ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ
«МЕТІНВЕСТ ПОЛІТЕХНІКА»

АВТОМАТИЗАЦІЯ ПРОЦЕСІВ ВИРОБНИЦТВА НА БАЗІ ІНТЕРНЕТУ РЕЧЕЙ:

методичні вказівки до виконання практичних робіт

Запоріжжя 2026

УДК 681.5:044.77(072)
А18

Рекомендовано Науково-методичною радою ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА»
(протокол № 5 від «27» лютого 2026 р.)

Укладач
Койфман Олексій Олександрович, доцент, канд. тех. наук.
Давиденко Олег Віталійович, аспірант

А18 Автоматизація процесів виробництва на базі Інтернету речей : методичні рекомендації до виконання практичних робіт з дисципліни «Автоматизація процесів виробництва на базі Інтернету речей» / уклад.: О. О. Койфман., О. В. Давиденко. Запоріжжя : ТОВ «ТЕХНІЧНИЙ УНІВЕРСИТЕТ «МЕТІНВЕСТ ПОЛІТЕХНІКА», 2026. 115 с.

Методичні вказівки включають методичні пояснення щодо порядку виконання практичних робіт, які наддадуть знання та вміння до застосовування спеціалізованого програмного забезпечення та цифрових технології для розв’язання складних задач і проблем автоматизації та комп’ютерно-інтегрованих технологій, з використанням інтелектуальних датчиків, Інтернету речей, цифрових двійників та мережевих технологій з урахуванням сучасних тенденцій розвитку галузі.

УДК 681.5:044.77(072)

ЗМІСТ

ВСТУП

Методичні вказівки до виконання практичних робіт розроблено в результаті плідної співпраці з доцентом, к.т.н., Пупена Олександр Миколайович, доцентом кафедри Автоматизації та комп'ютерних технологій систем управління ім. проф. А.П. Ладанюка Національного університету харчових технологій (https://www.iasu-nuft.pp.ua/about-5).

В основі практикуму використано відкритий ресурс з дисципліни «Технології Індустрії 4.0» (https://pupenasan.github.io/TI40/). Додатково отримано згоду автору на використання матеріалів в начальному процесі.

В якості звітної документації відповідно до виконаних практичних робіт у відповідні завдання завантажуються лістинги програм (потоков), та скріншоти результатів виконання програм.

Максимальна кількість балів, яку здобувач може отримати за одну практичну роботу, дорівнює 10 балам.

У таблиці наведено розподіл балів відповідно критерію оцінювання.

Кількість балів	Критерій оцінювання
10	Здобувач(ка) працював(ла) на практичних заняттях, приймав(ла) активну участь у виконанні завдання, виконав(ла) завдання у повному обсязі та завантажив(ла) звітні матеріали відповідно до методичних рекомендацій з семестровим графіком (до наступного заняття)
9-8	Здобувач(ка) працював(ла) на практичних заняттях, частково виконав(ла) завдання та завантажив(ла) звітні матеріали відповідно до методичних рекомендацій в Moodle пізніше терміну вказаного у семестровому графіку
7-6	Здобувач(ка) не працював(ла) на практичних заняттях, повністю виконав(ла) завдання та завантажив(ла) звітні матеріали відповідно до методичних рекомендацій в Moodle пізніше терміну вказаного у семестровому графіку
5-4	Здобувач(ка) не працював(ла) на практичних заняттях, частково виконав(ла) завдання та завантажив(ла) звітні матеріали відповідно до методичних рекомендацій в Moodle пізніше терміну вказаного у семестровому графіку
3-2	Здобувач(ка) не працював(ла) на практичних заняттях, частково виконав(ла) завдання та завантажив(ла) звітні матеріали не в повному обсязі відповідно до методичних рекомендацій в Moodle пізніше терміну вказаного у семестровому графіку
0	Здобувач(ка) був(ла) відсутня на практичних заняттях та не завантажив(ла) звіт (сертифікат) в Moodle

ПРАКТИЧНА РОБОТА №1. ОСНОВИ РОБОТИ З NODE-RED

1.1 Мета

Ознайомлення з інструментом програмування Node-RED для об'єднання апаратних пристроїв, API та онлайн-послуг.

Встановлення та налаштування Node-RED на платформі Windows.

Створення та розгортання базових потоків (flow) в Node-RED.

Використання різних вузлів (nodes) для обробки повідомлень.

Знайомство з вузлами типу change, delay, та function.

Інсталяція та налаштування модуля node-red-dashboard для розробки веб-інтерфейсів користувача.

1.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:

- інсталяція Node-RED під Windows;
- знайомство з Node-RED;
- Підключення та ознайомлення з модулем node-red-dashboard;
- робота з поштою (не обов’язкова частина);
- робота з Modbus;
- робота з JS об’єктами та обробка системної інформації;
- експорт потоку.

1.3 Хід роботи

1.3.1 Інсталяція Node-RED під Windows

Node-RED - це інструмент програмування для об'єднання апаратних пристроїв, API та онлайн-послуг новими та цікавими способами.

Редактор функціонує на основі браузера, а програма виглядає як об’єднані в потоки (flow) вузли (node), що вибираються з палітри.

Після редагування потоки можуть бути розгорнуті в середовище виконання в один клік. Вбудована бібліотека дозволяє зберігати корисні функції, шаблони або потоки для повторного використання.

Легке середовище виконання побудоване на Node.js, що використовує подіємо-керовану неблокуючу модель. Це робить його ідеальним для роботи на краю (Edge) мережі на недорогих апаратних засобах, таких як Raspbery PI, а також у хмарі.

З більш ніж 225 000 модулів у репозиторію пакетів Node, легко збільшити діапазон вузлів палітри для додавання нових можливостей.

Потоки, створені в Node-RED, зберігаються за допомогою JSON, які можна легко імпортувати та експортувати для спільного використання з іншими.

Онлайн бібліотека потоків дозволяє вам поділитися своїми найкращими потоками зі світом.

Підтримувані платформи: Windows 11, Windows 10, Windows 8, Windows 7.

1.3.1.1 Завантаження Node.JS

Завантажити msi-файл Node.JS LTS версії

1.3.1.2 Встановлення Node.JS

Запустити на виконання msi-файл від імені адміністратора і встановити Node.JS, при виклику діалоговий вікон все залишати за замовченням.

1.3.1.3 Визначення версії npm

Після інсталяції запустіть командний рядок (CMD), у якому введіть

node --version && npm –version

повинно вивести версію node() - npm ()

npm (Node Package Manager) - це менеджер пакунків для мови програмування JavaScript. Для середовища виконання Node.js є менеджером пакунків за замовчуванням. Включає в себе клієнт командного рядка, який також називається npm, а також онлайн-базу даних публічних та приватних пакунків, яка називається реєстром npm. Реєстр доступний через клієнт, а доступні пакунки можна переглядати та шукати через веб-сайт npm. Менеджер пакунків та реєстр керуються npm, Inc.

1.3.1.4 Інсталяція Node-RED

Інсталюйте Node-RED . Інсталювання проводиться з використанням npm з командою install. Наберіть в командному рядку

npm install -g --unsafe-perm node-red

почнеться процедура інсталяції.

1.3.1.5 Запуск Node-RED.

Запустіть Node-RED з командного рядка

node-red

Можуть бути показані повідомлення про пропозицію розблокування брандмауером, з якими треба погодитись.

1.3.1.6 Відкриття редактору.

Відкрийте браузер, перейдіть до редактору Node-Red, за посиланням http://127.0.0.1:1880/ .

Для того, щоб Node-RED виконувався, вікно з командним рядком не можна закривати.

1.3.2 Знайомство з Node-RED

Програма створена на Node-RED складається з **потоків** (**Flow**), які виконуються як умовно незалежні програми. Потоки – це зв'язані між собою інформаційними **дротами** (**wires**) **вузли** (**Node**), що виконують певну функцію. Таким чином, ідеологія програмування Node-RED дещо схоже на побудову програм на мові FBD, що є стандартною для програмування ПЛК (IEC 61131-3). Тим не менше, між цими мовами є значні відмінності.

1.3.2.1 Ознайомлення з редактором Node-RED.

Відкрийте в браузері редактор Node-RED, якщо він ще не відкритий. Ознайомтеся з його основними частинами (рис. 1.1).

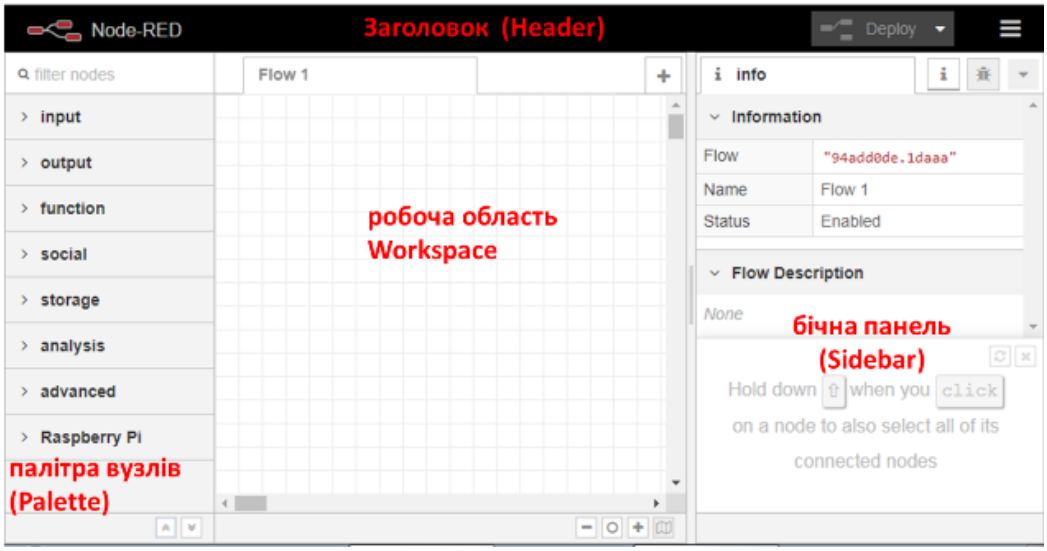


Рисунок 1.1 - Вигляд редактору Node-RED

Детально про роботу з редактором Ви можете ознайомитися з довідника.

1.3.2.2 Розміщення вузлів inject і debug

Виберіть з палітри і розмістить на робочій області вузли
- **input->inject**
- **output->debug**

З'єднайте їх між собою. Повинно вийти як на рис. 1.2. Блакитні кружечки значать, що зміна в вузлах ще не відобразилася в середовищі виконання, так як змінена програма не була в ньому розгорнута.

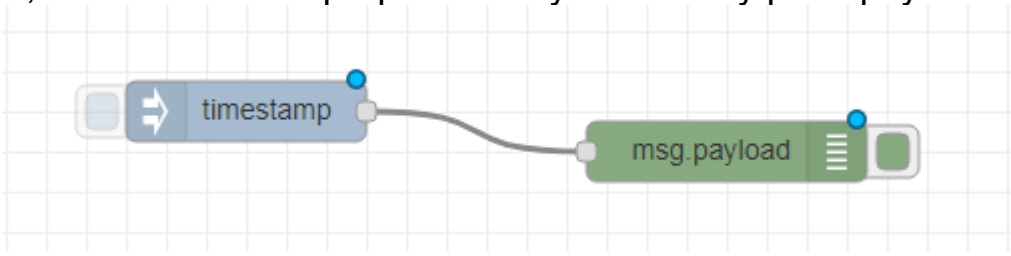


Рисунок 1.2 - Вигляд програми

1.3.2.3 Розгортання

В заголовку виберіть пункт **Deploy->Modified Nodes** (рис. 1.3), після чого натисніть **Deploy (Розгортання)**.

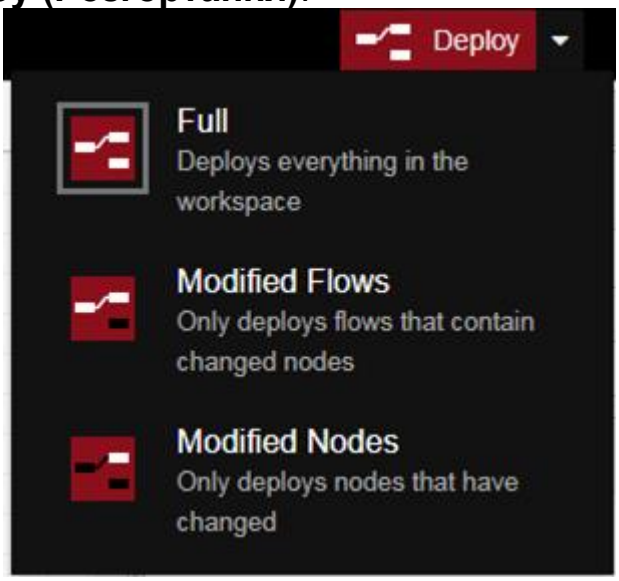


Рисунок 1.3 - Вибір варіанту розгортання

При вдалому розгортанні з'явиться повідомлення.
А в робочому просторі вузли вже будуть без блакитних кружечків.

1.3.2.4 Відображення вікна повідомлень

Для перевірки роботи програми, на бічній панелі треба відобразити вікно **Debug messages** (налагоджувальні повідомлення) шляхом натиснення кнопки з «жуком».

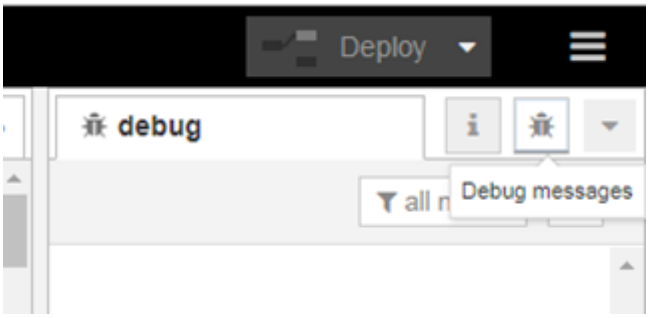


Рисунок 1.4 - Показати вікно повідомлень

Ліворуч вузла типу Inject з назвою «timestamp» є кнопка, яка приводить до ініціювання розрахунку ланцюжка вузлів, що починаються з нього. Натисніть на цю кнопку.

В результаті з'явиться повідомлення про успішне вприскування (Inject) а на панелі повідомлень з'явиться повідомлення (рис. 1.5).

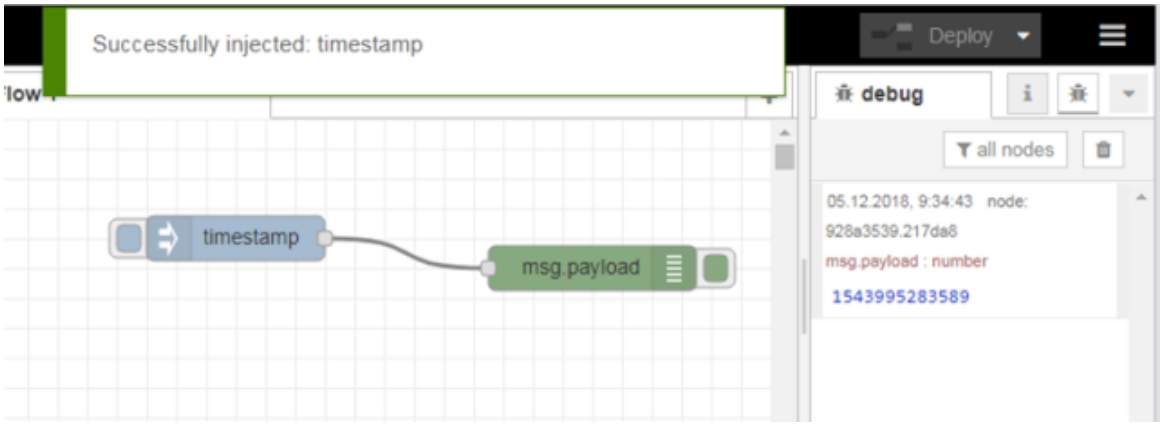


Рисунок 1.5 - Результат виконання програми

На цьому прикладі розглянемо, як виконується програма.

У більшості випадків перерахунок вузлів починається тоді, коли на його вхід подається повідомлення (message). Повідомлення – це прості об'єкти (типу структурні змінні) JavaScript що можуть мати будь який набір властивостей. Тобто в даній програмі після перерахунку вузла з іменем «timestamp» буде сформовано об'єкт-повідомлення (**msg**) і переданий по дроту вузлу з іменем (msg.payload).

На вхід вузла з іменем «timestamp» повідомлення не надходять, бо він є ініціатором розрахунку. Всі вузли палітри що входять в групу **Input** є ініціаторами розрахунку. Ініціація вузлів типу **Inject** відбувається шляхом ручного запуску по кнопці, або через певні інтервали часу, що

налаштовується у вузлі. Ініціювання повідомлення це формування полів **msg** та відправка його іншим вузлам по дротам. Повідомлення, надіслане вузлом Inject, має властивості **payload** (корисне навантаження) та **topic** (тема). За замовченням Inject записує у властивість topic відмітку часу (timestamp – кількість мілісекунд з 1980 року).

Вузол типу Debug «msg.payload» використовується для відображення повідомлень на бічній панелі Debug. Таким чином, після отримання повідомлення, цей вузол відображає його зміст на бічній панелі.

1.3.2.5 Налаштування Inject на періодичне оновлення

Змініть налаштування властивостей вузлів, як показано на рис. 1.6: змініть імена вузлів, вкажіть тему (topic) та періодичність оновлення для вузла типу Inject. Вікно налаштування з'являється по подвійному кліку по вузлу.

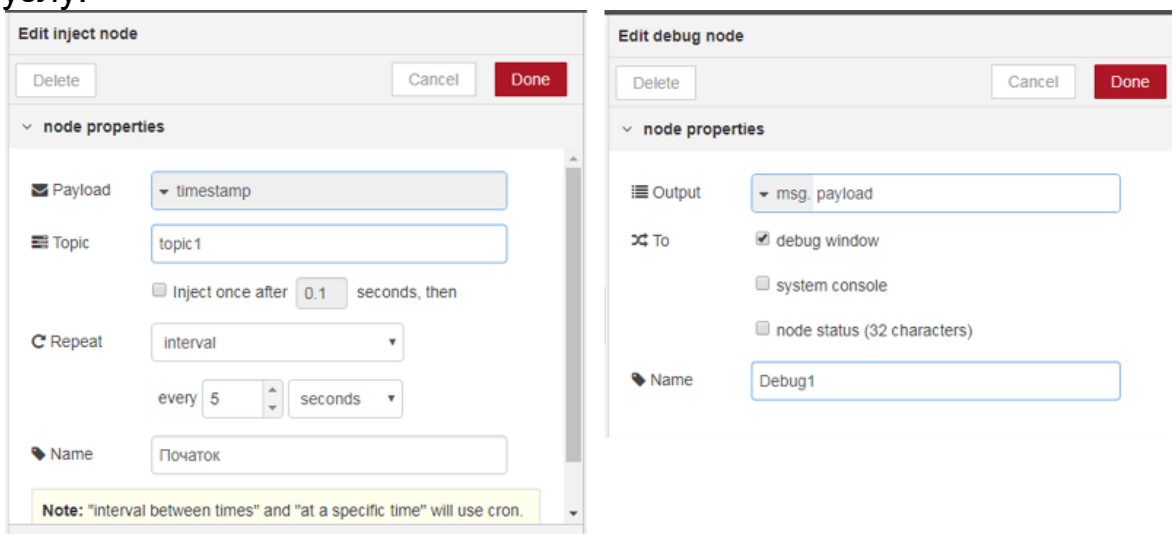


Рисунок 1.6 - Налаштування вузлів

Зробіть розгортання, та проаналізуйте зміст виведених у вікні Debug повідомлень.

1.3.2.6 Налаштування Inject на текстове повідомлення

Змініть вузол «Початок» так, щоб він формував корисне навантаження текстом «Це текстове повідомлення» (рис. 1.7), та проаналізуйте як воно виводиться на вікно Debug.

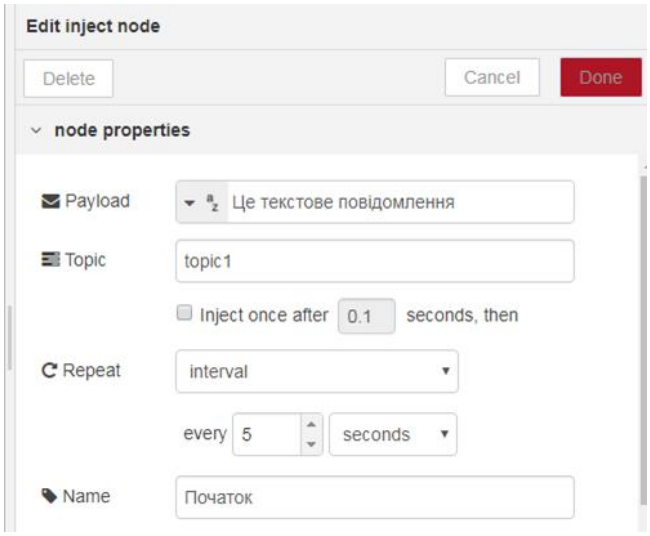


Рисунок 1.7 - Формування текстового повідомлення

1.3.2.7 Використання вузлів типу change та delay

Змініть вузол «Початок» так, щоб він знову формував корисне навантаження відміткою часу (Timestamp). Розгорніть (deploy) програму та проконтролюйте щоб відмітка часу кожні 5 секунд відображалася у вікні повідомлень

Ознайомтеся з роботою вузлів типу **change** та **delay** в інструкції користувача.

