

# Information Technology for Logistics

## **Part 2: N8N**

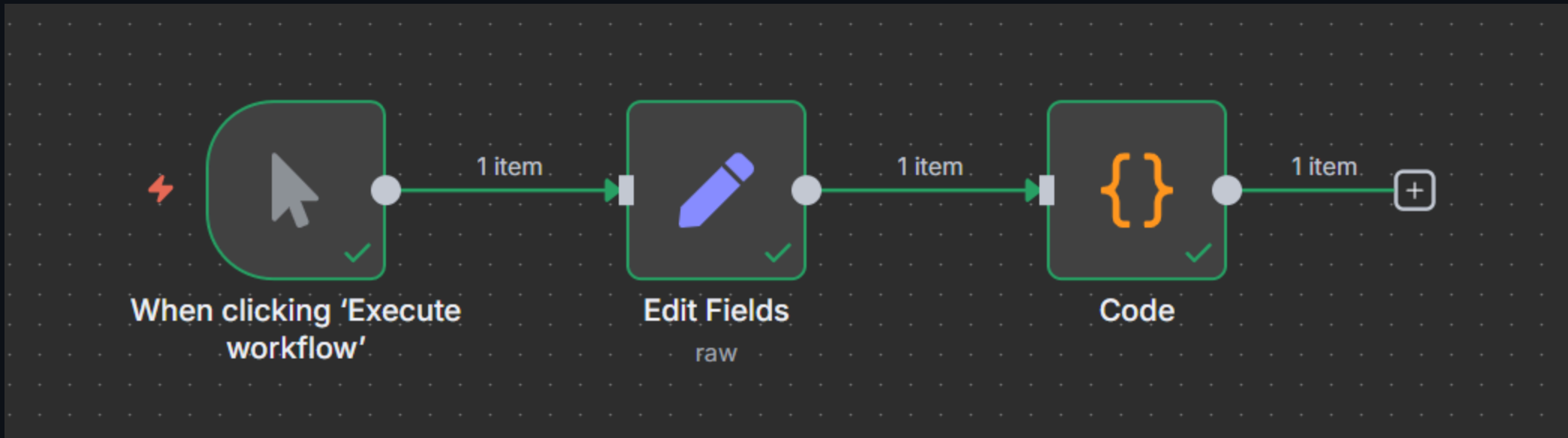
## What is it?

**n8n** is an open-source workflow automation tool that allows users to automate tasks and connect different apps, services, and APIs using a simple, visual interface.

