FlowFuse

eBook

The Ultimate Beginner Guide to a Professional Node-RED

Updated December 2024

Introduction

Node-RED is the open source, low-code programming environment that makes it easy to connect hardware, sensors, actuators and services using a visual development environment. This makes it possible for even non-developers to build applications, which has led to its wide popularity. A large library of nodes and flows makes it trivial to connect different types of analog and digital services, including everything from industrial protocols such as OPC-UA, Modbus, and Siemens S7, and databases to weather services. The Node-RED dashboard makes it easy to visualize data in dashboards, and unlike other dashboards, Node-RED dashboards can include actions and events that respond to the data.

Node-RED allows developers to connect a wide variety of endpoints, such as industrial PLCs, APIs, databases, enterprise applications and online services. Users of Node-RED visually drag 'nodes' from a palette that represents the endpoints and then connect them into flows to accomplish the desired task.

Node-RED applications are accessible via a web browser and can be used for a wide variety of use cases, including:

- IoT and IIOT use cases, such as collecting data from different pieces of factory equipment or geographically dispersed devices.
- Creating dashboards that visualize data and allow for event triggers to occur based on the data.
- Automation and integration of digital platforms from chatbots to data integration pipelines.
- Extract, transform and integrate data from many different sources in the enterprise.
- Home automation.
- Integrating data with machine learning models.

Node-RED is an open source project hosted at the OpenJS Foundation. It is made available under the Apache Software License so individuals can use Node-RED free of charge. Participation in the open source project is encouraged to help support the Node-RED community.

This ebook will take you through the history of Node-RED, an indepth look at its key features in an industrial setting, and how you can get started. It is a good starting point for anyone interested in learning more about this popular integration software and making citizen development a reality in your enterprise.

Table of Content

Chapter 1

History of Node-RED

In January 2013, FlowFuse's co-founder and CTO, Nick O'Leary, could have never foreseen that his fun little proof-of-concept project would become Node-RED, an open source low-code environment with millions of deployments in IoT and automation.

He was working in IBM's Emerging Technology Group playing around with capturing data from devices and doing interesting things with it, long before IoT became the ubiquitous term it is today. The team focused on very fast-paced, short, proof-of-concept projects, and were afforded time to learn new skills, innovate and work on side projects.

Nick's background working in the MQTT protocol space before it was known outside of IBM led him to a side project: He wanted some way to visualize mapping messages on an MQTT infrastructure to see how they come in on one topic and get sent out in another.

Starting with web visualization technology and Node.js runtime, relatively new at the time, he spent a day or two putting together a little demo of an application that would connect to an MQTT broker that had an API. Asking what mappings it had, the application drew a visualization in the browser which became a very early version of the topic explorer.

Nick wanted to make it more interactive. Not having the terms nailed down yet, he wanted to drag a blob onto the screen, draw a line, configure it, and then hit a button to apply the transformation. Just 24 hours later, he had a simple browser-based application that could define and apply mappings between MQTT topics.

It very quickly became useful.

Dave Conway Jones, Nick's colleague, took the code to the next step, adapting it to add a serial node to work with the project data he was collecting. Over the next few weeks, as part of the team's work on proof of concepts for customer engagements, they hard-coded different nodes into the palette and the utility of this new application was clear.

With the blessing of IBM management to spend more time on the project, Nick spent a few days redesigning the code to make it easier to write in new nodes, unlocking the ability to quickly add in the function node, change node, and switch node, which became the basic building blocks of the tool.

It became Node-RED when the application was submitted as an idea to map web services visually as part of IBM's new public cloud offering, and it started gathering more traction within the company.
To get it into the hands of a wider audience, IBM supported the decision to make Node-RED open source, and it was published on npm and GitHub. Internally, Node-RED had a one-click deployment to IBM Cloud and was used by its developers to demo the IBM services available on the cloud.

In late 2013, Nick O'Leary demoed Node-RED at a London IoT meetup and word spread among his peers in that community. A week later, at an open source hardware conference, Nick entered a workshop and was shocked to see Node-RED on everyone's screens. The facilitator had seen Node-RED and reworked his workshop so that people didn't have to worry about writing lines of code and were able to do useful things much more quickly.

Word was also spreading among the companies using IBM Cloud. One voiced concerns that Node-RED might have been an IBM-specific technology and suggested that we consider moving it to an open source foundation. This led to Node-RED becoming one of the founding projects in the relaunch of the Node.js Foundation. Having independent governance allowed companies like Hitachi to contribute to the project and have an equal voice in its development.

For software developers, time spent writing boilerplate code is not time adding value to the application they're building. With low-code, Node-RED abstracts all that boilerplate so they can focus on the business problem.

Device manufacturers paid attention when Node-RED was installed on the Raspberry Pi image, with its low-code accessibility attracting a broad range of people from systems engineers building automation to IoT hobbyists.

Now with millions of deployments, Node-RED continues to collect, transform, and integrate data through visualized dashboards. And, as this open source community grows it remains rooted in the two pillars of extensibility and a low-code, with an ever-expanding flow library to empower users to collaborate and build their projects.

Chapter 2

Getting Started with Node-RED

Node-RED can run on most modern computer systems including your local computer running Windows, MAC or Linux, devices or in the cloud. The Node-RED website has instructions on how to install it to run:

- Locally
- On Raspberry Pi
- Using Docker
- From the source on GIT
- BeagleBone Boards
- Android
- AWS
- Microsoft Azure

However, the easiest way to get started is to get Node-RED running on FlowFuse Cloud. When you sign up as a new user you are enrolled in a free trial and a Node-RED instance will be started for you within a minute. Once that instance has booted up you access Node-RED by pressing "Open Editor".

Creating your First Flow

A Flow is represented as a tab within the editor workspace and is the main way to organise nodes. The term “flow” is also used to describe a single set of connected nodes informally. So a flow (tab) can contain multiple flows (sets of connected nodes). A Node is the basic building block of a flow. Nodes are triggered by either receiving a message from the previous node in a flow or by waiting for some external event, such as an incoming HTTP request, a timer or a GPIO hardware change. They process that message, or event, and then may send a message to the next nodes in the flow. A node can have at most one input port and as many output ports as it requires.

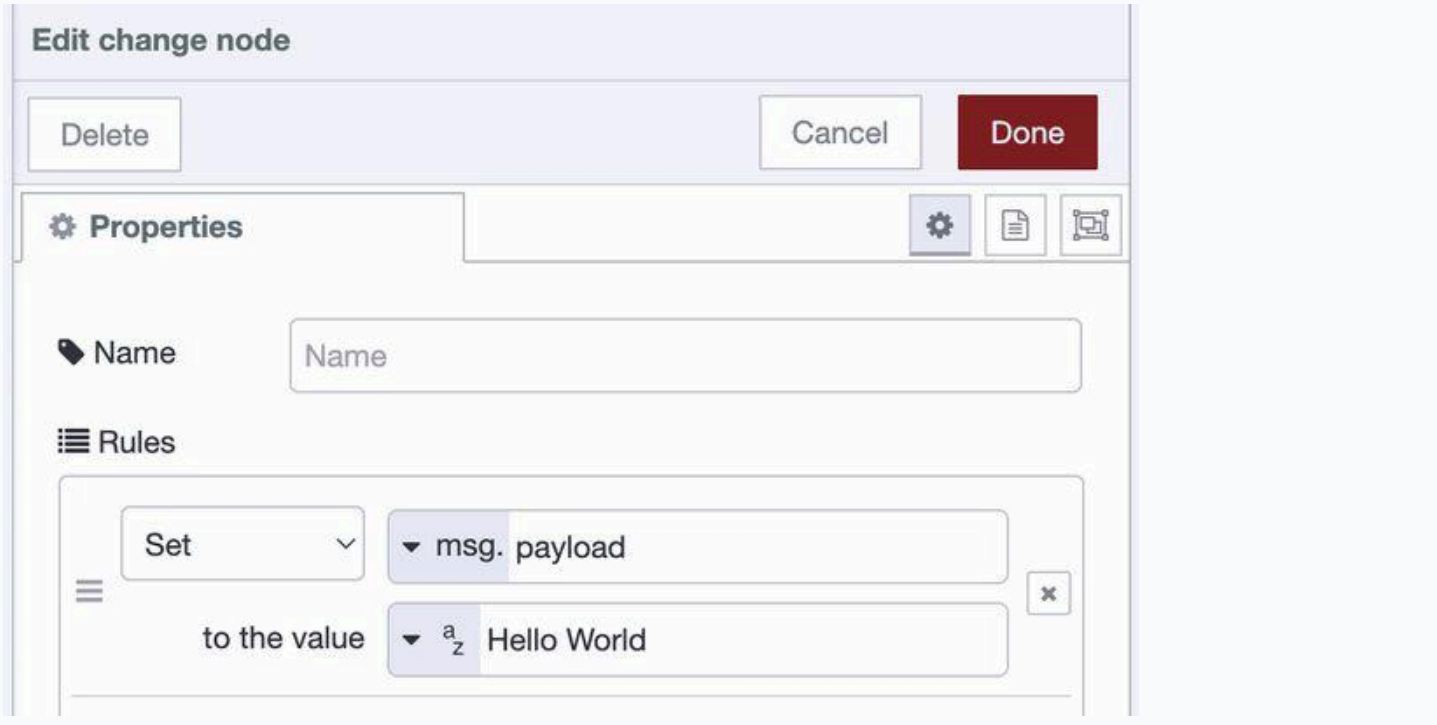
Let's take a look at how to create a simple flow.

In this example, we'll create a simple "Hello World" endpoint. To do this, we'll use the http in, http response, and the change nodes, which can be found in the common nodes menu on the left of Node-RED.

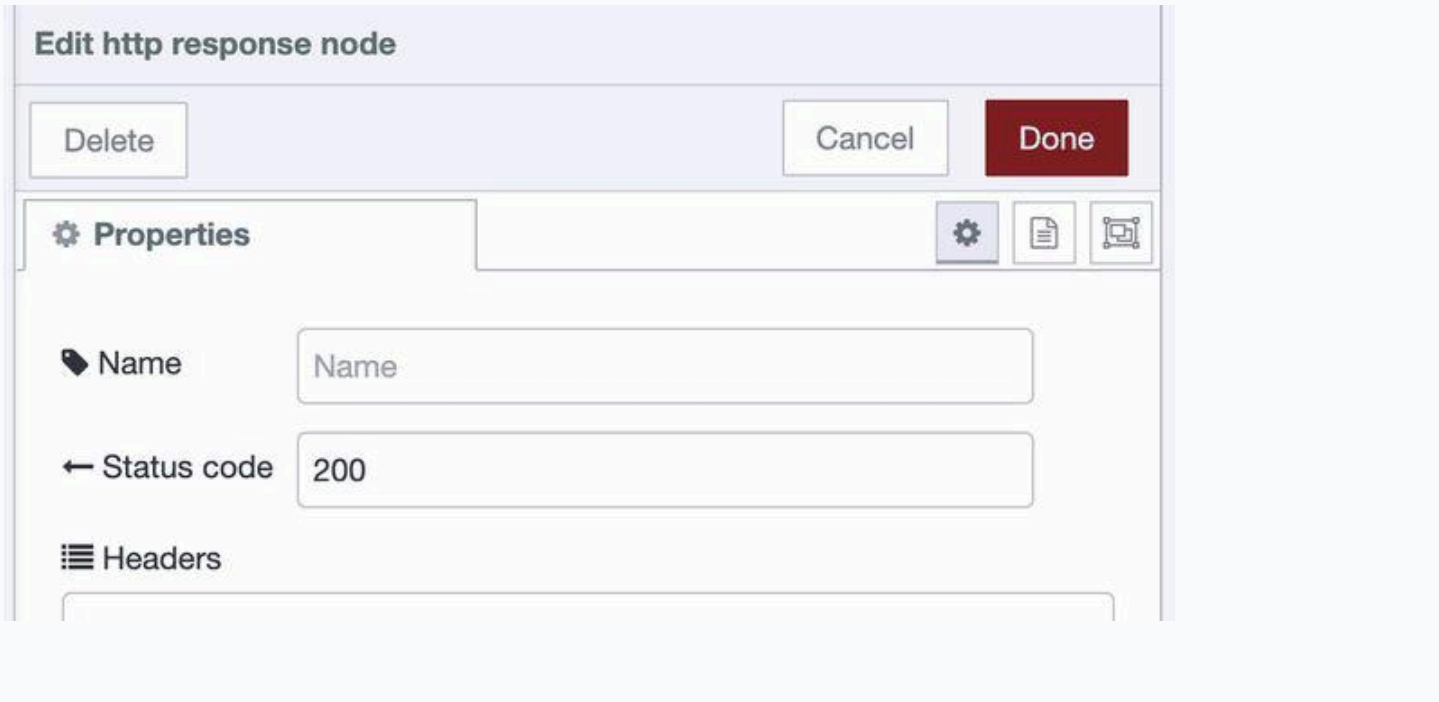
First, drag an http in node into the editor. This node will listen for incoming HTTP requests. Next drag in the "change" and the http response node into the editor. Connect the http in node to the change node and connect the change node to the http response node. Hopefully, your flow looks similar to this:

To configure the http in node, double-click on it to open its properties. Here, you can set the URL that the node will listen to, as well as the method (GET, POST, etc.). In this example, we'll set the URL to /hello and the method to GET.

