Bootcamp Manual for Introcuction to IOT

Editied by: Shrutkirthi S Godkhindi, Deeksha P Rao, T V Prabhakar

@Zenlab, DESE, IISc

Table of Contents

1

SECTION

Introduction to Nano BLE 33 Sense

The Arduino Nano 33 BLE Sense combines a tiny form factor, different environment sensors and the possibility to run AI using TinyML and TensorFlow™ Lite. Whether you are looking at creating your first embedded ML application or you want to use Bluetooth® Low Energy to connect your project to your phone, the Nano 33 BLE Sense will make that journey easy.

Table 1: **Comparison of the literature**

USB Connector	Micro USB	
Connectivity	**Bluetooth®**	NINA-B306
Sensors	IMU Microphone Gesture, light & proximity Barometric & pressure Temperature & humidity	LSM9DS MP34DT05 APDS9960 LPS22HB HTS221
Communication	UART I2C SPI	RX/TX A4(SDA), A5(SCL) D11 (COPI), D12 (CIPO), D13 (SCK). Use any GPIO for Chip Select (CS).
Clock speed	Processor	nRF52840 64MHz
Memory	nRF52840	256 KB SRAM, 1MB flash

It is built upon the nRF52840 microcontroller and runs on **Arm® Mbed™ OS**. The Nano

33 BLE Sense not only features the possibility to connect via **Bluetooth® Low Energy** but also comes equipped with **sensors** to detect color, proximity, motion, temperature, humidity, audio and more.
The microcontroller on the Arduino Nano 33 BLE Sense runs at 3.3V, which means that you must never **apply more than 3.3V** to its Digital and Analog pins. Care must be taken when connecting sensors and actuators to assure that this limit of 3.3V is never exceeded. Connecting higher voltage signals, like the 5V commonly used with the other Arduino boards, will damage the Arduino Nano 33 BLE Sense. Arduino Nano 33 BLE Sense provide 30 pinouts as shown in Fig1. The sensors are pointed out in Fig 2.

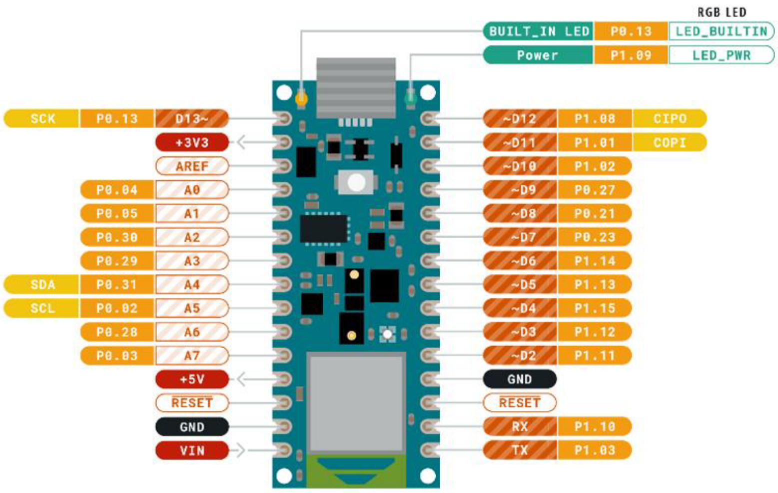


Figure 1: Arduino Nano BLE Pinout

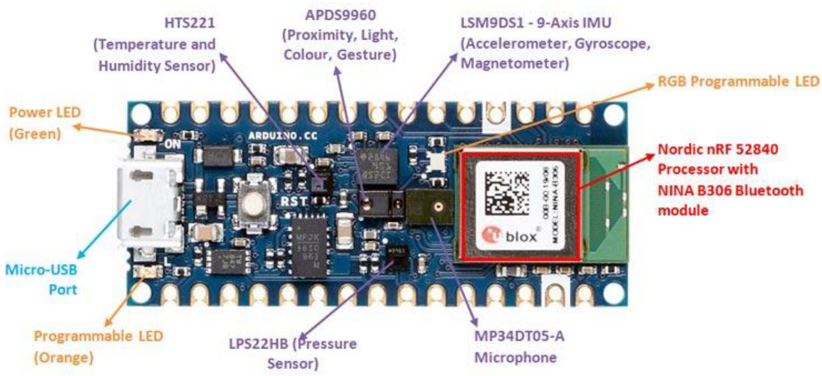
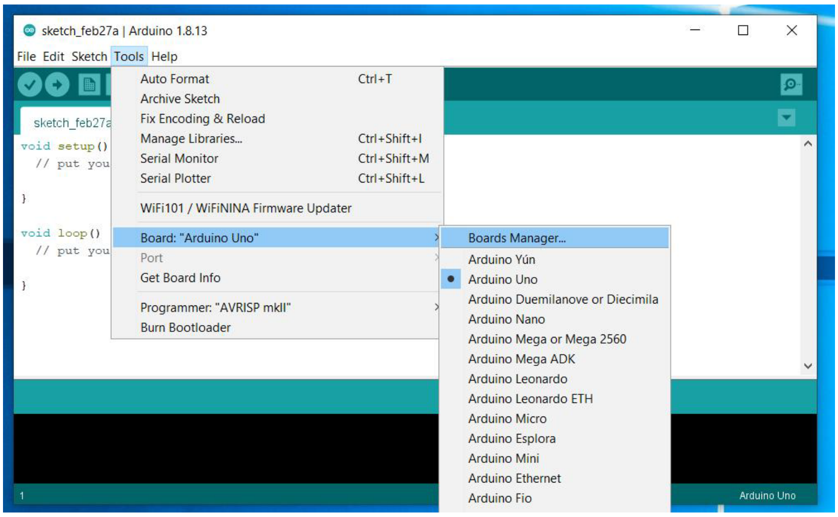


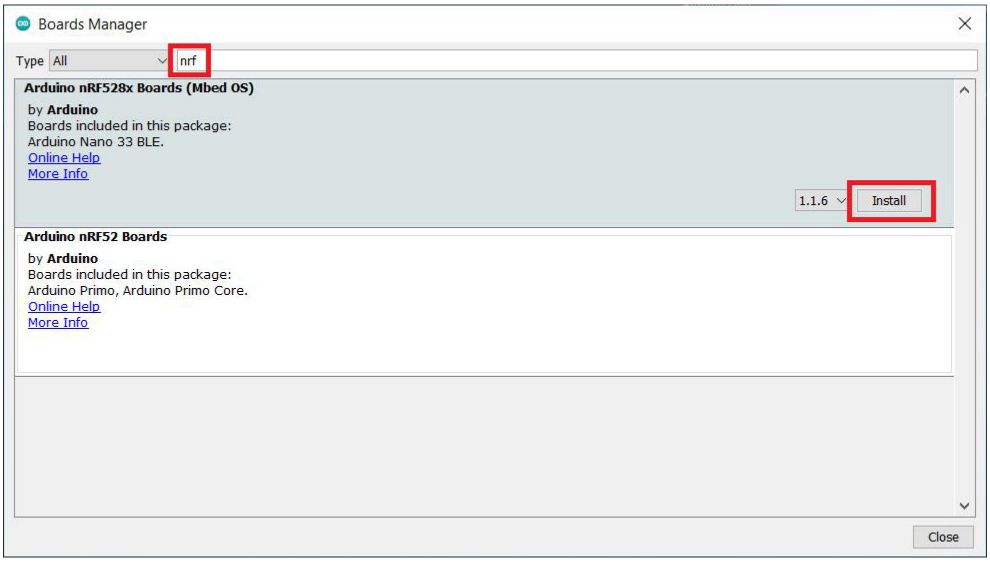
Figure 2: sensors

1.1 Adding Support for Arduino Nano 33 BLE Sense to Arduino IDE

Arduino IDE does not know anything about Arduino Nano 33 BLE Sense by default and you must add support for this board. Go to **Tools -> Board -> Board Manager**.