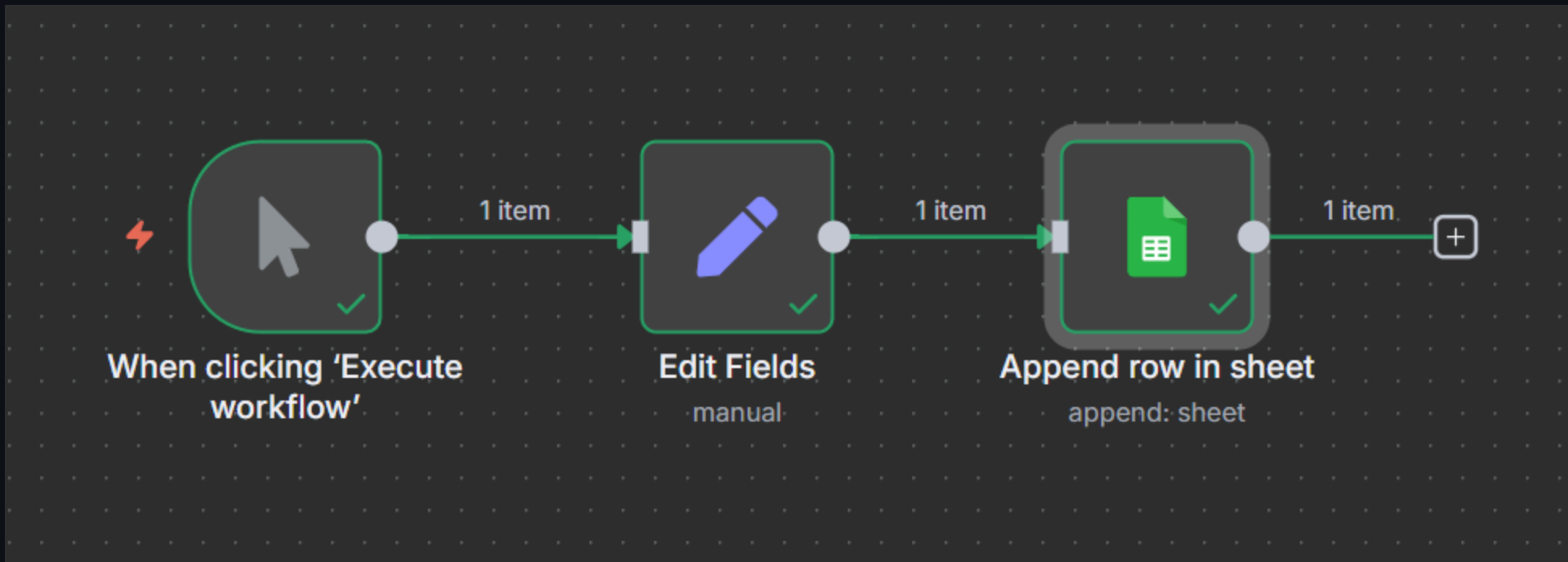
# Installation

## Guide

# Basic Flow 1



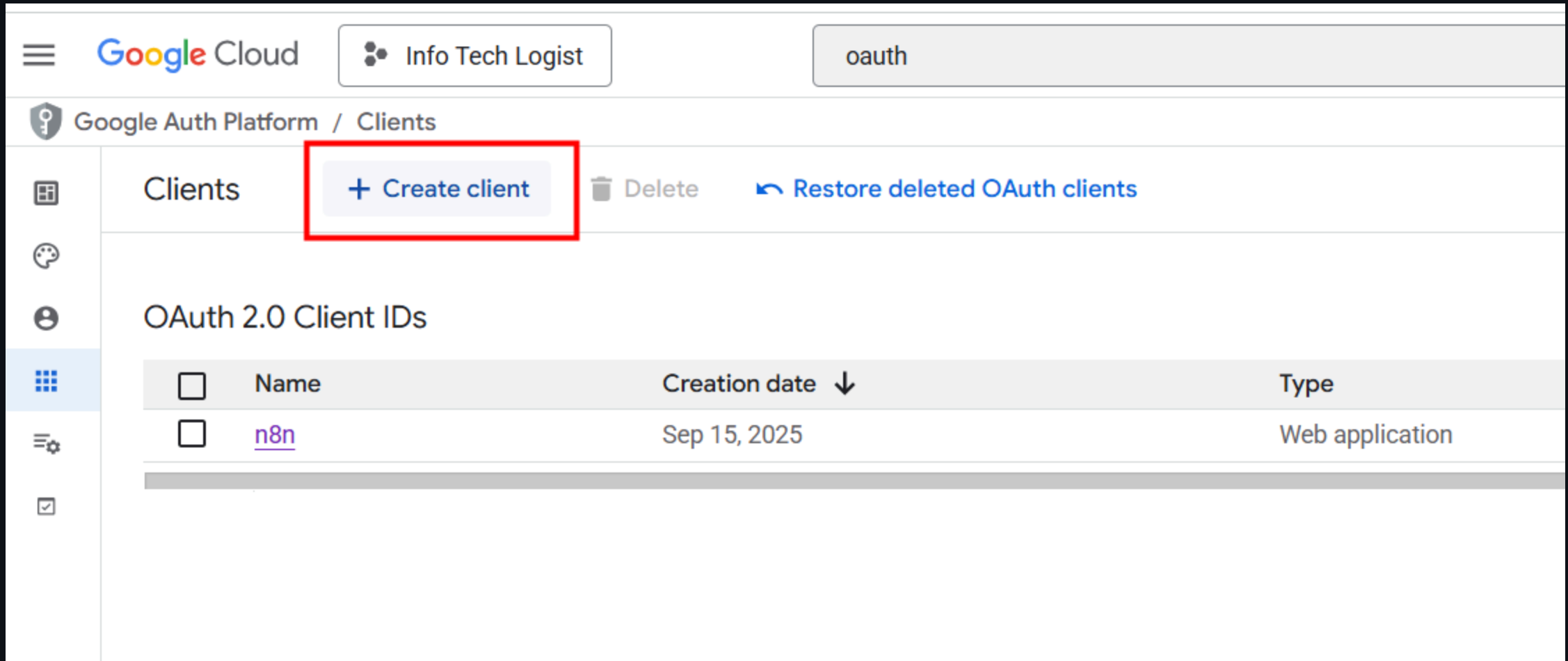
# Google Sheet



# Google Sheet

- Create a new `project` in Google Cloud Platform
- Enable `Google Sheet API`
- Create `App` from the `Consent Screen`
- Create a `client` from the `App`
- Add a test `user`
- Enable `Google Drive API`

# Create Client



The screenshot shows the Google Cloud console interface for managing OAuth 2.0 Client IDs. The top navigation bar includes the Google Cloud logo, the user profile 'Info Tech Logist', and a search bar containing 'oauth'. The main heading is 'Google Auth Platform / Clients'. Below this, there's a 'Clients' section with a '+ Create client' button highlighted by a red rectangle. To the right of this button are 'Delete' and 'Restore deleted OAuth clients' links. The main content area is titled 'OAuth 2.0 Client IDs' and contains a table with one client entry.

<input type="checkbox"/>	Name	Creation date ↓	Type
<input type="checkbox"/>	<a href="#">n8n</a>	Sep 15, 2025	Web application



# Add Test User

Google Cloud Info Tech Logist oauth

Google Auth Platform / Audience

Audience

External ?

Make internal

OAuth user cap ?