Now we need to set what the endpoint will respond with, we will do that in the change node. Double-click the change node then add "Hello World" to the field which says "to the value". It should look like this:



To configure the http response node, double-click on it to open its properties. Here, you should set the "Status Code" to be 200. This is not vital for the demo to work but it's good practice to return the correct codes when something connects to an API. Status code 200 means the API responded OK. This is how your http response node should look:



You can read more about HTTP response codes in this article.

Testing Your Flow

Now that we have our flow set up, we can deploy it by clicking the "Deploy" button in the top right corner of the editor. Once the flow is deployed, you can test it by opening up a web browser, if you installed Node-RED using npm navigate to "http://localhost:1880/hello". If you are working on FlowFuse take the URL of your project and add "/hello" to the end, it should looks something like this "https://your-instance-name.flowfuse.cloud/hello". You should see "Hello World!" displayed in the browser.

Debug Output

One of the most powerful features in Node-RED is the ability to debug your flow, this can be done by adding a debug node to your flow and connecting it to the nodes you want to debug. When a message is sent through the connected node, the debug node will print the message in the debug sidebar in the right side of the editor. This can be very helpful when trying to understand how a flow is working or troubleshoot any issues.

The Palette Manager

In addition to the built-in nodes, Node-RED also has a palette manager feature which allows users to easily install and manage additional nodes from the community. To access the palette manager, go to the menu in the top right corner and select "Manage Palette". Here, you can search for and install new nodes, as well as update or remove existing ones. This is a great way to extend the functionality of Node-RED and add new capabilities to your flows.

Import the flow

If you want to view this flow you can import it using the code below. Copy the code then select Import from the top right menu in Node-RED. Paste the code into the field then press Import.

Unset

```
[
 {
  "id": "a742e7a95697bb40",
  "type": "http in",
  "z": "9e9af3caa4dc14d3",
  "name": "",
  "url": "/hello",
  "method": "get",
  "upload": false,
  "swaggerDoc": "",
  "x": 180,
  "y": 200,
  "wires": [
   [
    "883e7d597f7c7c4b"
   ]
  ]
 },
 {
  "id": "aca024dcb79bdb92",
  "type": "http response",
  "z": "9e9af3caa4dc14d3",
  "name": "",
  "statusCode": "200",
  "headers": {},
  "x": 500,
  "y": 200,
  "wires": []
```

```
 },
 {
   "id": "883e7d597f7c7c4b",
   "type": "change",
   "z": "9e9af3caa4dc14d3",
   "name": "",
   "rules": [
     {
       "t": "set",
       "p": "payload",
       "pt": "msg",
       "to": "Hello World",
       "tot": "str"
     }
   ],
   "action": "",
   "property": "",
   "from": "",
   "to": "",
   "reg": false,
   "x": 340,
   "y": 200,
   "wires": [
     [
       "aca024dcb79bdb92"
     ]
   ]
 }
]
```

Now that we've covered the basics, you should be able to create flows for your own use cases. You can check out the Node-RED Library to find new nodes, share your flows and see what others have done with Node-RED.

In addition, FlowFuse offers a curated collection of certified nodes, ensuring top-notch quality, security, and support. Explore FlowFuse Certified Nodes and benefit from a robust ecosystem while maintaining operational reliability.

Chapter 3

Node-RED Core Nodes

In Node-RED, the core nodes are the set of nodes that are included with the Node-RED runtime by default without the node install procedure. These nodes are maintained and supported by the Node-RED development team and are intended to provide the basic building blocks for creating Node-RED flows.

Core nodes include nodes for basic functionality like input/output, processing, and control flow. They are the foundation upon which more complex workflows can be built, and they are essential to the operation of Node-RED.

Following are some of the more widely used core nodes.

Inject

The Inject node is the beginning of many flows that are triggered manually. The box to the left of the node sends a message to connected nodes. For that reason it's often used for debugging too, to inject values at a point of choosing. The message item can be empty too. The message to send defaults to the timestamp as payload and an empty topic.

Message properties can be set to flow, or global variable values, and many other types, including JSONata expressions.

Flows can also be triggered once right after Node-RED starts the flows or with a delay. This is useful to set an initial state from a flow on boot.
An inject node can also start a flow based on a schedule. The schedules have capabilities mimicking cron. In the bottom section of the properties pane the repeat section one can select "at a specific time".

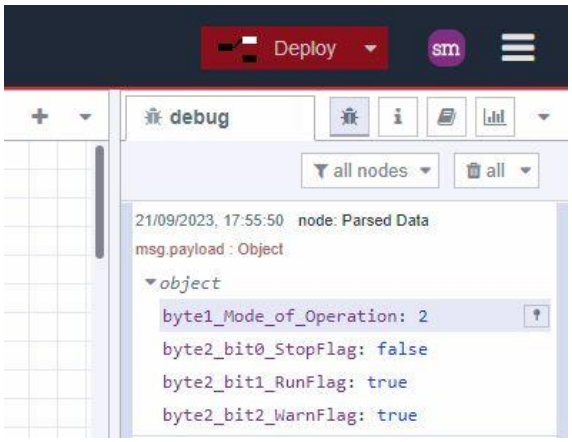
Repeating the measure on an interval is done by selecting "interval" in the "repeat" section. On a schedule requires an input higher than 1 and below 2^31. When the repeat value is 0 or below Node-RED will not display an error.

Examples of the Inject Node

Debug

The Debug node is used to understand the message and data traveling through your flows. During development, it is highly recommended to add Debug nodes at key points in your flow so that you have visibility and understanding of what is being passed around.

In the Node-RED editor, debug messages can be viewed in the right-hand sidebar panel, under the debug icon.



Examples of the Debug Node

Change

The Change node in Node-RED is used for modifying the content of messages within a flow. It allows you to add, remove, modify, or set message properties and payload values, making it a fundamental node for data transformation and manipulation. The Change node is essential for preparing data for further processing, formatting messages for specific outputs, and adapting data to suit the requirements of downstream nodes in a flow.

Examples of the Change Node

Spilt

Granular data processing is important in IoT use cases as multiple tags, for example, might be sent in one request to a server. Or, when a SQL query results in many results that need individual processing. In Node-RED, flows are message-based, but a message can be split in multiple messages if needed using the Split node. The split node is one of the core nodes in Node-RED, thus installed by default. It's a fundamental building block for powerful automation.

The Split node is used to divide a single message into multiple messages based on defined rules.
The Split node in Node-RED is used to split an incoming message object into several different message objects. The incoming object can be a simple string, an array, or an object. The Split node will split the object based on the following criteria:

- **String:** The Split node will split the string on a delimiter character. The delimiter character can be specified in the node's configuration.
- **Array:** The Split node will split the array into a series of messages, each containing one element of the array.
- **Object:** The Split node will split the object on the keys of the object. The keys of the object can be specified in the node's configuration.

The Split node can be used to process data in a variety of ways. For example, it can be used to split a string of text into a series of messages, each containing one word of the text. It can also be used to split an array of data into a series of messages, each containing one element of the array.

The following are some examples of how the Split node can be used:

1. Splitting a string of text into a series of messages, each containing one word of the text.
2. Splitting an array of data into a series of messages, each containing one element of the array.
3. Splitting an object on the keys of the object, to create a series of messages, each containing one key-value pair of the object.
4. Splitting a message on a delimiter character, to create a series of messages, each containing one part of the message.

The Split node is a powerful tool that can be used to process data in a variety of ways. It is a valuable addition to any Node-RED flow.

Examples of the Split Node

Delay

The Delay node allows you to introduce a delay in the flow of messages between nodes. It can be useful in various scenarios where you need to control the timing of message processing. For example, the delay node can limit the rate at which messages are processed downstream or throttle the flow of messaging. Both can be useful for interacting with external systems that may have limitations in place.

Here are some other use cases for using the Delay node:

Batch Processing: If you're dealing with a stream of incoming data that you want to process in batches, you can use the Delay node to introduce a delay between groups of messages. This can be helpful when you need to aggregate or analyze data in chunks.

Sequential Processing: Sometimes you need to ensure that messages are processed in a specific order. The Delay node can be used to enforce a sequence of message processing, especially when dealing with asynchronous systems that might not guarantee order.

Simulation and Testing: In testing and simulation scenarios, you might want to mimic real-world timing conditions. The Delay node can help you introduce delays that simulate actual conditions, allowing you to test how your system behaves over time.

Time-based Triggers: You can use the Delay node to trigger actions at specific time intervals. For instance, you might want to send a status update every hour or perform a cleanup task at the end of the day.

Circuit Breaker: The Delay node can be employed as a simple form of circuit breaker. If a downstream system is failing or experiencing issues, you can introduce a delay before retrying, giving the system some time to recover.

Examples of the Delay Node

Exec

Node-RED is written in Javascript, as are the custom nodes in the Flows Library. If you prefer to use programs written in other languages, Node-RED by default comes with the Exec node. This node allows you to run a command as if you're on the command line. The exec node has one input, and three outputs.

Examples of the Exec Node

Link

The Link In and Link Out nodes are used to help Node-RED developers to organize their flows to make them easier to understand. The Link nodes can be used to connect two sets of Nodes but the connection is only shown when one of the nodes is selected. This allows developers to group nodes that complete a specific function. The Link node will connect two groups but visually it isn't apparent until one of the nodes is selected.

Examples of the Link Node

Switch

The Switch node allows you to route messages based on certain conditions. It acts as a decision-making tool within your flow, allowing you to define rules for directing messages to different output branches.

Here are some common use cases for using the Switch node in Node-RED:

Message Filtering: You can use the Switch node to filter messages based on specific criteria. For example, you might want to filter out messages that don't meet a certain threshold or that don't contain certain keywords.

Conditional Routing: The Switch node enables you to route messages down different paths in your flow based on conditions. You can set up rules that determine which output branch a message should be sent to, depending on its content or properties.

Event Processing: If you're working with events or data streams, the Switch node can help you process different types of events differently. For instance, you might have events related to temperature and humidity readings, and you want to process them separately.

Value Conversion: In cases where you need to convert values from one format to another, the Switch node can route messages to different converters based on the incoming value's properties.

Error Handling: When working with data or APIs, you might receive error messages that need to be handled differently from regular data. The Switch node can direct error messages to a separate branch for appropriate handling.

Language or Region-Based Processing: In applications involving localization or multilingual support, the Switch node can route messages based on language or region information in the message.

Examples of the Switch Node

Function

Function nodes are an essential part of Node-RED. They allow you to write custom JavaScript functions that can be used in your Node-RED flows. The messages are passed in as a JavaScript object called msg. By convention, it will have a msg.payload property containing the body of the message.

The function is expected to return a message object (or multiple message objects) but can choose to return nothing to halt a flow. The On Start tab contains code that will be run whenever the node is started. The On Stop tab contains code that will be run when the node is stopped.

If the On Start code returns a Promise object, the node will not start handling messages until the promise is resolved.

Read more about the benefits and drawbacks of using Function nodes.

Comment

Maintaining flows over longer periods can save you time and interpretation errors if you add the comment node to your flows.

The 4 core benefits of adding the Comment node to your Node-RED flows are:

1. Improved readability: Comments can help to make flows easier to understand, especially when adding easy to miss context and explanation like implicit requirements of the flow. This can be especially helpful for complex flows with link nodes, or when you're editing in many tabs.
2. Enhanced maintainability: Comments can help to make flows easier to maintain by providing a record of the flow's purpose and functionality. This can be helpful when making changes to the flow or when troubleshooting problems, particularly if you collaborate with multiple team members.
3. Improved debugging: Comments can help to make debugging easier by providing information about the expected behavior. This can be helpful when tracking down errors and identifying the source of problems, or reasoning errors when previously developing the flow.
4. Increased documentation: Comments can be used to document the code, providing additional information about the code's purpose, functionality, and usage. Beyond what the flow or nodes do in the sequence, business logic and requirements documented next to the flow increased developer efficiency.

The comment node can be added to any open space in the editor, and it's advised to add a comment to all flow groups.

As the comment node will not take up more space if you write a larger comment, Node-RED allows you to be more explicit and elaborate further in your comments. ASCII diagrams or so do not distract from your flow, and as such you're encouraged to add these.

Examples of the Comment Node

Chapter 4

Centralizing Node-RED Managment

Managing multiple Node-RED instances can quickly become complicated as operations grow. Each new instance adds complexity, from configuration issues to security concerns. These challenges highlight the need for a centralized solution to simplify management and improve efficiency.

1. Deployment and Configuration Management: Setting up Node-RED instances on a server requires technical knowledge and ongoing maintenance. As the number of instances grows, maintaining them can become time-consuming and resource-intensive.

2. Egde Node-RED Management: Managing Node-RED instances on edge devices introduces additional challenges, such as the need for on-site troubleshooting when issues arise.

