESP8266 Board Package

Step 5: Setup ESP8266 Support

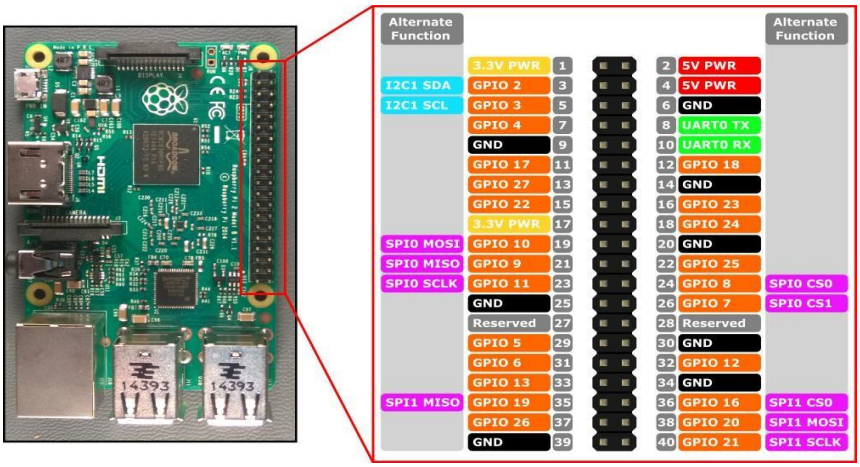
Step 6: Load the sketch

Step 7: Build and Run the code.

Step 8: Final Output

Demonstration of Setup & Working of Raspberry Pi. (Students have to prepare the Report for the same.)

Raspberry Pi is a series of small single board computers (SBCs) developed in the UK by the Raspberry Pi Foundation in association with Broadcom.
The Raspberry Pi project originally leaned towards the promotion of teaching basic computer science in schools and in deploying countries.
The original model became more popular than anticipated, selling outside its target market for uses such as robotics.
It is widely used in many areas, such as for weather monitoring because of its low cost, modularity, and open design. It is typically used by computer and electronic hobbyists, due to its adoption of HDMI and USB devices.
The first generation (**Raspberry Pi Model B**) was released in February 2012, followed by the simpler and cheaper **Model A**.
In 2014, the Foundation released a board with an improved design, **Raspberry Pi Model B+**. These first generation boards feature ARM11 processors, are approximately credit-card sized and represent the standard *mainline* form-factor.
The **Raspberry Pi 2** was released in February 2015 and initially featured a 900 MHz 32-bit quad-core ARM Cortex-A7 processor with 1 GB RAM.
Revision 1.2 featured a 900 MHz 64-bit quad-core ARM Cortex-A53 processor (the same as that in the Raspberry Pi 3 Model B, but underclocked to 900 MHz).
Raspberry Pi 3 Model B was released in February 2016 with a 1.2 GHz 64-bit quad core ARM Cortex-A53 processor, on-board 802.11n Wi-Fi, Bluetooth and USB boot capabilities
On Pi Day 2018, the **Raspberry Pi 3 Model B+** was launched with a faster 1.4 GHz processor, a three-times faster gigabit Ethernet (throughput limited to ca. 300 Mbit/s by the internal USB 2.0 connection), and 2.4 / 5 GHz dual-band 802.11ac Wi-Fi (100 Mbit/s). USB boot and network boot (an SD card is no longer required).
Raspberry Pi 4 Model B was released in June 2019 with a 1.5 GHz 64-bit quad core ARM Cortex-A72 processor, on-board 802.11ac Wi-Fi, Bluetooth 5, full gigabit Ethernet (throughput not limited), two USB 2.0 ports, two USB 3.0 ports, 1–8 GB of RAM, and dual-monitor support via a pair of micro HDMI (HDMI Type D) ports for up to 4K resolution.



Raspberry Pi3:

Make sure the Raspberry Pi board is switched off, motors are not connected and the batteries are not connected.
Make sure your sd card is in the Raspberry Pi securely.
Plug in your wifi dongle to a USB port on the Raspberry Pi.
Connect your Ethernet cable to your computer and to the Raspberry Pi.
Plug in the wall power adapter into the Raspberry Pi, and then plug it into the wall to turn the power on. Once the power is connected to the wall, the Raspberry Pi will be on.

On your computer, open Putty and enter the Host Name as raspberrypi.local and press open. If everything goes according to plan, you'll be prompted with a security prompt. Press "Yes" This will open a terminal and ask for a Username and Password. The username is "pi" and the password is "raspberry". After entering the credentials, you'll get logged on to the Raspberry Pi terminal.

Connecting Raspberry Pi to WiFi:

1 Put the Raspberry Pi OS SD card into your computer

2 Navigate to the boot directory

3 Add your wpa_supplicant.conf file

```
country=US # Your 2-digit country code
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
network={
   ssid="YOUR_NETWORK_NAME"
   psk="YOUR_PASSWORD"
   key_mgmt=WPA-PSK
}
```

Connecting to unsecured networks

```
country=US # Your 2-digit country code
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev # Include this line for Stretch
network={
   ssid="YOUR_NETWORK_NAME"
   key_mgmt=NONE
}
```

4 Put your SD card in the Raspberry Pi, boot, and connect

5 Troubleshooting

If your Pi hasn't connected to Wi-Fi, try these wpa_supplicant troubleshooting tips:
Double-check that the file was written in plaintext, without any special characters.
Double-check that the file *has* disappeared from your boot directory.
Connect the Pi to a TV or monitor via HDMI to ensure it is booting normally.
If you're using a Raspberry Pi Zero W, make sure you're attempting to connect to a 2.4GHz network (the Zero doesn't support 5G).
If you're using a Raspberry Pi Zero, make sure it's a Raspberry Pi Zero W, not a regular Zero (only the W supports Wi-Fi and Bluetooth).