Змініть програму, як показано на рис. 1.8, використовуючи вузли delay (“delay 1s”…”delay 4s”) та change (“set1”…”set5”).

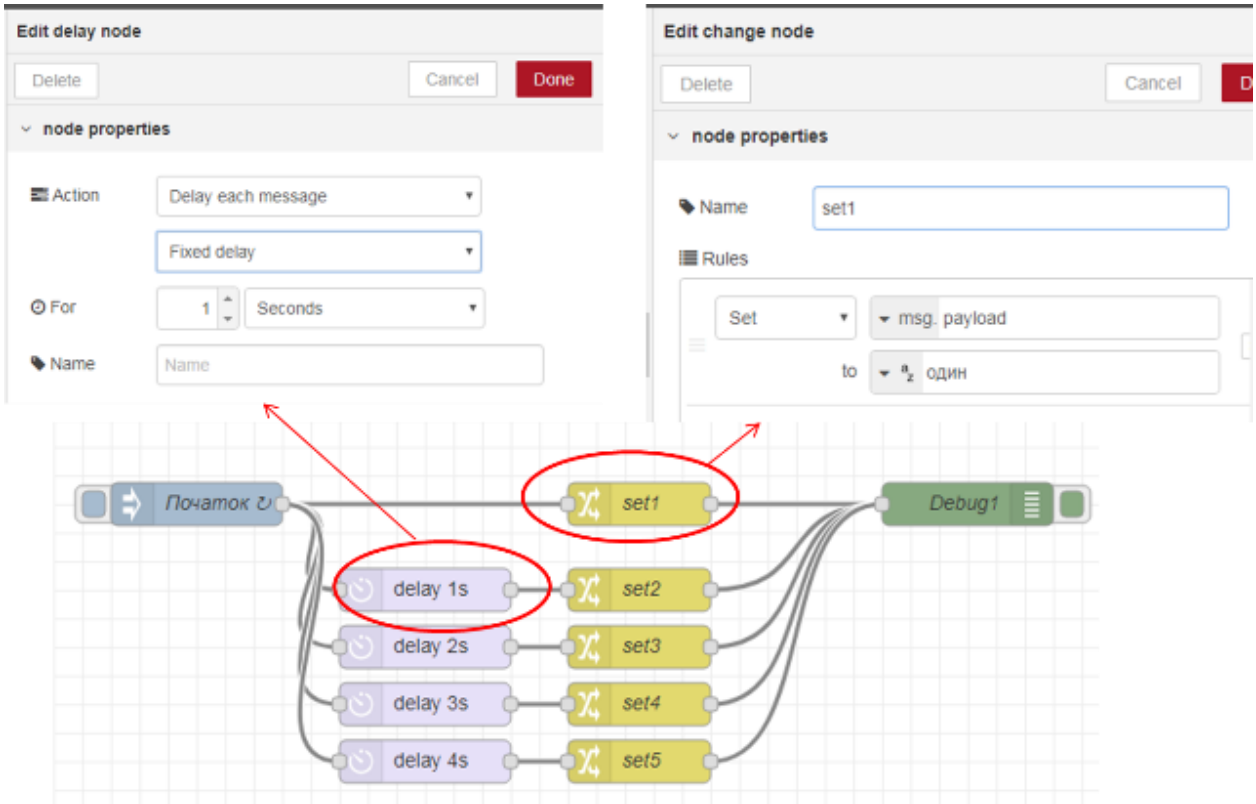


Рисунок 1.8 - Використання вузлів типу change та delay

Для вузлів delay виставте затримки:

- delay 1s – 1 seconds
- delay 2s – 2 seconds
- delay 3s – 3 seconds
- delay 4s – 4 seconds

Для вузлів change виставте правило рівним «set», та змініть властивості «to» на наступні текстові поля:

- set1 – один
- set2 – два
- set3 – три
- set4 – чотири
- set5 – п’ять

Розгорніть (deploy) програму та проконтролюйте щоб кожної секунди у вікні повідомлень виводилося конкретне повідомлення від «один» до «п’ять».

1.3.2.8 Ознайомлення з роботою вузлів типу function

Вузол function може обробляти повідомлення з використанням javascript. Змініть програму так, щоб відмітка часу виводилася в форматі дати та часу. Для цього використовується об’єкт типу Data та його метод toLocaleString(). Зробіть розгортання та переведіть вузол “Debug1” в режим приховання повідомлень.

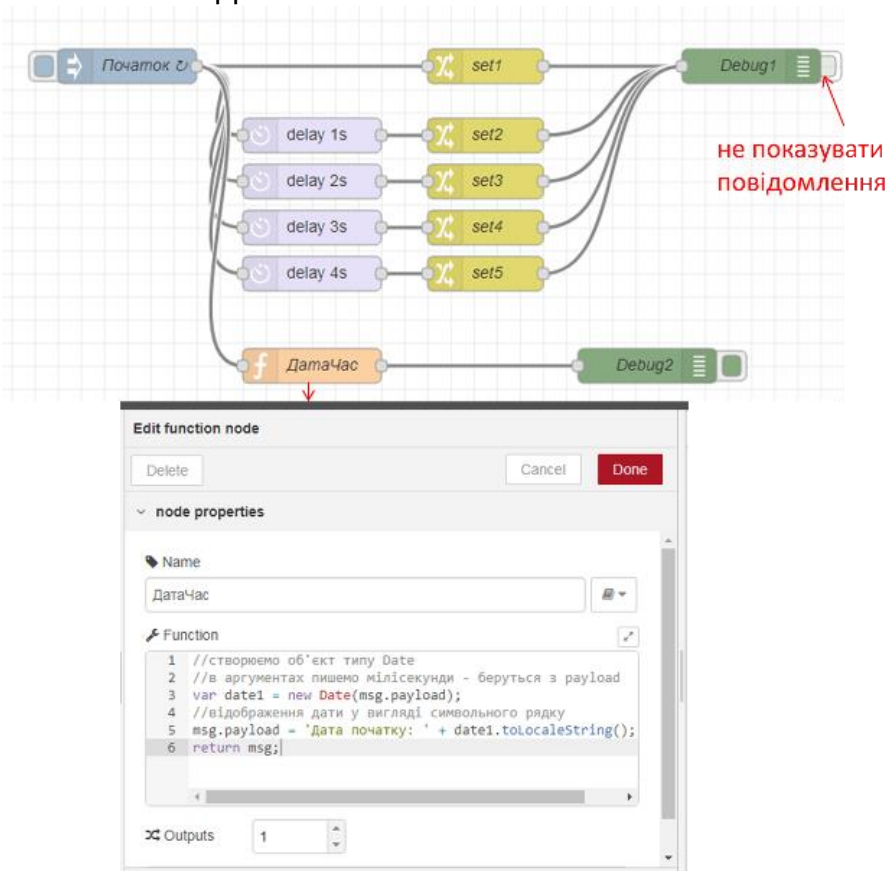


Рисунок 1.9 - Змінена програма з вузлом function

Про синтаксис **javascript** можна дізнатися тут (укр. мова).
Про об'єкт **Date** можна прочитати тут або тут.

1.3.3 Підключення та ознайомлення з модулем node-red-dashboard

Node-RED надає можливості розробки WEB-інтерфейсу користувача. Це робиться за допомогою модуля node-red-dashboard, який необхідно встановити. Додатково про node-red-dashboard можна прочитати у довіднику.

Node-RED дозволяє інсталювати та обновляти палітру вузлів. Це робиться через Manage Palette (рис. 1.10). Деталі інсталяції читайте в інструкції користувача.

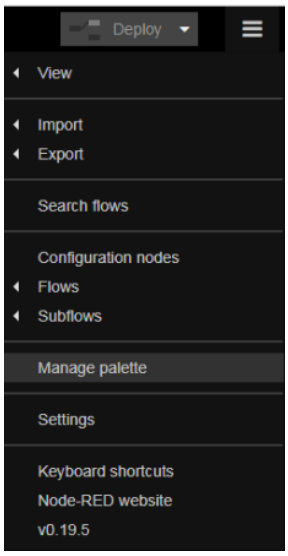


Рисунок 1.10 - Вибір Manage palette

1.3.3.1 Інсталяція node-red-dashboard

В налаштуваннях палітри на вкладці Install в поле фільтру введіть node-red-dashboard і інсталюйте даний пакет (рис. 1.11):

- натисніть кнопку install;
- підтвердьте інсталяцію у вікні повідомлення;
- після інсталяції закрийте вікно керування палітрою.

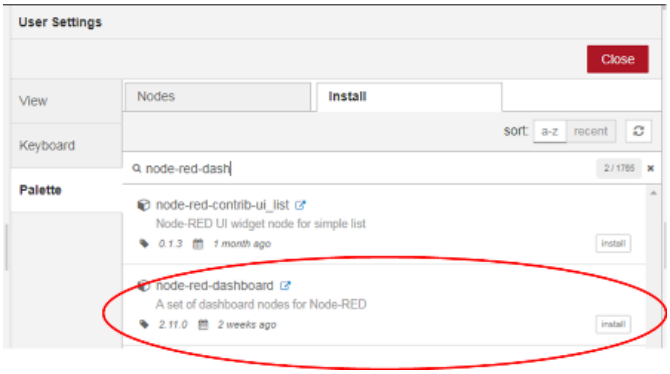


Рисунок 1.11 - Вибір Manage palette

Перевірте чи з’явилася в палітрі розділ Dashboard.

1.3.3.2 Добавлення закладок

Після встановлення у бічній панелі з’явилася нова іконка з зображенням діаграми (рис. 1.12) . Натисніть на ній.

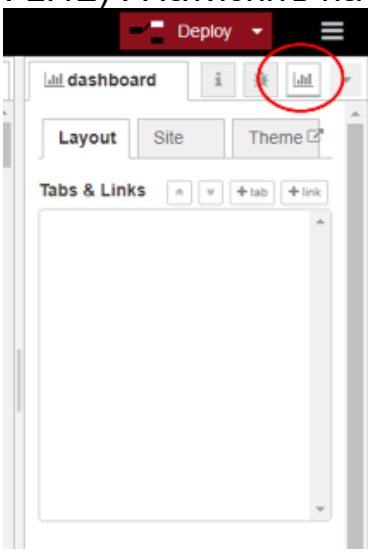


Рисунок 1.12 – Зображення нової іконки

В Layout додайте дві закладки (tab) та змініть їх назви як це показано на рис. 1.13.

Однак ім’я першої закладки повинно називатися Вашим прізвищем та ім’ям, наприклад «Іваненко Іван».

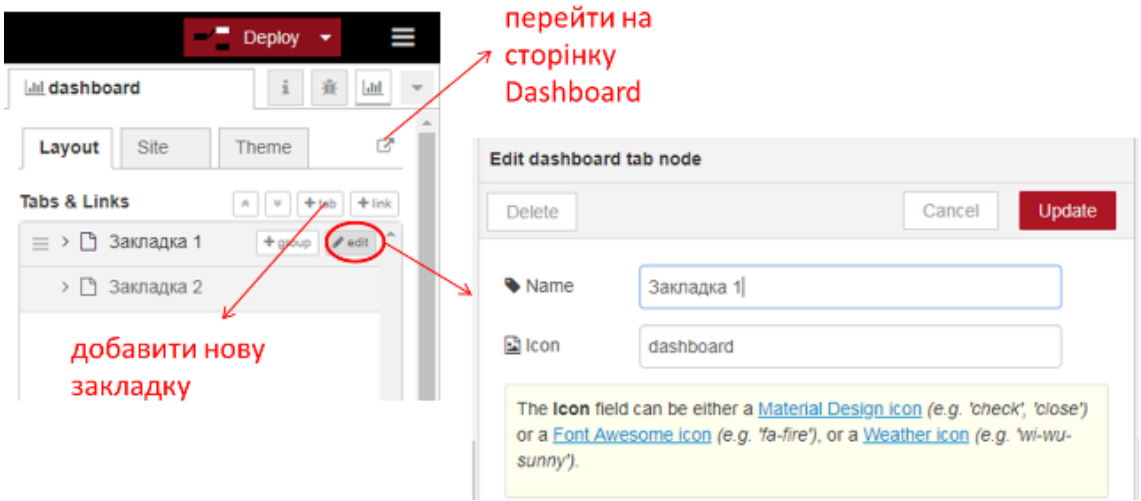


Рисунок 1.13 – Додавання нових закладок

1.3.3.3 Створення вузлу Dashboard text для виведення дати та часу

Модифікуйте програму, створивши вузол типу dashboard-> text і підєднавши його до вузла «ДатаЧас» (рис. 1.14). Налаштуйте вузол відповідно до рис. 14 **однак ім’я першої групи повинно називатися**

Вашим прізвищем «Іваненко». Створіть ще одну групу, яка повинна називатися Вашим ім'ям «наприклад Іван».

Після усіх налаштувань зробіть розгортання, відкрийте створений Dashboard, шляхом натискання кнопки переходу (рис. 1.13.), або ввівши в новій вкладці браузера http://127.0.0.1:1880/ui

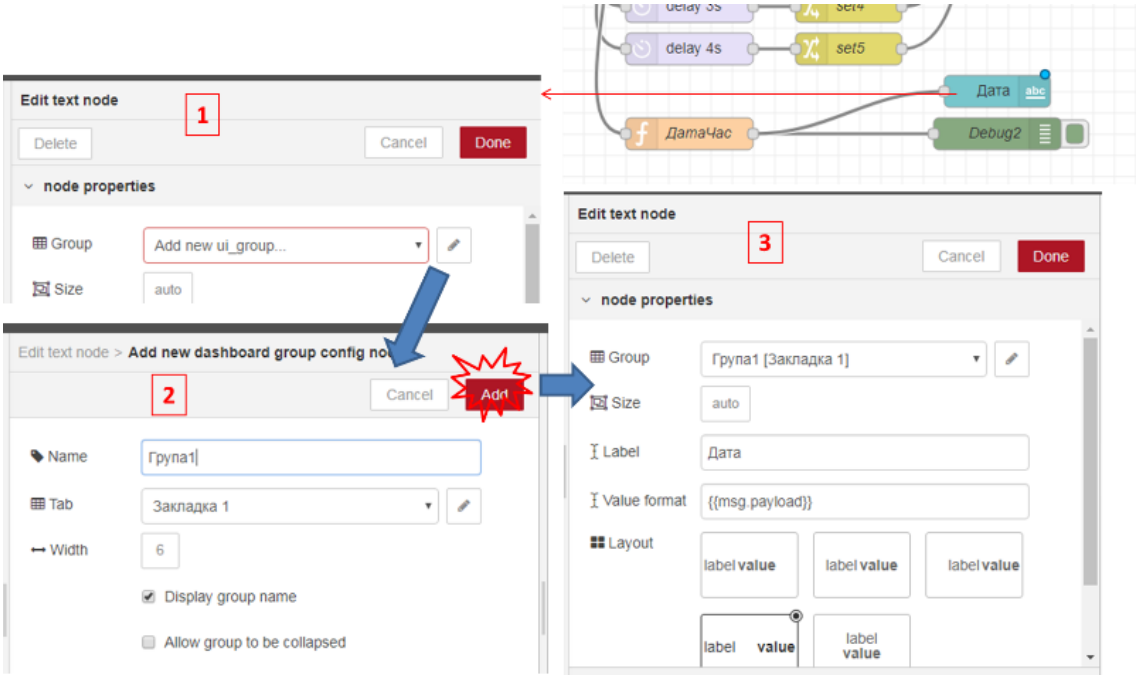


Рисунок 14 – Приклад модифікації програми

На вкладці повинно з'явитися щось типу такого, як показано на рис. 1.15

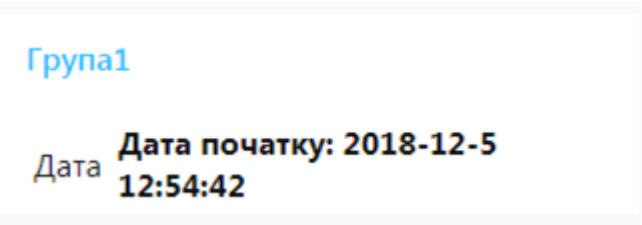


Рисунок 1.15 – Приклад виведення інформації на вкладці

1.3.3.4 Створення вузлу Dashboard text для виведення числа прописом

Аналогічним чином зробіть для відображення числа прописом (рис. 1.16).

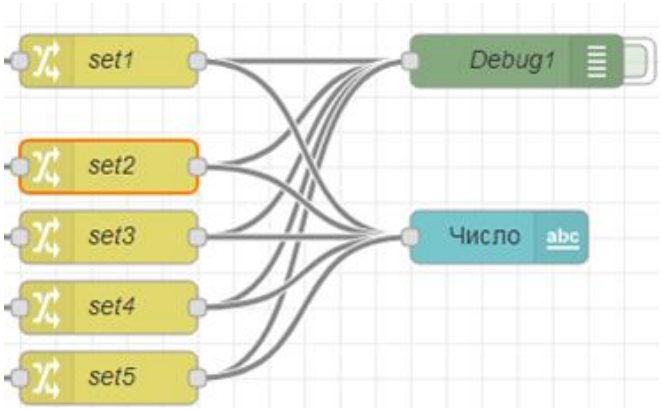


Рисунок 1.16 – Програмна реалізація відображення числа прописом

1.3.3.5 Використання вузлів Slider, Gauge, Audio out

Додайте до програми фрагмент, як показано на рис. 1.17.