3. Monitoring and Troubleshooting: Keeping track of the health and performance of multiple instances requires constant attention. Checking logs across different instances can become overwhelming.

4. Security Management: Each instance requires its own security settings. Ensuring that all instances are secure and up to date can be a difficult task, especially as the number of instances increases.

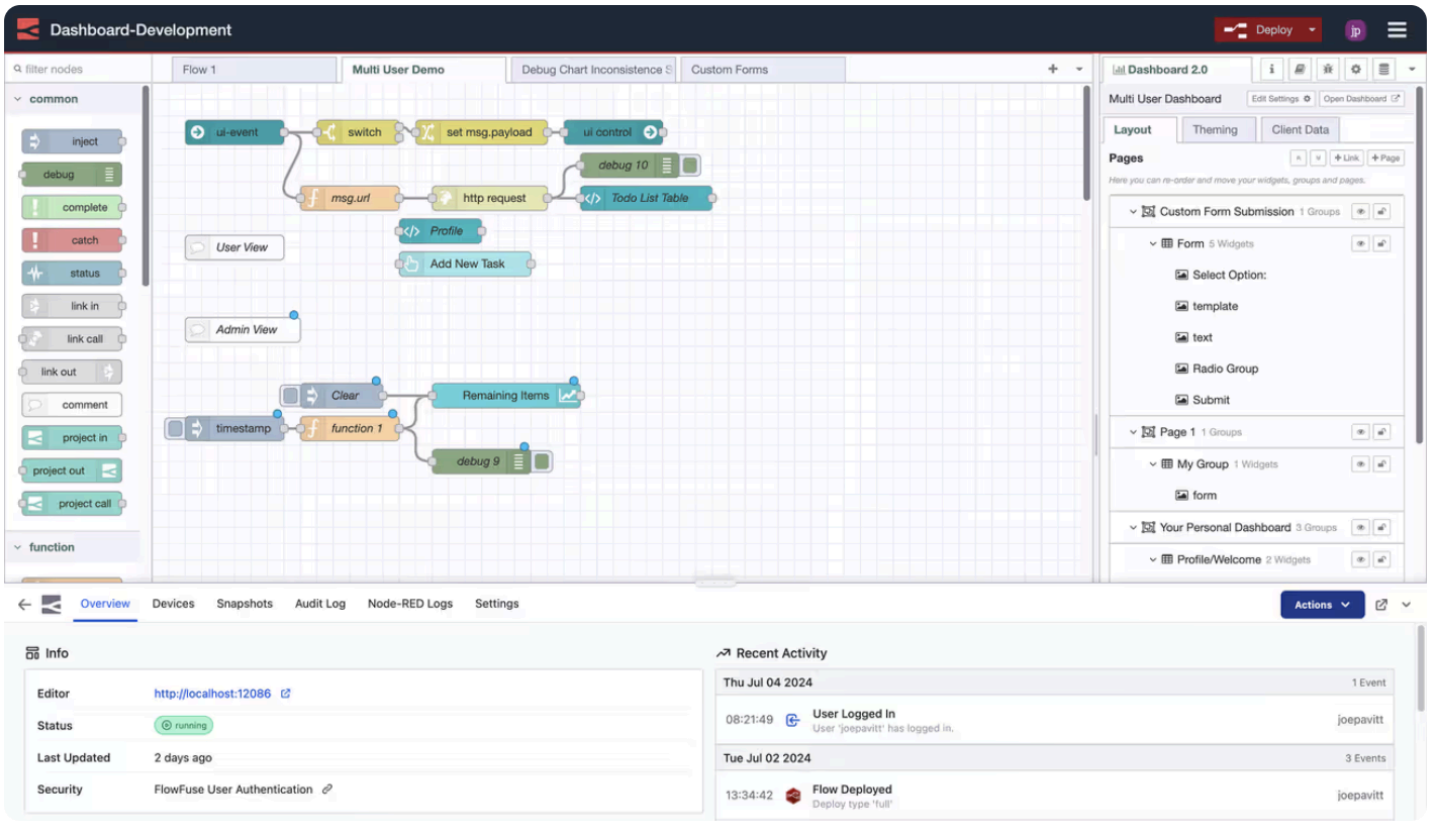
5. Backup and Recovery: Having a solid backup and recovery plan is critical. If a system crashes, you need a way to quickly restore it without losing important data.

6. Scaling: As applications grow in complexity, scaling Node-RED instances becomes necessary. This requires expertise in server management and the ability to handle multiple instances efficiently.

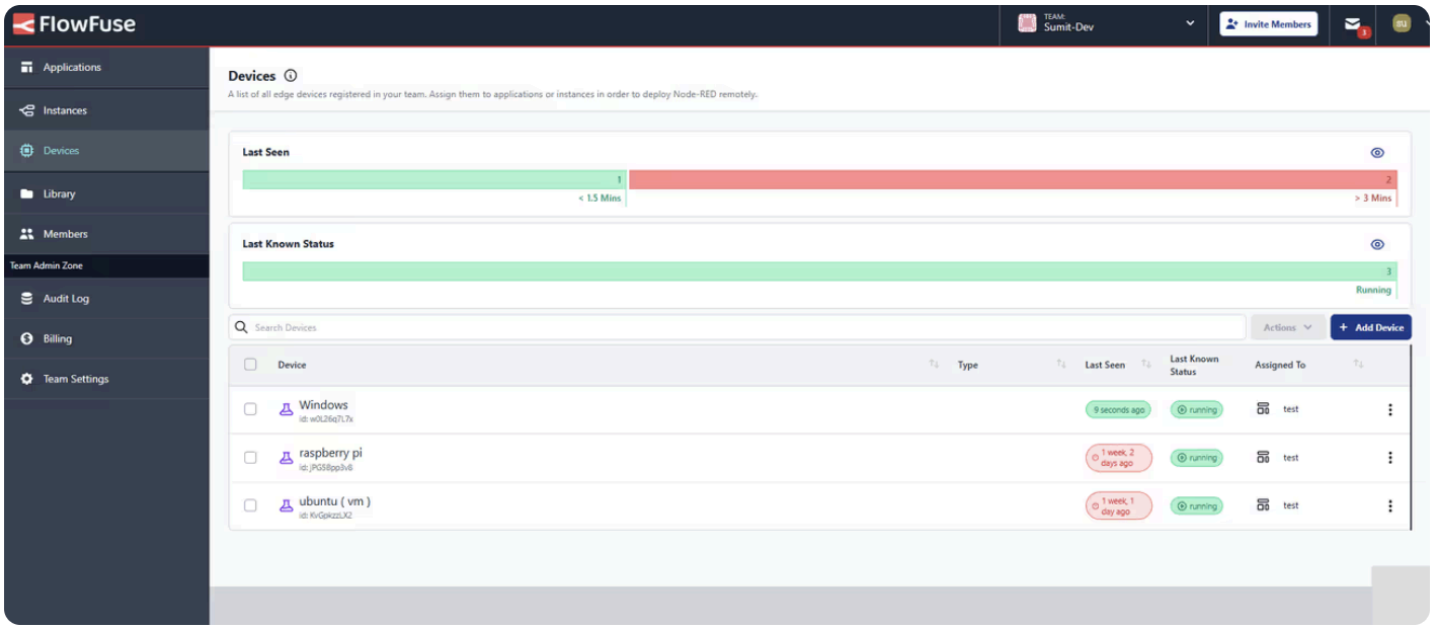
7. Ensuring High Availability: In production environments, keeping all Node-RED instances running smoothly and avoiding downtime is essential which also requires high technical exepertise

A centralized platform is essential to handle deployment, configuration, and management efficiently, providing a visual interface to maintain and update instances.

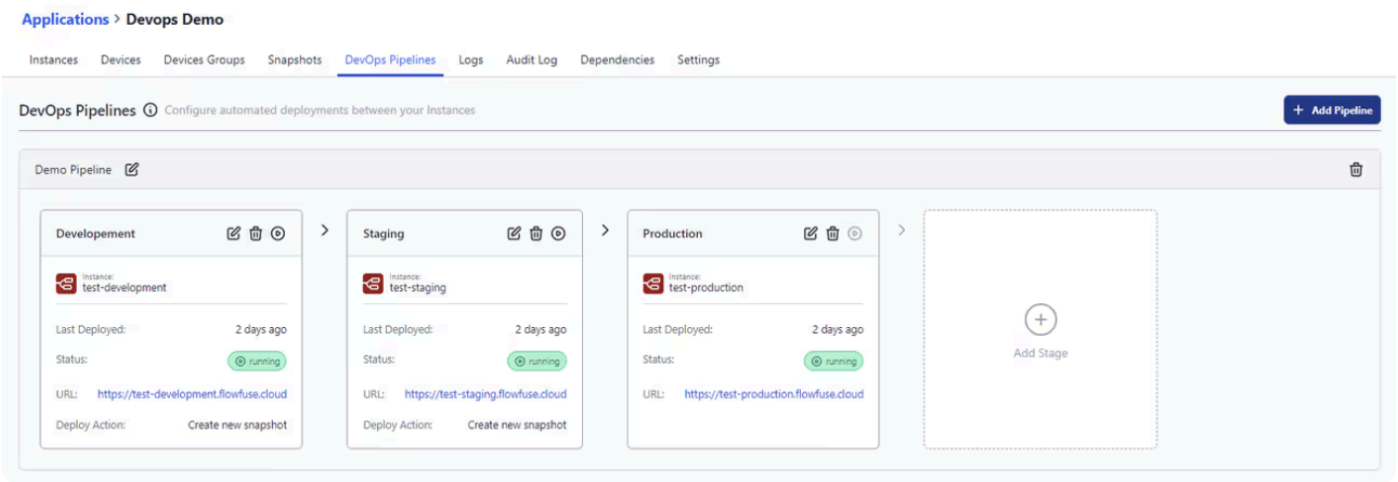
With FlowFuse, you can organize your Node-RED instances into teams for improved collaboration, allowing seamless teamwork on projects without the need to navigate between different instance locations physically. Create as many teams as needed, ensuring that instances are organized based on the team members assigned to them. Additionally, ensure that each member has the correct permissions they require through role-based access control (RBAC), providing precise management of access and responsibilities.



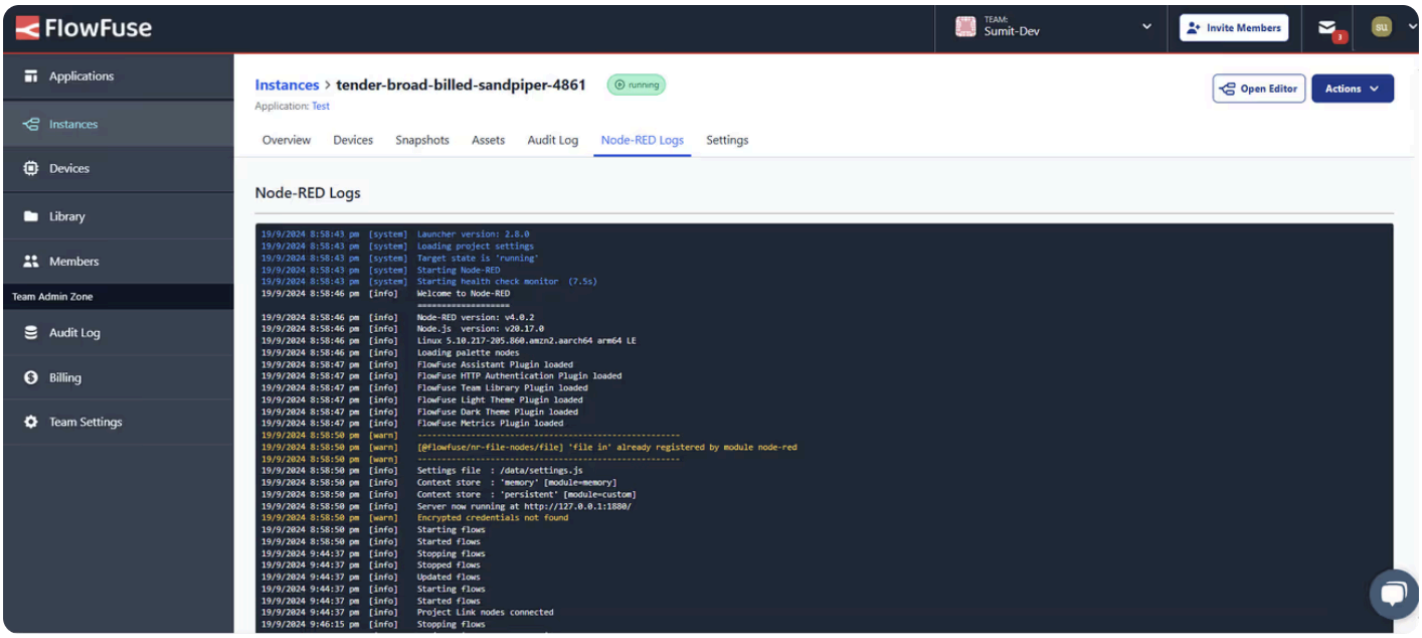
FlowFuse also simplifies the monitoring and controlling of edge devices through the FlowFuse Device Agent, which quickly connects your devices to the cloud platform and allows you to build and monitor applications remotely.