4. Implement sensor data collection using IoT gateways (Arduino/Raspberrypi/NodeMCU)

Code :

```
#include <DHT.h>
#include <ESP8266WiFi.h>
String apiKey = "APIKEY";
const char *ssid =  "Username";
const char *pass =  "Mrec@123";
const char* server = "api.thingspeak.com";
#define DHTPIN 0
DHT dht(DHTPIN, DHT11);
WiFiClient client;
void setup()
{
      Serial.begin(115200);
      delay(10);
      dht.begin();
      Serial.println("Connecting to ");
      Serial.println(ssid);
      WiFi.begin(ssid, pass);
     while (WiFi.status() != WL_CONNE


    {     delay(500);
         Serial.print(".");
    }
     Serial.println("");
     Serial.println("WiFi connected");
}
void loop()
{
     float h = dht.readHumidity();
     float t = dht.readTemperature();
           if (isnan(h) || isnan(t))
              {
                 Serial.println("Failed to read from DHT sensor!");
                  return;
              }
          if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
                {
                     String postStr = apiKey;
                      postStr +="&field1=";
                      postStr += String(t);
                      postStr +="&field2=";
                      postStr += String(h);
                      postStr += "\r\n\r\n";
                      client.print("POST /update HTTP/1.1\n");
                      client.print("Host: api.thingspeak.com\n");
                      client.print("Connection: close\n");
                      client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
                      client.print("Content-Type: application/x-www-form-urlencoded\n");
                      client.print("Content-Length: ");
                      client.print(postStr.length());
```

```
                        client.print("\n\n");
                        client.print(postStr);
                        Serial.print("Temperature: ");
                        Serial.print(t);
                        Serial.print(" degrees Celcius, Humidity: ");
                        Serial.print(h);
                        Serial.println("%. Send to Thingspeak.");
                    }
            client.stop();
             Serial.println("Waiting...");
          // thingspeak needs minimum 15 sec delay between updates
          delay(1000);
}
```

5. Develop your own Application that stores IoT data in open source IoT cloud platform analytic tools

Code :

```
#include <DHT.h>
#include <ESP8266WiFi.h>
String apiKey = "APIKEY";
const char *ssid =  "Username";
const char *pass =  "Mrec@123";
const char* server = "api.thingspeak.com";
#define DHTPIN 0
DHT dht(DHTPIN, DHT11);
WiFiClient client;
void setup()
{
      Serial.begin(115200);
      delay(10);
      dht.begin();
      Serial.println("Connecting to ");
      Serial.println(ssid);
      WiFi.begin(ssid, pass);
     while (WiFi.status() != WL_CONNECTED)
    {      delay(500);
          Serial.print(".");
    }
     Serial.println("");
     Serial.println("WiFi connected");
}
void loop()
{
     float h = dht.readHumidity();
     float t = dht.readTemperature();
           if (isnan(h) || isnan(t))
              {
                 Serial.println("Failed to read from DHT sensor!");
                  return;
              }
          if (client.connect(server,80))   //   "184.106.153.149" or api.thingspeak.com
                {
                     String postStr = apiKey;
                      postStr +="&field1=";
                      postStr += String(t);
                      postStr +="&field2=";
                      postStr += String(h);
                      postStr += "\r\n\r\n";
                      client.print("POST /update HTTP/1.1\n");
                      client.print("Host: api.thingspeak.com\n");
                      client.print("Connection: close\n");
                      client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
                      client.print("Content-Type: application/x-www-form-urlencoded\n");
                      client.print("Content-Length: ");
                      client.print(postStr.length());
                      client.print("\n\n");
                      client.print(postStr);
```

```
                    Serial.print("Temperature: ");
                    Serial.print(t);
                    Serial.print(" degrees Celcius, Humidity: ");
                    Serial.print(h);
                    Serial.println("%. Send to Thingspeak.");
                }
      client.stop();
       Serial.println("Waiting...");
  // thingspeak needs minimum 15 sec delay between updates
delay(1000)
```

6.Study of Streaming IoT data into Google cloud platform using Qwiklab environment.

In this lab, you learn how to perform the following tasks:

- Create Cloud Pub/Sub topics and subscriptions
- Use IoT Core to create a registry
- Start the MQTT Application on a simulator
- Stream data to Cloud Storage
- Use Dataprep to manipulate the data

Setup and Requirements:

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).

Note: Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

- Time to complete the lab---remember, once you start, you cannot pause a lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab to avoid extra charges to your account.

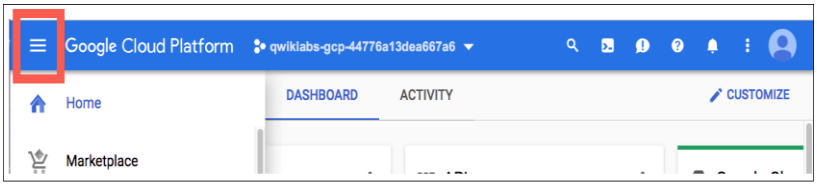
How to start your lab and sign in to the Google Cloud Console

1. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method.
   On the left is the **Lab Details** panel with the following:
   - The **Open Google Console** button
   - Time remaining
   - The temporary credentials that you must use for this lab
   - Other information, if needed, to step through this lab

   Click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.
   Tip: Arrange the tabs in separate windows, side-by-side.
   Note: If you see the **Choose an account** dialog, click **Use Another Account**.
   If necessary, copy the **Username** from the **Lab Details** panel and paste it into the **Sign in** dialog. Click **Next**.
   Copy the **Password** from the **Lab Details** panel and paste it into the **Welcome** dialog. Click **Next**.
   Important: You must use the credentials from the left panel. Do not use your Google Cloud Skills Boost credentials.**Note:** Using your own Google Cloud account for this lab may incur extra charges.
   Click through the subsequent pages:
   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top- left.



Ensure that the Dataflow API is successfully enabled

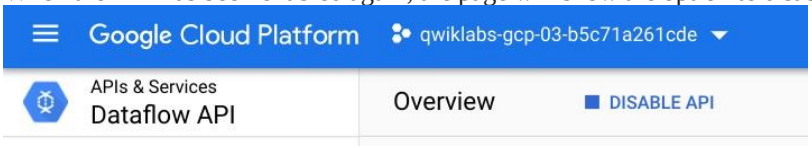
To ensure access to the necessary API, restart the connection to the **Dataflow API**.

1. In the Cloud Console, enter "Dataflow API" in the top search bar. Click on the result for **Dataflow API**.
2. Click **Manage**.
3. Click **Disable API**.

If asked to confirm, click **Disable**.

4. Click **Enable**.

When the API has been enabled again, the page will show the option to disable.



Cloud Pub/Sub setup and topics

Create a pub/sub topic for your streaming data.