While publishing status is set to "Testing", only test users are allowed to access the app. Allowed user cap prior to app verification is 100 users, which is counted over the entire lifetime of the app. [Learn more](#)

1 user (1 test, 0 other) / 100 user cap

Test users

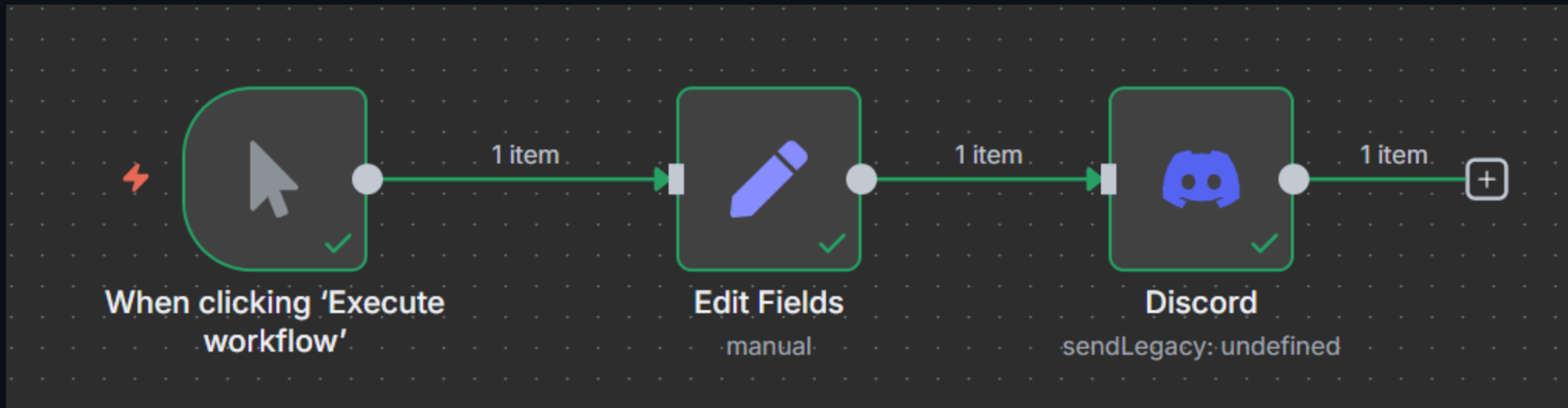
+ Add users

Filter Enter property name or value

User information
nnnpoo@gmail.com

Show less

# Notification



# Drink Ordering System

| Connecting Web Application to Other Services

# Install curl in php

- `sudo apt-get install php-curl`
- `sudo systemctl restart nginx`
- `php -m`

```
admin@pm1-ct102: ~  
(admin) admin@pm1-ct102:~$ php -m  
[PHP Modules]  
calendar  
Core  
ctype  
curl  
date  
exif  
FFI  
fileinfo  
filter  
ftp
```

# Modify **n8n** config

- `sudo nano /etc/supervisor/conf.d/n8n.conf`
- Add this line (*see next page*)

```
WEBHOOK_URL="https://pmX-ctXXX-n8n.iecmu.com",
```

- `sudo systemctl restart supervisor`

```
admin@pm1-ct102: /etc/supe x + v
GNU nano 7.2 n8r
[program:n8n]
directory=/home/admin
command=/home/admin/.nvm/versions/node/v22.18.0/bin/n8n start
autostart=true
autorestart=true
startsecs=10
user=admin
redirect_stderr=true
stdout_logfile=/var/log/n8n.log
environment=PATH=
    /home/admin/.nvm/versions/node/v22.18.0/bin:/usr/local/bi
    N8N_PORT=5678,
    N8N_SECURE_COOKIE=false,
    N8N_EDITOR_BASE_URL="https://pm1-ct102-n8n.iecmu.com",
    N8N_RUNNERS_ENABLED=true,
    DB_SQLITE_POOL_SIZE=2,
    WEBHOOK_URL="https://pm1-ct102-n8n.iecmu.com",
    TZ="Asia/Bangkok"
```

# Database

- Create a new database

```
CREATE DATABASE IF NOT EXISTS iedrink;
```

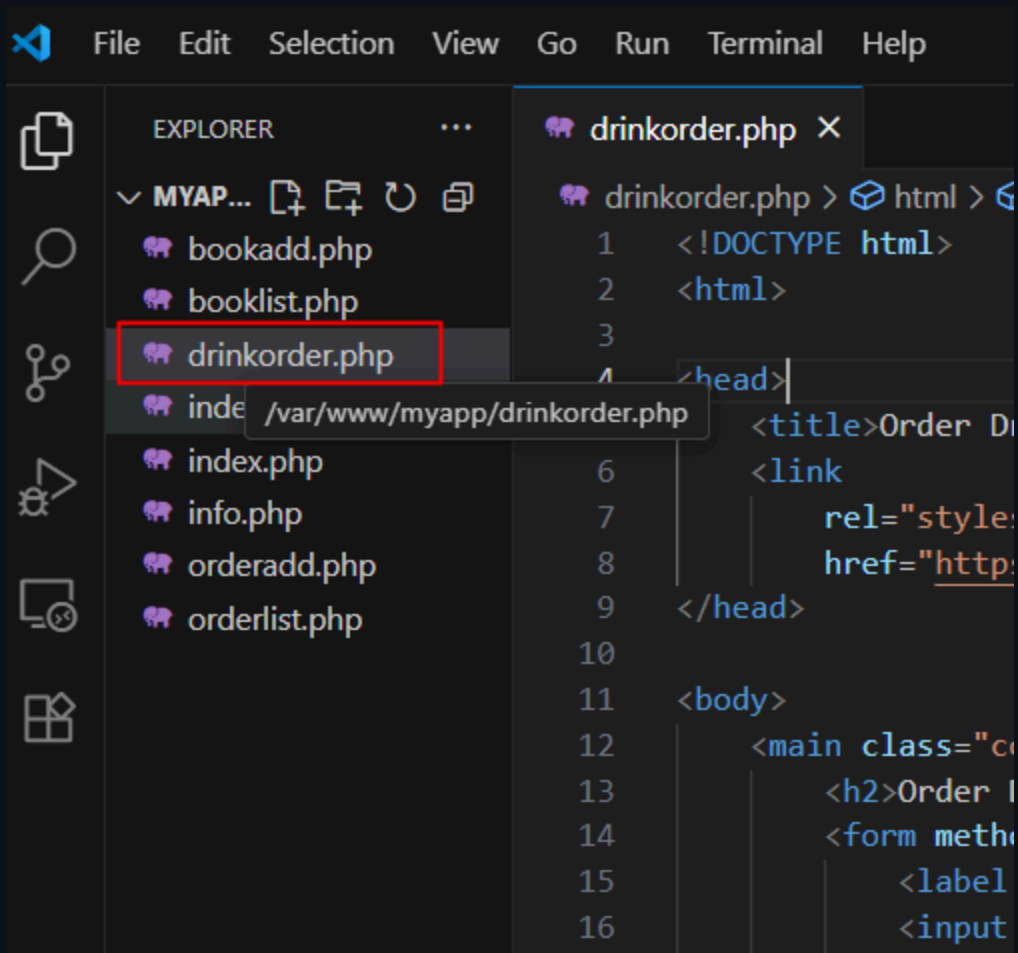
- Create a new table

```
CREATE TABLE IF NOT EXISTS iedrink.orders (  
  id INT NOT NULL AUTO_INCREMENT,  
  drink_name VARCHAR(100),  
  customer_name VARCHAR(100),  
  quantity INT,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,  
  PRIMARY KEY (id)  
);
```