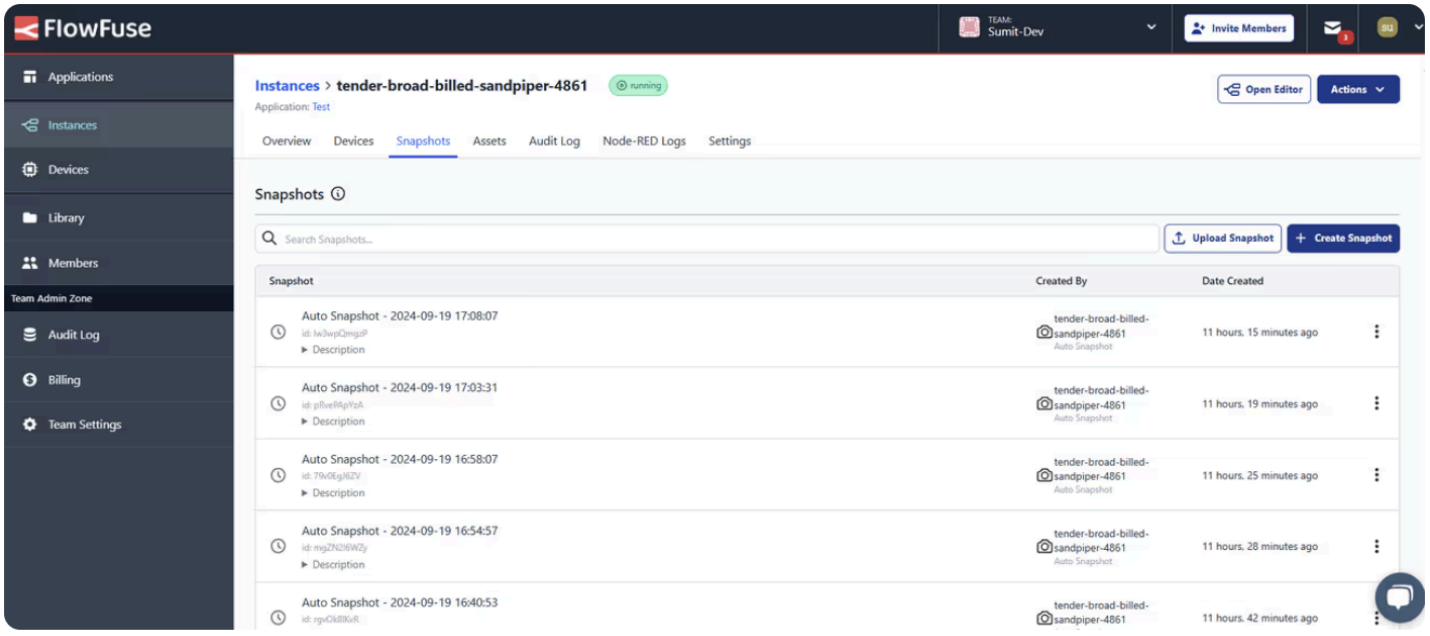
Create DevOps pipelines that ensure your application is well-tested and evaluated before deployment to production. Deploying the same flow to hundreds or thousands of devices becomes effortless with these pipelines.



Efficiently monitor logs for each instance and receive instant email alerts if any crashes occur, facilitating quick troubleshooting.



Add high availability features to your instances, ensuring smooth and efficient operation of your production applications. The platform includes an auto-snapshot feature that lets you recover from accidental changes to flows, ensuring you always have a backup of your application.



Chapter 5

Capturing Data from Edge Devices

While cloud computing has revolutionized data access and analysis, not all data can be accessed from the cloud. In many scenarios, data collection from the edge – the location where data is generated – is essential for real-time decision-making or process observability.

FlowFuse enables data to be collected through Node-RED. Data can be processed locally on the edge or sent on to other services. FlowFuse doesn't rely on continuous connections to the cloud, making it a good choice for locations with unreliable internet connectivity. Use cases like real-time monitoring of critical systems, proactive maintenance, and improved operational efficiency are now possible to implement.

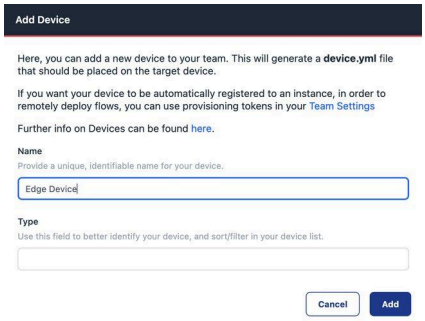
Installing the FlowFuse agent

To manage the capturing of data on the edge you need to first install the FlowFuse agent. It's installed on your device to manage the communication between the edge device and the FlowFuse server, manage the installation of Node-RED, and its execution environment, and facilitate communication between devices and the cloud.

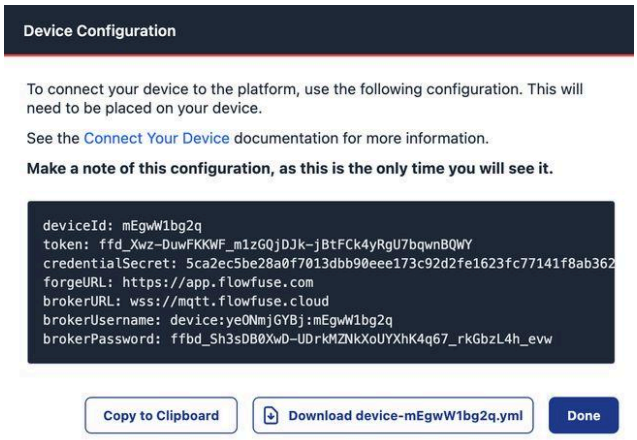
The device agent can run anywhere you can run a Docker container or Node.JS runtime (version 16.0+) can be installed.

Registering a device on FlowFuse

For the edge device to know what it's supposed to do, it needs to listen to the FlowFuse commands. The agent's configuration is provided by a device.yml file from FlowFuse. Go to the team you'd like to add an edge device to, and select "Devices" on the left-hand menu, followed by the "Add Device" button.



FlowFuse will prompt you to add a name (required), and a type (not required). When you've clicked Add you'll get a new dialog to download the required file.



Install the FlowFuse agent through Docker

If your device supports it, the fastest way to run the FlowFuse agent is with containers. Assuming you've already got Docker installed, there are two steps to follow: first, move the device YAML file downloaded from FlowFuse to the edge device and save it in /opt/flowfuse/device.yml. Start the agent by running:

```
Unset
docker run --mount type=bind,src=/path/to/device.yml,target=/opt/flowfuse-device/device.yml -p 1880:1880 flowfuse/device-agent:latest
```

Note that for production cases, ensure the container is restarted on reboot. Docker can do this for you, please follow their guide.

Install the FlowFuse agent with npm

To install the agent through NPM, you'll need a Node.JS version of 18.0 or later. Open a command prompt and run: npm install -g @flowfuse/device-agent.

This will install the FlowFuse Device Agent as a global npm module, making the flowfuse-device-agent command available in any directory on your system.
Once the installation is complete, you must configure the Device Agent to connect to your FlowFuse instance. In this guide, you've previously downloaded the device.yml file that's needed now.

On Linux or Mac, move the file to /opt/flowfuse-device/device.yml, and for Windows-based systems, move the file to c:\opt\flowfuse-device\device.yml.

Afterward, start the agent with: flowfuse-device-agent.

This will launch the Device Agent and connect it to your FlowFuse instance. The Device Agent will wait for instructions on which flows to run.

Programming flows for the edge

Now the agent is running, the FlowFuse platform will show it has contacted back to the platform and is ready to do some work. First, add it to the application and start the developer mode. That enables the device editor and provides you secure access to the editor anywhere in the world for everyone in the FlowFuse team with the right access role.

When the development is done, be sure to create a snapshot of the developed flows to create a point-in-time backup, or to roll the snapshot out to many other devices later.

Chapter 6

Data Visualization with Dashboards

The Node-RED Dashboard is a vital tool for creating live dashboards and user interfaces for Node-RED flows. The original version however was built on the no-longer-maintained Angular v1. This outdated foundation presented potential security issues that cannot be addressed with patches.
To counter this, FlowFuse launched a new version of the Dashboard. Dashboard 2.0 will maintain the core principles of open-source, community-driven development under the Apache 2.0 license, and is designed to safely usher the Node-RED community into the future of data visualization.

Dashboard 2.0 represents a comprehensive reconstruction of the original framework, now based on VueJS. The revamped version incorporates complete responsiveness, extending from desktop to mobile devices. Quality of life improvements have been implemented across the existing widget collection, with several new additions to enhance user experience.

Notable features include Dynamic Markdown, Tables & Notebooks, UI Chart improvements, and a custom video player. In addition, it introduces a groundbreaking feature – Personalized Multi-User Dashboards, exclusively available on Node-RED Dashboard 2.0 when running on FlowFuse Cloud. This new feature allows users to build dashboards that provide unique data to each user, build admin-only views, and track user activity.

Installation

FlowFuse Node-RED Dashboard 2.0 is available in the Node-RED Palette Manager. To install it:

- Open the menu in the top-right of Node-RED
- Click "Manage Palette"
- Switch to the "Install" tab
- Search node-red-dashboard
- Install the @flowfuse/node-red-dashboard package (not node-red/node-red-dashboard)

The nodes will then be available in your editor for you to get started.
If you want to use npm to install your nodes, you can instead follow these instructions.

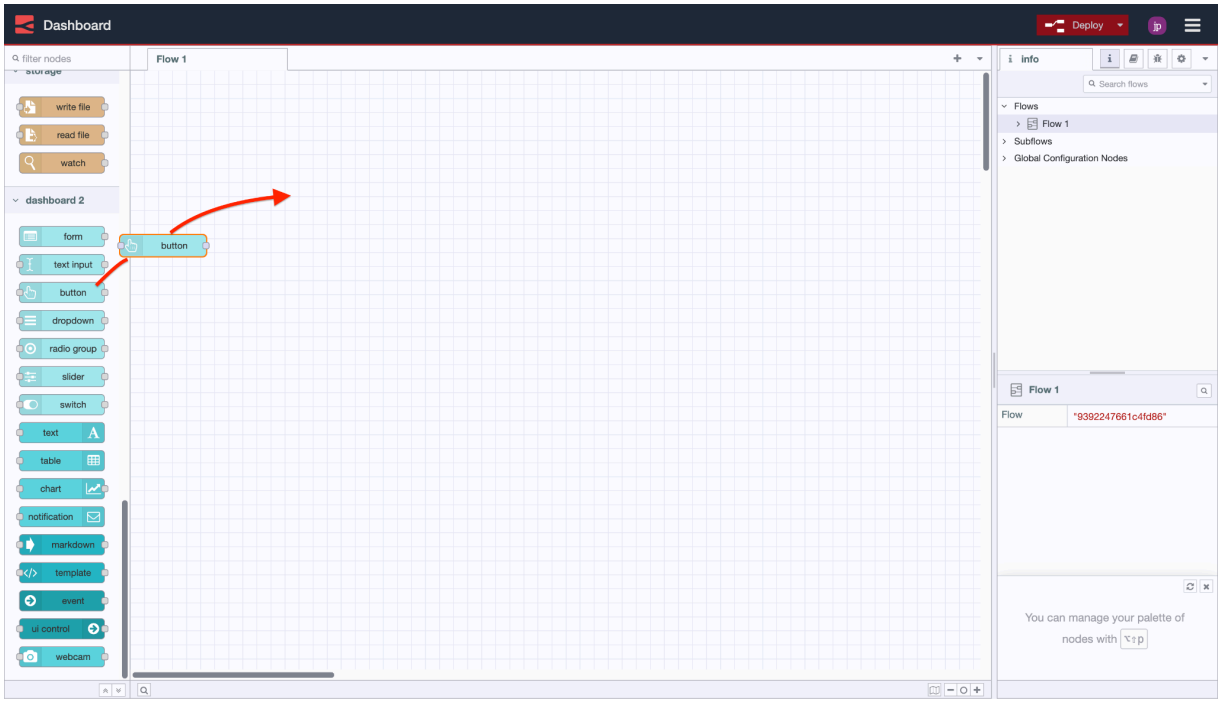
Dashboard Hierarchy

Each Dashboard is a collection of widgets (e.g. charts, buttons, forms) that can be configured and arranged in our own User Interface. The hierarchy of a Dashboard is as follows:

- **Base:** Defines the base URL (e.g. /dashboard) for your Dashboard.
- **Page:** A given page that a visitor can navigate to, URL will extend the base, e.g. /dashboard/page1. Each page can also have a defined, unique, Theme which controls the styling of all groups/widgets on the page.
- **Group:** - A collection of widgets. Rendered onto a page.
- **Widget:** - A single widget (e.g. chart, button, form) created in Dashboard.

Building a Dashboard

To get started, drop any widget from the flows on the left-side of Node-RED onto the editor.