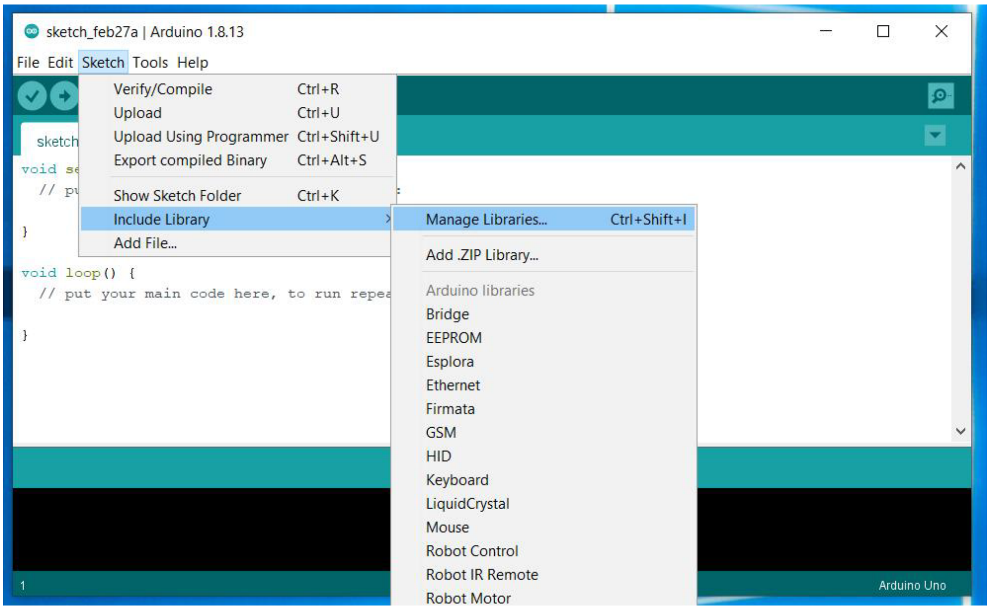


Search for “Arduino Mbed OS Nano Boards” and click Install.



Wait until installation complete and then close the Board Manager. Now we must install libraries for accessing sensors. Go to **Sketch -> Include Library -> Manage Libraries...** Search for

- HTS221 and install Arduino_HTS221 library
- LPS22HB and install Arduino_LPS22HB library
- LSM9DS1 and install Arduino_LSM9DS1 library.
- APDS9960 for and install Arduino_APDS9960 library.
- PDM library is installed along with the board only.



Basic Steps

- Open the Arduino IDE.
- Type the code.
- Go to **Tools -> Board -> Arduino Mbed OS Nano Boards -> Arduino Nano 33 BLE**
- **Go to Tools -> Port ->** Select the appropriate Serial Port.
- Compile the Code.
- Debug the Errors.
- Upload the Code.
- Go to **Tools -> Serial Monitor/Serial Plotter**, to check the Serial Output or give Serial Input.

2

SECTION

Tiny ML and Edge Computing - Bootcamp

2.1 TinyML and Edge Computing

TinyML refers to the deployment of machine learning models on ultra-low power devices such as microcontrollers and sensors. It focuses on enabling AI capabilities in resource-constrained environments with limited memory, power, and computational capacity. Applications include anomaly detection, predictive maintenance, and real-time environmental monitoring in IoT devices.

Edge Computing involves processing data closer to the source of generation (at the "edge" of the network) rather than relying on centralized cloud computing. This reduces latency, enhances data privacy, and minimizes bandwidth usage. Combining TinyML with Edge Computing enables real-time decision-making and analytics without continuous cloud connectivity, making it ideal for applications in autonomous systems, smart cities, and wearable devices.

2.2 Deployment Workflow for TinyML on Edge Devices

Overview of the steps involved in deploying TinyML models to edge devices:

1. **Data Collection**: Collect data from sensors or devices. This could be sensor readings from accelerometers, cameras, microphones, etc.

2. **Preprocessing and Feature Engineering**: Clean and preprocess the collected data. Depending on the application, this could involve normalizing, filtering, or transforming the raw sensor data.

3. **Model Development and Training**: Use machine learning frameworks like Edge Impulse to design and train your model.

4. **Model Optimization**: Perform model optimization techniques like quantization, pruning, and compression to reduce model size and computational requirements. TensorFlow Lite, for example, offers built-in tools to reduce the size of models while maintaining performance.

5. **Deployment to Edge Devices**: Deploy the optimized TinyML model to edge devices (such as microcontrollers, IoT devices, or wearables) using tools like Edge Impulse.

6. **Real-Time Inference**: Once deployed, the device runs the model to make predictions based on real-time sensor data. The model can run entirely on the edge device without relying on external servers or the cloud, enabling real-time decision-making.

2.3 Edge Impulse

Edge Impulse is a platform specifically designed to help developers build, deploy, and optimize machine learning models for edge devices. It provides a full development environment, including data collection, model training, and deployment tools, with minimal coding required. This makes it especially useful for IoT, embedded systems, and other low-power devices

2.3.1 Features

- Supports a wide range of sensors (accelerometers, microphones, etc.) for data collection.

- Provides tools to preprocess sensor data and train custom ML models.

- Offers seamless deployment to embedded systems and microcontrollers.

2.4 Arduino Nano 33 BLE Sense, Arduino Nicla Vision and Nicla Voice

2.4.1 Arduino Nano 33 BLE Sense

The Arduino Nano 33 BLE and the nRF52840 module (from Nordic Semiconductor) are popular edge computing platforms for TinyML applications.

- **Features**:

  1. Arduino Nano 33 BLE Sense is build upon the nRF52840 microcontroller and runs on Arm® Mbed™ OS.
  2. It has a built-in Bluetooth Low Energy (BLE) support for easy communication with other edge devices.

3. It can run lightweight TinyML models for local sensor data analysis.
4. It is compatible with TensorFlow Lite for Microcontrollers and other TinyML frameworks.
5. The board also comes equipped with sensors to detect color, proximity, motion, temperature, humidity, audio and more.
6. Compatible to program the board both online and offline with software tools like Arduino IDE, Arduino CLI and Cloud Editor.
7. This board can be programmed using MicroPython which is an implementation of the Python® programming language that comes with a subset of the Python® standard library.

- **Sensors on board**:

1. Arduino Nano 33 BLE Sense has a powerful 2.4 GHz Bluetooth® 5 Low Energy module from u-blox, with an internal antenna that can be used to transmit data between different devices.
2. LSM9DS1: An Inertial Measurement Unit (IMU) features a 3D accelerometer, gyroscope and magnetometer allowing to detect orientation, motion or vibrations.
3. APDS9960: A Proximity and Gesture Detection sensor also used to control the built-in RGB LED through hand gestures.
4. LPS22HB: Picks up on barometric pressure and allows for a 24-bit pressure data output between 260 to 1260 hPa.
5. MP34DT05: A built-in omnidirectional digital microphone to capture and analyze sound in real time to create a voice interface.

2.4.2 Nicla Voice

- **Features**:

1. The Nicla Voice features an nRF52832, a general-purpose multiprotocol System-on-Chip (SoC) from Nordic® Semiconductor.
2. The Arduino Nicla Voice has been designed as an ultra-low power board, being the perfect solution for industrial and IoT battery-powered applications.
3. The board integrates a Neural Decision Processor from Syntiant (NDP120), enabling it to run multiple AI algorithms.
4. It offers on-board Bluetooth® Low Energy connectivity and is compatible with Nicla, Portenta, and MKR products.
5. The Nicla Voice can also be battery-powered with long-lasting autonomy being perfect for autonomous power standalone application.
6. Compatible to program the board both online and offline with software tools like Arduino IDE, Arduino CLI and Cloud Editor.

