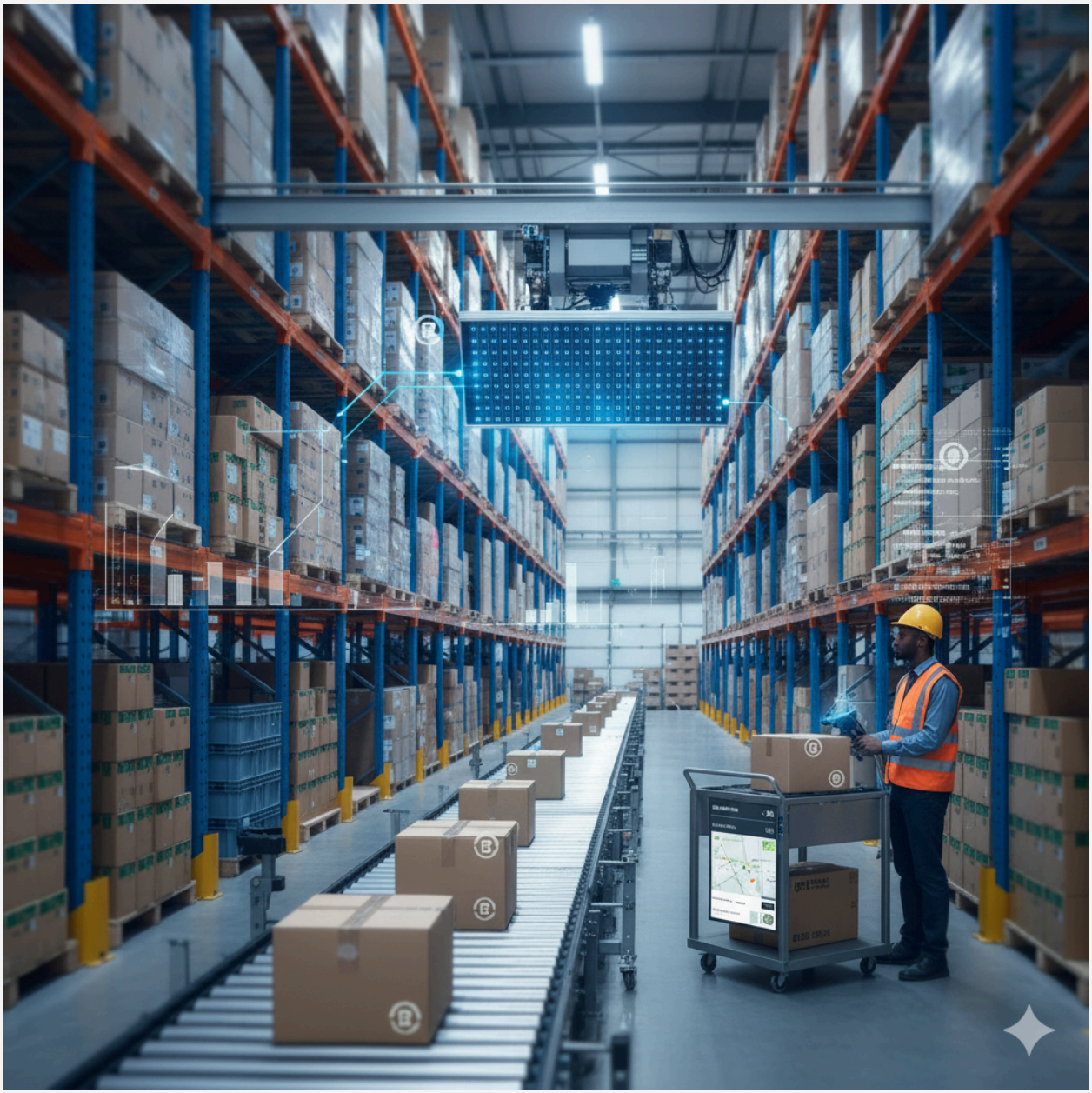

Product Description

The IoTReady API-ready RFID reader is an integral component in any data-driven manufacturing system. It enables you to easily implement key logistics functions such as part and product checks, presence/absence detection, flow tracking etc.