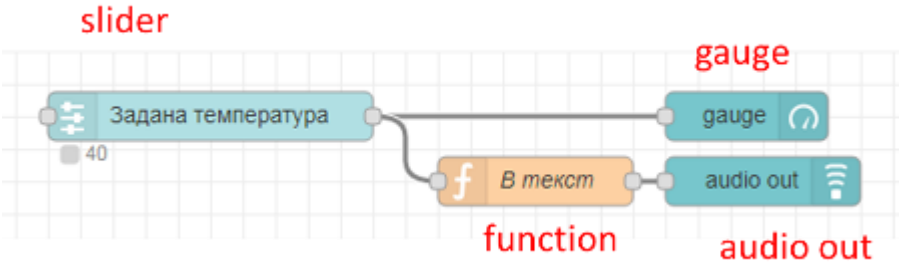


Рисунок 1.17 – Використання вузлів Slider, Gauge, Audio out

Налаштування вузлів показані на рис. 1.18-1.21

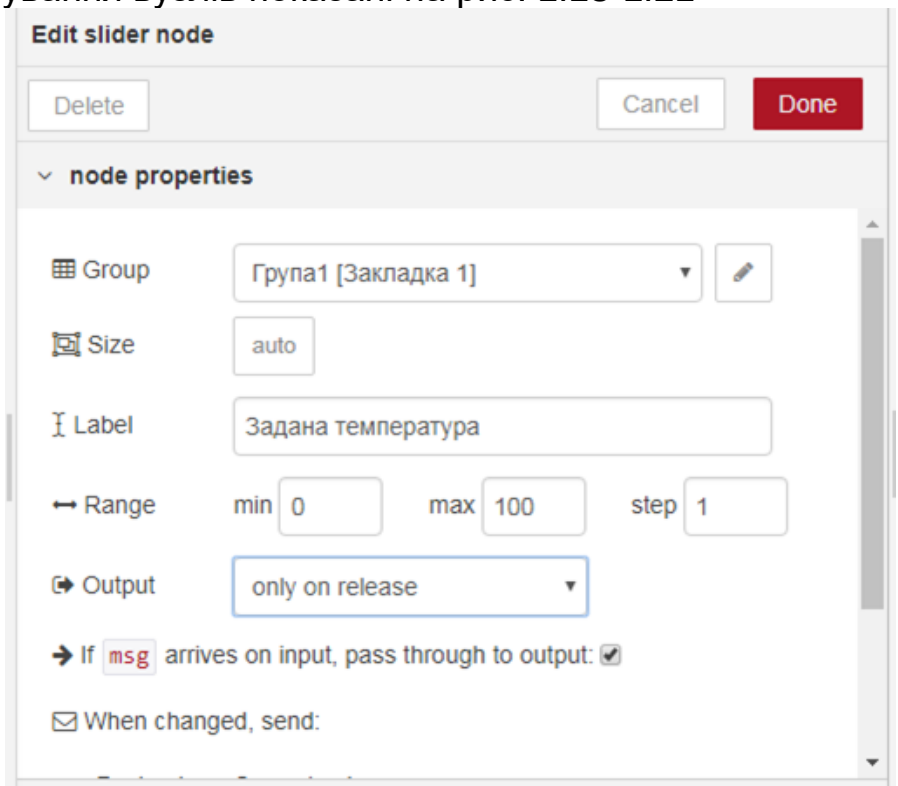


Рисунок 1.18 – Налаштування вузлу Slider

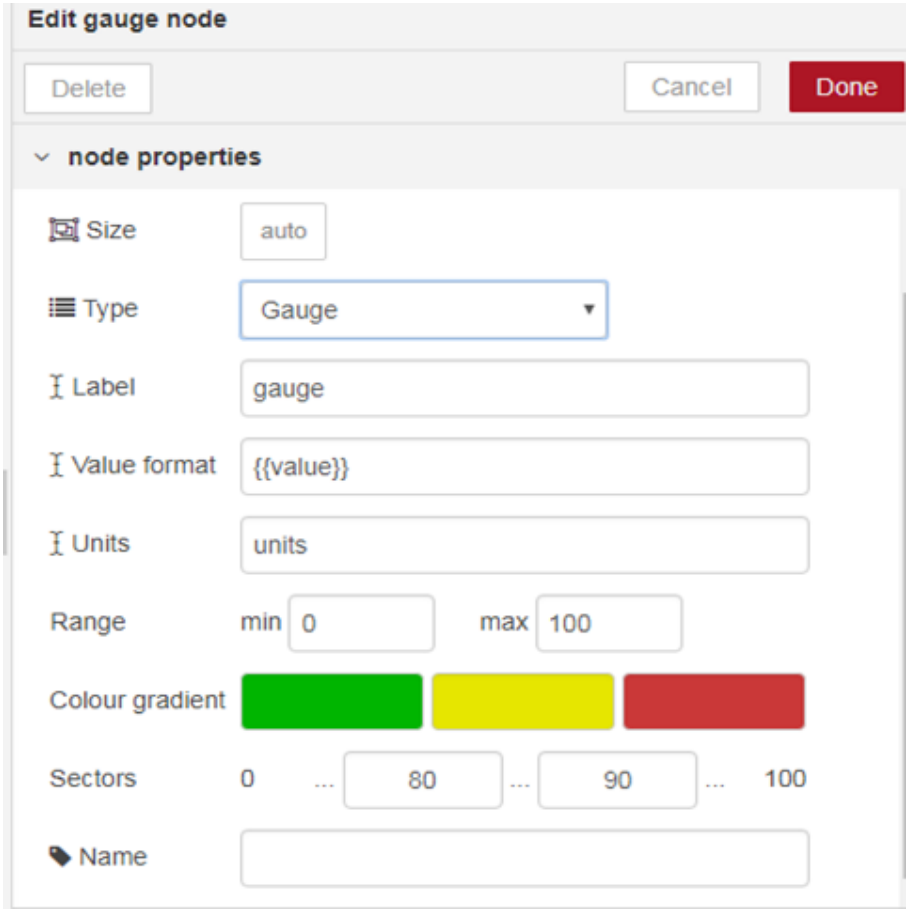


Рисунок 1.19 – Налаштування вузлу Gauge

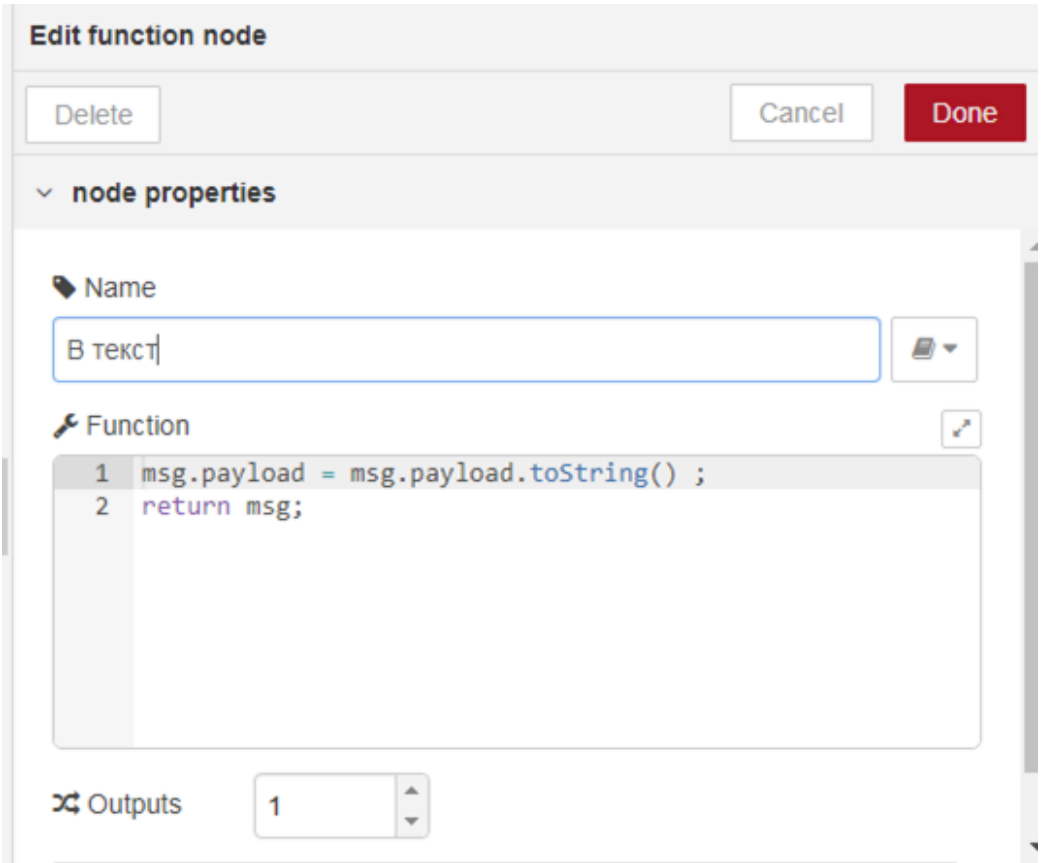


Рисунок 1.20 – Налаштування вузлу Function

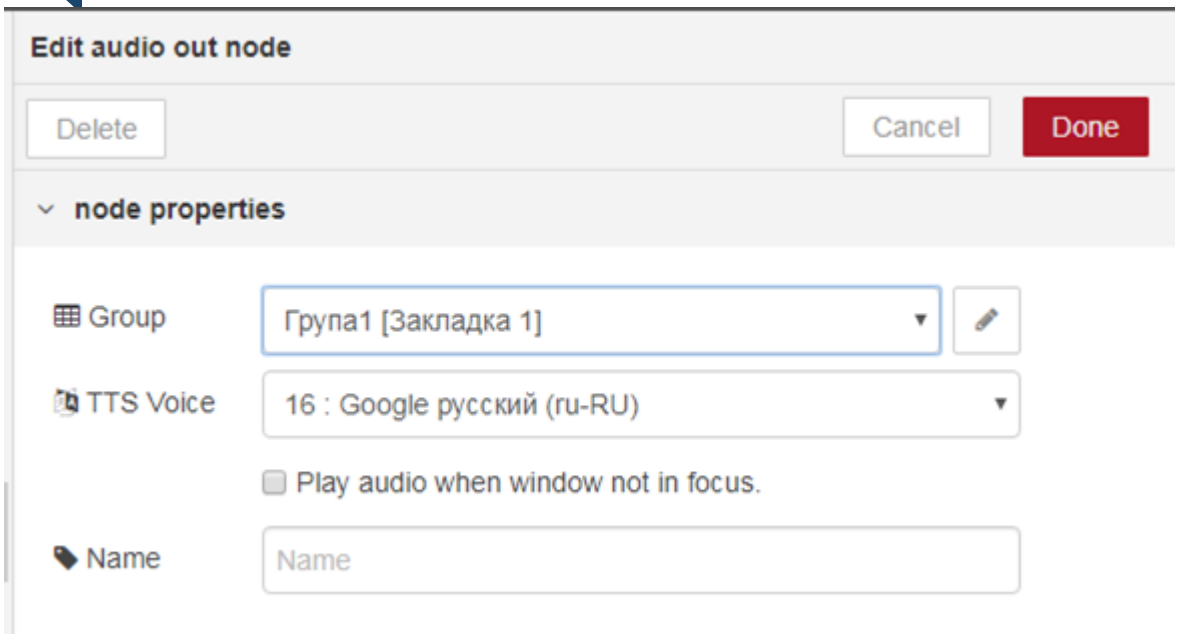


Рисунок 1.21 – Налаштування вузлу Audio

Зробіть розгортання і подивіться результат на вікні Dashboard.

1.3.3.6 Робота з вузлом switch

Ознайомтеся з принципами роботи вузла **switch**. Модифікуйте програму відповідно до наведеної на рис. 1.22. Вузли налаштуйте відповідно до рис. 1.23 – 1.26.

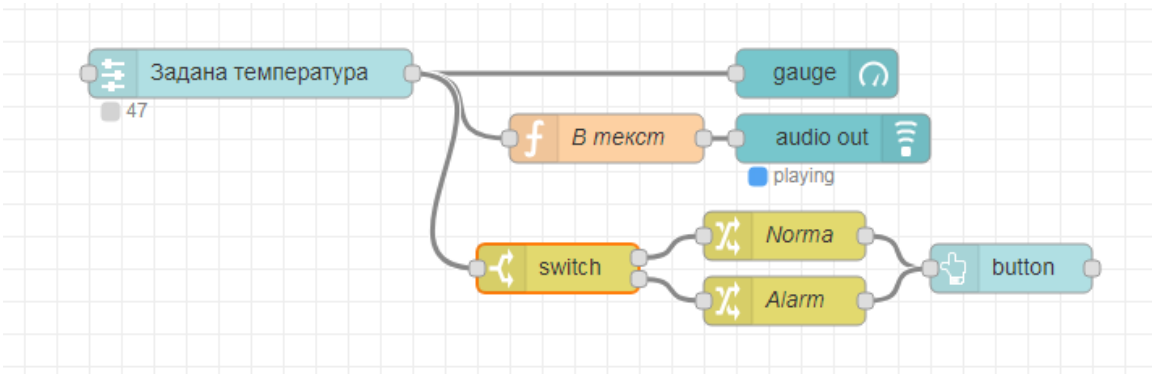


Рисунок 1.22 – Фрагмент модифікованої програми

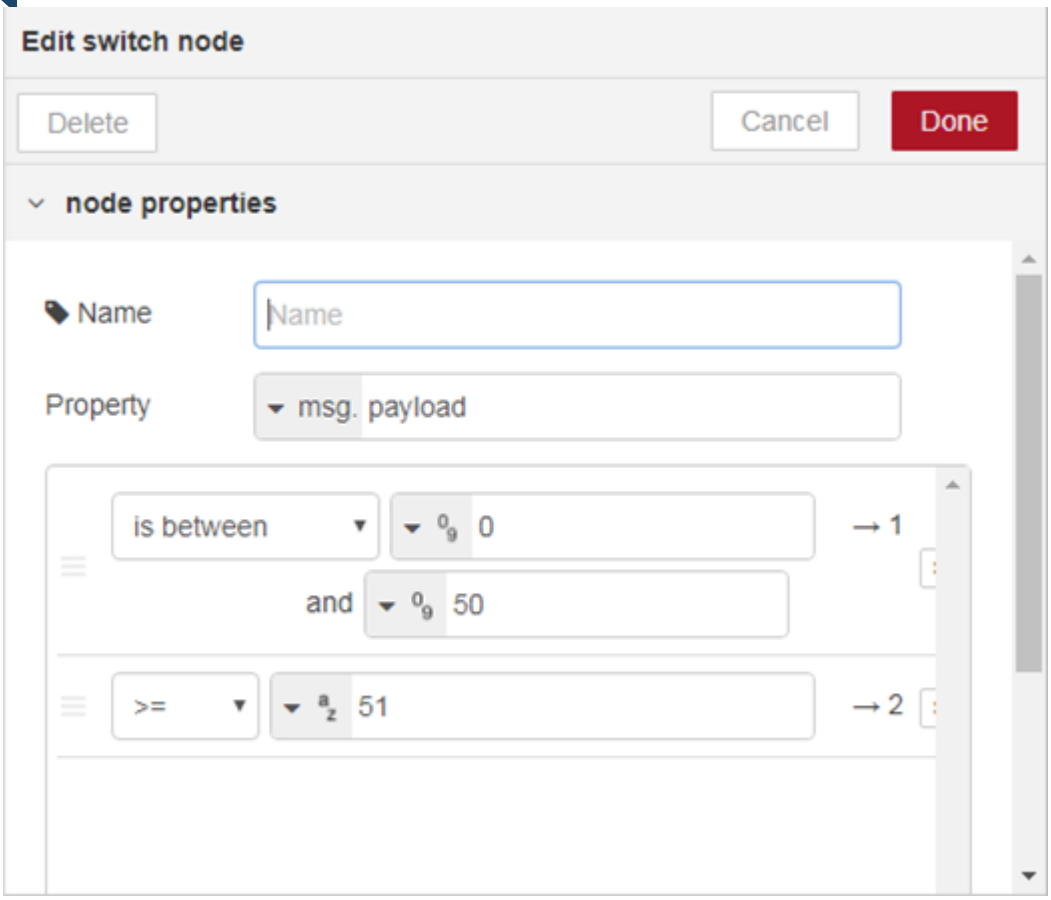


Рисунок 1.23 – Налаштування вузла switch

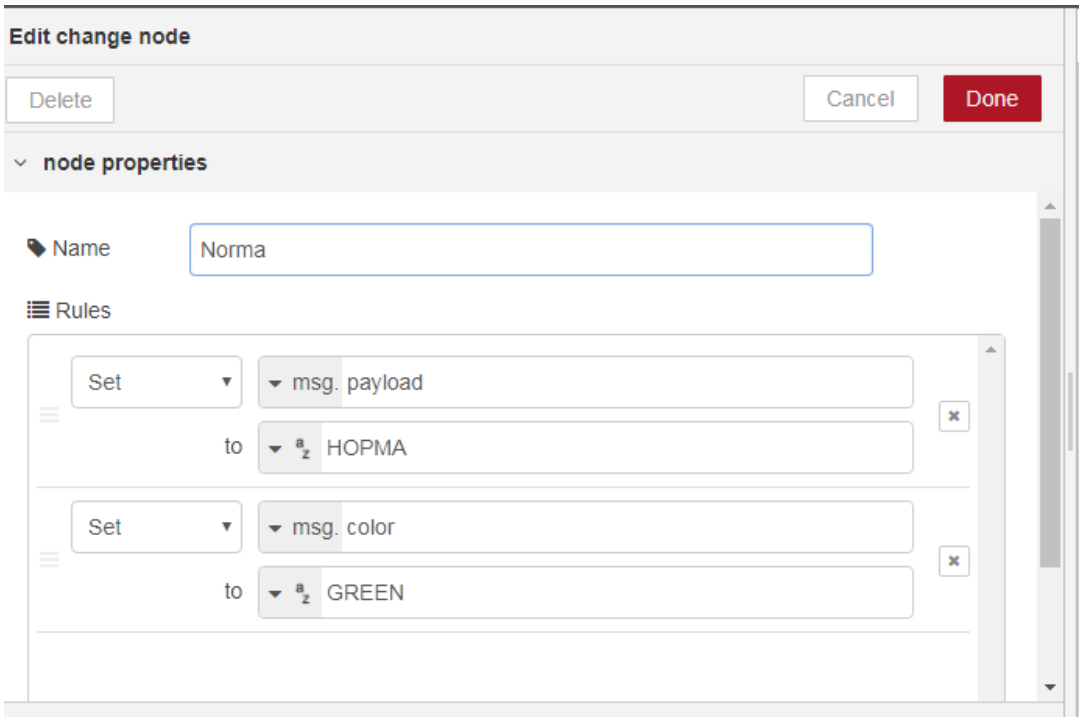


Рисунок 1.24 – Налаштування вузла norma

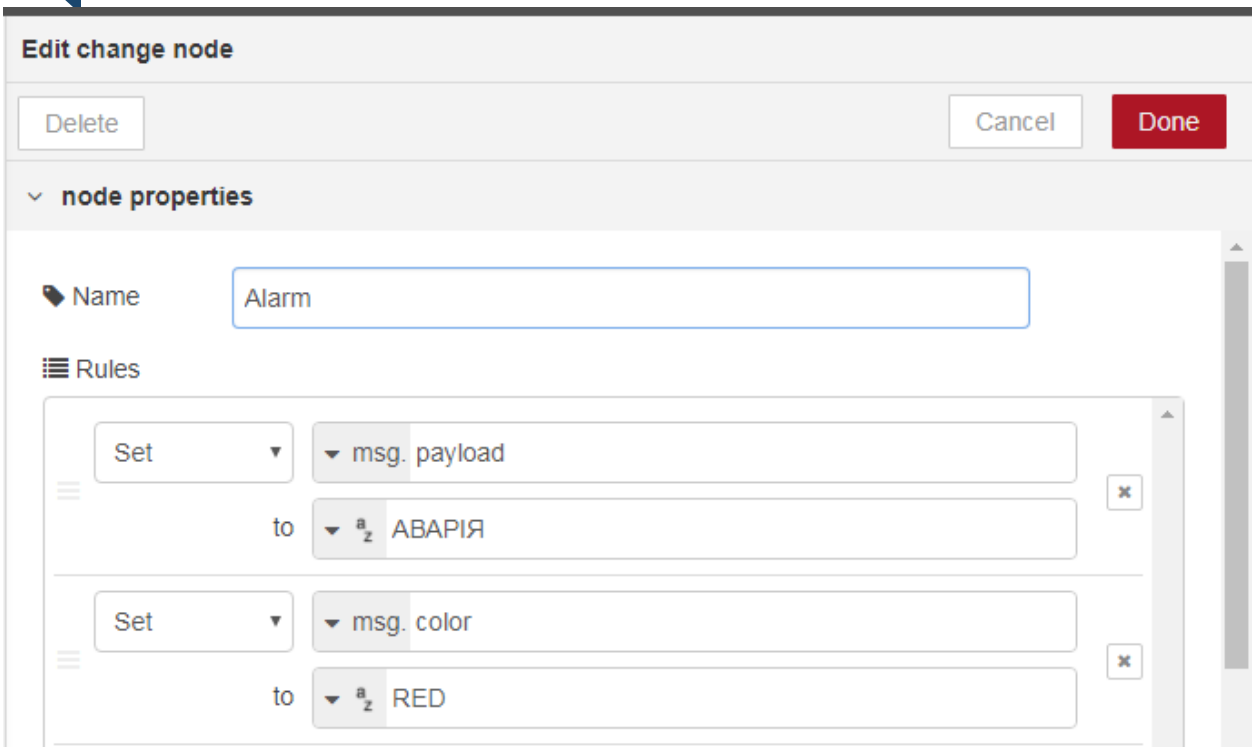


Рисунок 1.25 – Налаштування вузла Alarm

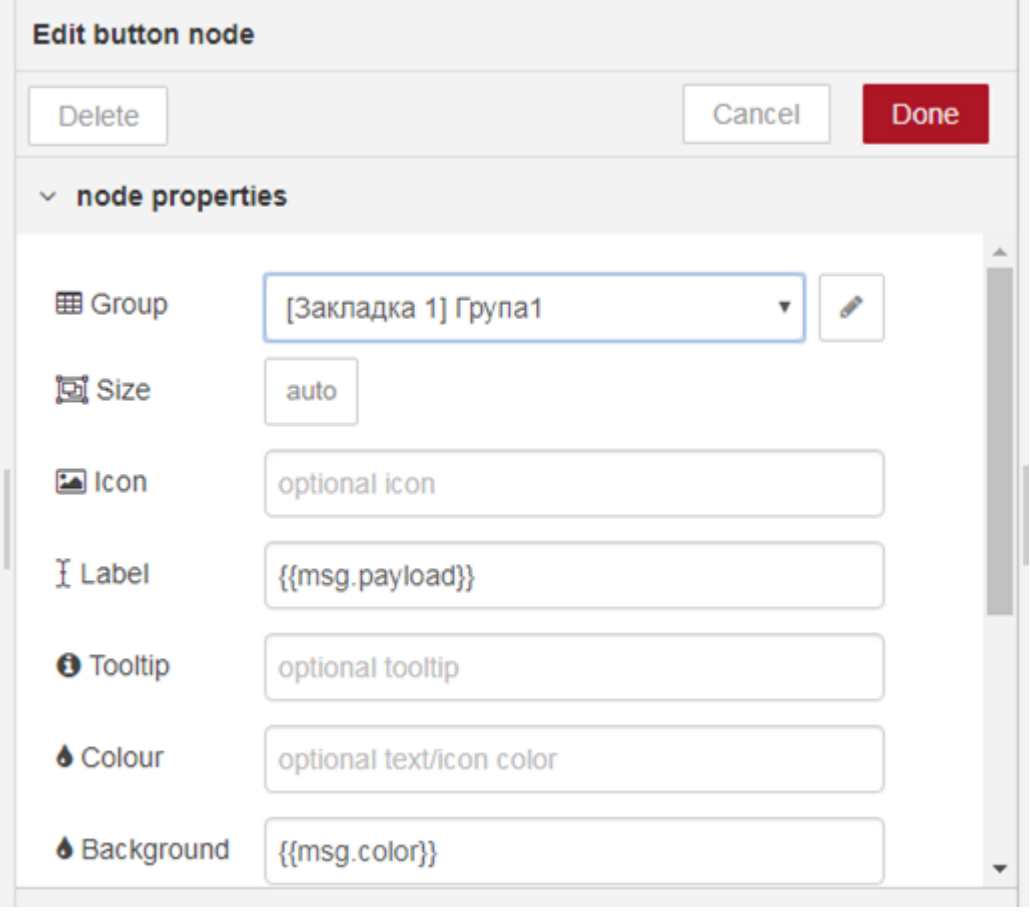


Рисунок 1.26 - Налаштування вузла Button

Зробіть розгортання проекту, перевірте як працює програма. Для цього на сторінці веб-інтерфейсу змініть значення заданої температури в діапазоні 0-50, а потім >50.