7. Arduino Nicla Voice is compatible with hardware like Portenta H7 and Portenta H7 Lite.

- **Sensors on board**:

1. IM69D130: A high-performance digital MEMS microphone from Infineon®.
2. BMI270: A 6-axis ultra-low power Inertial Measurement Unit (IMU).
3. BMM150: A low-power 3-axis digital geomagnetic sensor from Bosch®.
4. ANNA-B112 Bluetooth® module: A Bluetooth® 5.0 Low Energy system-in-package (SIP) with the powerful nRF52832 SoC integrated.

2.4.3 Arduino Nicla Vision

- **Features**:

1. The Arduino Nicla Vision is a ready-to-use, standalone intelligent camera for analyzing and processing images on the edge.
2. The Arduino® Nicla Vision is a powerful microcontroller equipped with 2MP color camera.
3. With WiFi and BLE connectivity, the board maximizes compatibility with professional and consumer equipment.
4. The board features an integrated microphone, distance sensor, smart 6-axis motion sensor and MicroPython support.
5. Compatible to program the board both online and offline with software tools like Arduino IDE, Arduino CLI, OpenMV IDE and Cloud Editor.
6. Arduino Nicla Vision is compatible with hardware like Portenta H7 and Portenta H7 Lite.

- **Sensors on board**:

1. The board uses the GC2145, a 2MP color camera.
2. LSM6DSOXTR: A 6-axis IMU which allows to obtain 3D gyroscopic and 3D accelerometer data.
3. STM32H747AII6 Dual ARM® Cortex® - M7 core up to 480 MHz + M4 core up to 240 MHz.
4. MP34DT06JTR: A digital MEMS microphone that is omnidirectional and operates via a capacitive sensing element with a high signal-to-noise. ratio.

2.5 Pre-requisites and installation

- **Visual Studio Installer**:
  https://www.techspot.com/downloads/7241-visual-studio-2019.html Install 2019 version of Visual studio installer using this link.

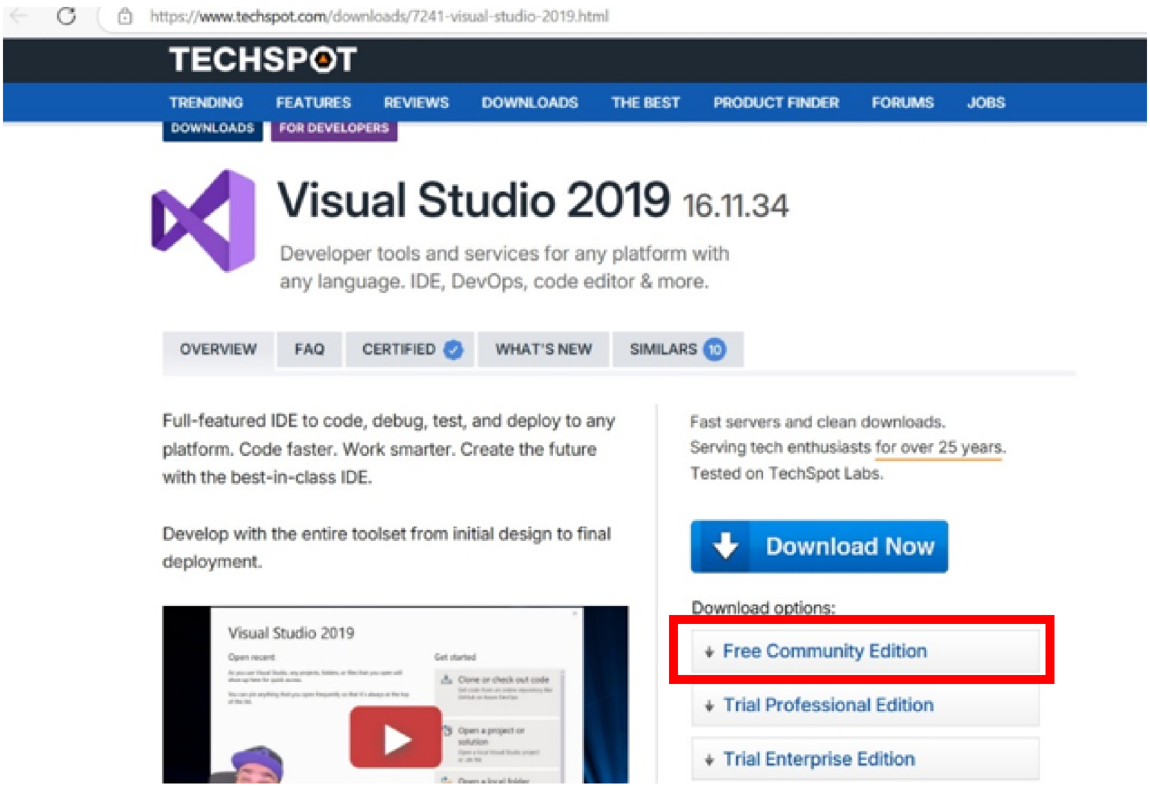
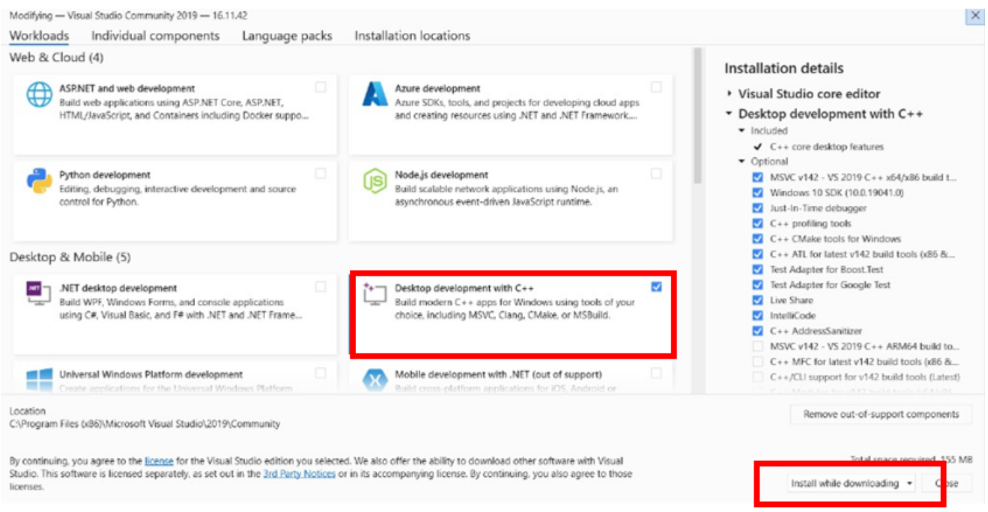