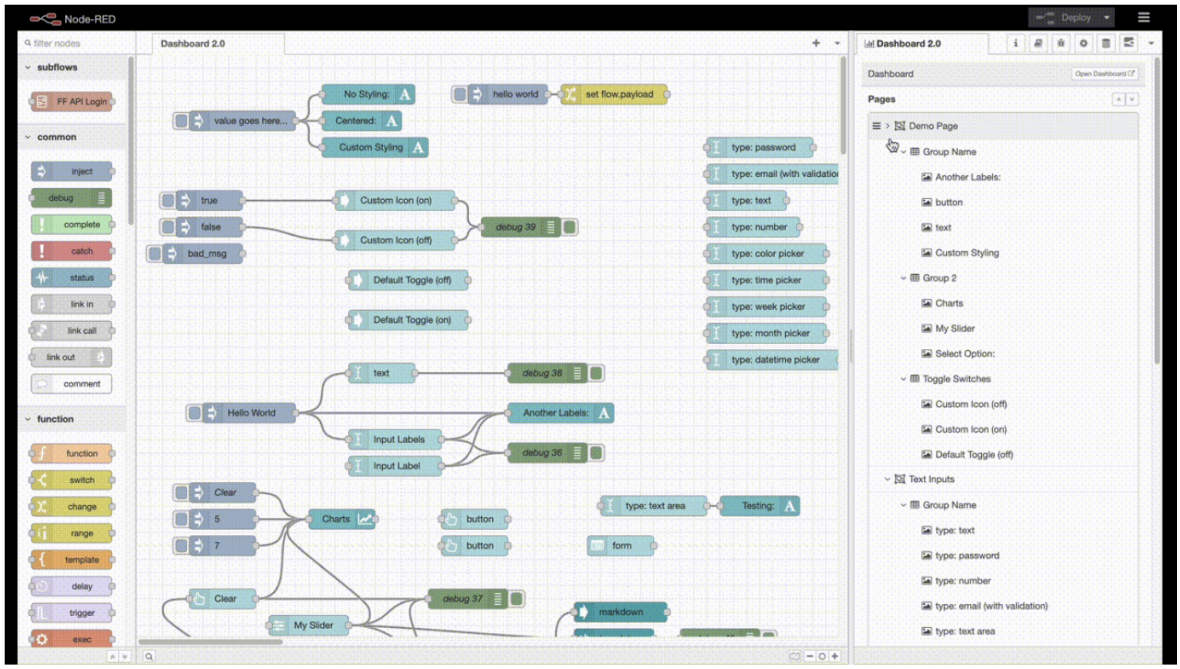
Initial Manual Step

For the first widget you add, you will need to double click your new widget, and create your first group, page, theme and base. Once you have created these, you can add more widgets to the same group freely, and add any more groups/pages as you need to.

Note: we do have a plan to automate this initial manual step of creating group/page/base.

Managing your Dashboard

Once you have created a few widgets, groups and/or pages, you may want to make use of the Dashboard 2.0 sidebar. Sidebars are available on the right-hand side of the Node-RED interface. The down chevron provides access to more if you don't see Dashboard 2.0 in the initial tabs.



The sidebar offers you a central place to manage your pages, groups and widgets. You can control the order that each of these appears in their respective parents, and even move widgets between groups, and groups between pages.

For more details on Dashboard 2.0, FlowFuse has published rich documentation that covers all available nodes, insights into the framework's architecture, and guidelines for contributing to the project. You can also watch the on-demand webinar.

Chapter 7

Securing Node-RED

When it comes to securing Node-RED applications and its editor, ensuring that your flows and data are protected from unauthorized access can feel like a daunting task. Even after investing considerable time, achieving the right level of security often remains a complex challenge. For enterprises, this goes far beyond access control— security is a cornerstone of protecting sensitive data, maintaining operational continuity, and meeting strict regulatory requirements. A robust security framework not only prevents breaches but also safeguards intellectual property, preserves trust, and shields the organization from costly cyber threats.

Here are 9 ways FlowFuse simplifies and strengthens your Node-RED deployments, ensuring you're fully protected without the hassle.

1. Default Security Measures to Keep Your Environment Safe

Securing your Node-RED applications is essential to protect sensitive data, proprietary business logic, and critical systems from unauthorized access or cyberattacks. Without proper safeguards, the risks of data loss, operational disruptions, and reputational damage are significant. FlowFuse implements robust security measures right from the start, ensuring your deployments remain safe. Data is encrypted during transmission, and rate limiting prevents traffic overloads, ensuring smooth operations.

secure tunnelling facilitates safe communication between your edge devices and the FlowFuse platform. To ensure that only authorized personnel have access to your FlowFuse team, strong login authentication measures are included, which can be further enhanced with multi-factor authentication as needed.

FlowFuse features a user-friendly interface that allows you to customize your security settings and design a strategy tailored to your needs.

To learn more about how FlowFuse safeguards your data privacy and security, see the detailed security statement.

2. *Single Sign-On (SSO) Integration*

Every organization relies on various tools and platforms to enhance productivity and efficiency. Providing seamless access to these resources is vital for optimizing workflows. That's where Single Sign-On (SSO) comes in, it simplifies the onboarding and offboarding processes.

With SSO, team members can log in using their existing credentials, eliminating the hassle of remembering multiple passwords. This streamlines their login experience and enables them to be productive from day one.

To implement SSO for your self-hosted FlowFuse, refer to the following resources:

- How to Set Up SSO SAML for Node-RED
- How to Set Up SSO LDAP for Node-RED

3. *Two-Factor Authentication (2FA)*

Given the vulnerabilities of passwords in today's digital landscape, Two-Factor Authentication (2FA) has become an indispensable security measure.

By enabling 2FA, even if someone gets hold of your password, they'll still require a second form of verification—like a code sent to your phone—to access your account. This simple yet powerful layer of security ensures your data is much safer from unauthorized access. Note however that when Single Sign-On (SSO) is enabled, 2FA will be replaced by SSO's authentication process.

To set up 2FA in FlowFuse, you'll need to head over to ***User Settings > Security > Two-Factor Authentication.*** It's as simple as clicking the "Enable Two-Factor Authentication" button, scanning the QR code displayed on the platform with your authenticator app, and then entering the code from your app back into FlowFuse. Once you've done that, 2FA will be up and running, adding that extra layer of security to your account.

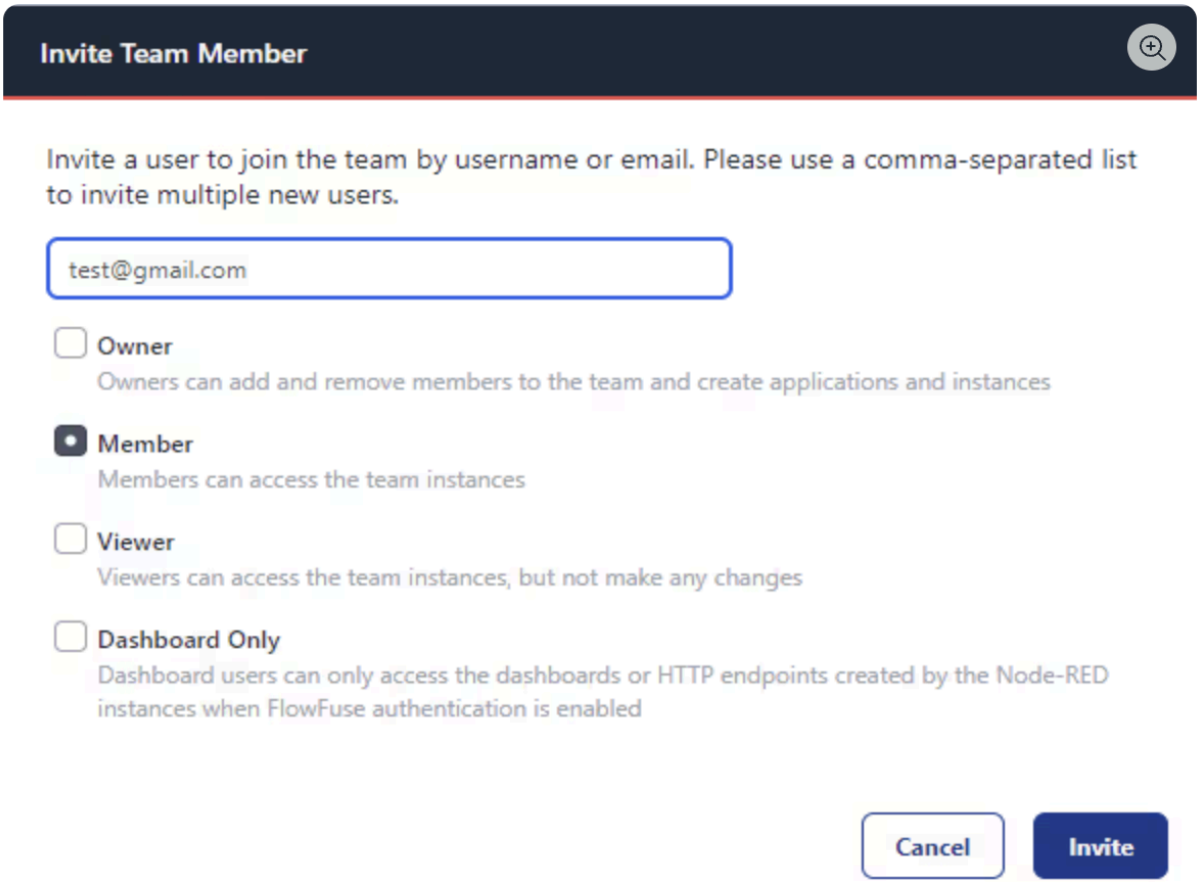
4. *Granular Role-Based Access Management*

With collaboration at its core, FlowFuse allows you to create teams and invite members to collaborate on projects. However, not all team members require access to every feature. Effective management is

essential, as some members might feel overwhelmed by unnecessary options, and there's a risk of accidental changes being made by those who are unfamiliar with the configurations and settings.

To address this, FlowFuse offers Role-Based Access Control (RBAC). When inviting team members, you can assign specific roles that provide the appropriate level of access for their work:

- **Owner:** Has full control over the team settings, applications, instances, and flows. Can invite users and change their roles.
- **Member:** Can access applications and instances and modify flows, but with limited permissions compared to the Owner. Cannot manage team, application, or instance settings or invite users.
- **Viewer:** Can view instances and flows but cannot make any changes. Ideal for users who need to monitor without editing capabilities.
- **Dashboard Only:** Restricted to accessing the dashboard or HTTP endpoint. This role is for users who only need to monitor status without making any changes.



If required, it is possible to change the roles of team members in the "members" page. This helps prevent unauthorized changes and ensures a more secure and efficient workflow.

5. Comprehensive Activity Audit Logs

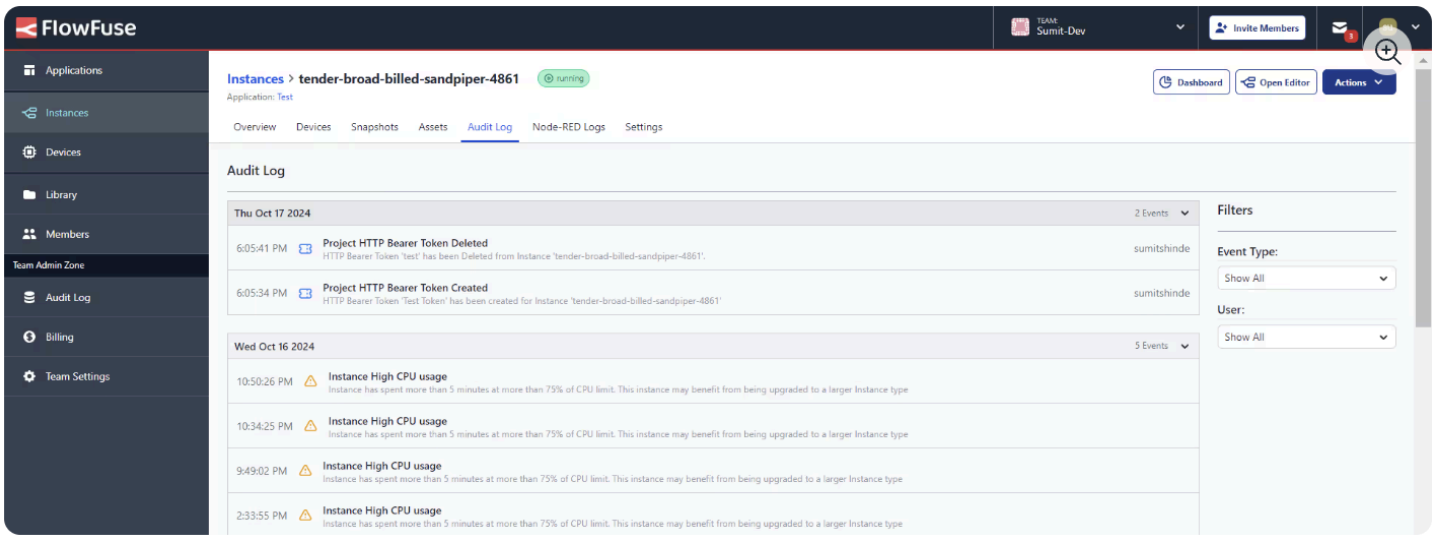
Audit Logs maintain a comprehensive record of all actions in the platform to enable a culture of openness and transparency with the proper security measures in place. These log details on who made changes, what was changed, and when it occurred, ensuring accountability and enabling teams to quickly identify any unauthorized access or mistakes that could jeopardize security.