Unlike competitive products, this reader comes ready to directly integrate into your workflow processes through a rich set of REST APIs. The reader supports EPCIS compatible event handling and directly generates XML formatted event information.

With WiFi/Ethernet connectivity, it provides easy access to setup and configuration functions. The protocol stack enables real-time updates of measurements to any destination application. Longevity is enabled through the use of over-the-air updates.

Built to IP64 specifications and capable of operating over industrial temperature ranges, this is an ideal device to deploy in all kinds of traceability and access control applications.

We offer 1 antenna, 4 antenna and 6 antenna variants currently. Contact us for custom configurations with up to 100 antenna variant (patent pending).

Key Features

- Simple BLE, WiFi & Ethernet interfaces allow you to easily configure the device and integrate into your system
- Self-hosted REST APIs

Function	API-ready RFID Reader
Use	Industrial, Logistics, Access Control
Operating Env.	-20C to +70C, <95% RH non-condensing

- Easily integrates with AWS IoT, SAP IoT or other similar brokers over HTTP(S), MQTT and optional EPCIS events.
- Easy to use, documented API (subject to change)
- Optional - Battery powered with automatic recharging of the battery
- Optional - API controlled LCD character display
- Up to 3m cable length for each antenna
    - Longer cables are supported but you may see a drop in read performance.

Communications Interfaces

Uses HTTPS/REST over WiFi/Ethernet as its default communications mechanism. This enables easy integration with minimal overhead.

Optional BLE interface for integration with mobile and desktop devices (select at time of ordering).

Device also supports MQTT over WiFi/Ethernet with configurable support for sending data to AWS IoT or SAP IoT.

4G and GPS interfaces available optionally.

System Integration

Simple Initialization over WiFI. On powerup, the system establishes a WIFI hotspot. Connecting to this WiFI enables one to set up the device using simple REST commands.

Supports multiple communication channels for sending scan data.

Protocol	EPCglobal UHF Class 1 Gen 2 / ISO 18000-6C
Operating Frequency	865MHz – 868MHz 902MHz – 928MHz (Optional)
Peak Read Speed	Up to 900 tags/sec
Offline Label buffer	10,000 labels @96 bit EPC
Max Range (tag dependent)	3dBi Circular polarization Ceramic antenna ~5m 9dBi Circular polarization plate antenna > 15m
Min Range	Software controllable to 5% of max range
Comm. Interface	WiFi, Ethernet, RS232 HTTPS/Websockets/MQTT
4G/GPRS Spec	Optional - LTE-TDD/LTE-FDD/GSM/GPRS/ EDGE
Outdoor Locationing	Optional - supports GPS, BDS, Galileo, and GLONASS
Mounting	Wall or pole mount
Base Construction	IP68 ABS enclosure
Battery	Optional - 7.4V/2800 mAH Rechargeable LiPo
Power Supply	12V DC

OTA Updates	OTA updates individually identifiable and upgradeable for each MAC address
Integration	Simple REST commands to set up and configure WiFI and other parameters
Diagnostics	All diagnostics can be optionally pushed to configurable REST endpoint

Note: Features are subject to change

WiFi API

Set up WiFi and POST URI

- When not configured and or unable to reach a configured WiFi network, the device will broadcast a hotspot: IoTReady-<mac_address>.
- Default password for the hotspot is ”getiotready”
- Default IP when the device is in hotspot mode: 192.168.0.1.
- Connect to this hotspot and make the following API calls to the above mentioned IP.
- All of the following APIs are also available after the device has connected to your preferred network - just use the new IP address.
    - Devices with displays will show the IP address.
    - Devices without displays will broadcast themselves using mDNS so you can discover them using any mDNS/Bonjour scanning application.
- Note: The RFID reader only supports connectivity with a 2.4GHz 802.11a/b/g network supporting WPA2-PSK.