# Web Application

- Create `drinkorder.php` ([Link](#))
  - Modify your database credential.
- Visit your page at <https://pmX-ctXXX-web.iecmu.com/drinkorder.php>





Visual Studio Code interface showing the Explorer sidebar and the main editor area. The Explorer sidebar lists files in a project named 'MYAP...', with 'drinkorder.php' highlighted. The main editor area displays the code for 'drinkorder.php'.

```
File Edit Selection View Go Run Terminal Help
```

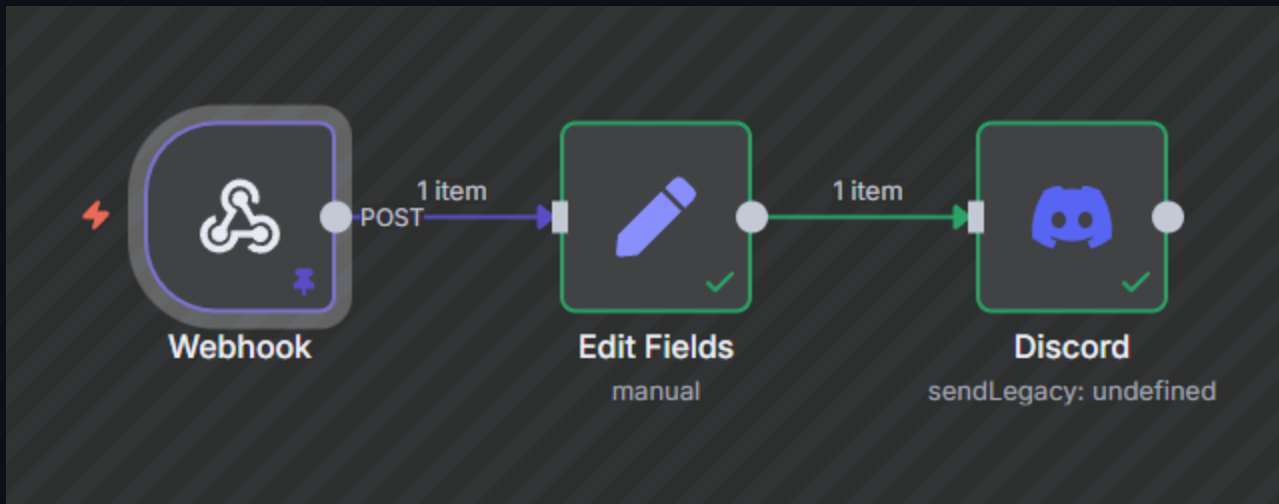
EXPLORER

- MYAP...
  - bookadd.php
  - booklist.php
  - drinkorder.php
  - index.php
  - info.php
  - orderadd.php
  - orderlist.php

drinkorder.php

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <title>Order D
6   <link
7     rel="style
8     href="http:
9 </head>
10
11 <body>
12   <main class="c
13     <h2>Order I
14     <form metho
15       <label
16       <input
```

# Webhook Flow



# Webhook

## Webhook

 Listen for test event

Parameters

Settings

Docs 

### Webhook URLs

Test URL

Production URL

POST https://pm1-ct102-n8n.iecmu.com/webhook-test/order

### HTTP Method

POST

### Path

  Fixed Expression

order


### Authentication

None

### Respond

Immediately

# Edit Fields

 Edit Fields

Execute step

ParametersSettingsDocs

Mode

Manual Mapping

Fields to Set

body.customer\_name

ABString

=

{{ \$json.body.customer\_name }}

body.drink\_name

ABString

=

{{ \$json.body.drink\_name }}

body.quantity

#Number

=

{{ \$json.body.quantity }}


body.created\_at

ABString

=

{{ \$json.body.created\_at }}

# Discord

 **Discord**

[Parameters](#) [Settings](#) [Docs](#)

[Execute step](#)

**Connection Type**  
Webhook

**Credential for Discord Webhook**  
Discord Webhook account

**Operation**  
Send a Message

**Message**  