Ця частина програми працює наступним чином. При зміні значення температури, в msg.payload значення поступає на обробку в вузол switch, де на один із 2-х виходів формується повідомлення в залежності від тієї умови, яка спрацювала.

При виконанні умови 0<msg.payload<50 (is between), повідомлення передається на перший вихід, до якого в свою чергу приєднаний вузол Norma (тип function->change). Той задає текстове значення для властивості msg.payload рівним «НОРМА» і формує нову властивість msg.color рівною “GREEN”. Далі msg поступає вузол «button», який використовується для відображення тексту в прямокутнику. Значення тексту задається полі Label, а колір в полі Background. При формуванні динамічних значень для вузлів, використовується формат angular фільтрів, в якому вказується підстановка в подвійних фігурних дужках.

Аналогічна обробка проводиться при спрацюванні у вузлі switch умови msg.payload>50. Повідомлення згенерується на другому виході, який активує перерахунок вузла “Alarm” що буде формувати текст та колір для кнопки.

Детальніше про використання фільтрів ангуляр дивіться тут або тут

1.3.4 Робота з поштою (не обов’язкова частина)

1.3.4.1. Визначення налаштувань поштового серверу

Для роботи з поштою необхідно дізнатися налаштування почтового сервера, який Ви будете використовувати для відправки повідомлень, і на якому є Ваш обліковий запис. **Ми рекомендуємо для тестування створити новий аккаунт на одному з поштових серверів.**

Випишіть налаштування:

Для вихідних повідомлень (відправки пошти):

- SMTP Server
- port

Для вхідних повідомлень (отримання пошти):

- POP3 Server або IMAP Server
- port

Інформацію про налаштування поштових сервісів можна отримати в довідці по цим серверам. Нижче наведений приклад деяких із найбільш вживаних:

Приклади налаштувань:

Поштовий сервіс	Для відправки (Сервер вихідних повідомлень)	Для отримання (Сервер вхідних повідомлень)	Примітка
www.ukr.net	SMTP Server: smtp.ukr.net	IMAP Server: imap.ukr.net	В налаштуваннях треба увімкнути IMAP/SMTP (рис. 27)
port	port: 465 або 2525	port: 993	
www.gmail.com	SMTP Server: smtp.gmail.com	IMAP Server: imap.gmail.com Вимагає SSL: так	В налаштуваннях треба увімкнути IMAP (рис. 28) Активувати доступ до додатків https://myaccount.google.com/lesssecureapps (рис. 29)
port	port: 465(SSL) або 587(TLS)	port: 993	

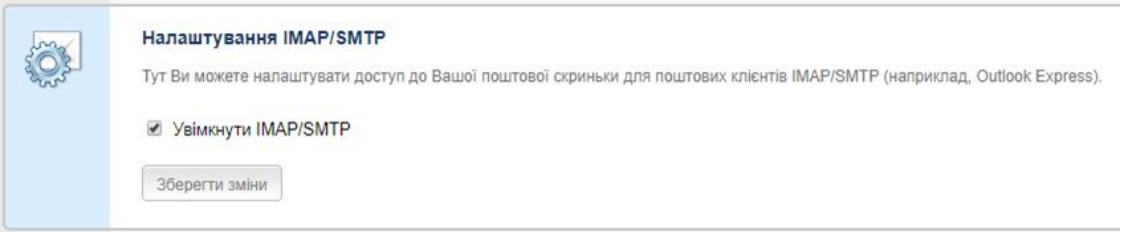


Рисунок 1.27 - Налаштування ukr.net для активації отримання та відправки пошти.

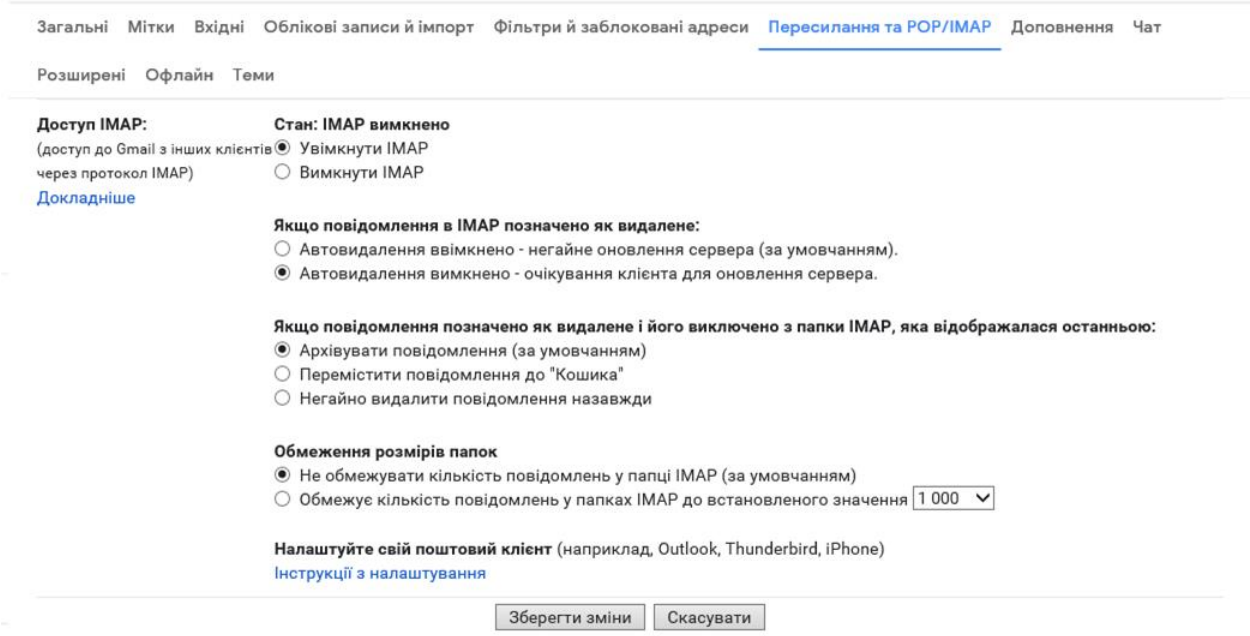


Рисунок 1.28 - Налаштування gmail.com для активації отримання та відправки пошти

Для сервісів Gmail навіть після активації дозволу на взаємодію з поштовими клієнтами через IMAP/SMTP, при використанні додатку NodeRED може прийти повідомлення про блокування доступу (рис. 1.29). Для активації доступу тимчасово (до закінчення практичної роботи) виставте дозвіл доступу до менш безпечних додатків (рис. 1.30) необхідно також дозволити https://myaccount.google.com/lesssecureapps

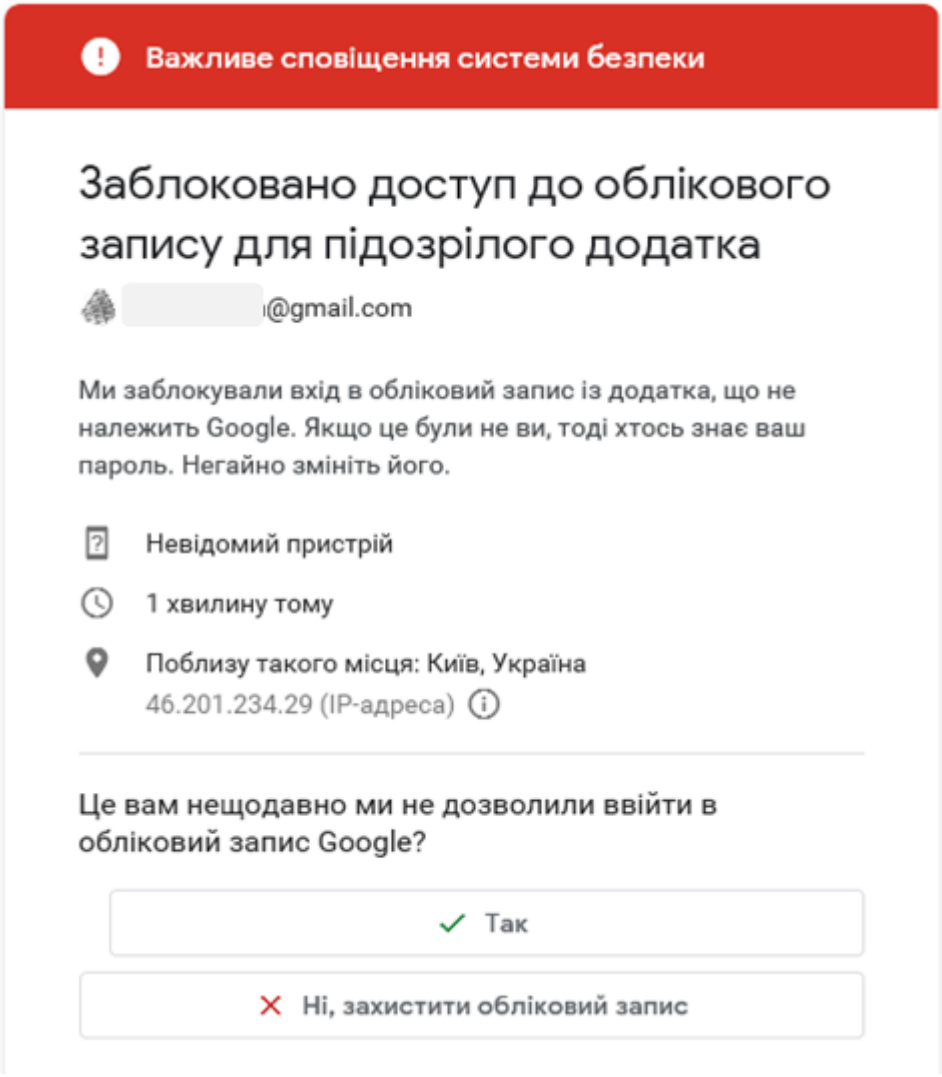


Рисунок 1.29 – Повідомлення gmail.com про блокування доступу

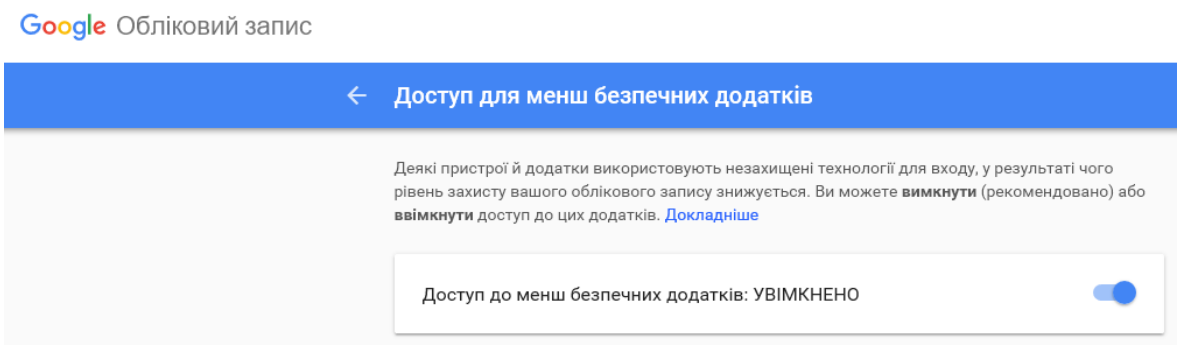


Рисунок 1.30 – Активація доступу до менш безпечних додатків (https://myaccount.google.com/lesssecureapps)

1.3.4.2 Модифікування програми для відправки поштових повідомлень

Модифікуйте програму фрагментом показаним на рис. 1.31. Налаштування вузів показано на рис. 1.32-1.33. У поля user-id та password необхідно вказати користувача та пароль для аккаунта. Інші налаштування беруться з визначених в попередньому пункті. В поле отримувача введіть власну поштову скриньку.

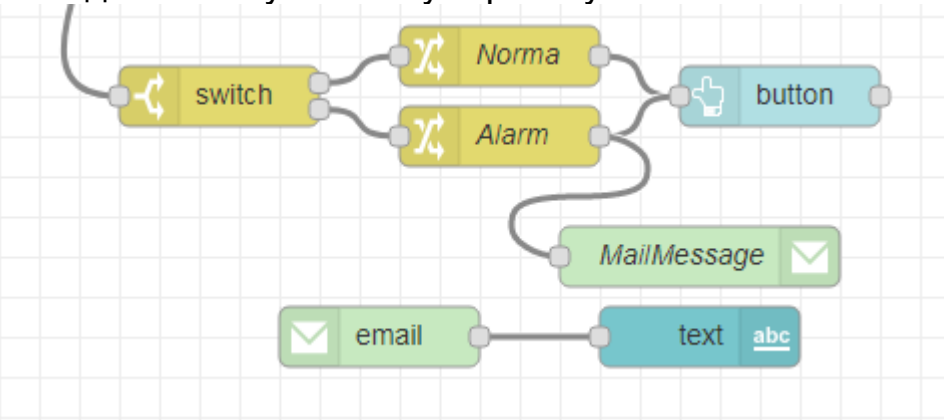


Рисунок 1.31 – Модифікований фрагмент програми

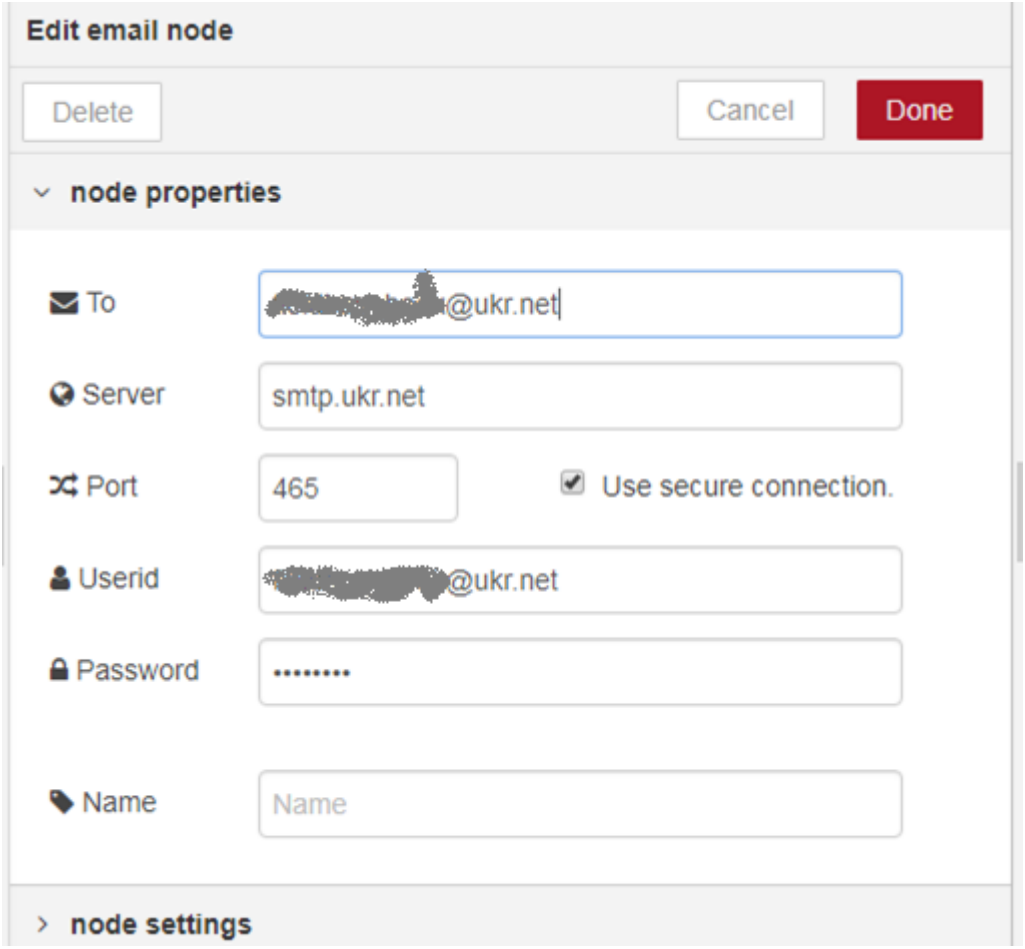


Рисунок 1.32 - Налаштування вузла MailMessage типу Social e-Mail

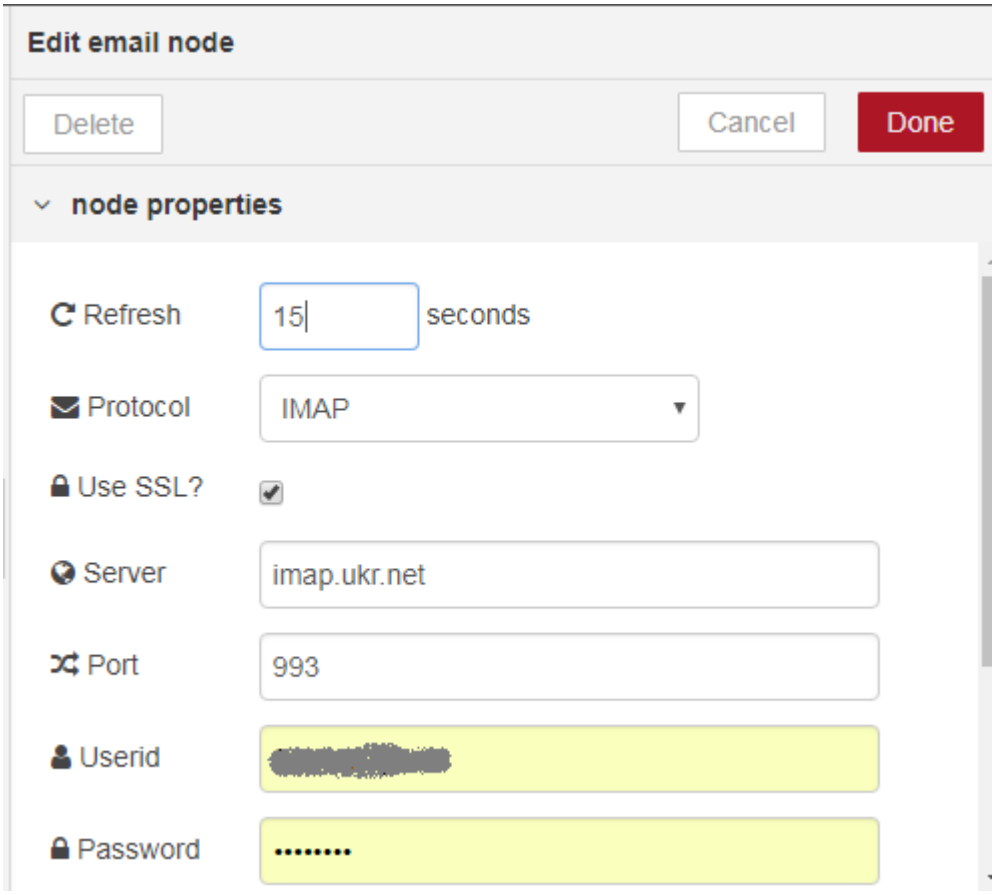


Рисунок 1.33 – Налаштування вузла MailMessage типу Social e-Mail-in

Зробіть розгортання проекту, встановіть значення заданої температури так, щоб згенерувалося повідомлення Alram. На пошту повинно прийти повідомлення, протягом 15 секунд воно повинно відобразитися на веб-сторінці в полі текст (рис. 1.34).