1. On the **Navigation menu**, click **Pub/Sub** > **Topics**.

In this lab, you learn how to perform the following tasks:

- Create Cloud Pub/Sub topics and subscriptions
- Use IoT Core to create a registry
- Start the MQTT Application on a simulator
- Stream data to Cloud Storage
- Use Dataprep to manipulate the data

Setup and Requirements:

Before you click the Start Lab button

Read these instructions. Labs are timed and you cannot pause them. The timer, which starts when you click **Start Lab**, shows how long Google Cloud resources will be made available to you.

This hands-on lab lets you do the lab activities yourself in a real cloud environment, not in a simulation or demo environment. It does so by giving you new, temporary credentials that you use to sign in and access Google Cloud for the duration of the lab.

To complete this lab, you need:

- Access to a standard internet browser (Chrome browser recommended).

Note: Use an Incognito or private browser window to run this lab. This prevents any conflicts between your personal account and the Student account, which may cause extra charges incurred to your personal account.

- Time to complete the lab---remember, once you start, you cannot pause a lab.

Note: If you already have your own personal Google Cloud account or project, do not use it for this lab to avoid extra charges to your account.

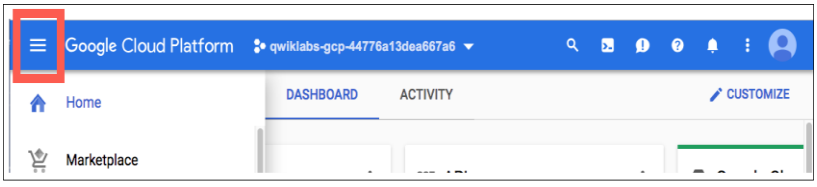
How to start your lab and sign in to the Google Cloud Console

2. Click the **Start Lab** button. If you need to pay for the lab, a pop-up opens for you to select your payment method.
   On the left is the **Lab Details** panel with the following:
   - The **Open Google Console** button
   - Time remaining
   - The temporary credentials that you must use for this lab
   - Other information, if needed, to step through this lab

   Click **Open Google Console**. The lab spins up resources, and then opens another tab that shows the **Sign in** page.
   Tip: Arrange the tabs in separate windows, side-by-side.
   Note: If you see the **Choose an account** dialog, click **Use Another Account**.
   If necessary, copy the **Username** from the **Lab Details** panel and paste it into the **Sign in** dialog. Click **Next**.
   Copy the **Password** from the **Lab Details** panel and paste it into the **Welcome** dialog. Click **Next**.
   Important: You must use the credentials from the left panel. Do not use your Google Cloud Skills Boost credentials.**Note:** Using your own Google Cloud account for this lab may incur extra charges.
   Click through the subsequent pages:
   - Accept the terms and conditions.
   - Do not add recovery options or two-factor authentication (because this is a temporary account).
   - Do not sign up for free trials.

After a few moments, the Cloud Console opens in this tab.

Note: You can view the menu with a list of Google Cloud Products and Services by clicking the **Navigation menu** at the top- left.



Ensure that the Dataflow API is successfully enabled

To ensure access to the necessary API, restart the connection to the **Dataflow API**.

5. In the Cloud Console, enter "Dataflow API" in the top search bar. Click on the result for **Dataflow API**.
6. Click **Manage**.
7. Click **Disable API**.

If asked to confirm, click **Disable**.

8. Click **Enable**.

When the API has been enabled again, the page will show the option to disable.

Cloud Pub/Sub setup and topics

Create a pub/sub topic for your streaming data.

1. On the **Navigation menu**, click **Pub/Sub** > **Topics**.
2. Click CREATE TOPIC
3. Type **iotlab** in Topic ID field.
4. Click **CREATE TOPIC**.

Setting topic permissions

You now have a pub/sub topic. To allow the project to publish this topic, add the project as a member/publisher.

1. To add members, click **ADD PRINCIPALS**.
2. Add the project as a member to the topic cloud-iot@system.gserviceaccount.com
3. Select the role of **Pub/Sub** > **Pub/Sub Publisher**, and then click **SAVE** to add the member.

Click **Check my progress** to verify the objective.

Create a pub/sub topic

Revisar mi progreso

Create a location for data storage

You need to create a storage folder to store the data streaming from the virtual device.

Create a storage bucket

1. On the **Navigation menu** click **Cloud Storage** > **Browser**.
2. Click **CREATE**.
3. Enter a unique bucket name for your bucket, then click **CREATE**.

Click **Check my progress** to verify the objective.

Create a Cloud Storage bucket.

Revisar mi progreso

Create a folder in your bucket

1. Click **CREATE FOLDER** in the bucket you just created.
2. For folder name, type **Sensor-Data**, and then click **CREATE**.

Click **Check my progress** to verify the objective.

Create Sensor-Data folder in Cloud Storage bucket

Revisar mi progreso

Start a Dataflow job

You now have a device publishing data, and your Google Cloud Project is authorized to receive this data. Now you can start a Dataflow job to save the data to your bucket.

Create a Dataflow job from a template

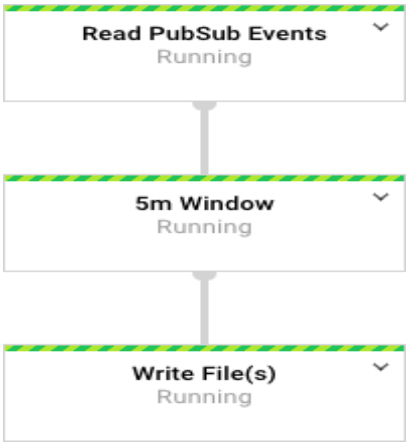
1. On the **Navigation menu** click **Dataflow**.
2. Click **CREATE JOB FROM TEMPLATE** at the top of the screen.
3. Enter the following values in the template.

Property	**Value (type value or select option as specified)**
Job name	sensor-data
Regional endpoint	<Region>
Dataflow template	Pub/Sub to Text Files on Cloud Storage

4. The template page will expand to display a series of textboxes. Some of the textboxes are optional and some are required. You will only modify the required textboxes. Be sure to replace <project-id> with your Google Cloud project ID and <bucket-name> with the name of the bucket you created.