```
Customer {{ $json.body.customer_name }} ordered {{ $json.body.drink_name }} for  
{{ $json.body.quantity }} at  
{{ $json.body.created_at }}.
```

# Discord

```
Customer {{ $json.body.customer_name }} ordered {{ $json.body.drink_name }} for  
{{ $json.body.quantity }} at  
{{ $json.body.created_at }}.
```

## Extra

| Can you write the data into Google Sheet too?

# MQTT

- MQTT is a Client Server publish/subscribe messaging transport protocol.
- It is light weight, open, simple, and designed so as to be easy to implement.
- Ideal for use in many situations
  - Machine to Machine (M2M)
  - **Internet of Things (IoT)**

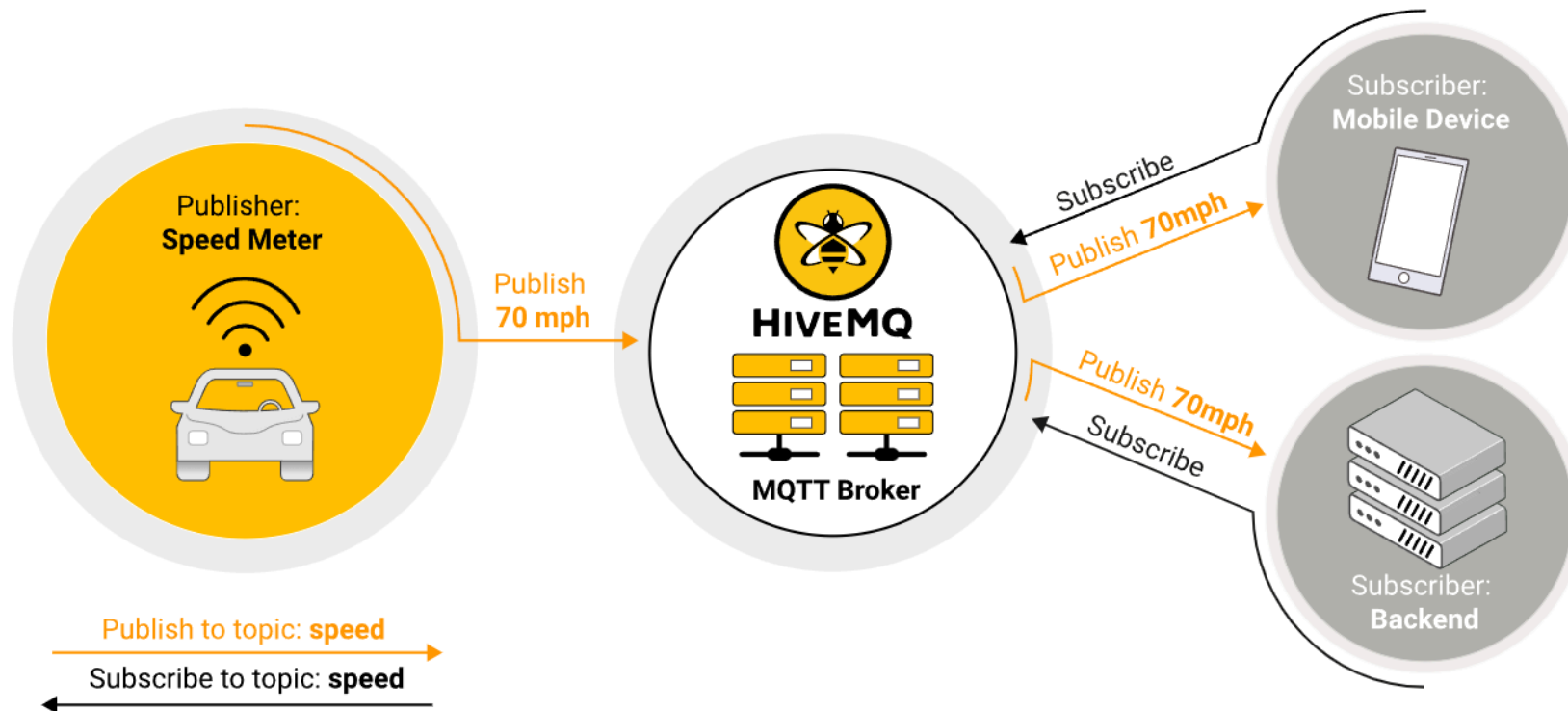


# Publish/subscribe pattern

- The publish/subscribe pattern (also known as `pub/sub`) provides an **alternative** to traditional client-server architecture.
  - In client-server architecture, a client communicates directly with an endpoint.

# Broker

- The connection between `publishers` and `subscribers` is handled by a third component (the `broker` ).
- The job of the broker is to filter all incoming messages and distribute them correctly to subscribers.



# Sensor

## | Storing Sensor Data

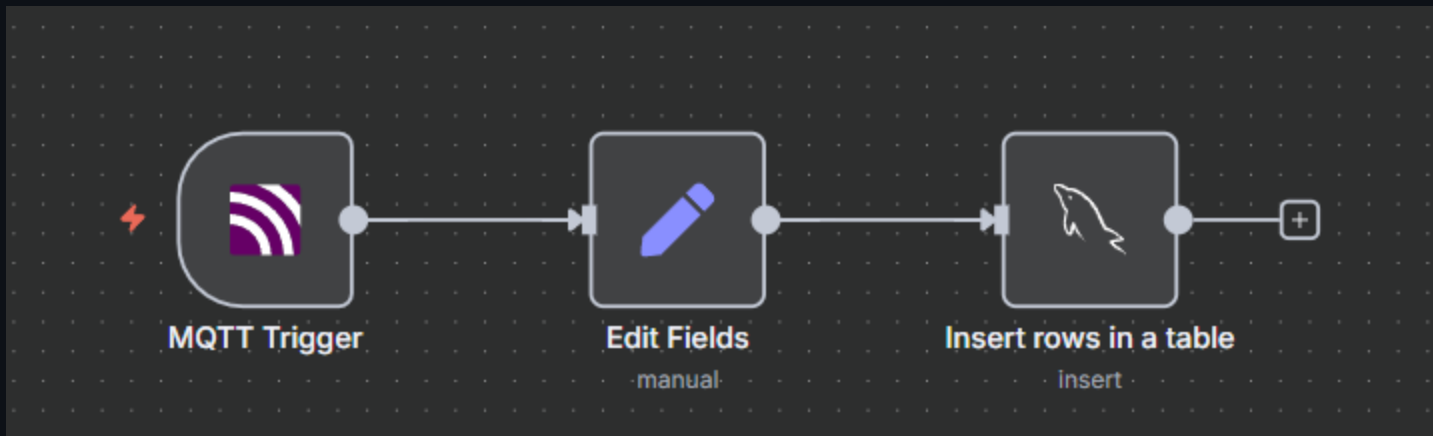
# Database

- Create a new database