Audit logs are provided at three different levels:

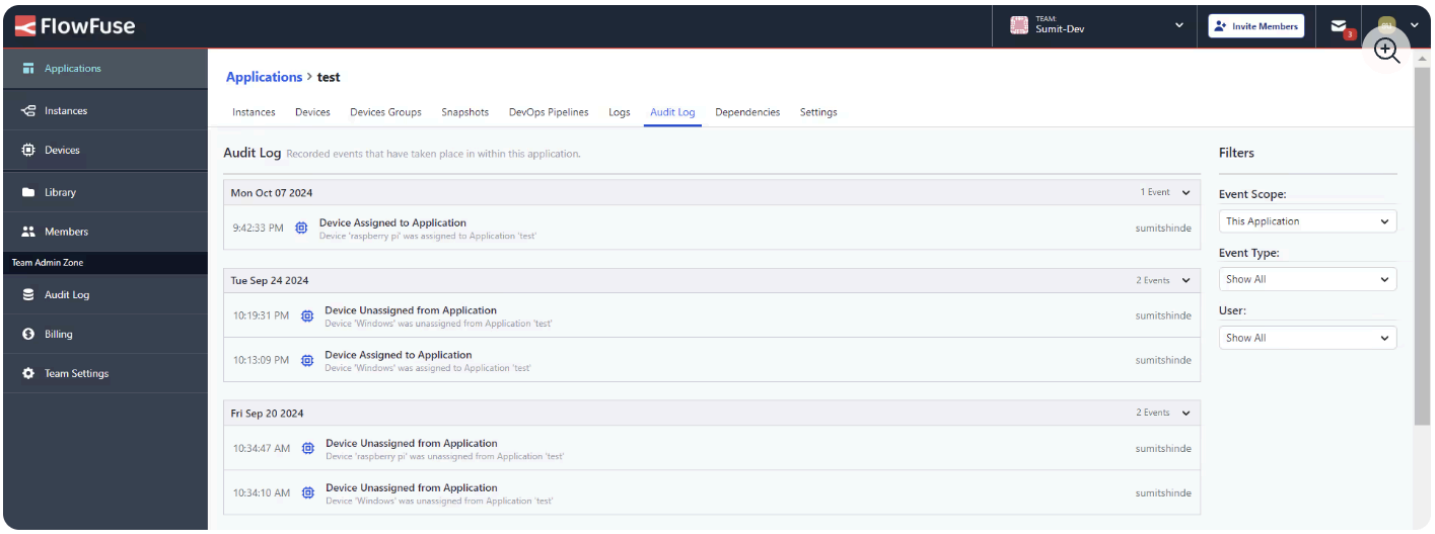
- Instance level, where all action logs related to a specific instance are recorded
- Application level, which groups logs from instances created within a particular application
- Team level, where all platform activities are documented but visible only to admins.

This layered approach helps organizations maintain secure workflows and demonstrates their commitment to transparency, ensuring that security concerns are effectively addressed without sacrificing openness.

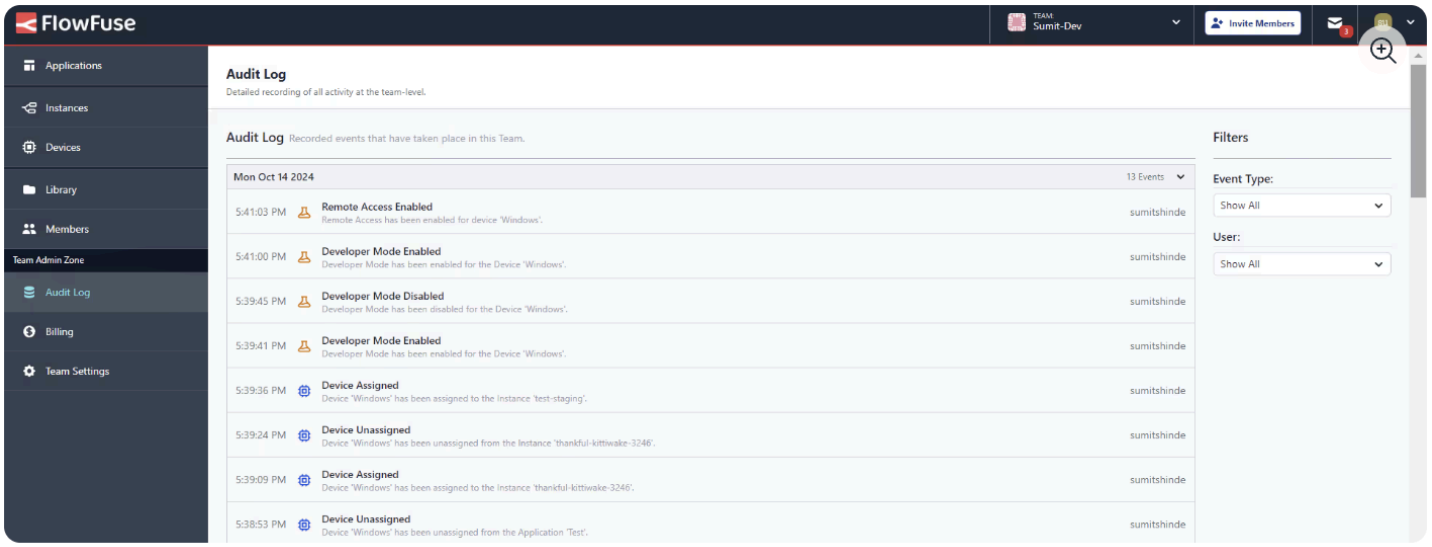
For instance-level logs, choose the specific instance you want to see and go to Audit Logs.



For application-level logs, select the application you want to view and navigate to Audit Logs.



For team-level logs, there will be an option labeled "Audit Logs" in the left sidebar, accessible only to admins.



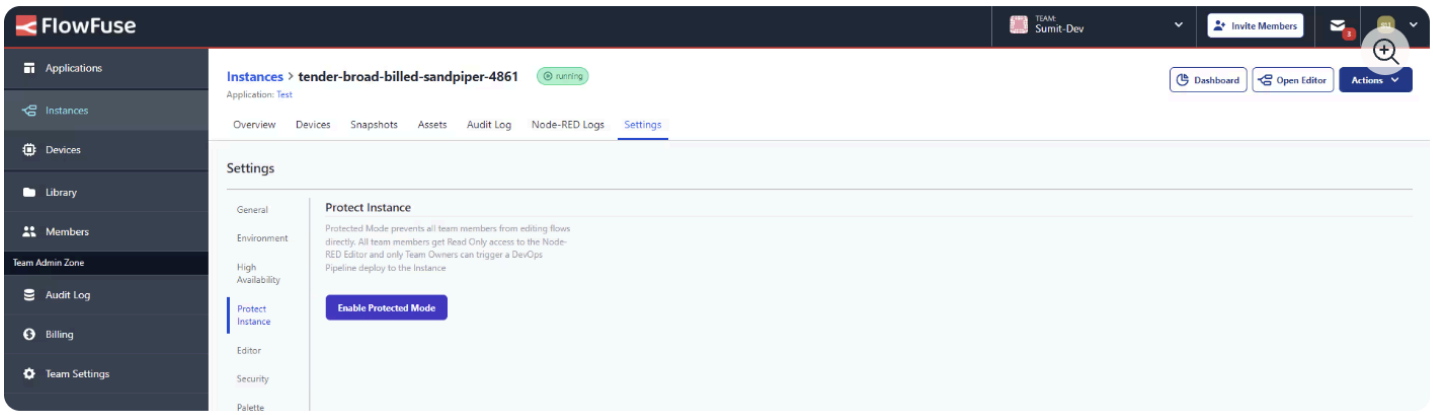
6. *Instance Protection Mode*

Imagine your Node-RED application running smoothly on the production line, seamlessly handling critical tasks and data flows. Now, picture the chaos that could ensue if someone from your team accidentally modified a flow. While the snapshot feature recovers previous changes, accidental

modifications may not be identified quickly. Even when they are discovered, and the snapshot is used to restore the previous state, it can still take seconds or even minutes to recover, resulting in costly downtime.

To prevent such scenarios, Instance Protection Mode allows you to set flows within your Node-RED instances to read-only, ensuring that modifications can only occur through a DevOps pipeline. This process guarantees that even the most critical flows can only be altered with thorough testing and approval.

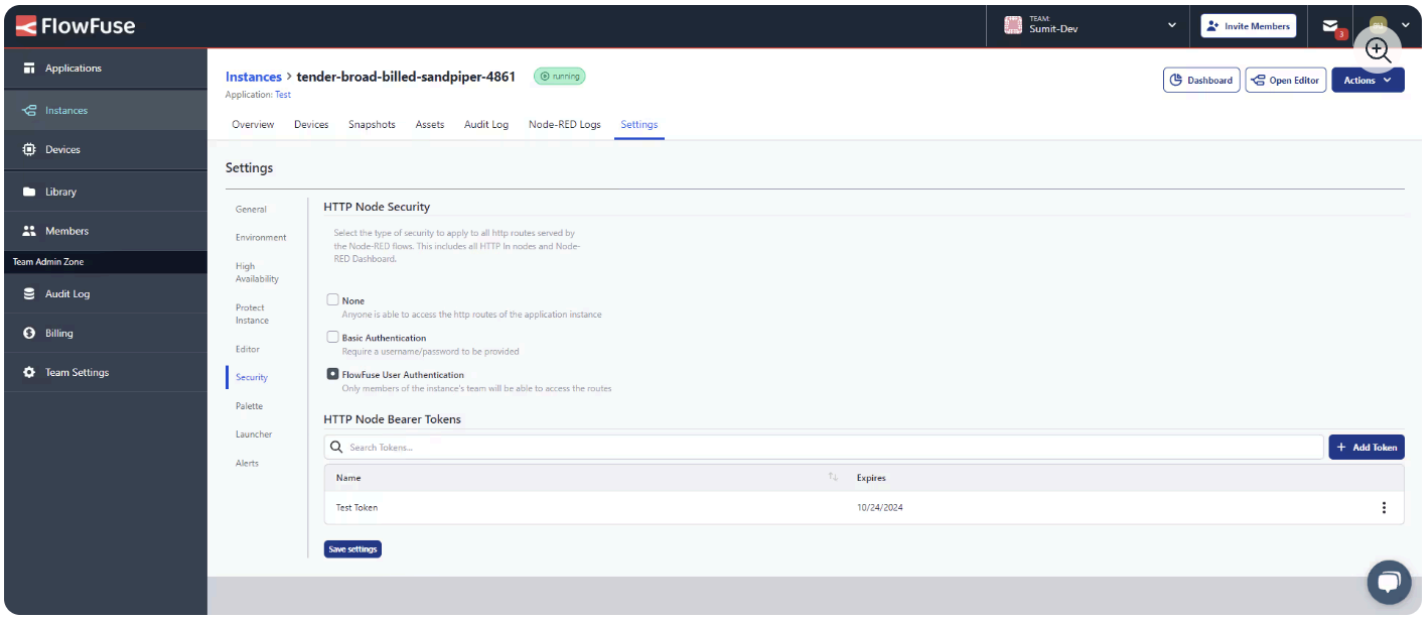
With Instance Protection Mode activated, team members can still view flows, but any attempts to modify them are blocked, providing an additional layer of security. This approach protects the integrity of your applications and fosters a controlled environment for making changes safely.



7. *Secure HTTP Nodes Endpoints*

HTTP is one of the most widely used protocols for enabling communication between different applications and services. In Node-RED, you can quickly create these APIs using HTTP-In nodes, which allow for communication. However, while this is convenient, ensuring that only authorized users can access your APIs is essential.

FlowFuse provides robust options for securing all HTTP endpoints served by Flow and the Node-RED Dashboard. To manage this, each instance has a dedicated interface that you can access by navigating to your ***instance -> Settings -> Security***. Here, you'll find several options for securing your APIs:



- **None (Default):** No authentication is enabled by default, which means anyone can access your endpoints.

- **Basic Authentication:** By selecting this option, two input fields will appear where you can enter a username and password. This ensures that only users with the correct credentials can access your APIs.

- **FlowFuse User Authentication:** This option allows all team members to use their unique usernames and passwords when requesting API.