Property	**Value (type value or select option as specified)**
Input Pub/Sub topic	projects/project-id/topics/iotlab
Output file directory in Cloud Storage	gs://bucket-name/Sensor-Data/ (note the slash at the end of the input text)
Output filename prefix	output-
Temporary Location	gs://bucket-name/tmp

5. Click **RUN JOB**. A Dataflow job will be kicked off—your console should now resemble the following:



Click **Check my progress** to verify the objective.
Set up a Cloud Dataflow Pipeline
Revisar mi progreso
Prepare your VM
In your project a pre-provisioned VM instance named **iot-device-simulator** will let you run instances of a Python script that emulate an MQTT-connected IoT device. Before you emulate the devices, you will also use this VM instance to populate your Cloud IoT Core device registry.
Connect to the iot-device-simulator VM instance

1. In the Cloud Console, go to **Navigation menu** > **Compute Engine** > **VM Instances**. You'll see your VM instance listed as **iot-device-simulator**.
2. To the right, click the **SSH** drop-down arrow and select **Open in browser window**.
3. In your SSH session, run the following command to initialize gcloud and login with your student account:

```
gcloud auth login --no-launch-browser
```

Se copió correctamente
content_copy

4. When you are asked "Do you want to continue (Y/n)?" enter **Y**.
5. Copy the URL output and paste it in a new browser tab and hit **Enter**.
6. Select the account you logged into this lab with, then click **Allow**.
7. Copy the verification code and return to the SSH window where you last left off (There will be a prompt to "Enter authorization code:"). Paste in the verification code and press **Enter**.
8. Enter this command to update the system's information about Debian Linux package repositories:

```
sudo apt-get update
```

Se copió correctamente
content_copy

9. Enter this command to make sure that various required software packages are installed:

```
sudo apt-get install python3-pip openssl git -y
```

Se copió correctamente
content_copy

10. Use **pip** to add needed Python components:

```
sudo pip3 install pyjwt paho-mqtt cryptography
```

Se copió correctamente
content_copy

11. Enter this command to add data to analyze during this lab:

```
git clone https://github.com/cagamboa123/training-data-analyst.git
```

Se copió correctamente
content_copy

12. Now run the following command to set your Project ID as an environment variable, replacing <YOUR_PROJECT_ID> with your Qwiklabs Project ID:

```
export PROJECT_ID=<YOUR_PROJECT_ID>
```

Se copió correctamente
content_copy

13. You must choose a region for your IoT registry. At the present time, these regions are supported:

- us-central1
- europe-west1
- asia-east1

To set an environment variable containing your preferred region, enter the following command replacing <YOUR_REGION> with the region nearest to you:

```
export MY_REGION=<YOUR_REGION>
```
Se copió correctamente
content_copy
Leave this session open, you will return to it shortly.

Open IoT Core

1. Back in the Console, open the **Navigation menu** and click **IoT Core**.
2. Click **CREATE REGISTRY**.
3. On the **Create a registry** page, specify the following and leave the remaining settings as their defaults:

Field	Value
Registry ID	iotlab-registry
Region	The region closest to you
Select a Cloud Pub/Sub topic	projects/project-id/topics/iotlab

4. Click **CREATE**.

Click **Check my progress** to verify the objective.
Create the device registry
Revisar mi progreso
Create a Cryptographic KeyPair
To allow IoT devices to connect securely to Cloud IoT Core, you must create a cryptographic KeyPair.
Return to your SSH session in the **iot-device-simulator** VM instance and enter in the following commands to create a key-pair in the appropriate directory:

```
cd $HOME/training-data-analyst/quests/iotlab/
openssl req -x509 -newkey rsa:2048 -keyout rsa_private.pem \
  -nodes -out rsa_cert.pem -subj "/CN=unused"
```
Se copió correctamente
content_copy
This openssl command creates an RSA cryptographic keypair and writes it to a file called rsa_private.pem.

Create the device and add it to the registry

1. In your SSH session in the **iot-device-simulator** VM instance, type:

```
cat rsa_cert.pem
```
Se copió correctamente
content_copy

2. Select and copy the entire certificate. Including all dashes at the beginning and end of the certificate.

Output example (do not copy)

```
-----BEGIN CERTIFICATE-----
MIIC+DCCAeCgAwIBAgIJAOJikTScq9oPMA0GCSqGSIb3DQEBCwUAMBExDzANBgNV
BAMMBnVudXNlZDAeFw0xODA4MTMxNjQ2MTNaFw0xODA5MTIxNjQ2MTNaMBExDzAN
zANBgNVBAMMBnVudXNlZDCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggE
BAL+wLyITE5Tjlh50I63ew3HdvoGty2aOpP04nMyOYZoooAw5o2rj5mkNb/hbkoMTkzo
6/5Jo0zgDYPVRpz2nGAhTfeQzPuvOfPZe7KPpZxYvmSN3pYT9kkiVo9pXwynG7q8kW
72Q9f0pffXS/VElPrC63Y9kcAgOyveZVX61qSokz4DVIj0Z6+1b1utxe2TnxR1q3Hce289
1re6qnxYp6Yuw0gVYtn8HdgEKKMqeSozqJP7dq8EvNkwY8BAUFU2NmuvwK2Z6hB1E
u0DImyhtKRxZ4pUbWuefC+P6GU2fB3rp4pR9Lc7xd5BuWXHgR6f0lV57elL9f1Q/iXippP
8RjhMCAwEAAaNTMFEwHQYDVR0OBBYEFF7808W+vP7vbgg6cS5Fky9xCstNMB8GA
1UdIwQYMBaAFF7808W+vP7vbgg6cS5Fky9xCstNMA8GA1UdEwEB/wQFMAMBAf8w
DQYJKoZIhvcNAQELBQADggEBAD9mSbWQRz8QHI947gGSMrsA+aO4dgWIujkypFw/p
7gSefleCCwGV4Wpfq6zoIjru9bnciWRLHZMKVbhptBDseyBnoPXxnJMgVYBAVzRRMhT
qPeo146Pv99dn3c310M2tkpQeQzP/wE9XFVqEud2sZCKXgXtydIsyTEX3wmG9s9m7f
6TJDknvJltOj1R7m+xO6GHPebK29x/r+LzPuYjIDYoG+mxLQUltDOM3v8QwZ4bneo+HI
BZX6FOBRb+x/fEE3EANCY3J5sKwRCxxXJ6l/Mts7aLUE6MrT8BM0n1fxnY7BX+6dvsJ
H/OeONG2tk3Y0ci/ly245NQyurqa3x35Ws=
-----END CERTIFICATE-----
```

3. Go back to the IoT Core tab click on **Devices**, then click **CREATE A DEVICE**.
4. In the **Device ID** field, enter **temp-sensor-buenos-aires**.
5. Expand **COMMUNICATION, CLOUD LOGGING, AUTHENTICATION** and in the Authentication section, set the fields as following.