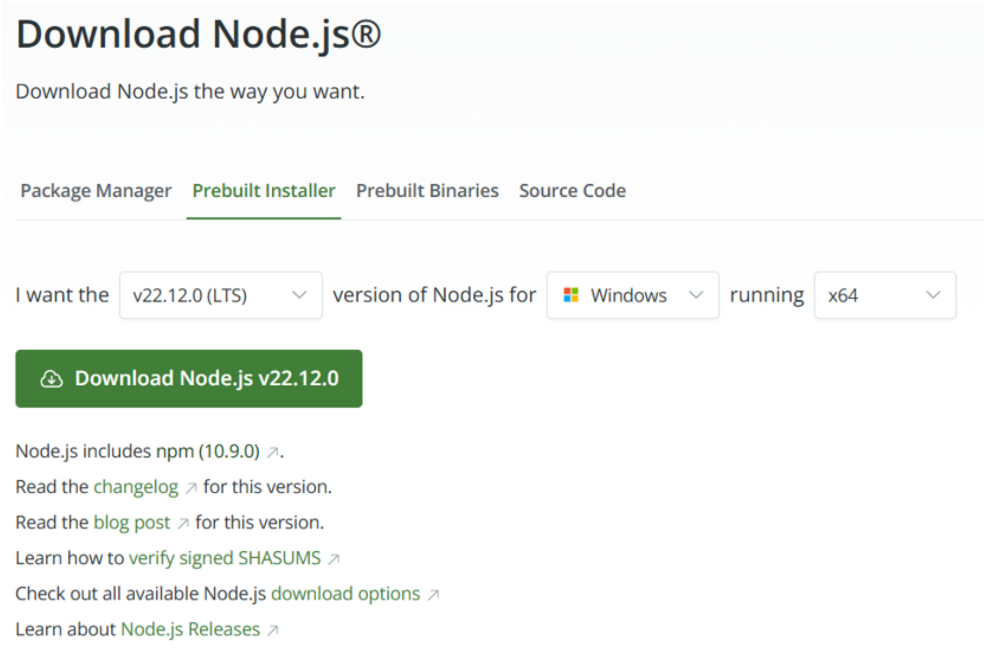
Figure 3: Install the Community edition

1. Select DESKTOP DEVELOPMENT WITH C++
2. Click on INSTALL on the right side of the screen

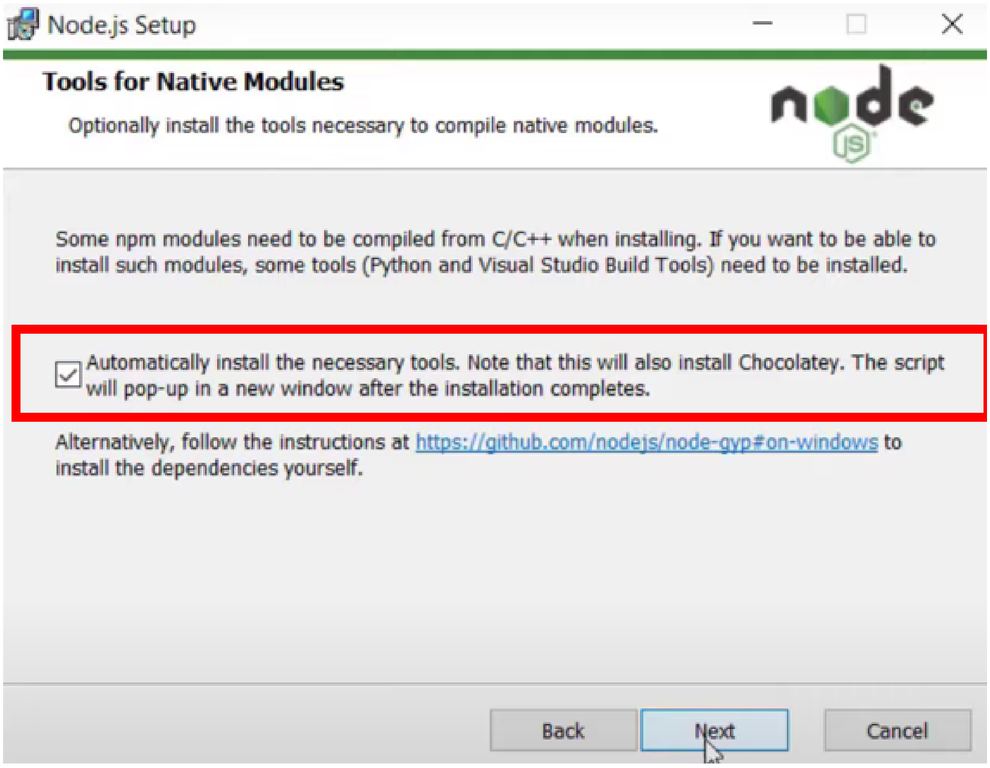


- **NodeJS**:
  Install the latest version(v2019 or above)

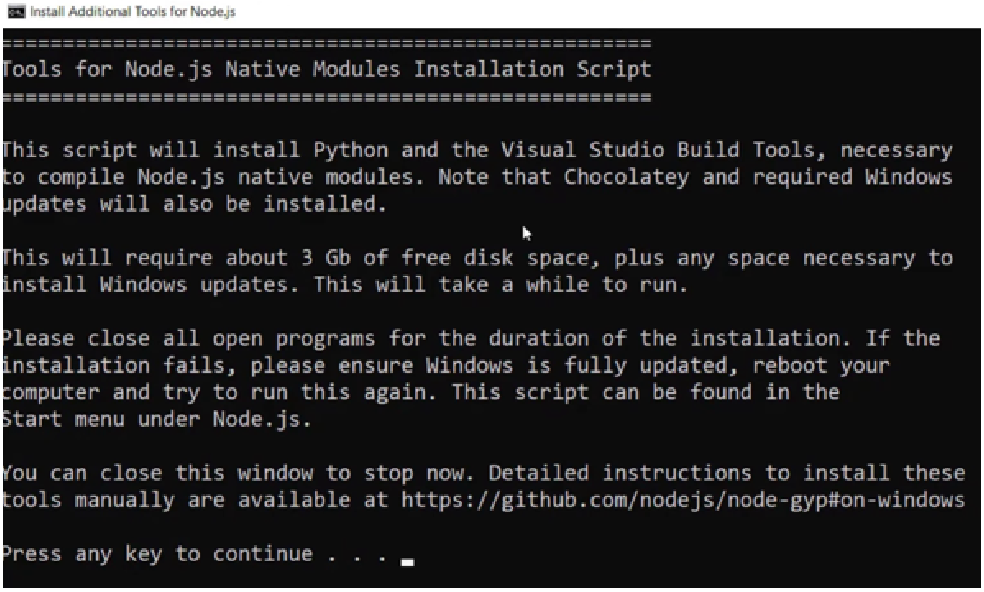
  https://nodejs.org/en/download/prebuilt-installer



1. Select- Install additional tools

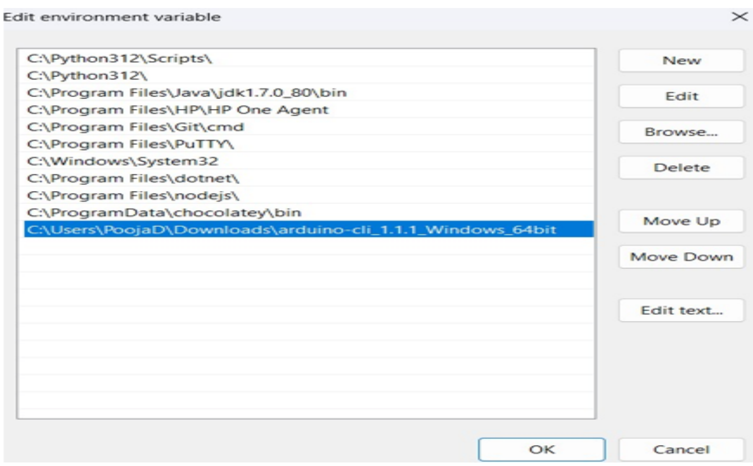


2. Once it is installed, new terminal will pop up, press ENTER twice



3. To verify nodejs installation run this command in terminal
   node - -version
   npm - -version
4. Make sure Chocolatey PATH is added in the Systems variables

- **Edge impulse CLI**:
  https://docs.edgeimpulse.com/docs/edge-impulse-cli/cli-installation
  1. Install using the link
  2. You can also install this CLI using this command in terminal
     npm install -g edge-impulse-cli
- **Arduino CLI**:
  https://arduino.github.io/arduino-cli/0.33/installation/
  1. Install Arduino CLI uisng this link according to your system configuration
  2. Extract the .zip after downloading
  3. Copy the extracted folder path and add it to SYSTEMS VARIABLES



3

SECTION

Gesture Recognition Using Arduino Nano 33 BLE Sense

The Arduino Nano 33 BLE Sense is equipped with an Inertial Measurement Unit (IMU), including an accelerometer and gyroscope, making it ideal for gesture recognition. Using the IMU data, you can classify gestures like waving, tapping, or shaking by leveraging platforms like Edge Impulse for model training and deployment.