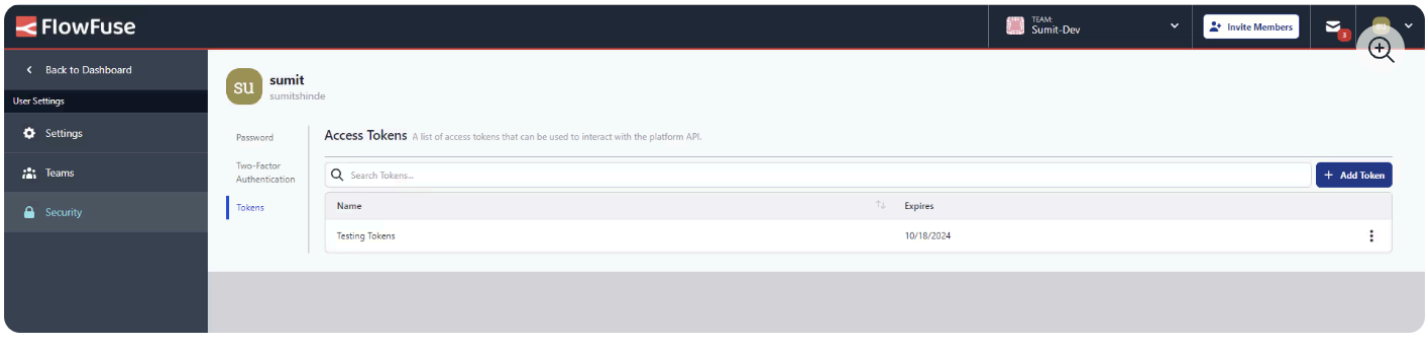
- **Bearer Tokens:** For more advanced users, there's an option to generate bearer tokens for secure API access without needing to send usernames and passwords. With this feature, you can also set expiration times for these tokens, ensuring that access is time-limited and reducing the risk of unauthorized use. To use bearer tokens, you must first enable FlowFuse User Authentication.

For more information, refer to [HTTP Authentication in Node-RED with FlowFuse.]

8. API Token Management for Secure Platform Interactions

Organizations need to create integrations for automation, monitoring, and efficient workflows. FlowFuse provides REST APIs that allow easy interaction with various platform parts, including users,

instances, teams, devices, and more. However, security settings are protected to ensure they can only be updated by admins or authorized team members directly on the platform, not via APIs.



Protecting against unauthorized access is crucial, especially since you control and monitor your entire factory and production lines through this platform. To safeguard this, we offer an interface similar to the one used for bearer tokens. You can access this by navigating to ***User Settings > Security > Tokens.***

For more information, refer to the [FlowFuse Platform API docs](#).

9. Software Bills of Materials

Node-RED is an open-source platform maintained by dedicated community members who ensure it operates smoothly and remains free of security vulnerabilities. Similarly, there exists a vast ecosystem of open-source packages, nodes, and libraries that we frequently use in our projects. While these packages are often excellent and enhance our capabilities, some may need a regular team or individual to update and monitor them. This can lead to potential risks, as outdated or unmaintained packages can introduce vulnerabilities into your applications.

To address these concerns, the Software Bill of Materials (SBOM) feature adds an extra layer of security and compliance. An SBOM is a detailed list of all an application's components. It provides a comprehensive view of all third-party libraries used in each application instance and their latest versions. This allows teams to monitor dependencies and make informed decisions about upgrades, ensuring effective dependency management and enhanced resilience against security threats.

For more information, see [Introdution to FlowFuse Software Bills of Materials.](#)

Chapter 8

Creating and Automating DevOps Pipelines

When deploying any update in your manufacturing or automotive process, it is important to perform testing and validation. The smallest errors can lead to a production outage. In such cases, having a quick and reliable deployment process is essential for maximising uptime and minimising downtime.

For developers using Node-RED, setting up a comprehensive DevOps pipeline can make all the difference. Lets look at how to build and automate DevOps pipelines specifically for Node-RED deployments.

First lets understand what a Dev-Ops pipeline is - a DevOps pipeline is a process that helps developers move their code from development to production smoothly. In a pipeline, each step depends on the one before it, ensuring every update is properly tested and ready for use. The most common stages of a pipeline are:

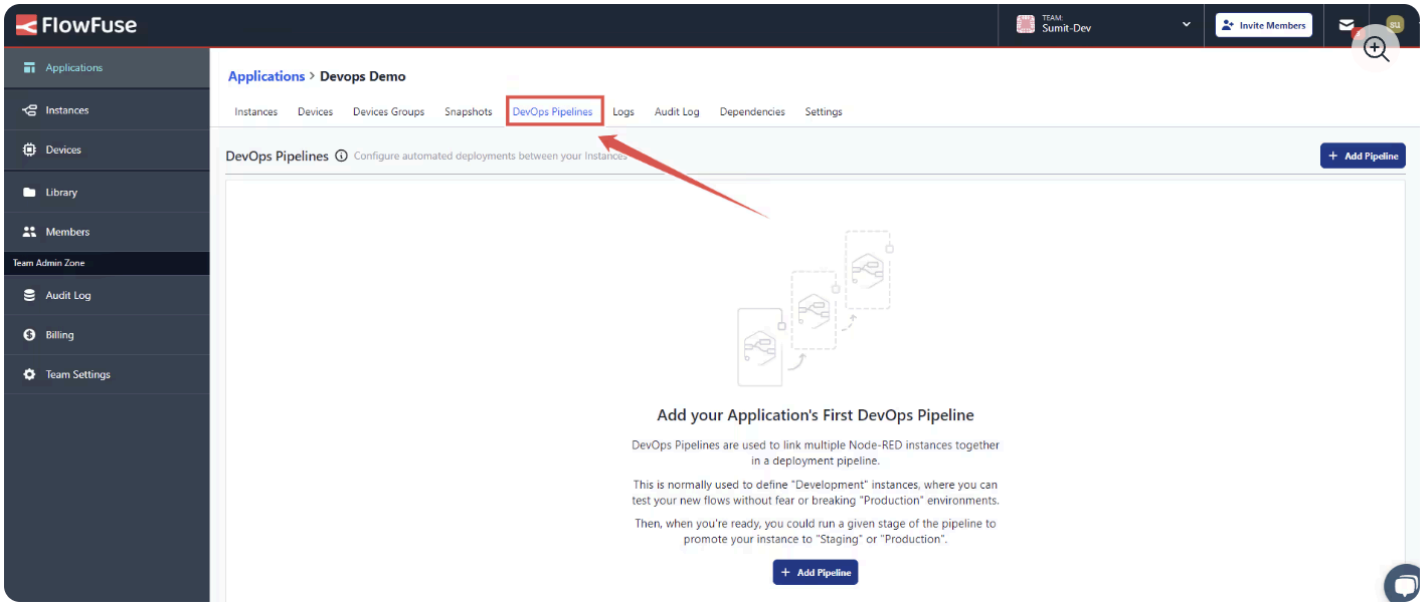
- **Development:** This is where the application flows are created. A level of testing and checking is carried out as a natural step of the development.
- **Staging/Testing:** In this stage, the code is deployed to a staging environment that closely mimics the live system. Here, the application goes through a level of QA and is tested in scenarios as close to real-world conditions as possible. This is stage designed to catch any issues that slipped through in the development stage.
- **Production:** When the flows are tested and confirmed in staging and are ready to be deployed, this stage can be operated to promote the flows to production.

How to Create DevOps Pipelines for Node-RED Deployments

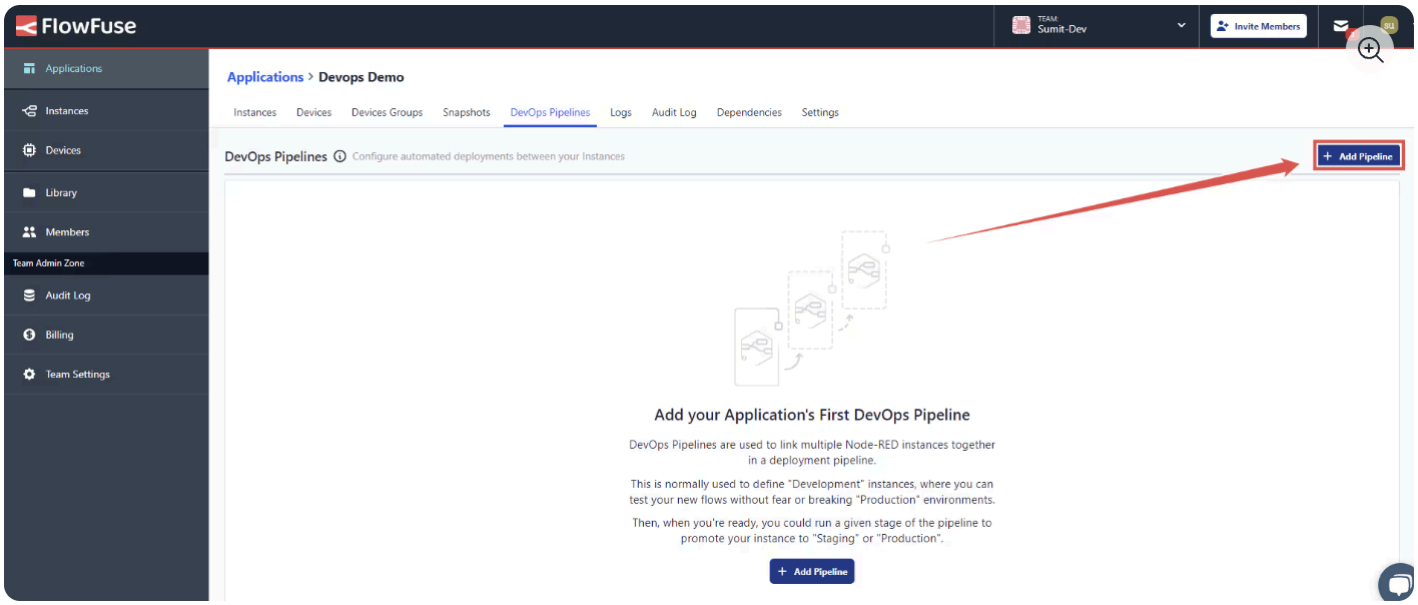
Creating DevOps pipelines manually for your Node-RED deployments can be time-consuming, expensive, and require considerable technical expertise. FlowFuse simplifies the creation of DevOps pipelines for Node-RED deployments.

FlowFuse enhances collaboration, security, and scalability for your Node-RED applications, making the deployment and management of edge devices seamless. With a centralized platform and an intuitive visual interface, FlowFuse allows you to connect, collect, transform, and visualize data effortlessly.

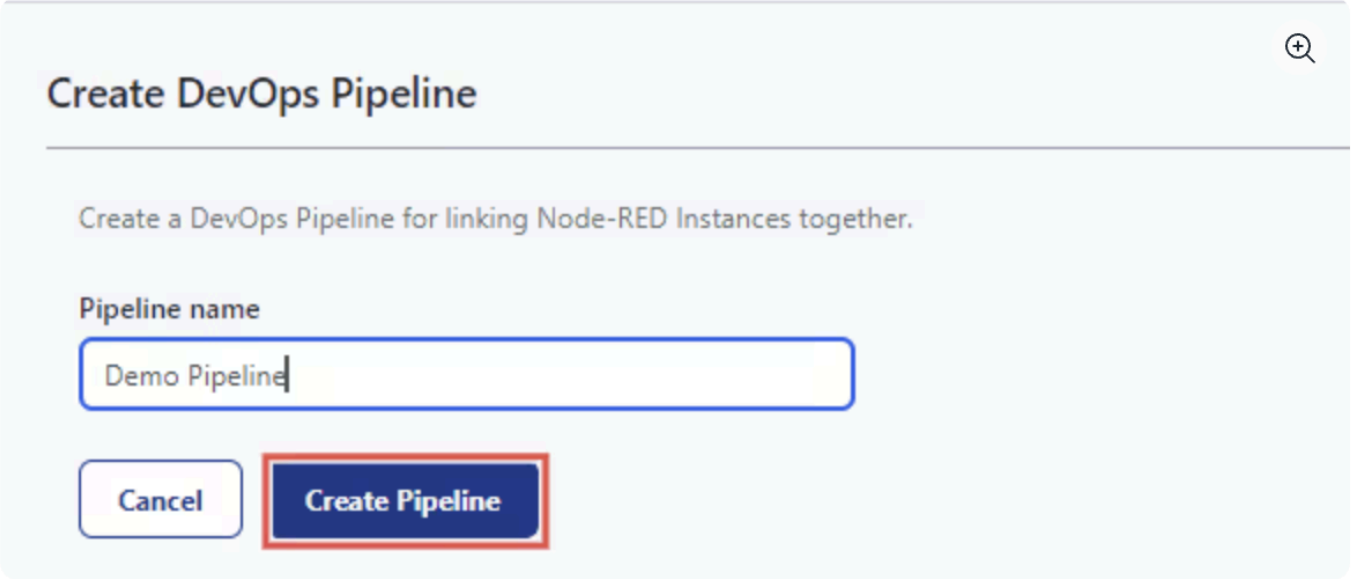
- Step1: Go to the FlowFuse platform and navigate to the application where your Node-RED instances are located. Ensure you have instances set up for all stages, including production devices or instances.
- Step 2: Switch to the DevOps Pipelines option from the top menu.