Field	Value
Input method	Enter manually
Public key format	RS256_X509
Public key value	Paste the certificate number you copied

6. Click **CREATE**.
7. Now add a second device. Use the back arrow in IoT Core to return to the Devices details page.

8. Click **CREATE A DEVICE**.
9. In the **Device ID** field, enter **temp-sensor-istanbul**.
10. Expand **COMMUNICATION, CLOUD LOGGING, AUTHENTICATION** and in the Authentication section, set the fields as following.

Field	Value
Input method	Enter manually
Public key format	RS256_X509
Public key value	Paste the certificate number you copied

11. Click **CREATE**.

Click **Check my progress** to verify the objective.
Create the device and add it to the registry
Revisar mi progreso

Run simulated devices

1. In your SSH session on the **iot-device-simulator** VM instance, enter these commands to download the CA root certificates from pki.google.com to the appropriate directory:

```
cd $HOME/training-data-analyst/quests/iotlab/
wget https://pki.google.com/roots.pem
```

Se copió correctamente
content_copy

2. Enter this command to run the first simulated device:

```
python3 cloudiot_mqtt_example_json.py \
  --project_id=$PROJECT_ID \
  --cloud_region=$MY_REGION \
  --registry_id=iotlab-registry \
  --device_id=temp-sensor-buenos-aires \
  --private_key_file=rsa_private.pem \
  --message_type=event \
  --algorithm=RS256 --num_messages=1000 > buenos-aires-log.txt 2>&1 &
```

Se copió correctamente
content_copy
It will continue to run in the background.

3. Enter this command to run the second simulated device:

```
python3 cloudiot_mqtt_example_json.py \
  --project_id=$PROJECT_ID \
  --cloud_region=$MY_REGION \
  --registry_id=iotlab-registry \
  --device_id=temp-sensor-istanbul \
  --private_key_file=rsa_private.pem \
  --message_type=event \
  --algorithm=RS256 --num_messages=1000
```

Se copió correctamente
content_copy
Telemetry data will flow from the simulated devices through Cloud IoT Core to your Cloud Pub/Sub topic. In turn, your Dataflow job will read messages from your Pub/Sub topic and write their contents to your BigQuery table.

Examine the stored data

Dataflow is collecting the data published by Pub/Sub and saving it in output files in the bucket and folder specified in the job template. The files are written every 5 minutes, and each begins with the prefix specified in the job template.

1. Back in the Console, open the **Navigation menu** and click **Cloud Storage**.
2. Click on the bucket you created.
3. Select the folder **Sensor-Data**. Dataflow is writing the data from the device to this folder. It writes a file every few minutes. If the folder is empty, wait a few minutes then click the "Refresh" link at the top of the Storage Browser page.
4. Open a file by clicking on its name. Click on the **Authenticated URL** link.

Your file contents should be similar to what is shown below.

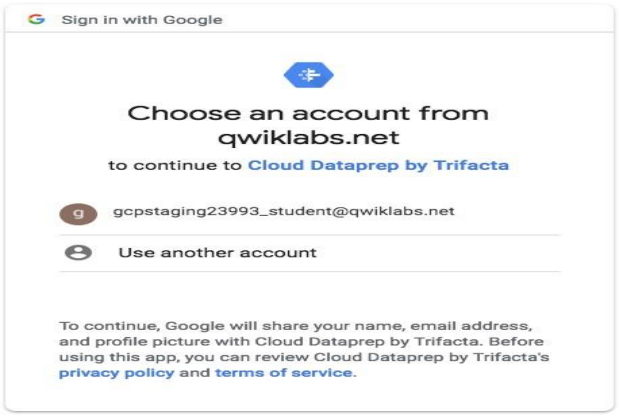
Output (do not copy)

Publishing message 51 of 1000: '{'device': 'temp-sensor-istanbul', 'timestamp': 'Mon Aug 13 22:12:28 2018', 'temperature': 23.215100358910885}'on_publishPublishing message 52 of 1000: '{'device': 'temp-sensor-istanbul', 'timestamp': 'Mon Aug 13 22:12:29 2018', 'temperature': 23.22890311742878}'on_publishPublishing message 53 of 1000: '{'device': 'temp-sensor-istanbul', 'timestamp': 'Mon Aug 13 22:12:30 2018', 'temperature': 23.237443887313777}'on_publishPublishing

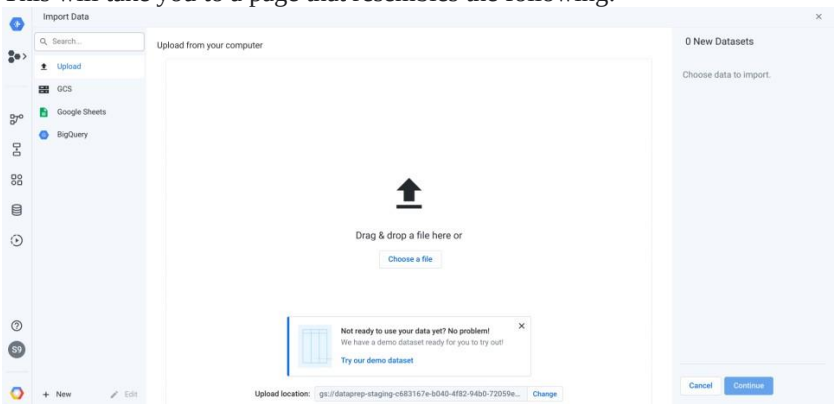
message 54 of 1000: '{'device': 'temp-sensor-istanbul', 'timestamp': 'Mon Aug 13 22:12:31 2018', 'temperature': 23.24313271883122}'

Create a Dataprep Flow

In this section you will create a Dataprep Flow. Open the **Navigation menu** and click on the **Dataprep** service.
Accept the "Google Dataprep Terms of Service". You will be prompted to share account information with Trifacta—check the box and click **Agree and Continue**. Then click **Allow** to let Trifacta access your project data.
You will be brought to a sign in page that looks similar to the following:



Click your Qwiklabs Google Cloud Username and then **Allow**. **Accept** the terms of service that follow.
Your storage location should be selected. Click **Continue** to finish setup.
You should now be on the Dataprep home page.
Click the **Import Data** button on the top-right part of the page.
This will take you to a page that resembles the following:



You are now in the Cloud Dataprep UI. You will now import the dataset from your Cloud Storage bucket.