3.1 Steps involved in Gesture Recognition using Arduino Nano 33 BLE Sense for deploying TinyML models on Edge Impulse

1. Install Arduino IDE: Download the latest version of the Arduino IDE using the following link.
   `https://www.arduino.cc/en/software`

2. Install Board Support Packages for Nano 33 BLE Sense:

   - Open Arduino IDE.
   - Go to Tools > Board > Boards Manager, and search for "Arduino Mbed OS Nano Boards by Arduino", and install it.

3. Install Required Libraries:

   - Go to Tools > Manage Libraries and install the following: "Arduino_LSM9DS1 by Arduino" (for IMU) "EdgeML-Arduino by edge-ml" (for deploying Edge Impulse models)

4. Install and Update the Firmware:

- The development board does not come with the right firmware yet. To update the firmware, download the latest Edge Impulse firmware using the following link:
  https://cdn.edgeimpulse.com/firmware/arduino-nano-33-ble-sense.zip, and unzip the files.
- Use a micro-USB cable to connect the development board to your computer. Then press RESET button twice to enter into the bootloader mode. The on-board LED should start pulsating.
- Open the downloaded flash script for your operating system (flash_windows.bat, flash_mac.command or flash_linux.sh) to flash the firmware.

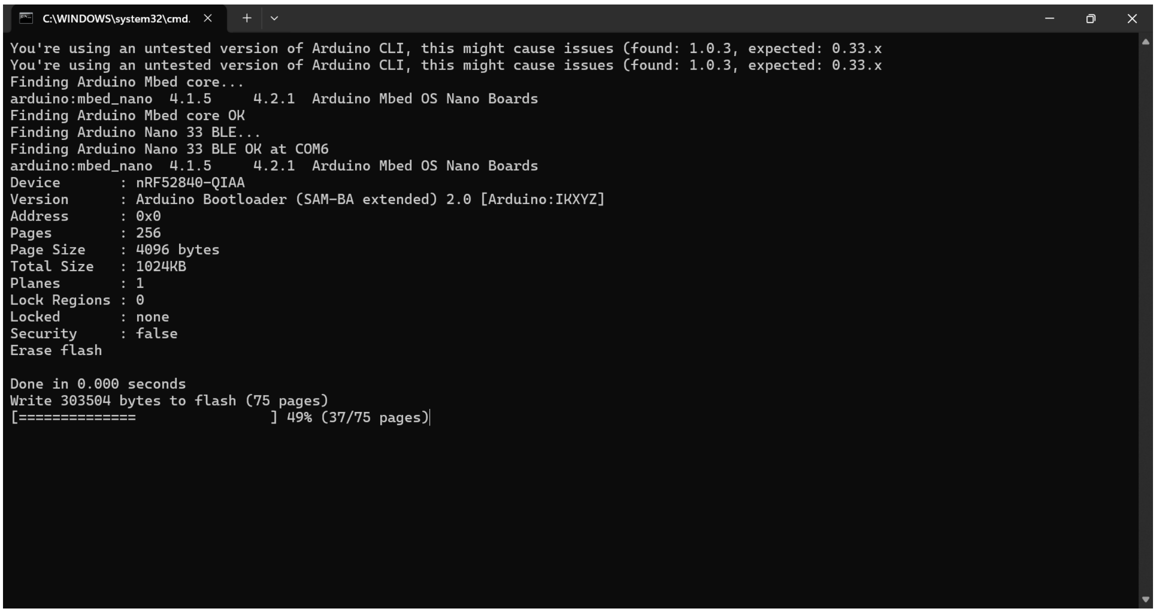


Figure 4: Install the Edge Impulse firmware for Nano 33 BLE Sense

- Wait until flashing is complete, and press the RESET button once to launch the new firmware.

5. Create a project in Edge Impulse

   - Sign Up/Log In: Go to Edge Impulse using the following link - https://edgeimpulse.com/ and create a free account.

6. Create a New Project:

   - Name the project (e.g., "Gesture_Recognition_Nano_33_BLE_Sense").
   - Set the data source as Accelerometer data.

7. Data collection: You need to collect accelerometer data for the gestures you want to recognize (e.g., tick, circle).

   - Open the "Data Acquisition" tab in Edge Impulse. Connect the Arduino Nano 33 BLE Sense to your computer using USB cable.

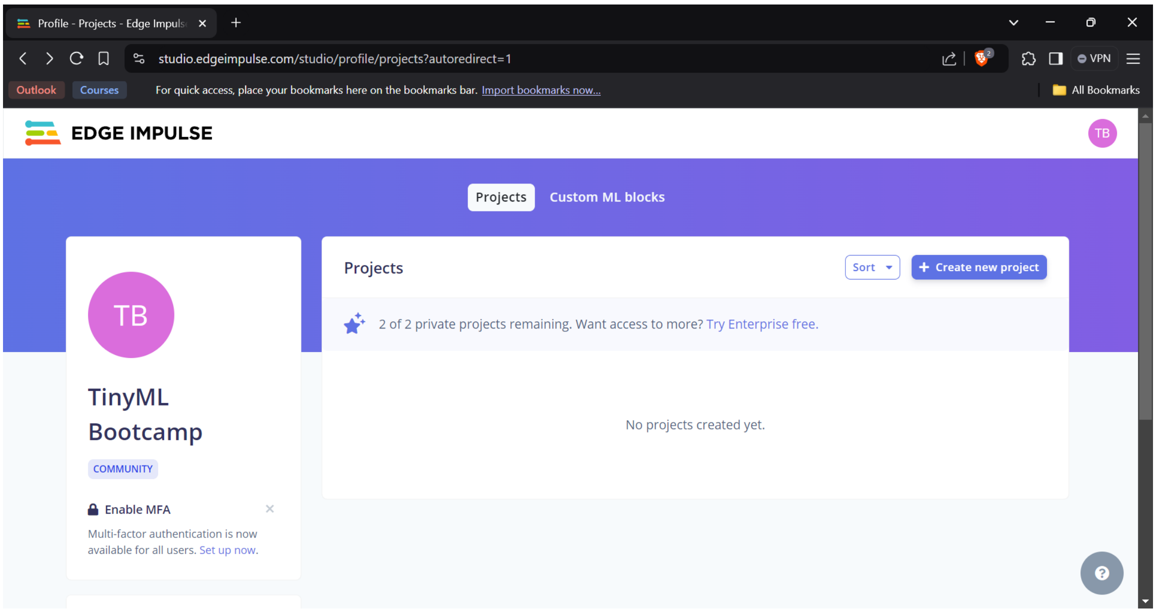


Figure 5: Edge Impulse Login Page

- Select "Connect using WebUSB". The device will be recognized as "Nano 33 BLE (COM6)" as shown below. Click on "Connect". The COM port varies for each system.
- In the Collect "Data section", the connected device's ID will be displayed.
- In the "Label" section, name the gesture you want to train the model with and select the "Sample Length" (in ms) between 1000 and 10000. By default The sensor is selected as "Inertial (Accelerometer/Gyroscope/Magnetometer)" and "Frequency" at 100Hz as shown below.
- Click on "Start Sampling" and move the Arduino Nano 33 BLE Sense in the desired gesture patterns (e.g., tick, circle).
- Label each sample appropriately (e.g., "Tick," "Circle").
- Collect an equal number of samples for each gesture.

8. Preprocess Data:

   - Go to the "Create Impulse" section under "Impulse Design" on the left side of the page. Click on "Add a Processing Block" and select "Spectral Analysis".

   - Click on "Add a Learning Block" and select "Classification". Click on "Save Impulse".