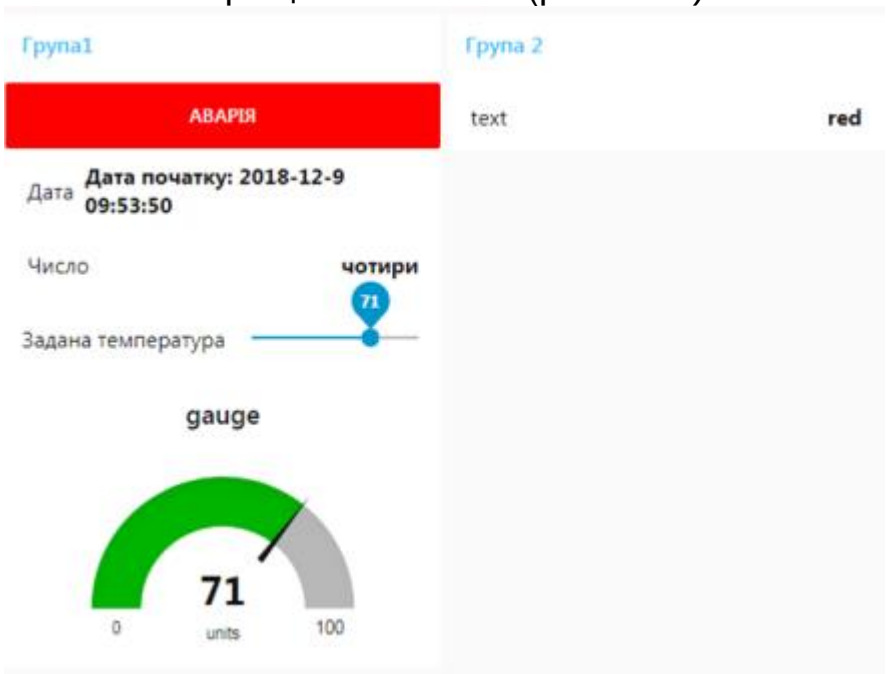


Рисунок 1.34 - Приклад зовнішнього вигляду UI.

1.3.5 Робота з Modbus

Node-RED, як правило використовується або на стороні Edge або в якості хмарного додатку. Якщо Node-RED використовується на стороні Edge в якості програми для концентратора або шлюзу чи маршрутизатору, наприклад на апаратній платформі Raspberry PI, він може збирати дані з різних пристроїв по протоколам промислових мереж. Найбільш поширеним і простим протоколом на сьогоднішній день є Modbus, тому в спільноті Node-RED розробили кілька бібліотек з ним. Таким чином, як варіант Raspberry PI буде взаємодіяти з пристроями по протоколу Modbus TCP/IP, а з іншого боку він буде взаємодіяти з хмарними додатками та сервісами (рис. 35)

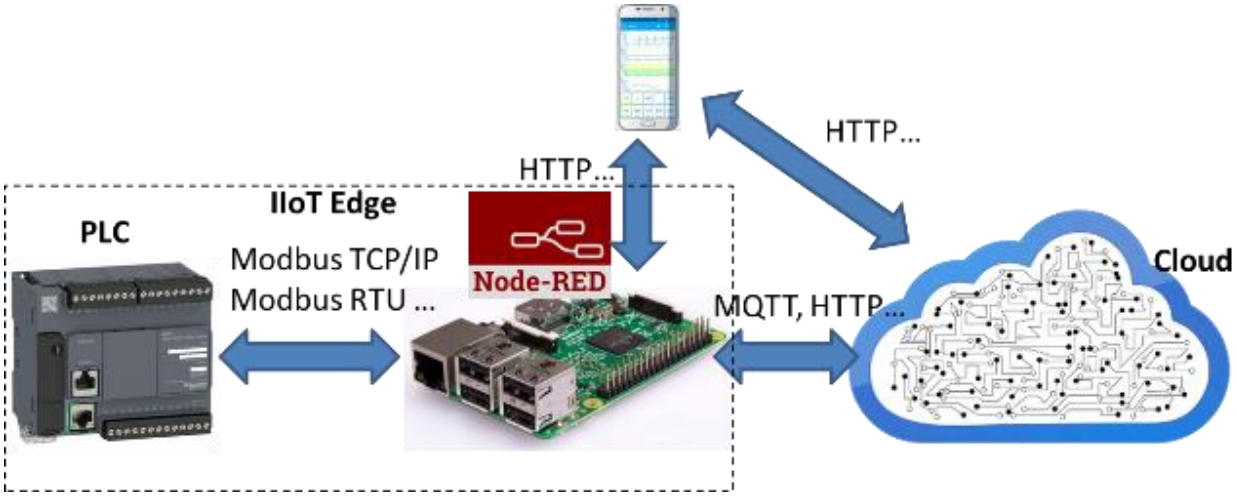


Рисунок 1.35 - Приклад структури рішення IIoT з використанням Raspberry PI та Node-RED.

Для тестування такого рішення можна на перших порах обійтися тільки програмними складовими. Замість ПЛК можна використати імітатор ПЛК, що підтримує Modbus TCP/IP, а замість Raspberry PI – віртуальну машину з ОС Raspbian з усім встановленим ПО, як в реальному залізі. Ще простіше – використовувати тільки Node-RED, який буде з'єднуватися з імітатором ПЛК або імітатором Modbus TCP/IP Server (рис. 36). У даній практичній роботі в якості імітатора PLC буде використаний пакет Mod_RSsim, який треба попередньо встановити. А в якості бібліотеки Modbus для Node-RED - node-red-contrib-modbustcp.

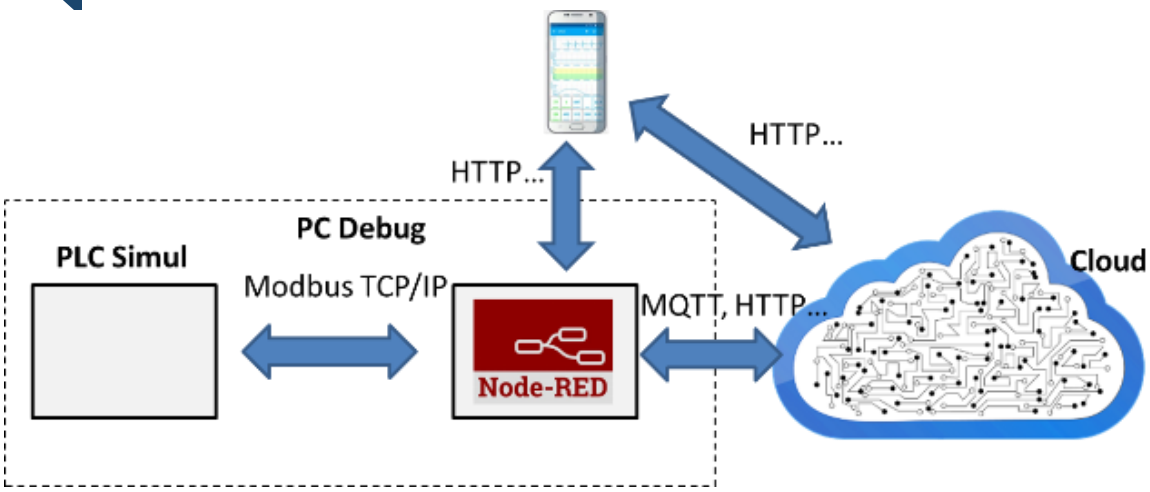


Рисунок 1.36 – Приклад структури рішення IIoT з використанням Raspberry PI та Node-RED.

1.3.5.1. Встановити пакет Modbus (node-red-contrib-modbustcp) використовуючи Manage Palette (рис. 37)

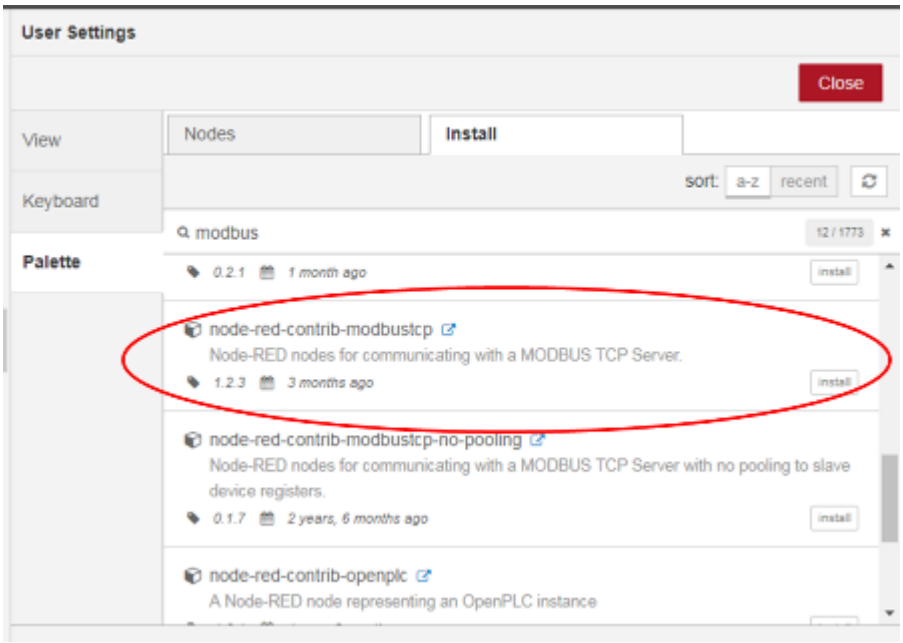


Рисунок 1.37 – Встановлення пакету node-red-contrib-modbustcp

1.3.5.2 Завантажити Modbus PLC Simulator (Mod_RSsim)

Завантажте Mod_RSsim2, перед завантаженням треба буде трохи почекати.

1.3.5.3 Запуск на виконання Modbus PLC Simulator

Запустіть на виконання Modbus PLC Simulator з того місця, куди Ви його завантажили

Виставьте значення в Prot: Modbus TCP

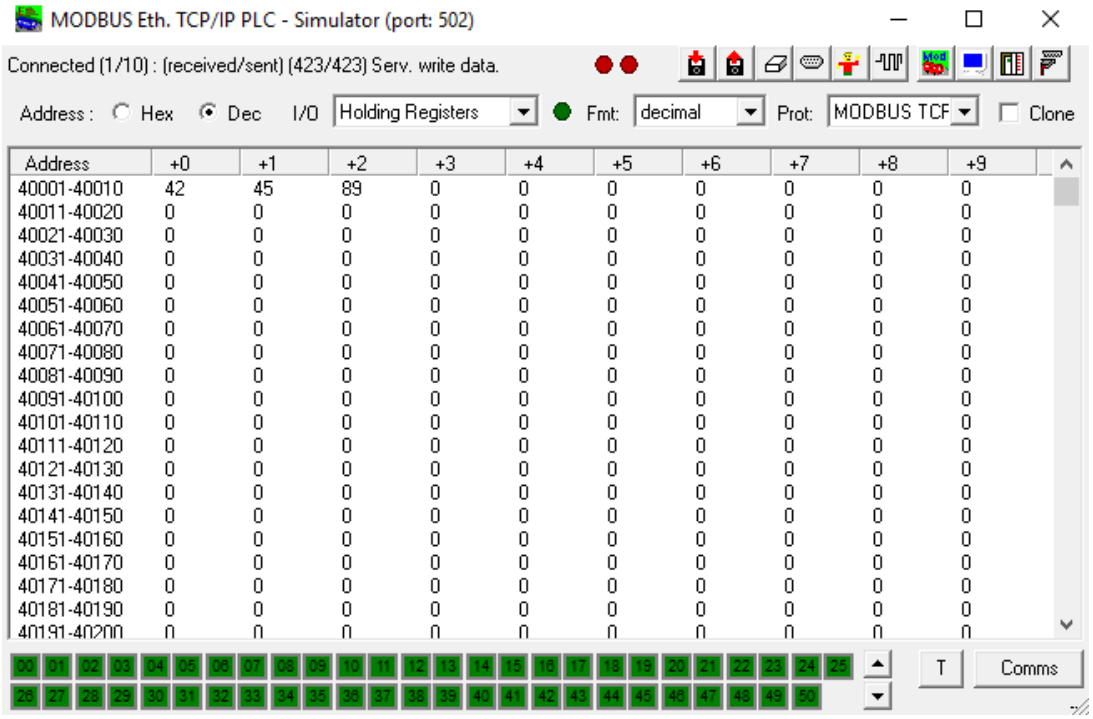


Рисунок 1.38 - Налаштування Mod_RSsim.

1.3.5.4 Ознайомлення з бібліотекою Node-RED

Ознайомтеся з правилами роботи з бібліотеки Modbus.

1.3.5.5 Вставка та перевірка елементу modbustcp-read

З розділу палітри Inputs вставте елемент modbustcp-read, зайдіть в налаштування. Праворуч поля Server натисніть кнопку з олівцем для створення нового серверу (рис. 1.39).

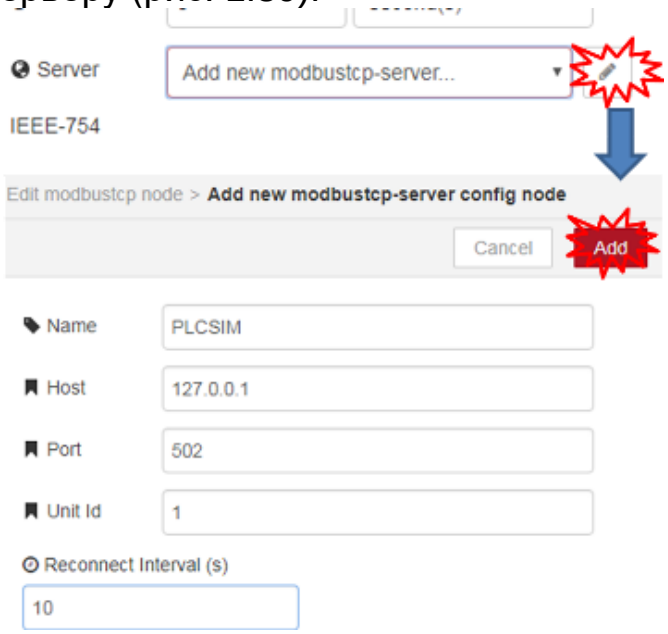


Рисунок 1.39 – Створення серверу Modbus TCP/IP.

Після створення серверу налаштуйте зчитування десяти Holding регістрів починаючи з 0-го, як це показано на рис. 40.

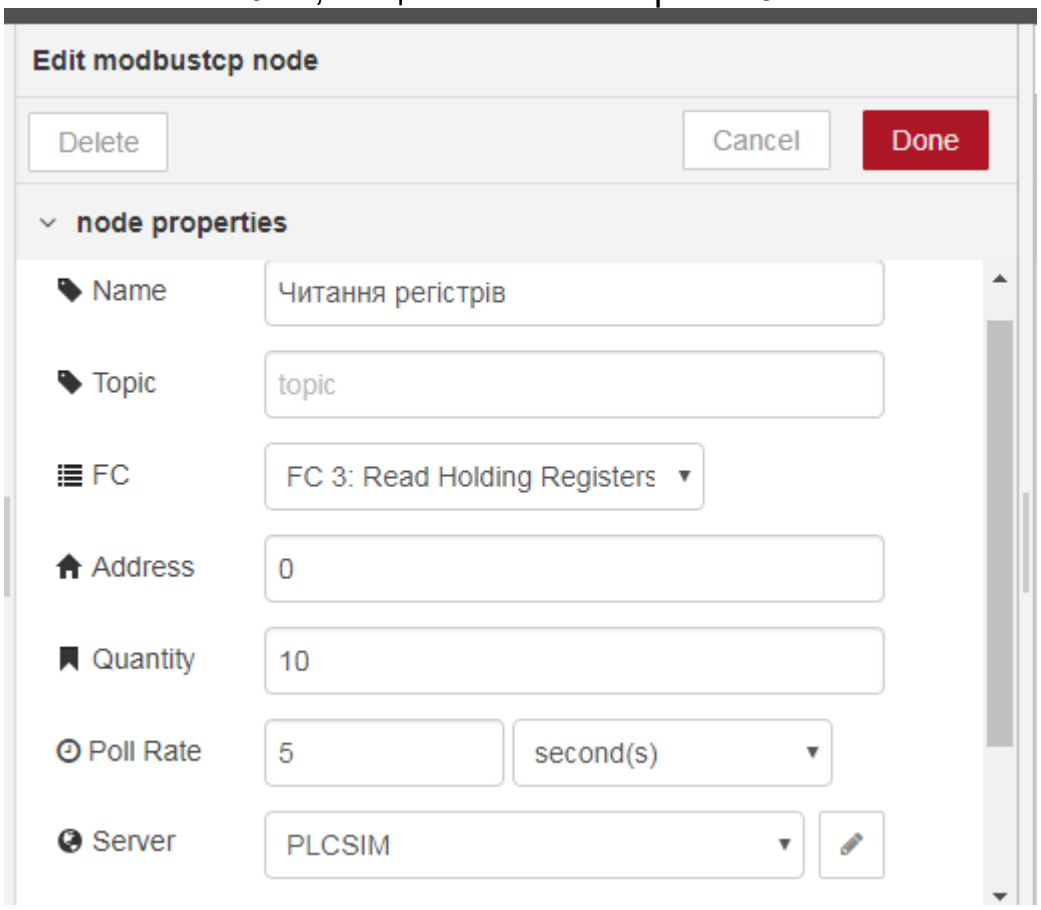


Рисунок 1.40 – Налаштування зчитування по Modbus TCP/IP.

Зробіть фрагмент програми, показаний на рис. 41. Зробіть розгортання проекту, деактивуйте усі виводи debug окрім останнього.

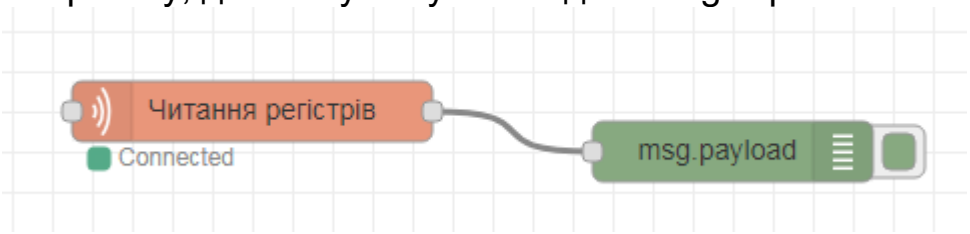


Рисунок 1.41 - Фрагмент програми зчитування.

Змініть значення перших десяти регістрів у програмі Mod_RSsim. Активуйте вікно виводу Debug, там повинні виводитися значення регістрів у вигляді масиву.

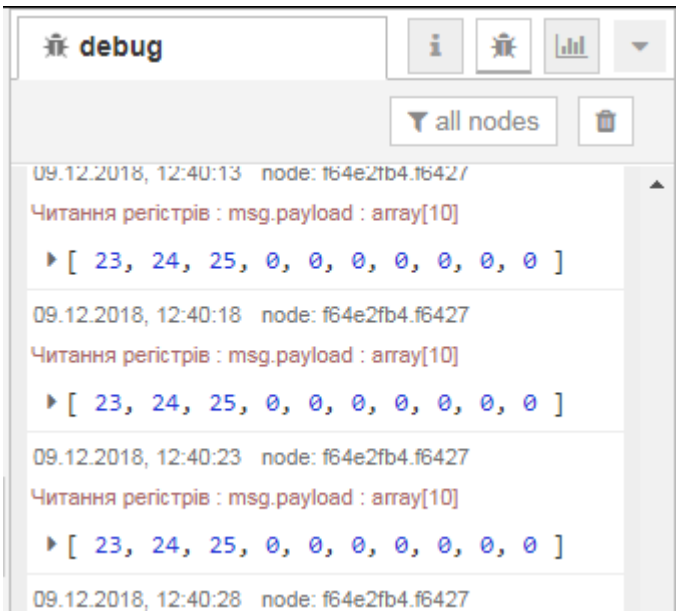


Рисунок 1.42 – Значення зчитуваних регістрів у вигляді масиву.

Зверніть увагу, що тепер msg.payload є масивом з десяти елементів (Array[10]), тому для роботи з цими значеннями, наприклад виводу на відображення на ВЕБ-сторінці необхідно їх попередньо обробити.

1.3.5.6 Модифікація програми

Модифікуйте програму відповідно до рис. 1.43 – 1.46. Зробіть розгортання та перевірте чи вірно відображаються значення.