1. On the left hand side, click **Cloud Storage**.
2. Click the name of your bucket, then click **Sensor-Data/**.
3. Click the **+** next to the files in the folder (should be ~2. Go back to Console and **Refresh** the bucket if you only see 1).
4. Check the **Add to new flow** box in the bottom-right corner.
5. Then click **Continue**. Name the flow "Sensor-Data" and click **OK**.

Read and step through the information panels that define a Dataprep flow.
Your Dataprep page should now resemble the following:

Create a Dataprep recipe

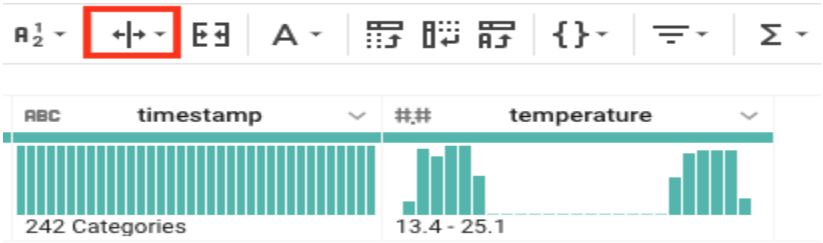
In this section you will create a Dataprep *recipe*. A recipe is a list of transformations and modifications of the data and data set. It will be applied to all the data presently in the dataset, as well as any new data.

1. Click on the first dataset and then click **Add** > **Recipe** in the right-hand side of the console.

2. Then click **Edit Recipe**.

Split a column

1. Read and step through the information panels that define a Transformer. Click the **Split Column** icon from the toolbar:



2. Click **On delimiter**
3. Click in the Column field, select **device**.
4. In the delimiter section type in 'temp-sensor-'
5. Click **Add**.

You should have a column called **device2** with locations of the devices listed (buenos-aires and istanbul).

Delete column

1. Click the expansion arrow on the column titled **device1**.
2. Click **Delete**.

Rename column

1. Click the expansion arrow on the column titled **device2**.
2. Select **Rename**, and in the New name field, enter 'Device location'.
3. Click **Add**.
4. The "device2" column should now be renamed to "Device location":

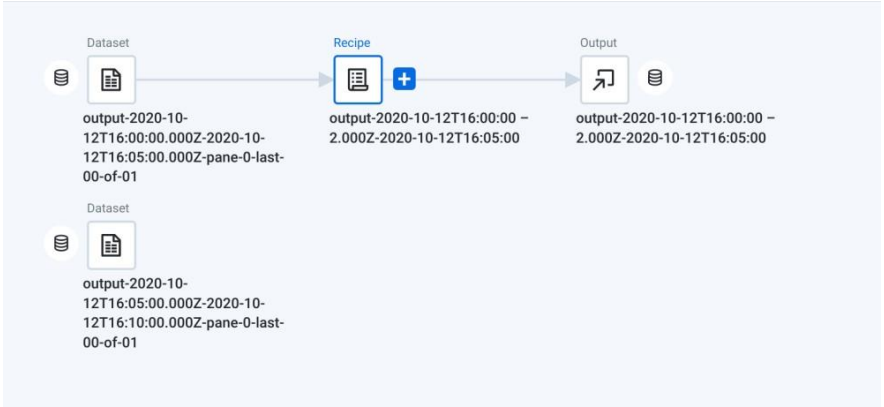
ABC Device location	ABC timestamp	#,# temperature
2 Categories	242 Categories	13.4 - 25.1
buenos-aires	Wed Sep 26 00:15:59 2018	15.79977724801916
buenos-aires	Wed Sep 26 00:16:01 2018	15.775931984925396
buenos-aires	Wed Sep 26 00:16:00 2018	15.787863403464552
buenos-aires	Wed Sep 26 00:15:58 2018	15.80888889949552
buenos-aires	Wed Sep 26 00:16:02 2018	15.757599846660394
istanbul	Wed Sep 26 00:16:03 2018	22.686660588654316
buenos-aires	Wed Sep 26 00:16:03 2018	15.752148123756156

Round values in a column

1. Click the expansion arrow on the column titled **temperature**.
2. Click **Calculate** > **Round** > **Round**.

You now have a recipe for handling data coming from Cloud Storage.

3. Click **Run** in the far right corner, then **Run** again in the lower right corner. This can take several minutes to complete. Watch the job status in the right-hand column.
4. When the job is done, click the **Sensor-Data** link back to your Flow under the **Job Summary** panel.
5. **Double click** the recipe icon in the flow and examine the data:

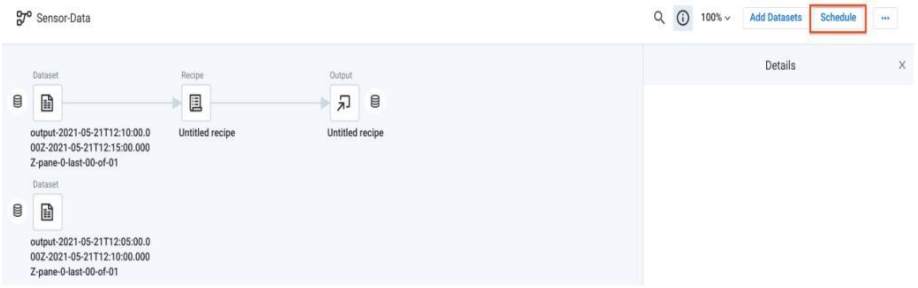


Schedule a Dataprep job

You now have a Dataprep dataset and a recipe for transforming the data. As you may have noticed, your devices are continuing to publish data and store it in Cloud Storage. You will now schedule a flow to download new data and execute the recipe.

1. Click on the Dataprep icon in the upper left corner:

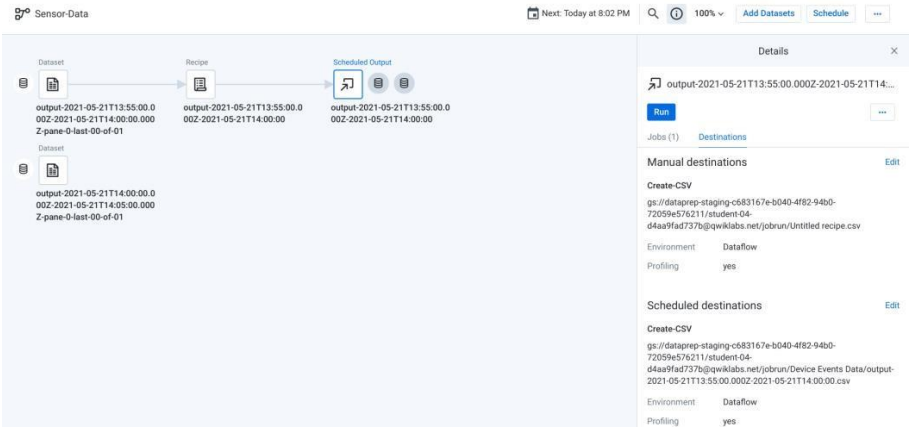
2. Click on **Sensor-Data**.
3. At the top of the Flow page, click **Schedule**.