```
CREATE DATABASE IF NOT EXISTS sensor;
```


```
CREATE TABLE IF NOT EXISTS sensor.light (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  sensor_name VARCHAR(50),  
  sensor_value FLOAT,  
  created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
);
```

# Database Record Flow





# MQTT

- Topic: Test00/Light/Value

 **MQTT Trigger** [Execute step](#)

[Parameters](#) [Settings](#) [Docs](#)

Credential to connect with

MQTT account  


Topics

Test00/Light/Value


Options


JSON Parse Body


☐

Add option 


# Edit Fields

 Edit Fields


 Execute step


Parameters Settings Docs 

Mode

Manual Mapping 

Fields to Set


sensor_value	# Number 
=	{{ \$json.message }}

sensor_name	AB String 
nr_sensor	

Drag input fields here or Add Field



# MySQL Node

 Insert rows in a table

Execute step

ParametersSettingsDocs

Credential to connect with

MySQL account

Operation

Insert

Table

From list

light

Data Mode

fx

defineBelow

defineBelow

Values to Send

Column

sensor\_name

Value

fx

{{ \$json.sensor\_name }}

Column

sensor\_value

Value

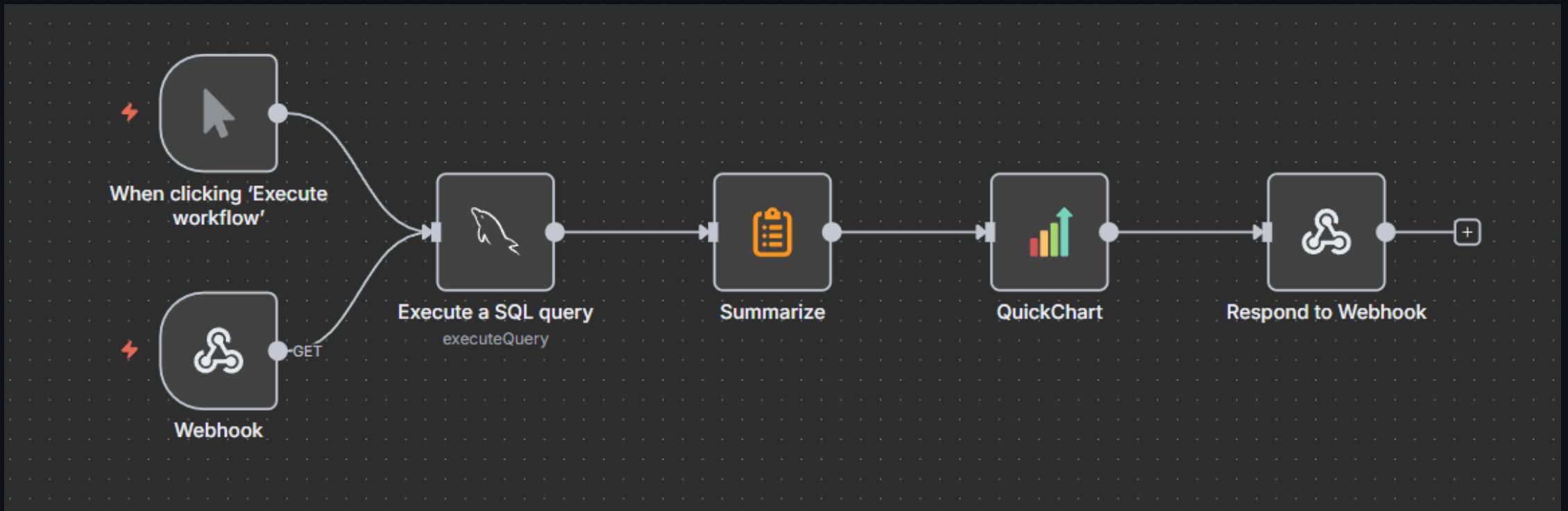
fx

{{ \$json.sensor\_value }}


Add Value

33

# Viewing Data



# MySQL

 Execute a SQL query

Execute step

Parameters

Settings

Docs

Credential to connect with

MySQL account

Operation

Execute SQL

Query