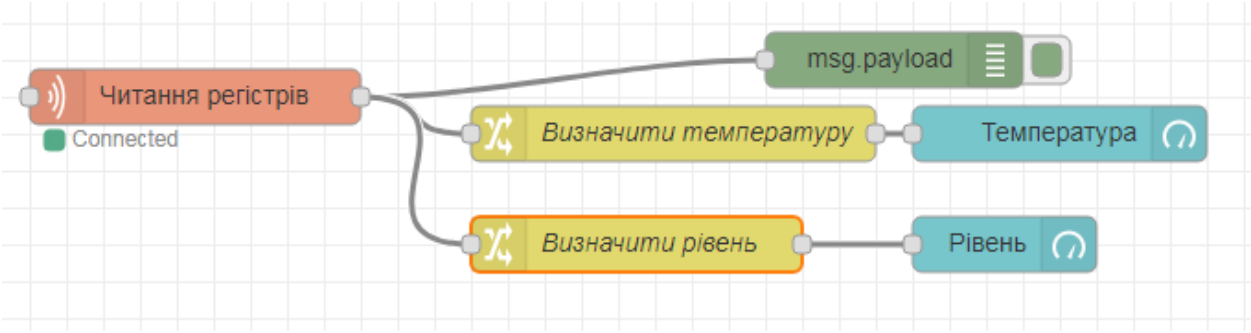


Рисунок 1.43 - Модифікована програма

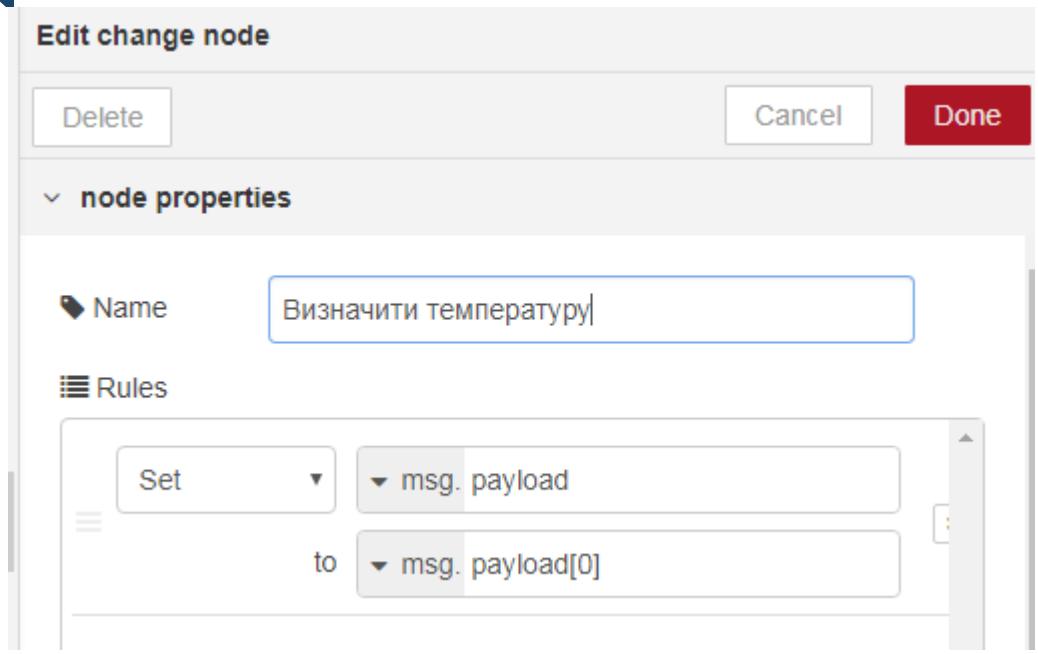


Рисунок 1.44 – Налаштування вузла «визначити температуру».

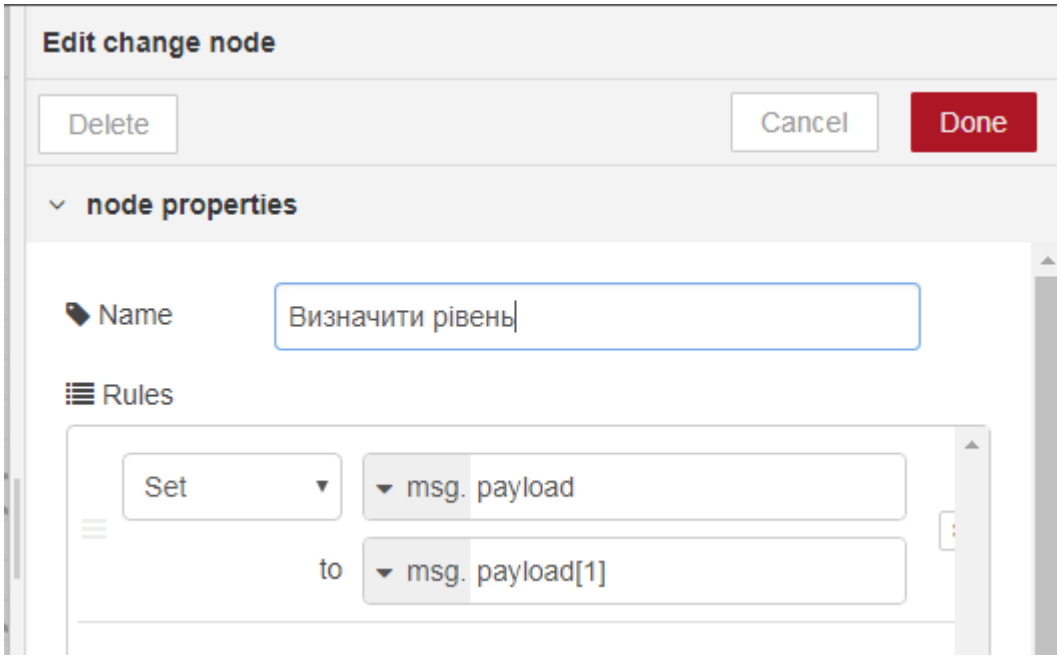


Рисунок 1.45 – Налаштування вузла «визначити рівень».

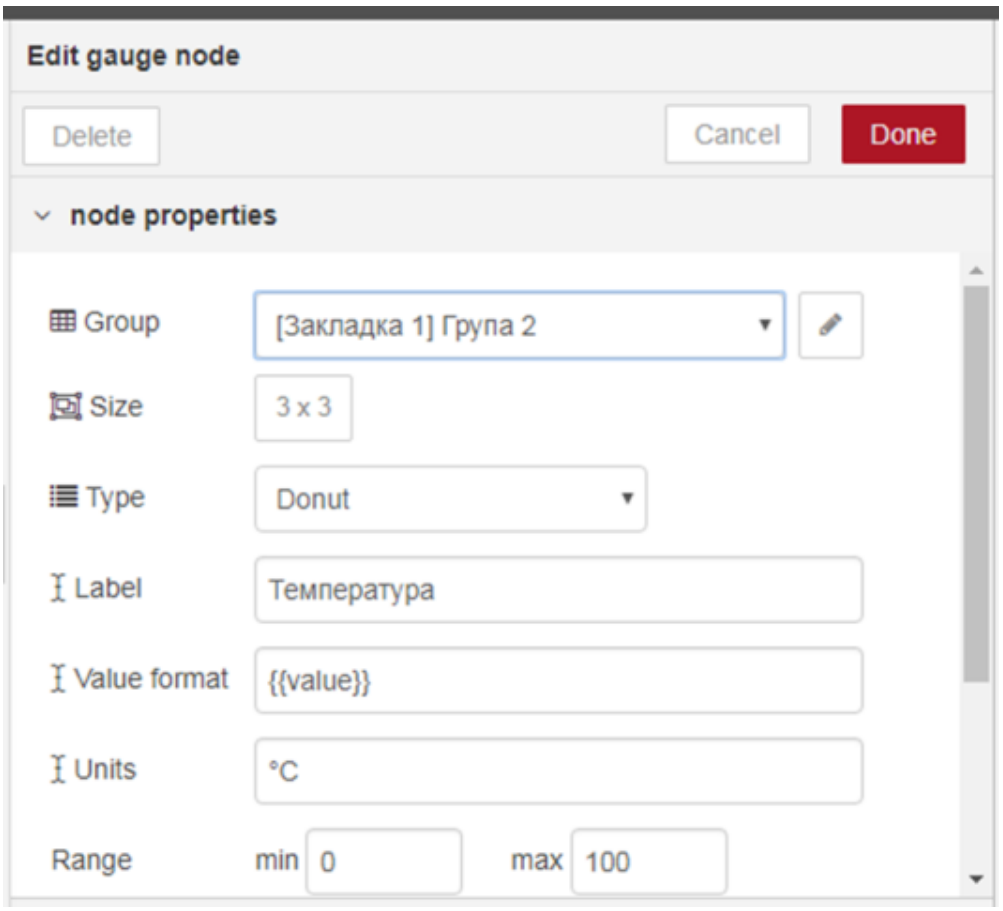


Рисунок 1.46 – Налаштування вузла «температура»

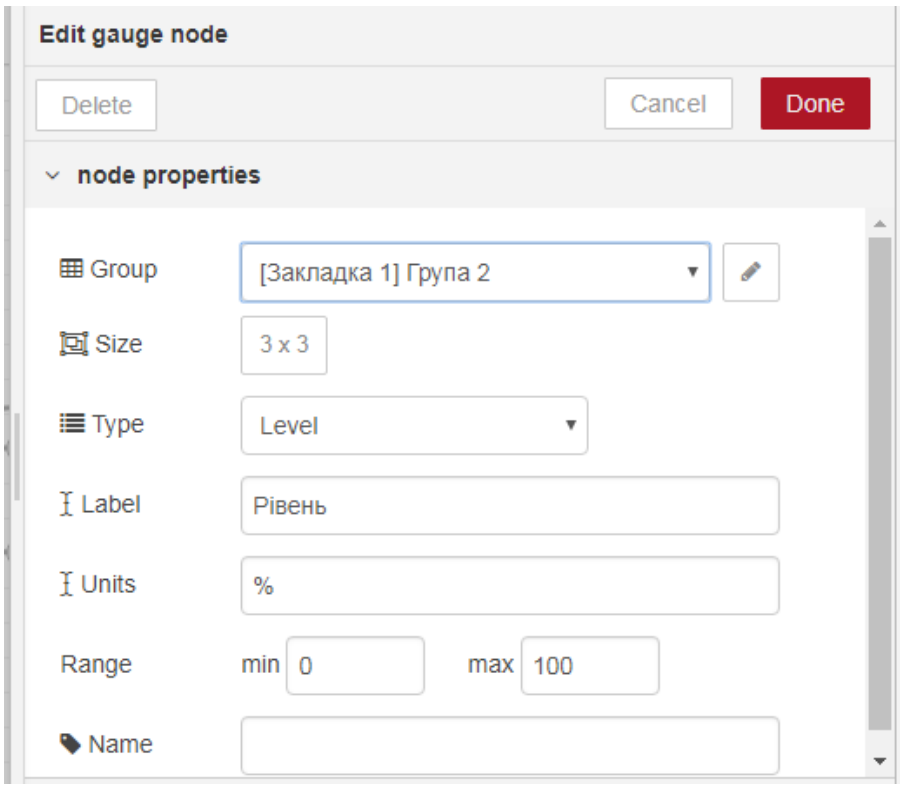


Рисунок 1.46-1 Налаштування вузла «рівень»

1.3.5.7 modbustcp-write

Для запису по Modbus використайте вузол modbustcp-write з розділу палітри outputs. Модифікуйте програму відповідно до рис. 1.47. Зробіть розгортання проекту і перевірте чи змінюється значення Holding регістру в Mod_RSsim при зміні його через елемент «Задана температура»

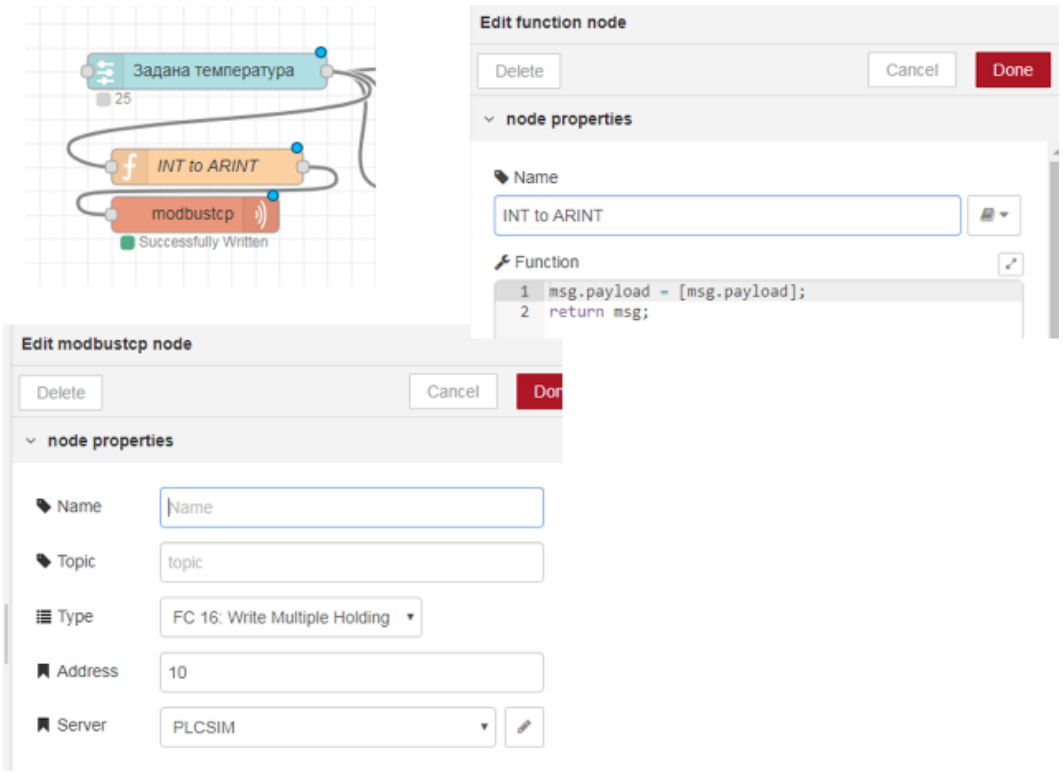


Рисунок 1.47 - Фрагмент програми та налаштування запису по Modbus

1.3.6 Робота з JS об’єктами та обробка системної інформації

1.3.6.1 Встановлення модуля node-red-contrib-os

Встановіть в Node-RED модуль node-red-contrib-os

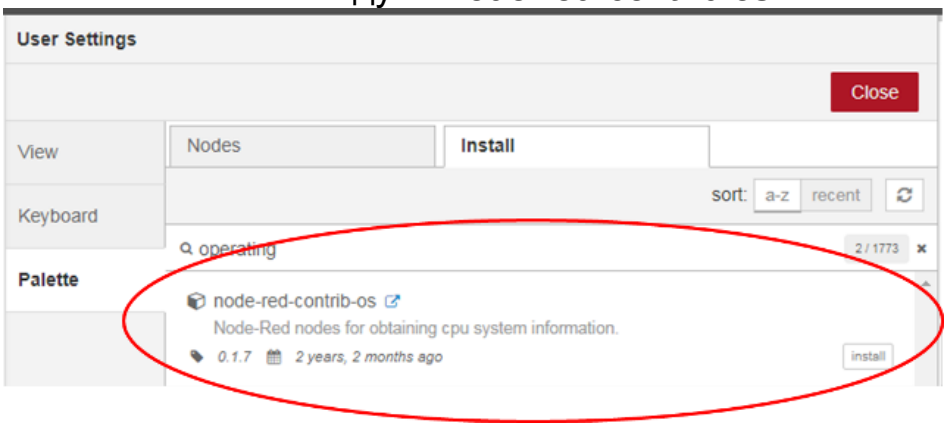


Рисунок 1.48 – Встановлення модуля node-red-contrib-os.

1.3.6.2 Використання вузлу Networkintf

З нововстановленого модуля використайте вузол типу Networkintf для створення фрагменту програми, як на рис. 1.49.

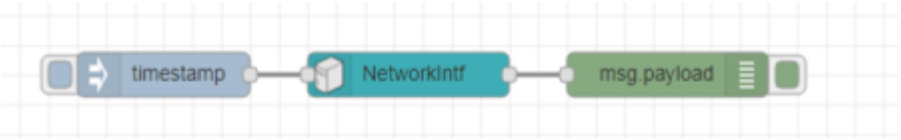


Рисунок 1.49 – Фрагмент програми для виведення інформації по наявним мережним картам

Зробіть розгортання програми, ініціюйте формування повідомлення, проаналізуйте вивід. Приклад виведеної інформації показаний на рис. 1.50. Як видно, інформація надається у вигляді JS об’єкту, який включає в себе об’єкт NetworkInterfaces, що в свою чергу вміщує кілька мережних інтерфейсів, що є масивами об’єктів, що представляють певний протокол..

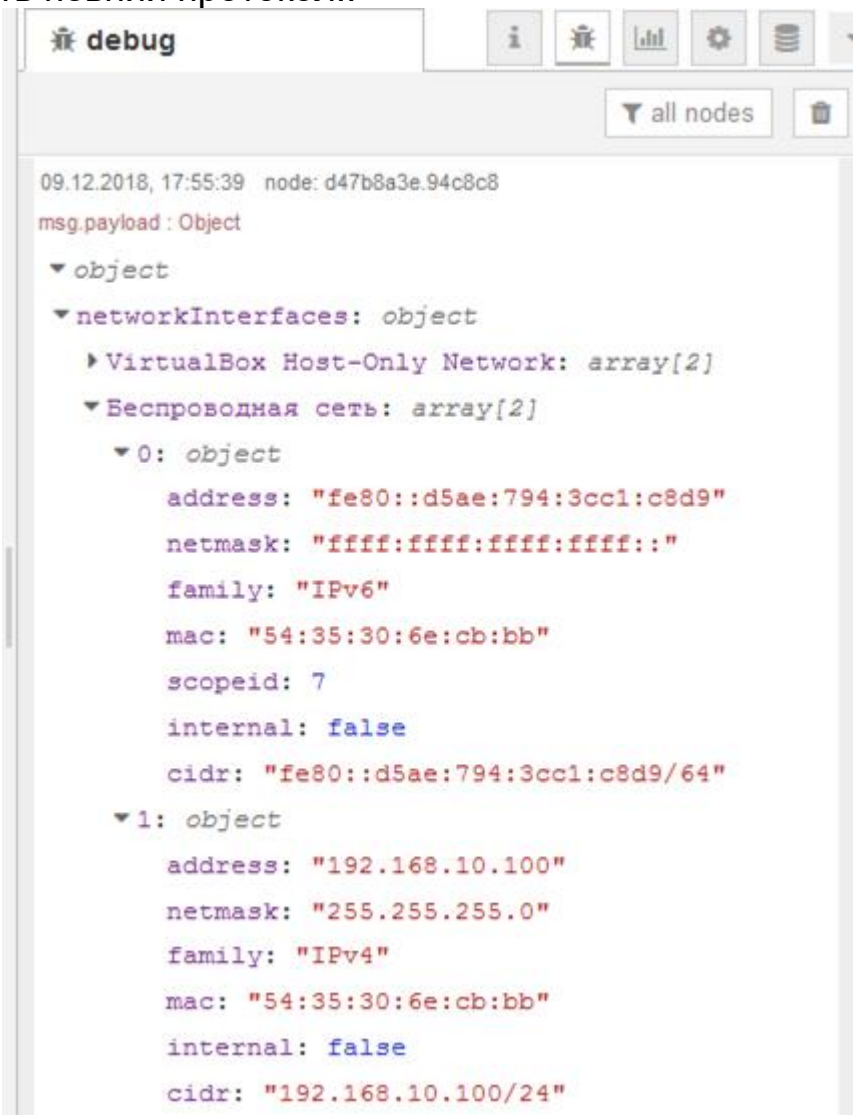


Рисунок 1.50 – Виведення інформації про мережу

Про об’єкти в JavaScript можна почитати тут або тут. Для перебору усіх властивостей об’єкту можна скористатися конструкцією for..in

1.3.6.3 Виведення інформації про мережні карти

Створіть програму, що буде виводити перелік MAC адрес для мережних карт, що встановлені на Вашому ПК (рис. 1.51 – 1.52).

Рисунок 1.51 – Виведення інформації про мережні карти

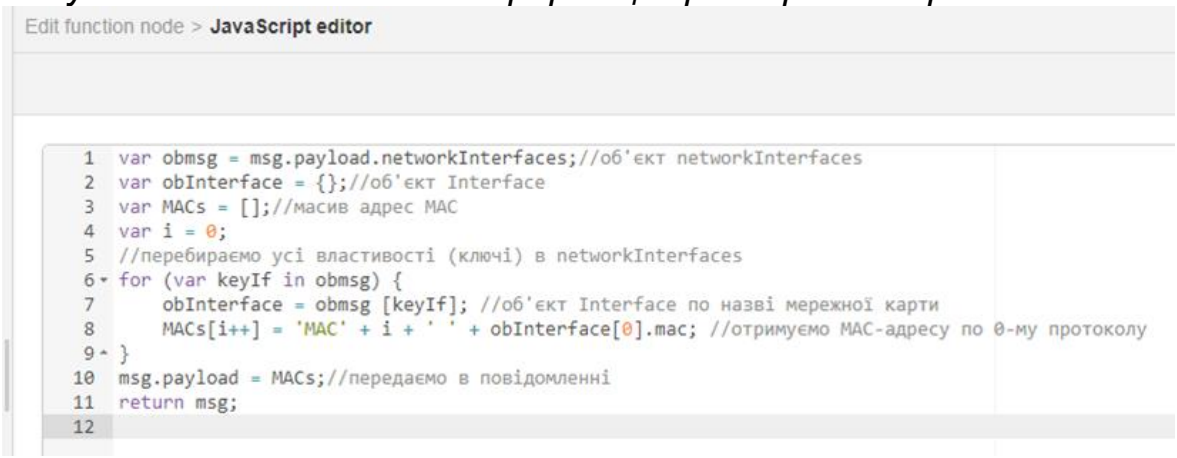


Рисунок 1.52 – Програма для вузла MACs.