WiFi Config

Use this API to configure the device to connect to your preferred network. We strongly recommend reserving a static IP address on your router for the device’s MAC address. Please ensure that you have verified WPA2-PSK and whitelisted the MAC address as well as any remote APIs you are going to make API calls to. These are the most common reasons for failures.

```
curl -X POST 'https://{{Device_IP}}/config' -H 'Content-Type: application/json' \
-d '{
   "cmd": "write",
   "ssid": "IoTReady",
   "pwd": "getiotready",
"logUri": "Your POST URI here",
      "authHdr": "Your postUri auth token here"
}'
```

Reboot the RFID reader once you get a “Node configuration saved” response from the API call.

Read Config

Read Config will list all the configurable parameters for the RFID reader

```
curl -X POST -H 'https://{{Device_IP}}/config' \
-d '{
```

```
   "cmd": "readAll"
}'
```

Set Read Power

The maximum read power that can be set is 3000 (30 dBi). Lower values reduce range.
The Default readpower is set at 3000.

```
curl -X POST -H 'https://{{Device_IP}}/config' \
-d '{
   "cmd": "write",
   “readPower”: 3000
}'
```

Continuous Scan

The device can be configured to start scanning on boot via a special ‘boot scan’ mode. The device will scan for ‘bootscanDuration’ milliseconds and pause for ‘bootscanInterval - bootscanDuration’ milliseconds.

```
curl -X POST -H 'https://{{ip}}/config' \
-d '{
   "cmd": "write",
   “bootscan”: true,
   “ bootscanType”:”asynchronous” ,
   “bootscanInterval”:  3500,
   "bootscanDuration": 3000,
"bootscanEpcEncoded": false,
}'
```

Triggered Scan

Alternatively, you can use APIs to start and stop scans. The `bootscan` flag must be enabled for this mode as well.

Curl for read config:

```
curl -X POST -H 'https://{{ip}}/config' \
-d '{
   "cmd": "write",
   “bootscan”: true
}'
```

Start scan

A scan can be started by setting the scan duration and pause interval to start the next scan.

```
curl -X POST -H 'https://{{ip}}/rfid \
-d '{
   "cmd": "startAsyncScan",
"scanDuration": 1000,
"scanInterval": 1500,
"epcEncoded": false
}'
```

Stop Scan

```
curl -X POST -H 'https://{{ip}}/rfid \
-d '{
   "cmd": "stopScan"
}'
```

Offline Mode

The device has a built-in offline log that will store scanned tags along with a timestamp. When the device is back offline, it will post this data to the remote API.

Payload Structure

The device will post data to your API (‘logURI’) in JSON format. The payload structure is shown below. This structure cannot be changed so please implement your API accordingly.

```
{
  "data": [
    {
      "device_id": "DEVICE_ID",
      "timestamp": "TIMESTAMP",
      "group_id": "GROUP_ID",
      "tags": [
        "Tag1",
        "Tag2",
        "Tag3",
        "Tag4",
```

```
      "Tag N"
    ]
   }
 ]
}
```

Sample Payload

```
{
 "data": [
  {
   "device_id": "24dcc3e0bdcb",
   "timestamp": "1720096047570",
   "group_id": "MB3RMB",
   "tags": [
    "3055EE62CF10000000159840"
   ]
  }
 ]
}
```

BLE API

This document describes the Bluetooth Low Energy (BLE) interface for communicating with ESP32-based RFID readers. The interface uses GATT characteristics to send commands and receive responses from the device.

Scan And Connect

The device will start a BLE advertisement soon after booting and initialising. The advertised name is '<mac_address>' e.g. 'd48c49ca0fd8'. You have two choices for UI:

- Scan for BLE devices and show a list to the user in your app for them to connect to OR
- Use a QR + BLE flow (Strongly recommended):
    - Show a QR scan button in your app
    - Prompt the user to scan the QR code on the side or top of the device,
    - Do a BLE scan to verify the device with that ID is advertising
    - Connect to the BLE device

There are 3 possible reasons for the device to not show up in the BLE scan:

- Device is already connected to a different mobile device
- Device is not powered on or you have used an incorrect power adapter (e.g. 5V instead of 12V)
- Device is faulty
    - Before arriving at this conclusion, please test the connection with nRF Connect - a robust third party app

Characteristics

Once connected, the following characteristics are available on the BLE interface (GATT).