     Go to "Spectral Features" on the left side tab and click on "Save parameters". Then click on "Generate features".
   - This block processes the raw IMU data into meaningful features for classification

9. Train the Model:

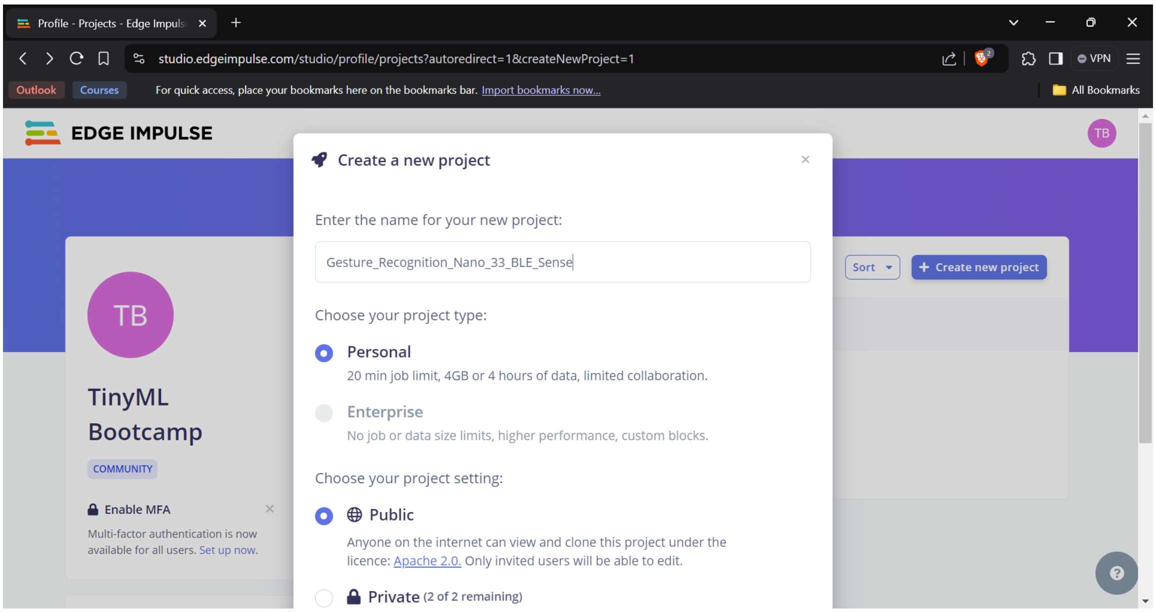


Figure 6: Create New Project on Edge Impulse

- Select the "Classifier" tab under the "Impulse design" section on the left side.
- Configure the model parameters like number of training cycles and and start training. Click on "Save and train" to begin model training.
- Evaluate its performance using metrics like accuracy, precision, and recall.

10. Export the model:

    - Go to the Deployment tab in Edge Impulse.
    - Select Arduino Library as the deployment target under "Configure your deployment" section.
    - Click on "Build" and Download the library and add it to your Arduino IDE (Sketch > Include Library > Add .ZIP Library). Select the downloaded .zip folder

11. Test the Model on Arduino IDE2:

    - Create a new Arduino sketch and include the exported library.
    - Connect the Arduino Nano 33 BLE Sense to your computer and upload the sketch to the board.
    - Open the Serial Monitor in Arduino IDE (set baud rate to 115200).
    - Perform the gestures, and the board should classify them in real time, displaying results in the Serial Monitor.

12. Alternate Method:

    - Select Arduino Nano 33 BLE during configure your deployment.
    - Download and unzip the the folder.

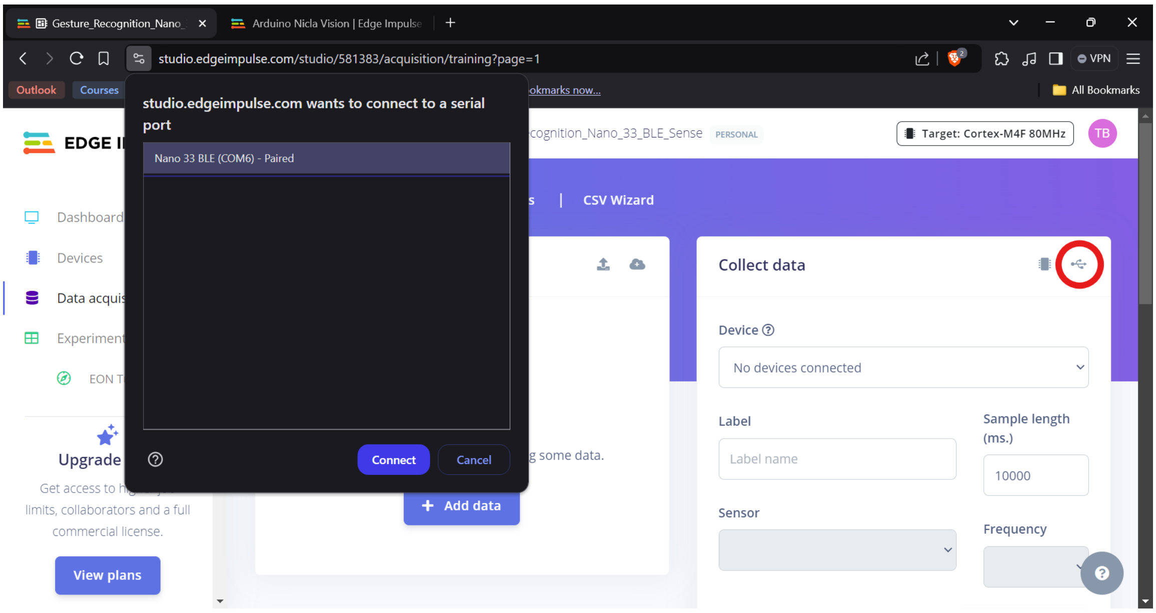


Figure 7: Connect Device to Edge Impulse

- Load the model using the flash file (.bat from windows, .sh for linux).
- run “edge-impulse-run-impulse" from the terminal to verify the model.