```
1 SELECT *
2 FROM light
3 WHERE created_at >= DATE_SUB(NOW(), INTERVAL 3 MINUTE);
```

Consider using query parameters to prevent SQL injection attacks. Add them in the options below

Options

No properties

Add option

# MySQL

```
SELECT *  
FROM light  
WHERE created_at >= DATE_SUB(NOW(), INTERVAL 3 MINUTE);
```

# Summarize

Summarize

Parameters

Settings

Docs

Execute step

Fields to Summarize

Aggregation

Append

Field

sensor\_value

Enter the field name as text

Include Empty Values

Fixed

Expression

☐

Aggregation

Append

Field

created\_at

Enter the field name as text

Include Empty Values

Fixed

Expression

☐

# Quick Chart

QuickChart

Execute step

ParametersSettingsDocs

Chart Type

Line Chart

Add Labels

From Array

Labels Array

fx

{{ \$json.appended\_created\_at }}

Data

fx

{{ \$json.appended\_sensor\_value }}

Put Output In Field


FixedExpression

data

The name of the output field to put the binary file data in

Chart Options

# Webhook

 Webhook

[Listen for test event](#)

[Parameters](#) [Settings](#) [Docs](#)

Webhook URLs

Test URL

Production URL

GET

https://pm1-ct102-n8n.iecmu.com/webhook-test/light

HTTP Method

GET

Path

light

Authentication


None

Respond

Using 'Respond to Webhook' Node

# Respond to Webhook

.....

 Respond to Webhook

Execute step

Parameters

Settings

Docs

Verify that the "Webhook" node's "Respond" parameter is set to "Using Respond to Webhook Node". [More details](#)

Respond With

Binary File

Response Data Source

Choose Automatically From Input

Options

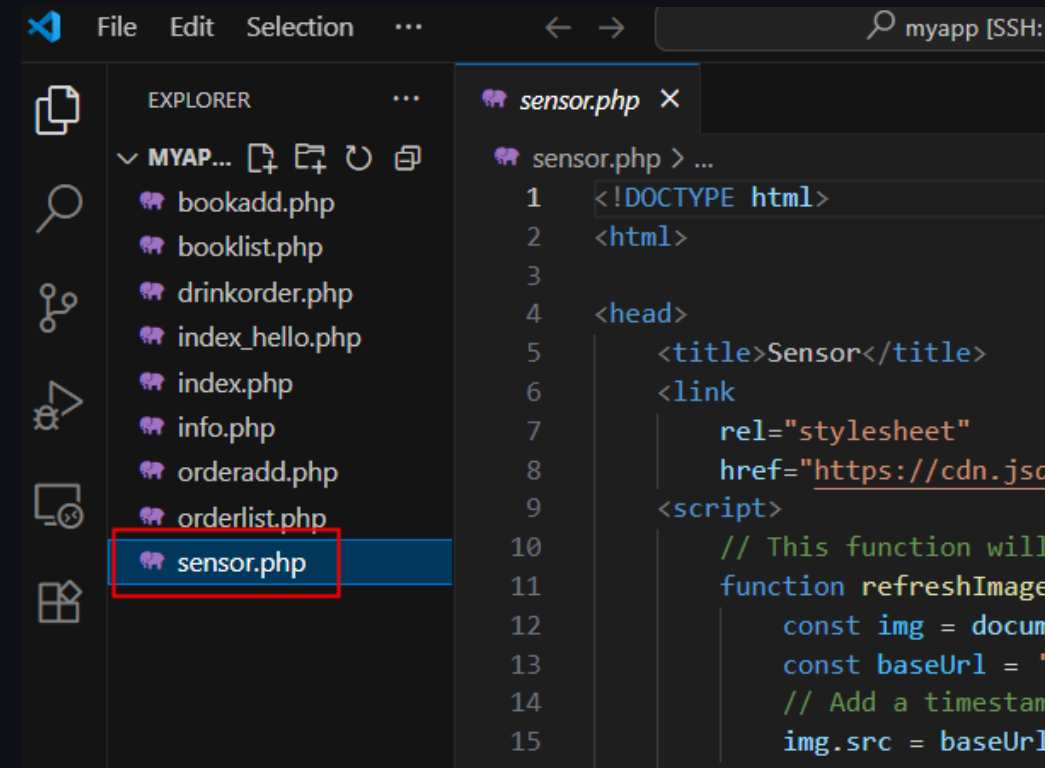
No properties

Add option



# Web Application

- Create `sensor.php` on the webserver.  
(Link)
  - Fix the webhook link accordingly.



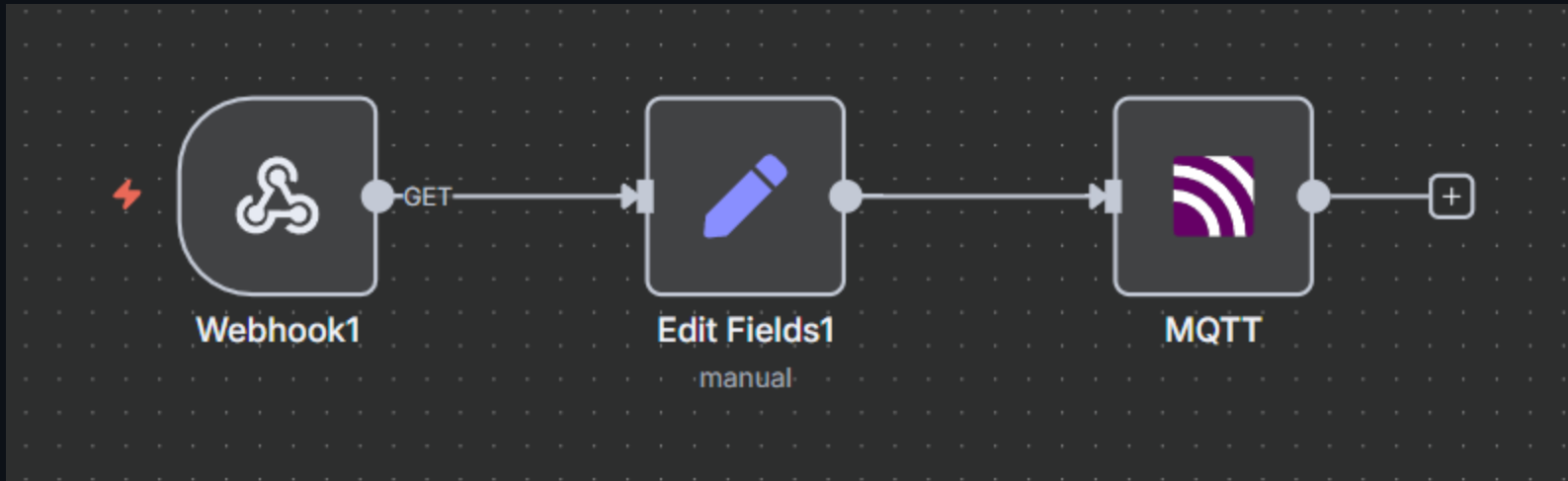
The screenshot shows the Visual Studio Code interface. On the left, the 'EXPLORER' sidebar displays a file tree for a project named 'MYAP...'. The files listed are: `bookadd.php`, `booklist.php`, `drinkorder.php`, `index_hello.php`, `index.php`, `info.php`, `orderadd.php`, `orderlist.php`, and `sensor.php`. The `sensor.php` file is selected and highlighted with a red rectangle. On the right, the editor window shows the content of `sensor.php`. The code is as follows:

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5     <title>Sensor</title>
6     <link
7         rel="stylesheet"
8         href="https://cdn.js
9     <script>
10         // This function will
11         function refreshImage
12             const img = docum
13             const baseUrl = '
14             // Add a timestan
15             img.src = baseUr
```