Про JSON читайте тут

У JavaScript та IoT часто використовуються дані типу JSON. Детально про JSON та їх обробку Ви можете почитати тут

1.3.7 Звіти та експорт потоку

1.3.7.1 Копії екранів

Зробіть копії екранів програми та візуалізації для звіту до Вашої роботи.

1.3.7.2 Експорт потоку

Відкрийте потік (Flow) створений в практичній роботі. Виділіть всі вузли шляхом комбінації Ctrl+A. Усі вузли повинні виділитися червоним контуром. Через меню бокової панелі зробіть експорт виділеного в буфер обміну (рис. 53).

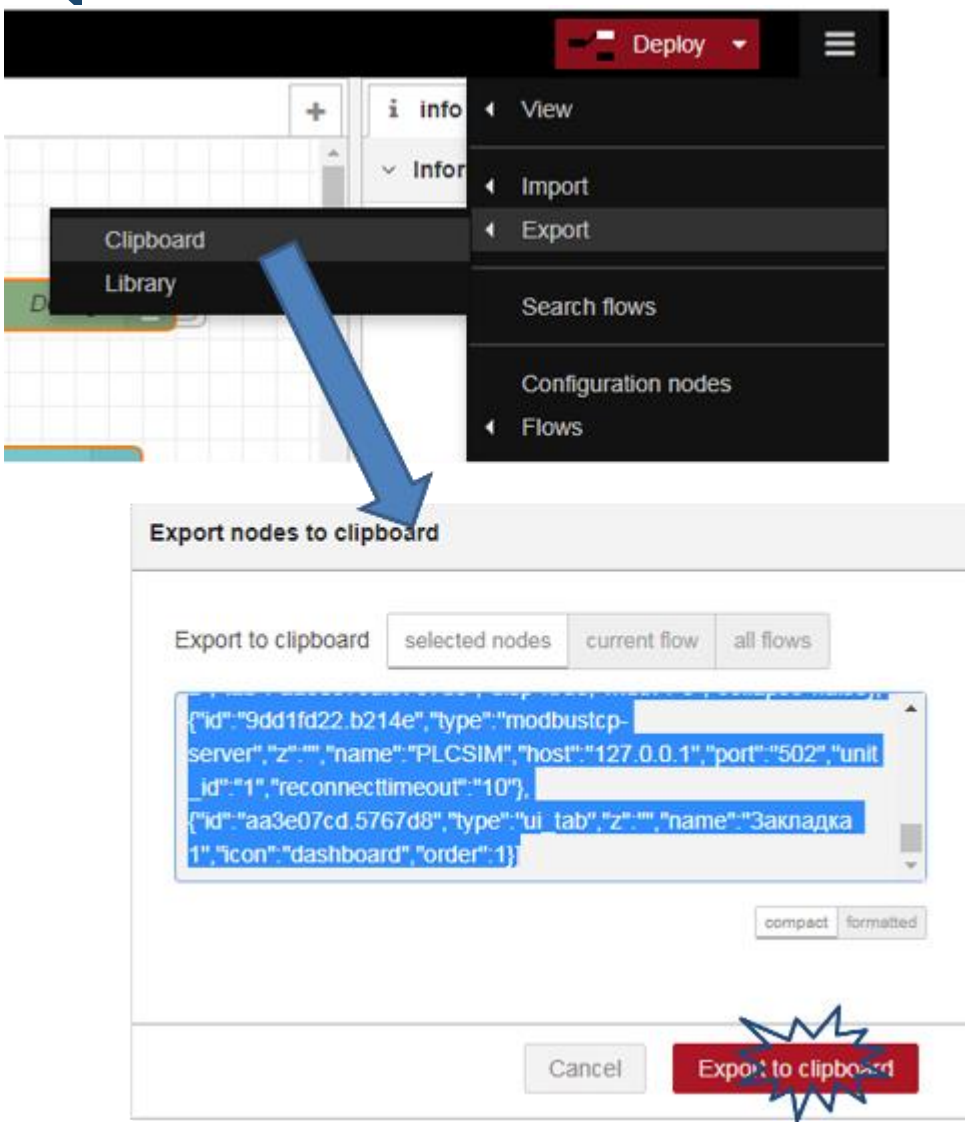


Рисунок 1.53 – Експорт потоку Node-RED в буфер обміну.

Створіть текстовий файл з назвою Lab1.txt і скопіюйте з буферу обміну текст експорту. Збережіть файл, він вам знадобиться для звіту та для захисту практичних робіт у майбутньому.

1.4 Питання для самоперевірки

1. Що таке Node-RED і для чого він використовується?
2. Як встановити Node-RED на платформу Windows?
3. Що таке потоки (flows) в Node-RED і як їх створювати?
4. Які типи вузлів (nodes) доступні в Node-RED і для чого вони використовуються?
5. Як використовувати вузол change в Node-RED?
6. Для чого використовується вузол delay і як його налаштувати?
7. Як створити та використовувати вузол function?
8. Як встановити та налаштувати модуль node-red-dashboard?

9. Як створювати веб-інтерфейси користувача за допомогою node-red-dashboard?

10. Як здійснюється обробка повідомлень у різних вузлах Node-RED?

1.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40*. URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).

2. Node-RED українською : довідник з Node-RED українською мовою / за ред. О. Пупени. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).

3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ : НУХТ, 2021. 170 с.

4. Лабораторна робота №1. Основи роботи з Node-RED. *TI40*. URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab1NodeRED.md (дата звернення: 09.07.2024).

5. Основи Node-RED. *TI40*. URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B5%D0%BA%D1%86/2_nodered.md (дата звернення: 09.07.2024).

6. Node-RED : Low-code programming for event-driven applications. *Node-RED*. URL: https://nodered.org/ (дата звернення: 09.07.2024).

7. Node-red-dashboard 3.6.5. *Node-RED*. URL: https://flows.nodered.org/node/node-red-dashboard (дата звернення: 09.07.2024).

8. Documentation. *Node-RED*. URL: https://nodered.org/docs/ (дата звернення: 09.07.2024).

ПРАКТИЧНА РОБОТА №2. ПРОТОКОЛИ IOT: MQTT

2.1 Мета

Мета практичної роботи полягає в ознайомленні з протоколом MQTT, вивченні принципів роботи MQTT-брокерів та клієнтів, а також освоєнні інструментів для моніторингу та тестування MQTT-з'єднань, таких як MQTT Explorer і HiveMQ WebSocket.

В результаті студенти повинні навчитися налаштовувати та використовувати MQTT-клієнти для публікації і підписки на повідомлення, а також інтегрувати Node-RED для обміну даними через MQTT.

Практичне застосування протоколу MQTT і інструменти для його тестування:

- MQTT дозволяє надійно передавати дані між пристроями з низькою затримкою, що є критичним для автоматизації;
- інструменти, такі як MQTT Explorer, дозволяють легко відслідковувати і налагоджувати процеси обміну повідомленнями;
- використання Node-RED для інтеграції різних систем та пристроїв через MQTT забезпечує гнучкість і масштабованість автоматизованих рішень.

2.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:

- використання тестових клієнтів та брокерів для зв’язку по MQTT;
- зв’язок Node-RED з іншими пристроями по MQTT;
- зв’язок MQTT-клієнта з мобільного телефону.

2.3 Хід роботи

2.3.1 Використання тестових клієнтів та брокерів для зв’язку по MQTT

Одна із областей застосування MQTT – це обмін між пристроями та програмами, що підключені до Інтернет. У даному практичному занятті використовується загальнодоступні брокери, наприклад test.mosquitto.org або mqtt.eclipse.org. Слід звернути увагу, що їх використання є безкоштовним, але вони не гарантують безперебійну роботу сервісу, тому їх не слід використовувати для реальних рішень, що потребують надійних з’єднань та цілодобового використання. За необхідності використання надійних сервісів, слід користуватися іншими брокерами власними, або хмарними.

Також в роботі використовуються тестові клієнти:

http://www.hivemq.com/demos/websocket-client

http://mqtt-explorer.com/

2.3.1.1 Завантаження, встановлення та запуск MQTT Explorer

Завантажте та встановіть MQTT Explorer
Запустіть на виконання MQTT Explorer.
Виберіть наперед-сконфігуроване з'єднання mqtt.eclipse.org, подивіться на налаштування і натисніть Connect.
Якщо з'єднання не працює, перевірте аналогічно наперед-сконфігуроване з'єднання test.mosquitto.org .

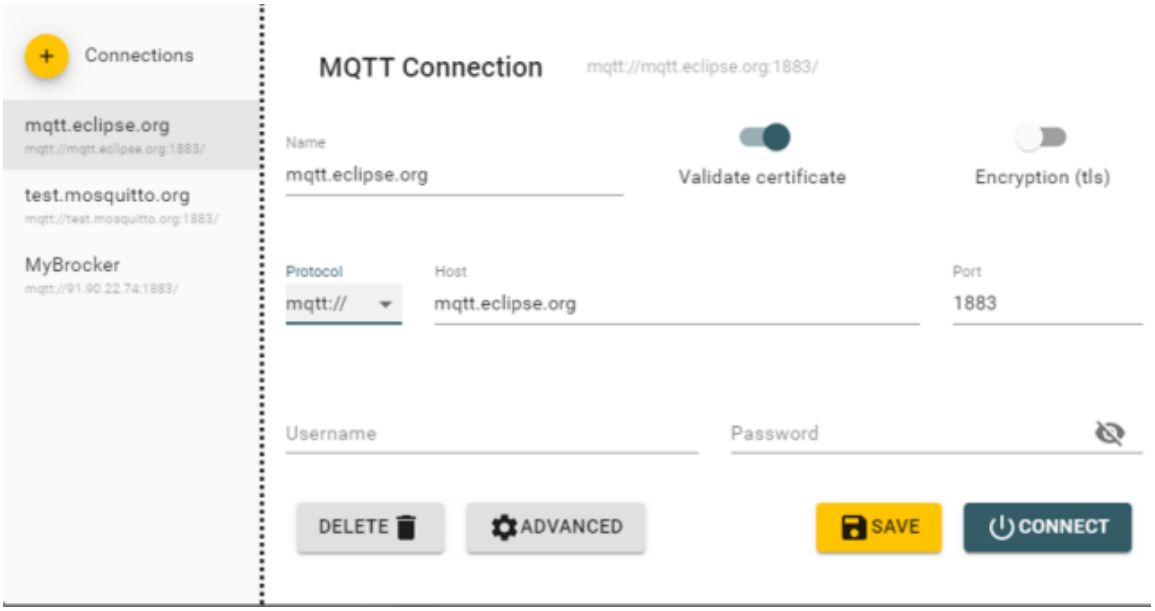


Рисунок 2.1. – Налаштування MQTT Explorer

Після з'єднання Ви побачите усі теми, які публікуються на брокері.
Введіть фільтр $SYS для відображення тільки системних повідомлень.

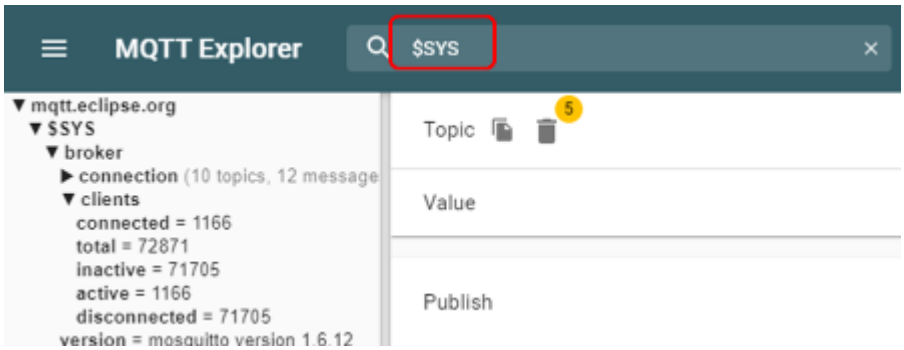


Рисунок 2.2 – Фільтрування системних повідомлень на брокері

Зробіть огляд гілок та значень в $SYS
Знайдіть і виберіть тему clients/connected, який показує кількість підключених клієнтів. У деталізації History Ви побачите перелік усіх

повідомлень, які були отримані з початку сеансу а також їх значення у вигляді графіку.

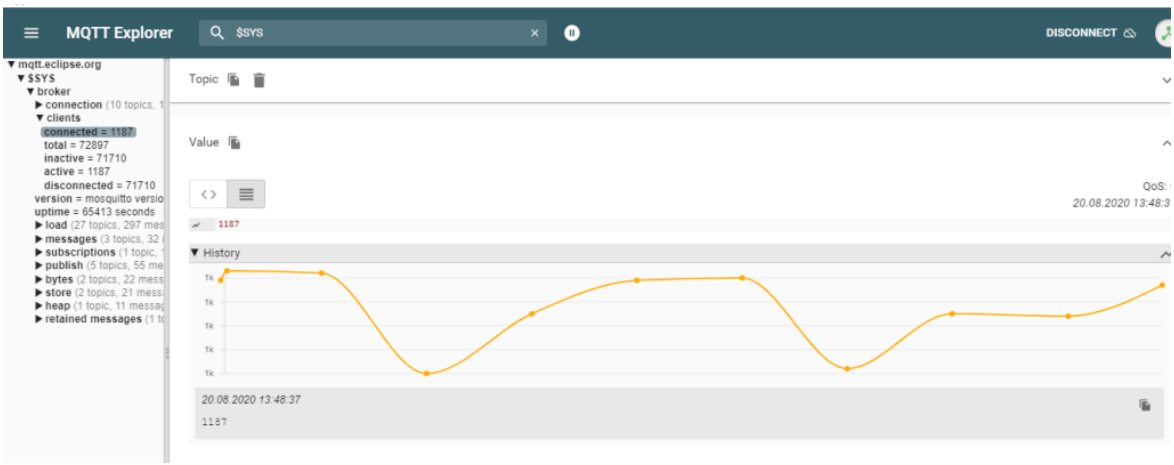


Рисунок 2.3 – Деталізація переліку повідомлень

Натисніть Disconnect. Зайдіть в налаштування Advanced. Подивіться налаштування: у списку тем вказано фільтр підписки на усі теми. На кожну тему підписка вказана з QoS=0

Перевірте підключення до test.mosquitto.org

2.3.1.2 Робота з HiveMQ Вебсокет-клієнтом

У браузері зайдіть на http://www.hivemq.com/demos/websocket-client
На сторінці Вебсокет-клієнта в полі Host введіть:

- **mqtt.eclipse.org**
- в полі Port **80** (over WebSocket),

Якщо **mqtt.eclipse.org** не працює, на сторінці Вебсокет-клієнта в полі Host введіть:

- **test.mosquitto.org**
- в полі Port **8081** (over WebSocket),
- залиште виставленою опцію SSL

після чого натисніть кнопку Connect. Повинен з'явитися напис Connected.

Натисніть Add New Topic Subscription і в полі Topic задайте

$SYS/broker/clients/connected

Тепер у полі Messages виводимуться повідомлення з даної теми.

У випадку відсутності зв'язку з брокером зробіть перевірку на test.mosquitto.org.

2.1.3 Публікація і підписка для власного повідомлення

На онлайновому клієнті HiveMQ створіть нову тему для публікації:
myname/device1/val

де myname - це якесь придумане ім'я, яке має бути унікальне в адресному просторі брокера.

Задайте:

- QoS=1, виставіть опцію Retain;
- в поле Message пишіть якесь числове значення;
- зробіть публікацію;
- відкрийте MQTT Explorer, в Advanced налаштуйте фільтр на ваші публікації, які задаються полем myname ;
- знайдіть це повідомлення і передивіться його значення;
- в HiveMQ ще кілька раз введіть різні значення і зробіть публікацію.

2.3.1.4 Відкриття сторінки з варіантом на тестовому сервері

Перейдіть на http://edu.asu.in.ua:1880/ui/#/0 (надалі, **тестовий сервер**) виберіть вкладку і групу елементів **з вашим варіантом**, маєте побачити (рис. 2.4):

- повзунок для керування клапаном;
- тренди температури, позиції клапану, та секундної пилоподібної кривої (0-100);
- круговий індикатор температури.

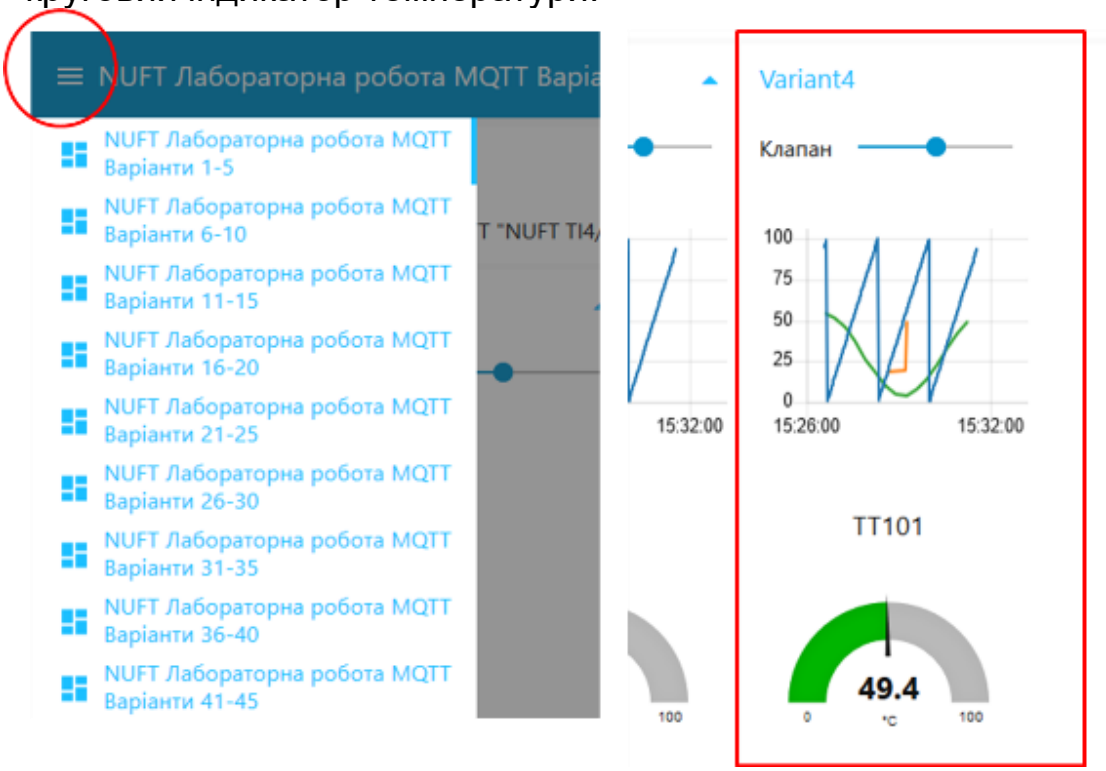


Рисунок 2.4 – Вигляд сторінки з варіантом на тестовому сервері

2.3.1.5 Перевірка підключення до тестового варіанту

У MQTT Explorer відключіться від брокера.
Виберіть брокер test.mosquitto.org .
Зайдіть в налаштування Advanced, де:

- видаліть усі теми
- добавте нову тему для підписування NUFT TI4/#
- вкажіть QoS=0
- натисніть ADD
- натисніть Back

Натисніть Connect для підключення до брокера.
На тестовому сервері (http://edu.asu.in.ua:1880/ui/#/0) змініть якісь значення повзунків.
У MQTT Explorer мають з'явитися відповідні записи.

2.3.1.6 Зміна даних на тестовому сервері через MQTT

У MQTT Explorer на панелі Publish в полі Topic впишіть “NUFT TI4/Variant**X**/TT101”, де **X** – номер вашого варіанту.
QoS задайте рівним 0.
Виберіть тип повідомлення RAW.
У полі Message введіть значення від 10.5, натисніть Publish.
Перейдіть на тестовий сервер, подивіться як змінюється значення на круговому індикаторі.
Зробіть поступове введення 30, 75, 50, з періодичністю 5 секунд, після кожного натискайте Publish. Подивіться як змінюється значення на тренді.