4. Specify the following:

Field	**Value**
Timezone	Select your timezone
Frequency	**hourly**
Minutes past the hour	Specify 2 minutes from your present time

5. Click **Save**. You will see a message that says 'No scheduled destinations set...'. You will set the destination in the following steps.
6. Select the **Output** icon for your recipe:

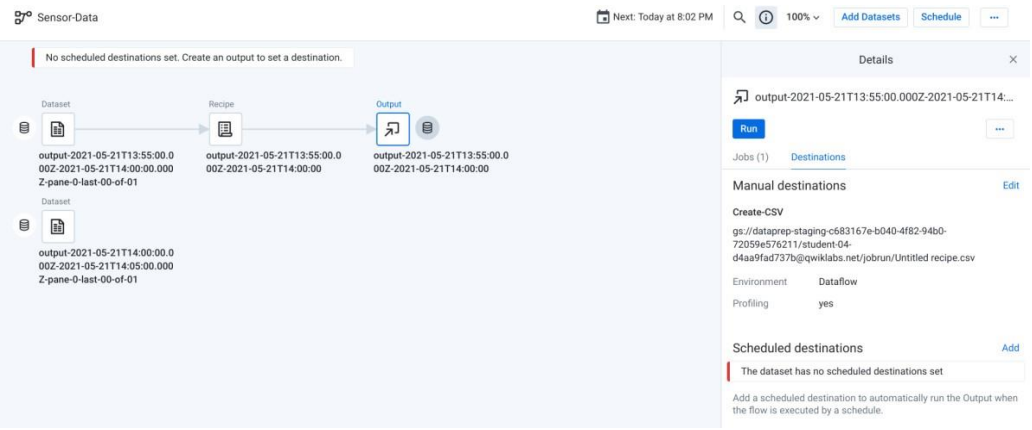


7. In the **Details** panel, click the **Scheduled Settings** tab.
8. Under **Publishing Actions**, click **Add Action**.
9. Ensure **Cloud Storage** is selected on the left hand panel.
10. Click **Create Folder**.
11. Specify the following:

Folder Name	**Device Events Data**

12. Click **Create Folder**.
13. Click **Create a new file**.
14. Click **Add**.
15. Click **Save Settings**.

Your flow should look like the image below. Note the calendar icon, this signifies the flow is scheduled.



Examine the data

You have created a Dataprep flow and you have created a schedule for it to be regularly updated. Examine the data after a scheduled job execution.

1. Click the **Job history** icon from the left-hand menu:

You can run the job manually. If you run it manually, you won't see the auto update run.

To run a job manually, do the following:

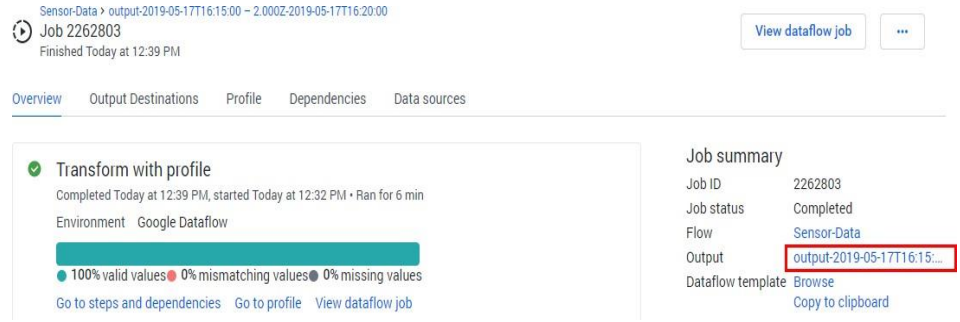
1. Click on the **Flows** page. Then click **Sensor-Data**.
2. Click on the second recipe icon and then click **Add** > **Recipe**.
3. Click on **Edit Recipe.**
4. Click on **Run**(upper right corner).
5. Click on **Run**(bottom right corner).
6. Click on the **Job history** icon from the left-hand menu.
2. The Jobs page lists the jobs that have been completed. Click on the **job number** as soon as it is completed. This time the job will take a little longer because it has more data to process.

Once you click on the finished job, the metadata for the dataset is shown. Moving your cursor over the bar graphs will give you details about each data point.

Dataprep assumes the dataset is large, therefore it does not show the entire dataset. The default is to show initial values. There are cases where this may not be acceptable.

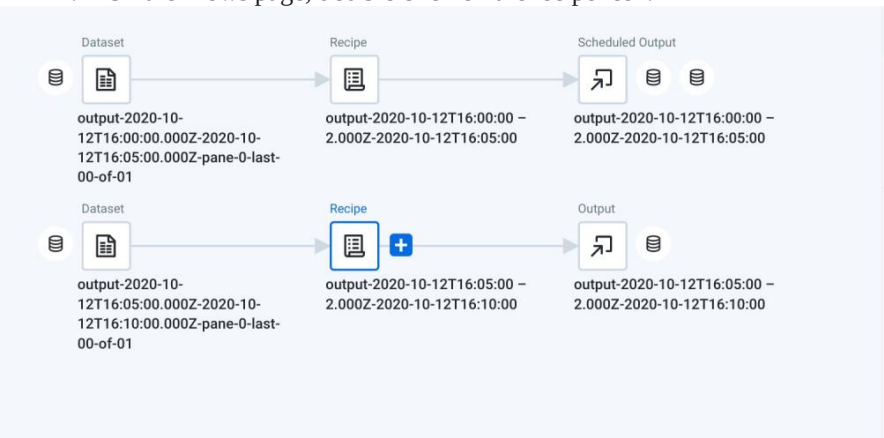
Change dataset sample

1. Find the dataset name in the information bar at the top of the page. Click on the dataset name.



Your dataset output name will likely differ from the pictures.