Characteristic Name	UUID	Access	Description
Weight	d3edcb21-d7aa-4731-bad8-946651538502	Read	Weight data from connected scale (optional)
Scan	d3edcb21-d7aa-4731-bad8-946651538501	Read	RFID scan results
System Commands	d3edcb21-d7aa-4731-bad8-946651538519	Read/Write	System-level commands

Config Cmd	d3edcb21-d7aa-4731-bad8-94665153851a	Read/Write	Device Configuration Commands
OTA Trigger	d3edcb21-d7aa-4731-bad8-946651538516	Write	OTA firmware control
OTA Data	d3edcb21-d7aa-4731-bad8-946651538517	Write	OTA Firmware Data Transfer
RFID Commands	d3edcb21-d7aa-4731-bad8-946651538518	Read/Write	RFID-specific commands
Peripheral Cmd	d3edcb21-d7aa-4731-bad8-94665153851b	Read/Write	Control peripherals like LCD Display, RGB LEDs, Buzzer

System Commands

Characteristic UUID: d3edcb21-d7aa-4731-bad8-946651538519

Get Version

```
{"cmd": "version"}
```

Returns the firmware version of the device.

Reboot Device

```
{"cmd": "reboot"}
```

Reboots the device.

Config Commands

Characteristic UUID: d3edcb21-d7aa-4731-bad8-94665153851a

Configure Weight-Triggered Scanning

```
{
 "cmd": "write",
```

```
 "weightTriggered": false,
 "scaleResolution": 10,
 "stabilityFactor": 3,
 }
```

Configures weight-triggered RFID scanning.

Parameters: - weightTriggered: Enable/disable weight triggering (boolean) - scaleResolution: Weight resolution - stabilityFactor: Weight stability factor

Configure RFID Boot Scan

```
{
 "cmd": "write",
 "bootscan": false,
 "bootscanType":"asynchronous"
 "bootscanInterval": 3000,
 "bootscanDuration": 1000,
 "bootscanEpcEncoded": false,
 "bootscanBank": 0
 }
```

Configures RFID scanning on device boot.

Parameters: - bootscan: Enable/disable RFID boot scan (boolean) - bootscanType: decides RFID boot scan Type synchronous or asynchronous - bootscanInterval: Interval between boot scans in milliseconds - bootscanDuration: Duration of boot scan in milliseconds.

Configure Empty Scan Filtering

```
{"cmd": "write", "skipEmpty": false}
```

Controls whether empty scans are reported.

Parameters: - skipEmpty: Skip empty scan results (boolean)

Configure Duplicate Detection

```
{
 "cmd": "write",
 "skipEmpty": false,
 "checkDuplicates": 2,
 "metadata": false
 }
```

Configures duplicate tag detection and metadata reporting.

Parameters: - skipEmpty: Skip empty scan results (boolean) - checkDuplicates: Duplicate check mode (2 = enabled) - metadata: Include metadata in responses (boolean)

Read Current Configuration

```
{"cmd": "read"}
```

Reads and returns the current device configuration.

OTA Commands

Characteristic UUID: d3edcb21-d7aa-4731-bad8-946651538516

Initiate OTA Update

```
{"cmd": "doOta"}
```

Triggers an Over-The-Air firmware update.

OTA Data

Characteristic UUID: d3edcb21-d7aa-4731-bad8-946651538517

The characteristic accepts firmware data in chunks. See this Python script for reference.

RFID Commands

Characteristic UUID: d3edcb21-d7aa-4731-bad8-946651538518

Set Temperature Limit

```
{"cmd": "setTempLimit", "tempLimit": 80}
```

Sets the maximum operating temperature for the RFID module (in °C). Scans will automatically stop above this threshold.

Parameters: - tempLimit: Temperature limit in Celsius (integer)

Get Temperature Limit

```
{"cmd": "getTempLimit"}
```

Returns the configured temperature limit.

Get Current Temperature

```
{"cmd": "getTemp"}
```

Returns the current temperature of the RFID module.