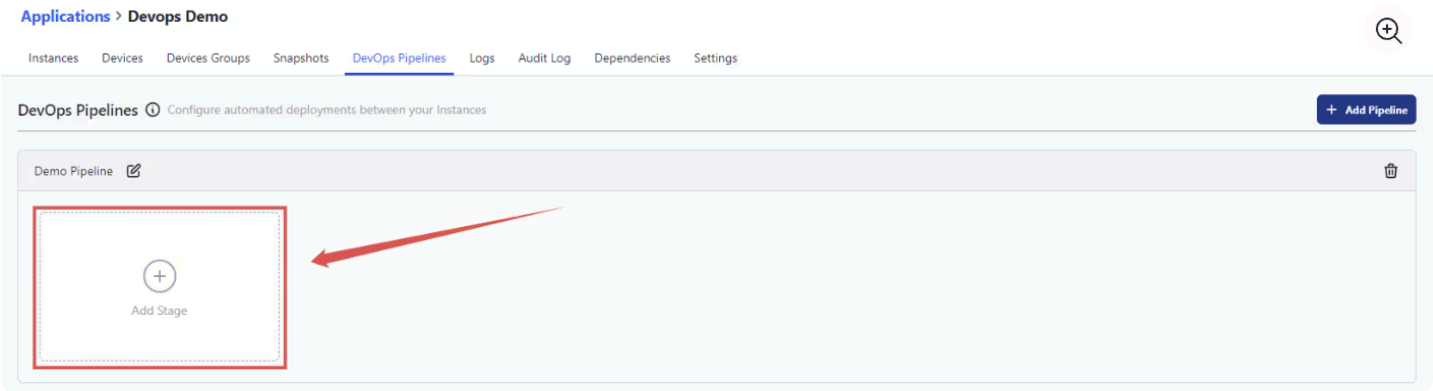
- Step 3: Click the Add Pipeline button in the top right corner to create the pipeline.



- Step 4: Enter the name for your pipeline and click Create Pipeline.

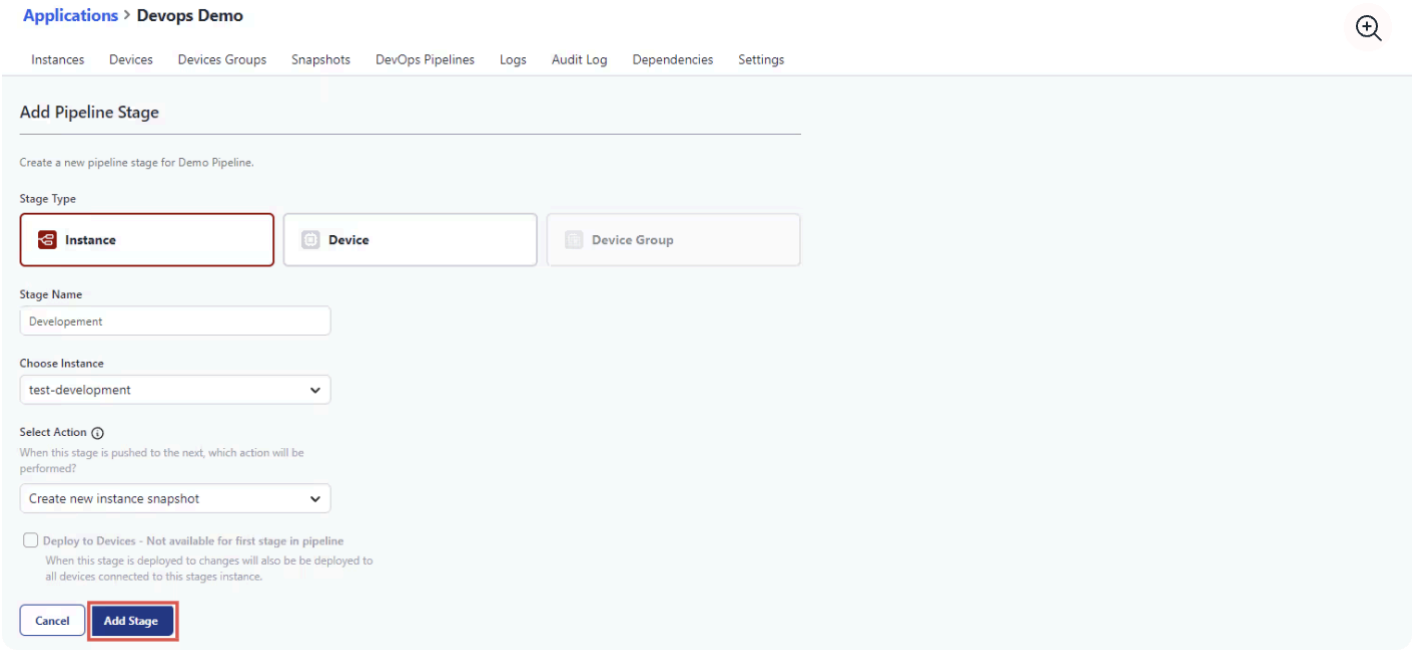


- Step 5: You'll see an option to create stages by clicking Add Stage.



- Step 6: In the window that opens, select the stage type based on whether it's an instance, device, or device group.
- Step 7: Enter the name for the stage in the Stage Name field.
- Step 8: Choose an instance, device, or device group for the stage.
- Step 9: Next, configure which action should be performed when this stage is pushed to the next:
    - Create New Snapshot: Generates a new Snapshot using the current flows and settings.
    - Use Latest Instance Snapshot: Uses the most recent existing snapshot of the instance. The deployment will fail if no snapshot exists.

    - Prompt to Select Snapshot: Prompts at deploy time to select which snapshot from the source stage should be copied to the next stage.
- Step 10: Check the option Deploy to Devices if you want changes to be deployed to all devices connected to this stage's instance when the stage is deployed.

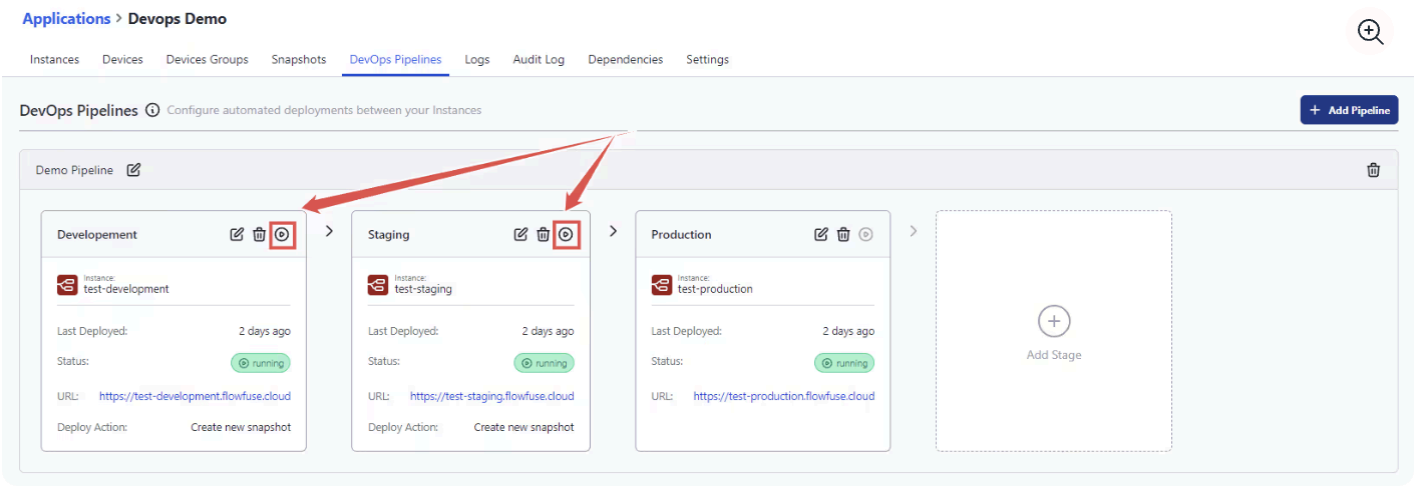


Once you've created your initial stage, you can add more stages by following the same process. This flexibility allows you to tailor your DevOps pipeline to meet the specific needs of your Node-RED deployment.

For example, in development, you might have a Node-RED instance in the cloud to build your application. During staging, you could test the setup with a single device. Finally, in production, you can deploy the tested application to thousands of devices in a device group, saving time and ensuring smooth deployment at scale.

Running a Pipeline Stage

Once your pipeline is set up, you can run it to deploy your changes across each stage. Here's how:



1. Click the "Run Pipeline" button for the current stage to start the deployment. This button is available for all stages except the last one.

2. After clicking, the deployment automatically progresses to the next stage on the right. Since each pair of stages operates independently, you need to click the "Run" button for each stage to continue the deployment.

Pressing the "Run Pipeline" button for the current stage creates a new snapshot that includes all settings, environment variables, and flows for that stage. This snapshot is then copied and deployed to the next stage, but any existing environment variable keys in the target stage will remain unchanged.

When creating a pipeline, you can include only one Device Group, and it must be in the final stage. This ensures all changes are fully tested and verified before reaching production, guaranteeing a safe and reliable deployment.

Chapter 9

Using Snapshots for Version Control

Effective collaboration on Node-RED projects necessitates a robust version control system. Without it, conflicts and accidental overwrites can jeopardize critical flows. Implementing version control for Node-RED flows can introduce complexity, particularly in fast-paced development environments. The frequent need to push changes to maintain synchronization can be time-consuming and disruptive.

FlowFuse streamlines this process by providing snapshot capabilities, allowing users to create backups, revert to previous versions, and mitigate risks.

Snapshots provide point-in-time backups of your flows. In the context of Node-RED, snapshots automatically capture the state of your work, ensuring that you can quickly restore them if needed. It captures the following:

- **Flows:** The flow, including all nodes and config nodes of your Node-RED flows.
- **Credentials:** Any sensitive information used within flows using config nodes.
- **Environment Variables:** Environment variables you have used or defined within that Node-RED instance.
- **Packages:** The packages you have installed, including 3rd party contribution nodes and Node.js packages.
- **Runtime Settings:** The configurations that govern the behavior of your Node-RED runtime.

Creating snapshots is straightforward and can be done in just a few steps. See this guide on how to manage snapshots in the FlowFuse platform.

There is always the risk of forgetting to create snapshots before making changes and though it is straightforward, it can be time consuming to do each time. To simplify this process, FlowFuse also introduces the Auto Snapshot feature.

Auto Snapshots automatically create backups whenever you deploy changes, ensuring that your work is continuously backed up without requiring manual intervention. These snapshots are labeled as "Auto snapshot - yyyy-mm-dd hh:mm:ss" for easy identification.

This feature allows you to focus on developing your Node-RED flows with the assurance that your changes are securely saved. If necessary, you can disable Auto Snapshots for devices only from the

Developer Mode tab. This can be helpful to avoid excessive data usage when a device is in the field or on a cellular connection, or to prevent reaching the limit of auto snapshots with unnecessary snapshots.

Note: A limit of 10 auto snapshots is maintained, with the oldest one being deleted when a new one is created. Also this feature is only available to Team and Enterprise users of the FlowFuse Platform.

Common Mistakes to Avoid

- **Neglecting Regular Backups:** Don't skip those snapshots! Regularly create backups, especially before making significant changes, moving devices through instances or applications, or disabling developer mode. Think of it as your safety net—ensuring you can always bounce back to a stable state if unexpected issues pop up.

- **Overlooking Auto-Snapshot Limits**: Did you know that your auto snapshots have limits? Be mindful of how many you can retain, as older auto snapshots will be automatically deleted. If you have important auto snapshots, either rename them to avoid automatic deletion or download and save them locally to keep them safe. Remember, manually created snapshots have no limits, so take advantage of that!

- **Secret Key:** The secret key used to encrypt your snapshot is crucial for when you later need to upload that snapshot. When downloading snapshots locally, securely store that key. Losing it could mean losing access to your snapshot and your ability to recover your work. Treat it like a password—keep it safe!

- **Taking a Snapshots of the wrong thing:** When a device is owned by an instance it typically runs the same flows however, if a device flows may have been modified directly in developer mode it will have different flows to the instance that owns it. In this case, before you move the device or switch it out of developer mode, it is recommended that you take a snapshot directly from the device itself. Direct device snapshots are performed on the Developer Mode tab of the device.

- **Deploying Unverified Snapshots:** A little caution goes a long way! Always review and verify the details of a snapshot before deploying it. Jumping into deployment without checking can lead to unexpected behavior or the loss of critical configurations. Take the time to ensure everything is in order.

About FlowFuse

FlowFuse is a platform that empowers any engineer; mechanical, electrical, or software engineer, to build, deploy, manage, and secure enterprise-grade applications. We enable IT experts and operational engineers to collaborate on data acquisition, merging of data streams, create data visualization, and interactive applications.

Our platform enhances the capabilities of Node-RED by providing security and compliance guardrails, increasing developer productivity, and centralized operations for scalability and reliability.

At FlowFuse we are committed to providing the best tools and support to seamlessly integrate into your existing business, offering an adaptable foundation for complex application development.
Contact us to find out how we can help develop innovative applications that revolutionize your operations.