2. On the Flows page, **double click** on the recipe icon.



3. In the upper right corner, click the **Samples** icon:

4. Click **GOT IT!** when the panel opens up in the right-hand column. This will display all the samples available.
5. Click **Random**.
6. Select **Full** to pull samples from the entire dataset.

7. Click **Collect**.
8. Click on the Dataflow symbol next to the progress bar (it might take a few seconds for this to appear; if taking too long, refresh your browser):

Go back to the Cloud Console. The Dataflow page shows the job as running. The time to complete the job depends on how much data has accumulated in the **Sensor-Data** folder, but should only take a couple of minutes.

9. Click **Navigation menu** > **Cloud Storage** > **Browser**.
10. Click on the bucket you created.
11. Click **Sensor-Data**.

You should now see a list of files, each written five minutes apart. These files contain all the data published by the devices:

Name	Size	Type	Storage class	Last modified
output-2018-09-26T00:15:00.000Z-2018-09-26T00:20:00.000...	52.75 KB	text/plain	Multi-Regional	9/25/18, 5:20 PM
output-2018-09-26T00:20:00.000Z-2018-09-26T00:25:00.000...	65.83 KB	text/plain	Multi-Regional	9/25/18, 5:25 PM
output-2018-09-26T00:25:00.000Z-2018-09-26T00:30:00.000...	65.79 KB	text/plain	Multi-Regional	9/25/18, 5:30 PM
output-2018-09-26T00:30:00.000Z-2018-09-26T00:35:00.000...	65.66 KB	text/plain	Multi-Regional	9/25/18, 5:35 PM
output-2018-09-26T00:35:00.000Z-2018-09-26T00:40:00.000...	66.06 KB	text/plain	Multi-Regional	9/25/18, 5:40 PM
output-2018-09-26T00:40:00.000Z-2018-09-26T00:45:00.000...	65.67 KB	text/plain	Multi-Regional	9/25/18, 5:45 PM
output-2018-09-26T00:45:00.000Z-2018-09-26T00:50:00.000...	57.74 KB	text/plain	Multi-Regional	9/25/18, 5:50 PM

Cleanup

Because you're in a Qwiklab, when your lab completes all of your resources will be deleted. However, it's good to know how to clean up a project! The following steps specify an orderly shutdown of a Dataflow job:

1. Click **Navigation menu** > **Dataflow**.
2. Click on the **sensor-data** job.
3. Then Click **STOP**.

4. When the page pops up, select **Drain** > **STOP JOB**.

7.Write a program to implement the linear regression for a sample training dataset stored as a .CSVfile. Compute the accuracy of the classifier, considering few test datasets

```
import pandas as pd
path_to_file = 'home/projects/datasets/student_scores.csv'
df = pd.read_csv(path_to_file)
df.head()
df.shape (25, 2)
df.plot.scatter(x='Hours', y='
print(df.corr())
print(df.describe())
y = df['Scores'].values.reshape(-1, 1)
X = df['Hours'].values.reshape(-1, 1)
print(df['Hours'].values) # [2.5 5.1 3.2 8.5 3.5 1.5 9.2 ... ]
print(df['Hours'].values.shape) # (25,)
print(X.shape) # (25, 1)
print(X)
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
SEED = 42
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = SEED)
print(X_train) # [[2.7] [3.3] [5.1] [3.8] ... ]
print(y_train) # [[25] [42] [47] [35] ... ]
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
print(regressor.intercept_)
print(regressor.coef_)
 def calc(slope, intercept, hours):
    return slope*hours+intercept

score = calc(regressor.coef_, regressor.intercept_, 9.5)
print(score) # [[94.80663482]]
score = regressor.predict([[9.5]])
print(score) # 94.80663482
y_pred = regressor.predict(X_test)
df_preds = pd.DataFrame({'Actual': y_test.squeeze(), 'Predicted': y_pred.squeeze()})
print(df_preds)
from sklearn.metrics import mean_absolute_error, mean_squared_error
import numpy as np
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
print(f'Mean absolute error: {mae:.2f}')
print(f'Mean squared error: {mse:.2f}')
print(f'Root mean squared error: {rmse:.2f}')

Scores', title='Scatterplot of hours and scores percentages');
```

8.Build a decision tree classifier for weather prediction dataset. Compute the accuracy of the classifier, considering few test datasets

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
%matplotlib inline
dataset = pd.read_csv("D:/Datasets/bill_authentication.csv")
dataset.shape
dataset.head()
X = dataset.drop('Class', axis=1)
y = dataset['Class']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
from sklearn.tree import DecisionTreeClassifier
classifier = DecisionTreeClassifier()
classifier.fit(X_train, y_train)
y_pred = classifier.predict(X_test)
from sklearn.metrics import classification_report, confusion_matrix
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

10.Predicting Humidity using Decision Tree Algorithm

```
import pandas as pd
from sklearn.metrics import accuracy_score
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
data = pd.read_csv("../input/daily_weather.csv")
data.head()
data.columns
data.isnull().any().any()
data.isnull().sum()
data[data.isnull().any(axis = 1)]
del data['number']
data.columns
before_rows = data.shape[0]
data = data.dropna()
after_rows = data.shape[0]
print("The number of dropped rows are {}".format(before_rows - after_rows))
clean_data = data.copy() # New data frame to avoid confusion
clean_data['high_humidity_label'] = (clean_data['relative_humidity_3pm'] > 24.99) * 1
print(clean_data['high_humidity_label'])
y = clean_data[['high_humidity_label']].copy()
y
clean_data['relative_humidity_3pm'].head()
y.head()
time = '9am'
features = list(clean_data.columns[clean_data.columns.str.contains(time)])

# we do not need relative humidity at 9am
features.remove('relative_humidity_9am')

features
X = clean_data[features].copy()
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.33, random_state = 324)
# type(X_train)
#  type(X_test)
# type(y_train)
# type(y_test)
```

```
# X_train.head()
# #y_train.describe()
y_train.describe()
X_train.describe()
humidity_classifier = DecisionTreeClassifier(max_leaf_nodes = 10, random_state = 0)
humidity_classifier.fit(X_train, y_train)
type(humidity_classifier)
predictions = humidity_classifier.predict(X_test)
type(predictions)
predictions[:10]
y_test[['high_humidity_label']][:10]
accuracy_score(y_test, y_pred = predictions)
from sklearn.metrics import mean_squared_error
mean_squared_error(y_test, y_pred = predictions)
```