Get Power Settings

```
{"cmd": "getPower"}
```

Returns the current read and write power settings.

Set Power Settings

```
{"cmd": "setPower", "readPower": 1600, "writePower": 1600}
```

Configures the RFID read and write power levels.

Parameters: - readPower: Read power in centibels (e.g., 1600 = 16.00 dBm) - writePower: Write power (e.g., 1600 = 16.00 dBm)

Start Synchronous Scan

```
{
 "cmd": "startScan",
 "scanInterval": 10000,
 "scanDuration": 1000,
 "epcEncoded": false
 }
```

Starts a synchronous RFID tag scanning operation.

Parameters: - scanInterval: Time between scans in milliseconds - scanDuration: Duration of each scan in milliseconds.

Stop Synchronous Scan

```
{"cmd": "stopScan"}
```

Stops the ongoing synchronous RFID scan.

Start Asynchronous Scan

```
{
 "cmd": "startAsyncScan",
 "scanInterval": 1500,
 "epcEncoded": false
 }
```

Starts an asynchronous RFID scanning operation.

Parameters: - scanInterval: Time between scans in milliseconds

Stop Asynchronous Scan

```
{"cmd": "stopAsyncScan"}
```

Stops the ongoing asynchronous RFID scan.

Peripheral Commands

Characteristic UUID: d3edcb21-d7aa-4731-bad8-94665153851b

Control Display, RGB LED, and Buzzer

```
{
 "peripheral": {
 "display": ["", "show", "me"],
 "rgb": [[25, 0, 0],[0, 25, 0],[0, 0, 50],[25, 0, 0],[0, 25, 0],[0, 0, 50]],
 "buzzer": {
 "time_on": 200,
 "time_off": 50,
 "count": 2
 }
 }
 }
```

Controls multiple peripherals simultaneously.

Parameters: - display: Array of 4 strings for display lines (if supported) - rgb: Array of RGB values [[R, G, B]] where each value is 0-255 - buzzer: - time_on: Buzzer on time in milliseconds - time_off: Buzzer off time in milliseconds - count: Number of beep cycles

Payload Format

Payload format is configurable using the `metadata` flag in the Config payload.

Metadata Enabled + Weight Scale Connected (Full Payload)

```
{
  "device_id": "BLE_NAME",
  "timestamp": "1234567890123",
  "group_id": "unique_group_identifier",
  "tags": [
   {
    "tag": "E280116060000209A1E23456",
    "rssi": -45,
```

```
    "antenna": 1,
    "weight": 1250
   },
   {
    "tag": "E280116060000209A1E23457",
    "rssi": -52,
    "antenna": 1,
    "weight": 1250
   }
  ]
}
```


Metadata Disabled + Weight Scale Connected (No RSSI/Antenna)


```
{
  "device_id": "BLE_NAME",
  "timestamp": "1234567890123",
  "group_id": "unique_group_identifier",
  "tags": [
   {
    "tag": "E280116060000209A1E23456",
    "weight": 1250
   },
   {
    "tag": "E280116060000209A1E23457",
    "weight": 1250
   }
  ]
}
```

Metadata Enabled + Weight Scale Not Connected (No Weight)


```
{
 "device_id": "BLE_NAME",
 "timestamp": "1234567890123",
 "group_id": "unique_group_identifier",
 "tags": [
  {
   "tag": "E280116060000209A1E23456",
   "rssi": -45,
   "antenna": 1
  },
  {
   "tag": "E280116060000209A1E23457",
   "rssi": -52,
   "antenna": 1
  }
 ]
}
```


Metadata Disabled + Weight Scale Not Connected (Minimal Payload)


```
{
 "device_id": "BLE_NAME",
 "timestamp": "1234567890123",
 "group_id": "unique_group_identifier",
 "tags": [
  {
   "tag": "E280116060000209A1E23456"
  },
  {
   "tag": "E280116060000209A1E23457"
  }
 ]
}
```

Reach Us

IoTReady
F3, 3rd Floor, Rich Homes,
Richmond Road, Bengaluru,
Karnataka - 560025

+91 74119 67890
sales@iotready.co