# Actuator


| Controlling devices

# Control Flow




# Webhook

 Webhook1

 Listen for test event

Parameters

Settings

Docs 

Webhook URLs

Test URL

Production URL

GET

https://pm1-ct102-n8n.iecmu.com/webhook-test/led

HTTP Method

GET

Path

led

Authentication

None

Respond

Immediately


44


# Webhook

We want to send the following (using query parameter)

- ON: <https://pmX-ctXXX-n8n.iecmu.com/webhook/led?led=on>
- OFF: <https://pmX-ctXXX-n8n.iecmu.com/webhook/led?led=off>


# Edit Fields

 Edit Fields1




Parameters

Settings


Docs 

Mode


Manual Mapping 

Fields to Set

led


AB String 

= {{ \$json.query.led }}



Drag input fields here or Add Field

Include Other Input Fields




Fixed Expression

Options

# MQTT

- Topic: Light/TestMQTTin

 MQTT

Execute step

Parameters Settings Docs

Credential to connect with

MQTT account

Topic

Light/TestMQTTin

Send Input Data

☐

Message

Fixed Expression

fx

{{ \$json.led }}

Options

No properties

Add option

# Web Application

- Uncomment

```
<html>
<head>
<script>
  // This function will refresh the image every second
  function refreshImage() {
    const img = document.getElementById('sensor-img');
    const baseUrl = 'https://pm1-ct102-n8n.iecmu.com/webhook/light';
    // Add a timestamp to bypass browser cache
    img.src = baseUrl + '?t=' + new Date().getTime();
  }

  // Turn LED ON (Uncomment for control)
  // -----
  // function turnLedOn() {
  //   fetch('https://pm1-ct102-n8n.iecmu.com/webhook/led?led=on', {
  //     method: 'GET'
  //   })
  //   .then(response => {
  //     alert('LED turned ON');
  //   });
  // }

  // Turn LED OFF (Uncomment for control)
  // -----
  // function turnLedOff() {
  //   fetch('https://pm1-ct102-n8n.iecmu.com/webhook/led?led=off', {
  //     method: 'GET'
  //   })
  //   .then(response => {
  //     alert('LED turned OFF');
  //   });
  // }

  // Start refreshing after the page loads
  window.onload = function() {
    refreshImage(); // Initial load
    setInterval(refreshImage, 2000); // Repeat

    // Uncomment for control
    // -----
    // document.getElementById('btn-on').addEventListener('click', turnLedOn);
    // document.getElementById('btn-off').addEventListener('click', turnLedOff);
  };
</script>
</script>
```