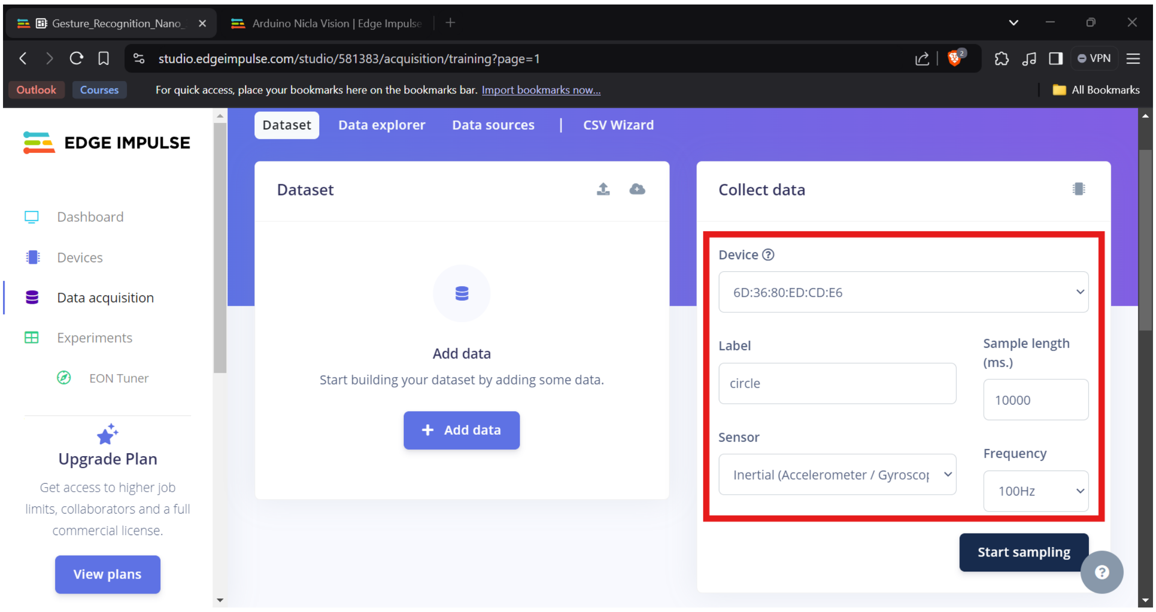


Figure 8: Configure the Parameters to Collect Data

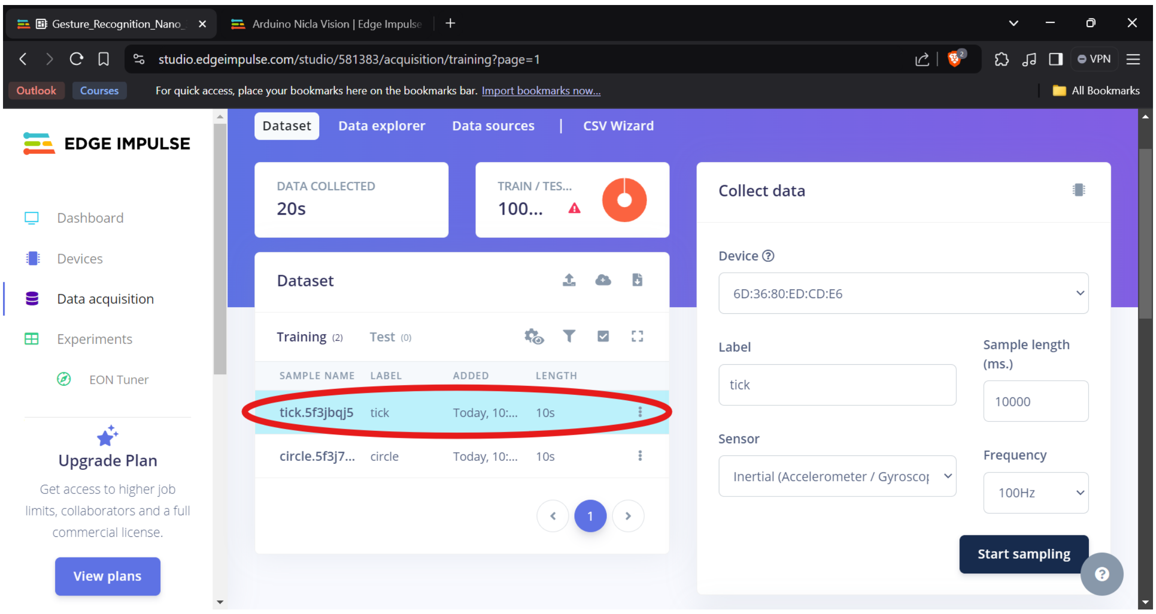


Figure 9: Sample Gesture Data Collection

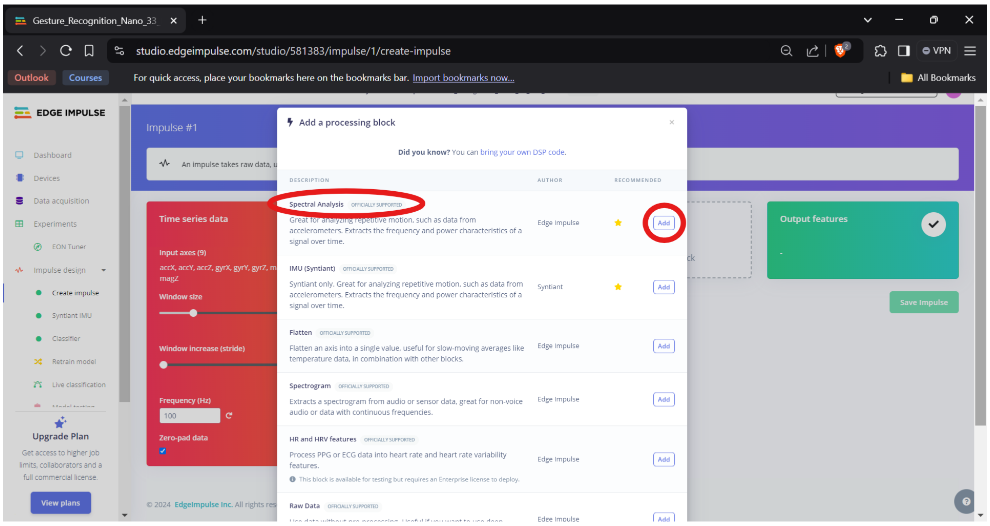


Figure 10: Adding Processing Block

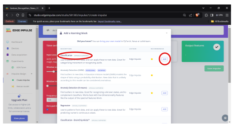


Figure 11: Adding Learning Block

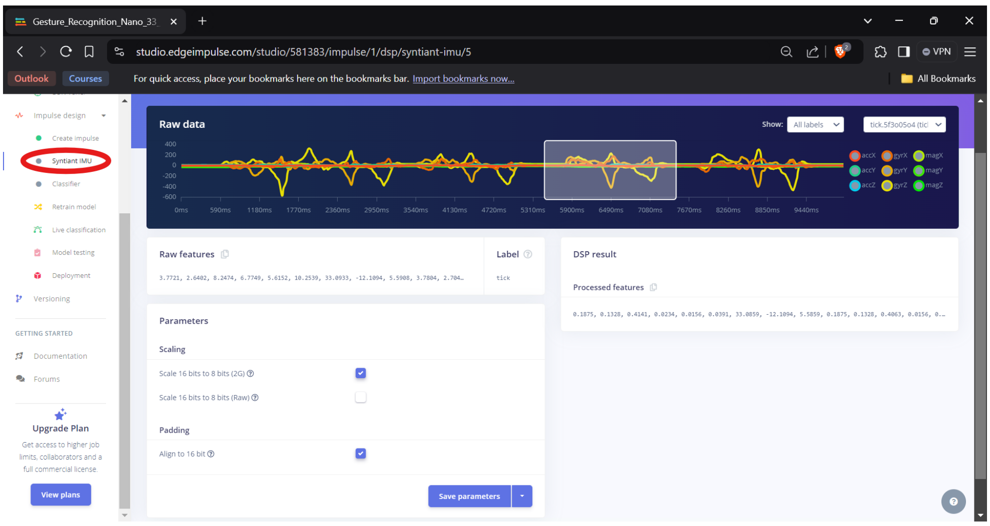


Figure 12: Save Parameters

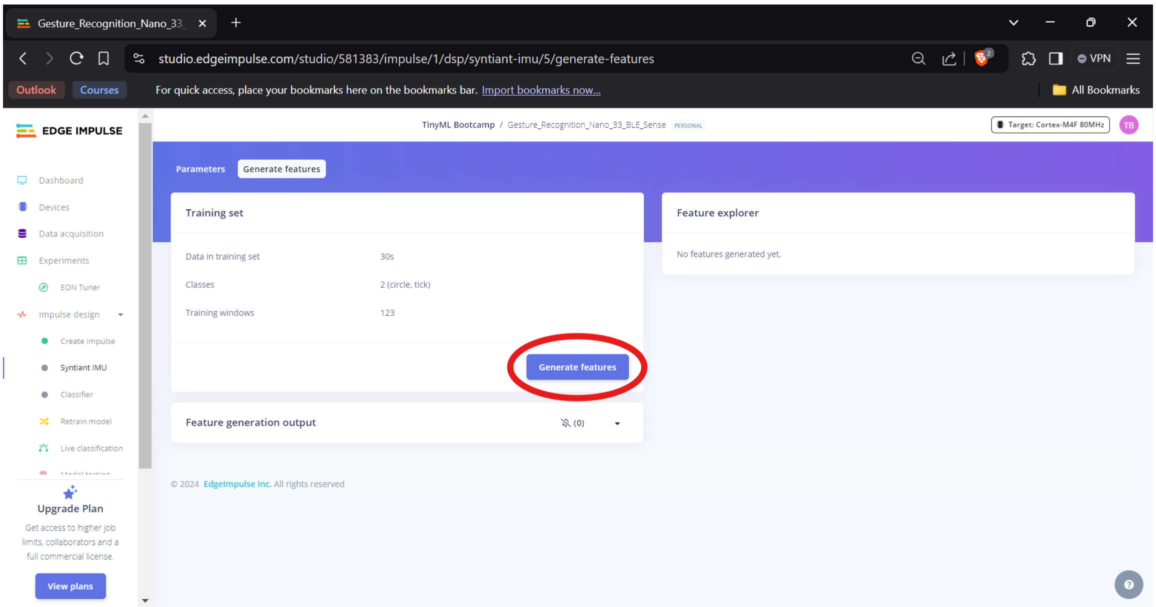


Figure 13: Generate Features

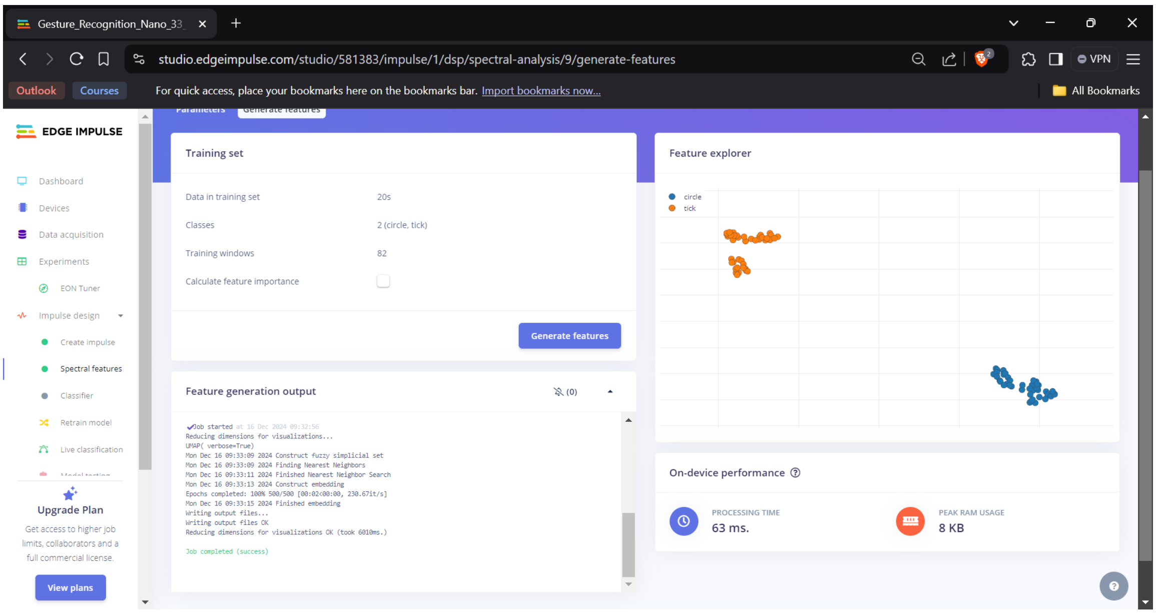


Figure 14: Generated Features

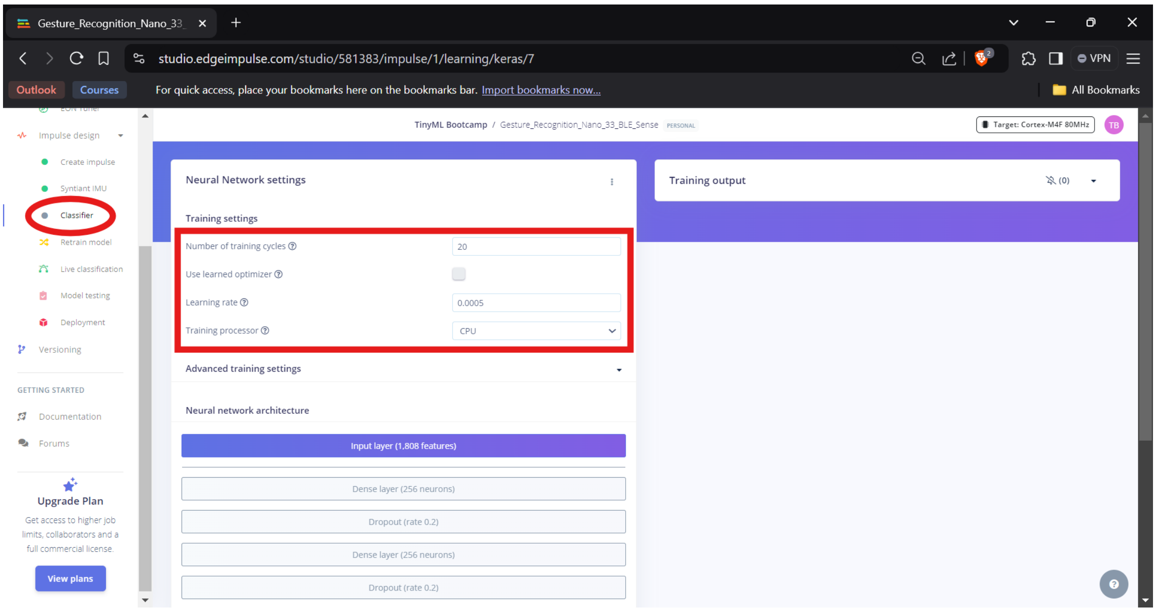


Figure 15: Set Model Parameters for Training

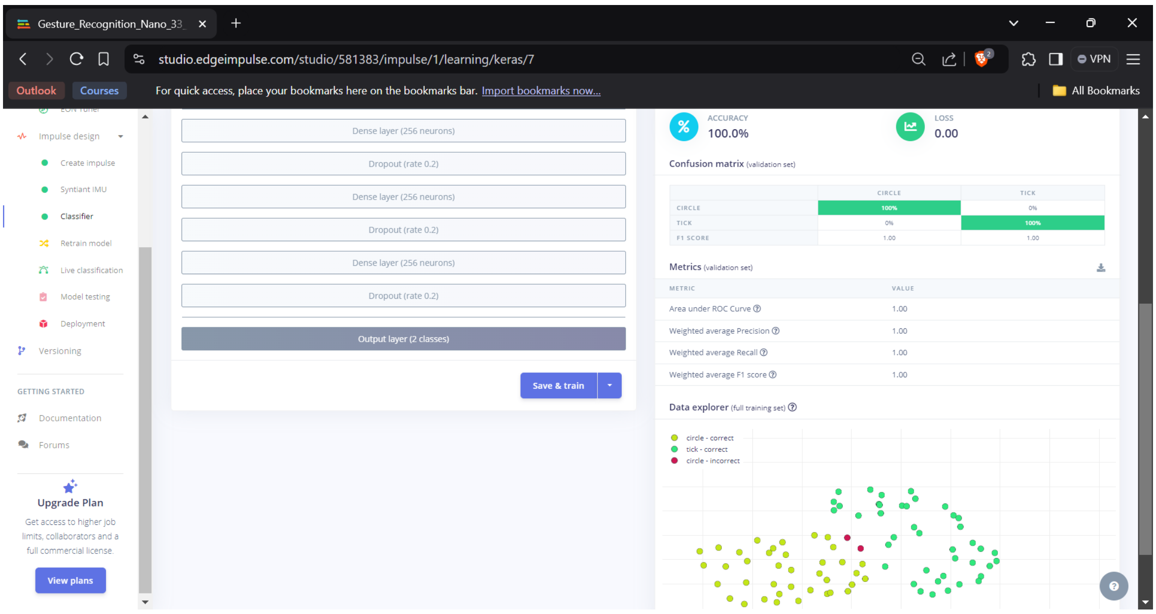


Figure 16: Model’s Performance on the Validation Data