2.3.2 Зв’язок Node-RED з іншими пристроями по MQTT

2.3.2.1 Налаштування отримання даних по MQTT

Відкрийте Node-RED.
Створіть нову вкладку з іменем MQTT та деактивуйте існуючі.
З розділу палітри Network вставте вузол MQTT In. В налаштуваннях Server добавте новий брокер MQTT з назвою mosquitto і адресою серверу http://test.mosquitto.org

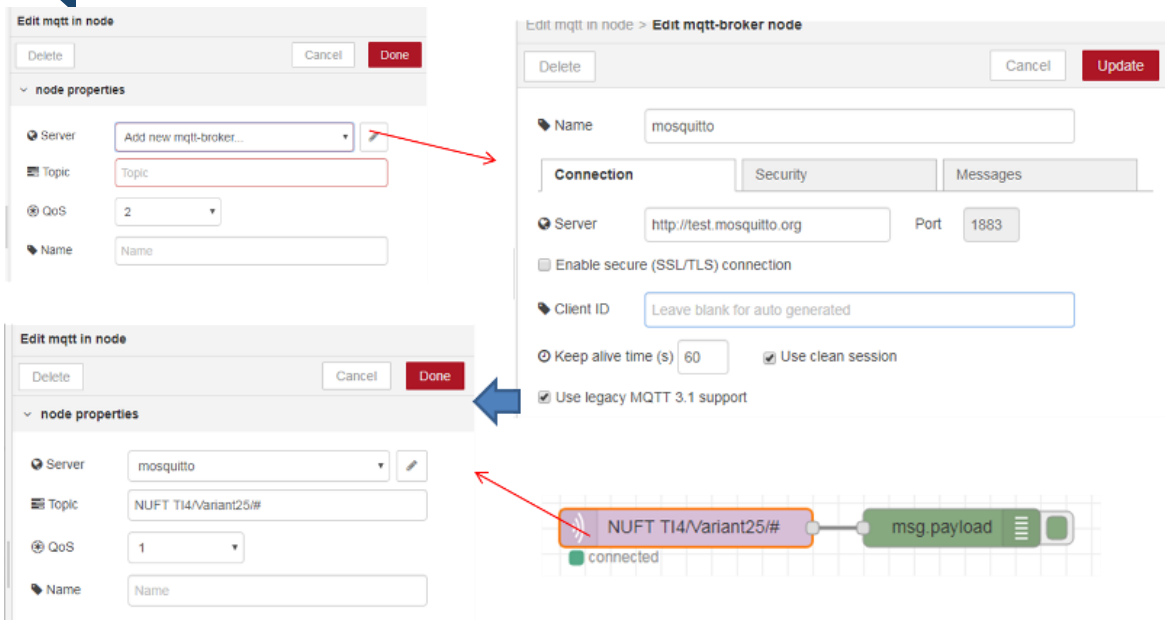


Рисунок 2.5 - Налаштування брокеру в MQTT

У полі Topic вузла MQTT in введіть NUFT TI4/VariantX/# де **X** – номер вибраного варіанту. Це значить, що цей вузол підписується на всі теми з даної гілки.

Використайте вузол Debug для виведенню повідомлень. Зробіть розгортання, дочекайтеся коли вузол «MQTT in» покаже статус «Connected».

У випадку відсутності зв’язку з брокером зробіть спробу пізніше.

2.3.2.2 Тестування отримання даних по MQTT

Активуйте на боковій панелі режим відображення повідомлень відлагодження. Змініть значення на тестовому сервері для клапану зі свого варіанту.

Використовуючи MQTT Explorer задайте значення температури. Зробіть аналіз виведених повідомлень на бічній панелі.

2.3.2.3 Тестування відправки даних по MQTT

Використовуючи вузли «Slider» з Dashboard та «MQTT out» самостійно реалізуйте зв'язок локального графічного інтерфейсу з віртуальним приладом на тестовому сервері, що показує TT101 для вашого варіанту.

Використовуючи вузли «gauge» і «MQTT in» самостійно реалізуйте зв'язок локального графічного інтерфейсу з повзунком завдання ступені відкриття клапану на тестовому сервері для вашого варіанту.

Програма та зовнішній інтерфейс матиме вигляд приблизно як на рис. 2.6. Для відображення підписів використовуйте теми а для формату відображення чисел ангулярні фільтри.

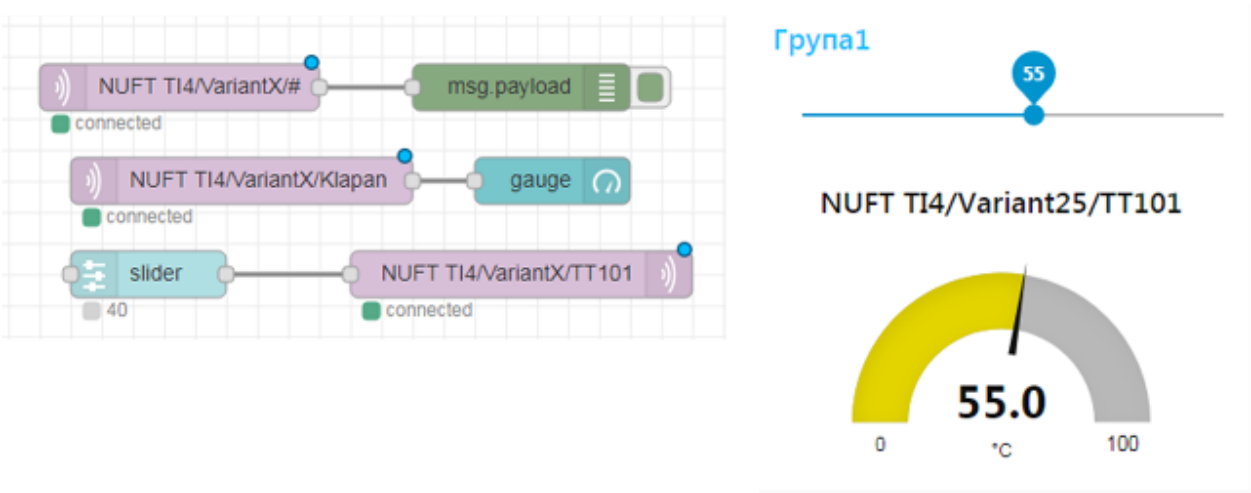


Рисунок 2.6 – Фрагмент програми в Node-RED для роботи з MQTT

2.3.2.4 Генерування синусоїди

Модифікуйте програму так, як це показано на рисунку нижче. Перейдіть на тестовий сервер, подивіться результат.

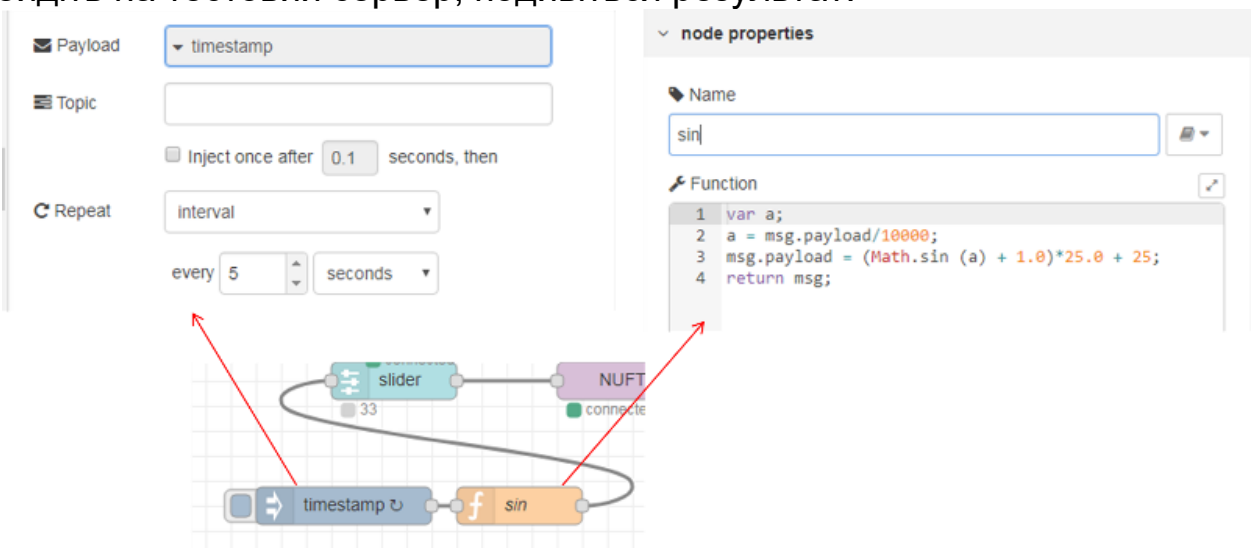


Рисунок 2.7 – Фрагмент програми для генерування синусоїди

Зробіть копії екранів з графіками синусоїди

2.3.2.5 Реалізація "коротко-замкнутого" з'єднання видавця і абонента в Node-RED

Для тестування можливостей MQTT в Node-RED рекомендується зав'язати видавця з абонентом у тому самому потоці але з різними підключеннями.

У Node-RED створіть новий потік (вкладку).

Реалізуйте там наступний фрагмент програми, при умові що:

– в налаштуваннях сервера вибираєте існуюче підключення і змінюєте його налаштування Messages.

– слово pupenasan змінюєте на власну унікальну назву типу myname , який був вибраний Вам в п.1.3.

– Для MQTT-in та MQTT-out пропишіть різні підключення Client1 та Client2.

– Обов'язково вкажіть унікальні ідентифікатори для клієнтів.

– Періодичність оновлення верхнього потоку (властивість Inject) задайте рівним 10 секунд.

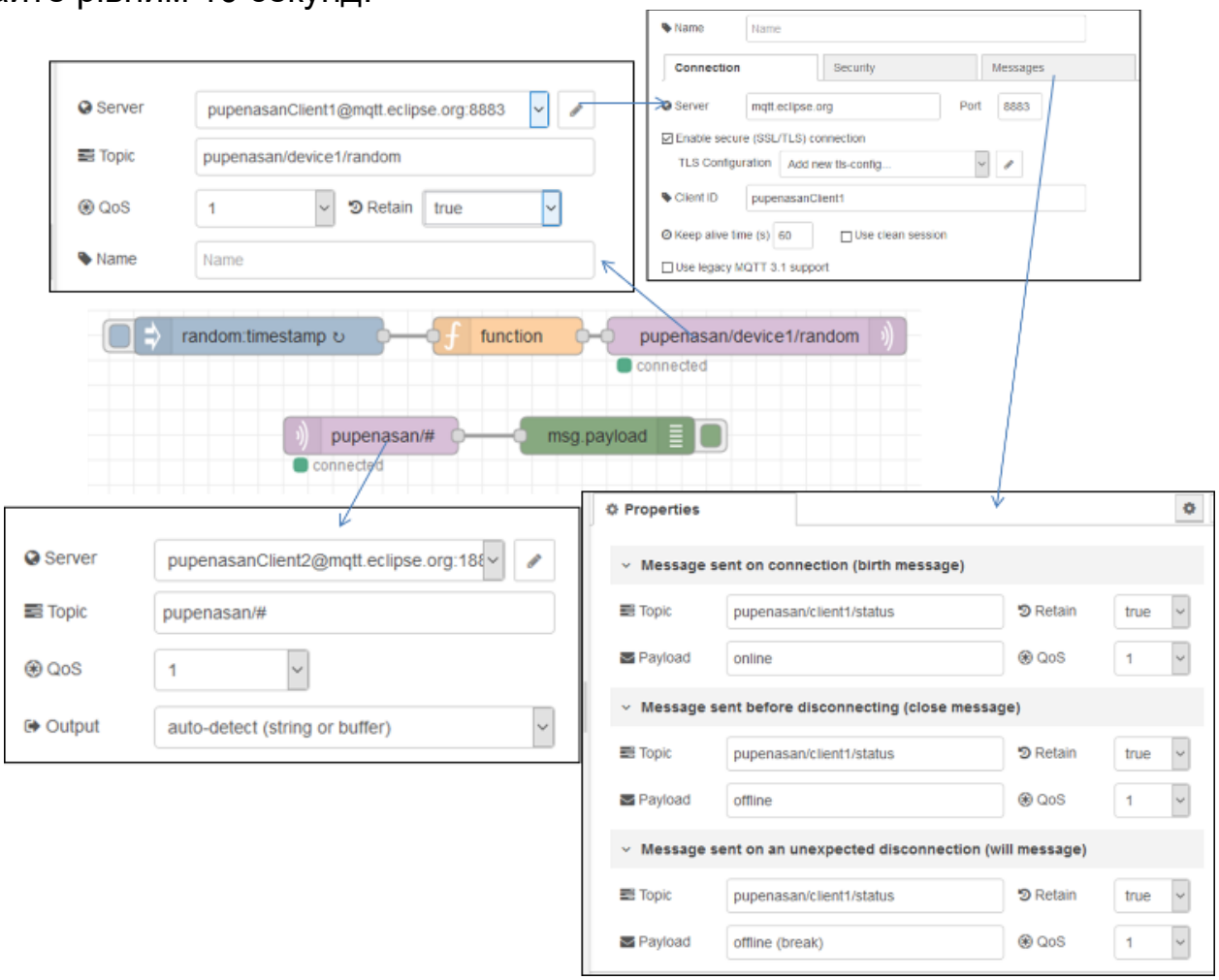


Рисунок 2.8 – Реалізація "коротко-замкнутого" з'єднання видавця і абонента в Node-RED

Програма функції має наступний вигляд:

```
let rad = context.get ("rad") || 0;
rad = (rad>6.28) ? 0 : rad + 0.1;
msg.payload = (Math.sin (rad)+1)/2*100;
context.set ("rad", rad);
return msg;
```

– запустіть потік на виконання, перевірте що повідомлення відображаються на бічній панелі

– У MQTT Explorer у налаштуваннях підключення Advanced, підпишіться на гілку myname/# з QoS1

– перевірте що тема myname/device1/random оновлюється

– перевірте що статус myname/client1/status рівний online

2.3.2.6 Перевірка роботи LWT-повідомлення

У налаштуваннях підключення вказане повідомлення останньої волі LWT. Воно відправиться тільки при некоректному обриву з'єднання. Для імітації такого обриву можна тимчасово відключити мережу, після чого завершити роботу Node-RED, щоб при появі мережі він не обновив статус. Саме після підключення мережі при вимкнутому Node-RED за допомогою MQTT Explorer можна буде визначити, повідомлення останньої волі, яке має бути рівним offline (break). Слід зауважити, що після останнього відправленого вузлом MQTT-out повідомлення повинно пройти щонайменше 1,5 часу Keep alive time, який дорівнює 60 секунд.

Вимкніть підключення до комп'ютерної мережі, щоб Node-RED не міг відправляти дані

Зупиніть виконання Node-RED (наприклад через комбінацію CTRL+C в консолі, з якої він запускався)

Включіть мережу.

За допомогою MQTT Explorer подивіться статус myname/client1/status, десь через хвилину-півтори він повинен стати в значення offline (break)

Запустіть Node-RED дочекайтеся, коли він запуститься

myname/client1/status повинен знову стати online

2.3.2.7 Перевірка роботи черги повідомлень для QoS=1

При знятій опції Use Clean Session , що значить Persisten connection, а також QoS>=1 і постійному ID-клієнта, повідомлення що не були отримані абонентом під час його відключення, зберігаються в буфері брокера. Після повторного підключення він повертає їх клієнтові.

Зайдіть в налаштування конфігураційних вузлів, знайдіть вузол MQTT-broker що відповідає з 2-ге підключення (до якого підключений MQTT-in) і деактивуйте вузол (відключіть Enable)

Зробіть розгортання проекту, очистіть усі повідомлення на бічній панелі

Дочекайтеся десь хвилини, активуйте знову конфігураційний вузол, і зробіть розгортання.

Повинно пройти кілька повідомлень одразу. Проаналізуйте їх зміст. Спробуйте пояснити чому саме такі повідомлення прийшли.

2.3.3 Зв'язок MQTT-клієнта з мобільного телефону

Для виконання даної частини практичного завдання необхідно мати пристрій з Андроїдом або iOS. Даний пристрій буде використовуватися як мобільний клієнт MQTT.

2.3.3.1 Встановлення MQTT Client для мобільного телефону

Встановіть безкоштовний додаток MQTT Client

- Приклад для Андроїд “IoT MQTT Panel” завантажити тут
- Приклад для iOS «MQTTool» завантажити тут

2.3.3.2 Добавлення з'єднання з MQTT брокером

Запустіть на виконання.

Додайте з’єднання з MQTT брокером (MQTT Host).

Наприклад, для застосунку “IoT MQTT Panel” це робиться в розділі Connection, де означуються ті самі налаштування, що і в попередніх пунктах. Додатково також треба добавити Device.

2.3.3.3 Добавлення та тестування інтерфейсу користувача

Створіть інтерфейс користувача шляхом добавлення графічних елементів та означте для них теми (Topic) відповідно до вибраного варіанту.

Перевірте роботу, змінюючи значення клапану на тестовому сервері.

2.4 Питання для самоперевірки

1. Розкажіть про принципи функціонування протоколу MQTT.
2. Які можливості утиліти MQTT Explorer використовувалися в даній практичній роботі?
3. Які можливості сервісу HiveMQ Вебсокет-клієнту використовувалися в даній практичній роботі?
4. Розкажіть про принципи публікації і підписки в MQTT. Як це налаштовується в клієнтах?
5. Розкажіть про принципи використання MQTT в Node-RED.
6. Розкажіть про принципи функціонування сервісу LWT в MQTT. Як цей сервіс використовувався в практичній роботі?
7. Розкажіть про призначення QoS.
8. Розкажіть про принципи функціонування HTTP API та REST.
9. Які відкриті сервіси HTTP API і як використовувалися в даній практичній роботі?

2.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40.* URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).

2. Node-RED українською : довідник з Node-RED українською мовою. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).

3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ : НУХТ, 2021. 170 с.

4. Лабораторна робота №2. Протоколи IoT: MQTT. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab2_1.md (дата звернення: 09.07.2024).

5. Протокол MQTT. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B5%D0%BA%D1%86/MQTT.md (дата звернення: 09.07.2024).

6. HiveMQ Cloud. URL: http://www.hivemq.com/demos/websocket-client (дата звернення: 09.07.2024).

7. MQTT Explorer. URL: http://mqtt-explorer.com/ (дата звернення: 09.07.2024).

ПРАКТИЧНА РОБОТА №3. ПРОТОКОЛИ IOT: ВИКОРИСТАННЯ WEB API ТА WEB-СОКЕТІВ

Увага! Більшість ресурсів є захищеними та потребують автентифікації, шифрування і т.п. У цій роботі усі інтерфейси є відкритими, тому не можуть в чистому вигляді використовуватися в промислових умовах!!!

3.1 Мета

Метою практичної роботи є знайомство принципами роботи протоколу HTTP та використання інструментів веб-розробника для аналізу та тестування HTTP запитів, створення HTTP клієнта в Node-RED для відправлення та обробки запитів; робота з відкритими WEB API для отримання даних та їх обробки, налаштування Web-сокетів для з'єднання мобільного застосунку та Node-RED, а також для обміну даними в реальному часі.

3.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:

- роботою HTTP та використанням інструментів Веб-розробника;
- реалізація клієнта HTTP в Node-RED;
- використання відкритого WEB API;
- використання WEB-сокетів для з'єднання мобільного застосунку та Node-RED;
- доробка застосунку в Node-RED для фіксації штрих-кодів в базі даних та виведення по ним інформації.

3.3 Хід роботи

3.3.1 Знайомство з роботою HTTP та використанням інструментів Веб-розробника

Ознайомтеся з принципами роботи HTTP (даються в лекційному матеріалі).

Для перевірки роботи протоколу HTTP, відлагодження та тестування HTML, CSS і JavaScript в завантажених сторінках використовуються різноманітні інструменти Веб-розробника (WEB Developer Tools). Деякі з них є частиною браузерів, таких як Google Chrome або Mozila FireFox. Якщо Ви звикли працювати з Google Chrome, ознайомитися з вкладками і можливостями Ви можете за наступним посиланням. В даній практичній роботі використовується інструменти WEB Developer Tools, що вбудовані в Mozilla FireFox. Зокрема, для

аналізу і тестування HTTP запитів використовується Монітор мережі, документація українською мовою для якого доступна за посиланням. У прикладах в практичній роботі використовується українська версія FireFox, яку можна завантажити за наступним посиланням https://www.mozilla.org/uk/firefox/download/thanks/. При виконанні практичної роботи дозволяється використовувати і інші інструменти.

Для перевірки роботи запитів HTTP використовуватимуться сервіси сайту https://zxing.org/w/decode.jspx. Ці сервіси дають можливість декодувати штрих-коди та QR-коди що передаються на сайт у вигляді зображення. Слід відмітити, що є багато програм, які можуть це робити в онлайн, дана використовується тільки в якості прикладу.

3.3.1.1 Перевірка роботи ВЕБ-застосунку https://zxing.org

Відкрийте сторінку в браузері https://zxing.org/w/decode.jspx і перевірте її роботу. Для цього завантажте будь-який файл з зображенням штрих-коду, наприклад звідси (рис. 3.1).

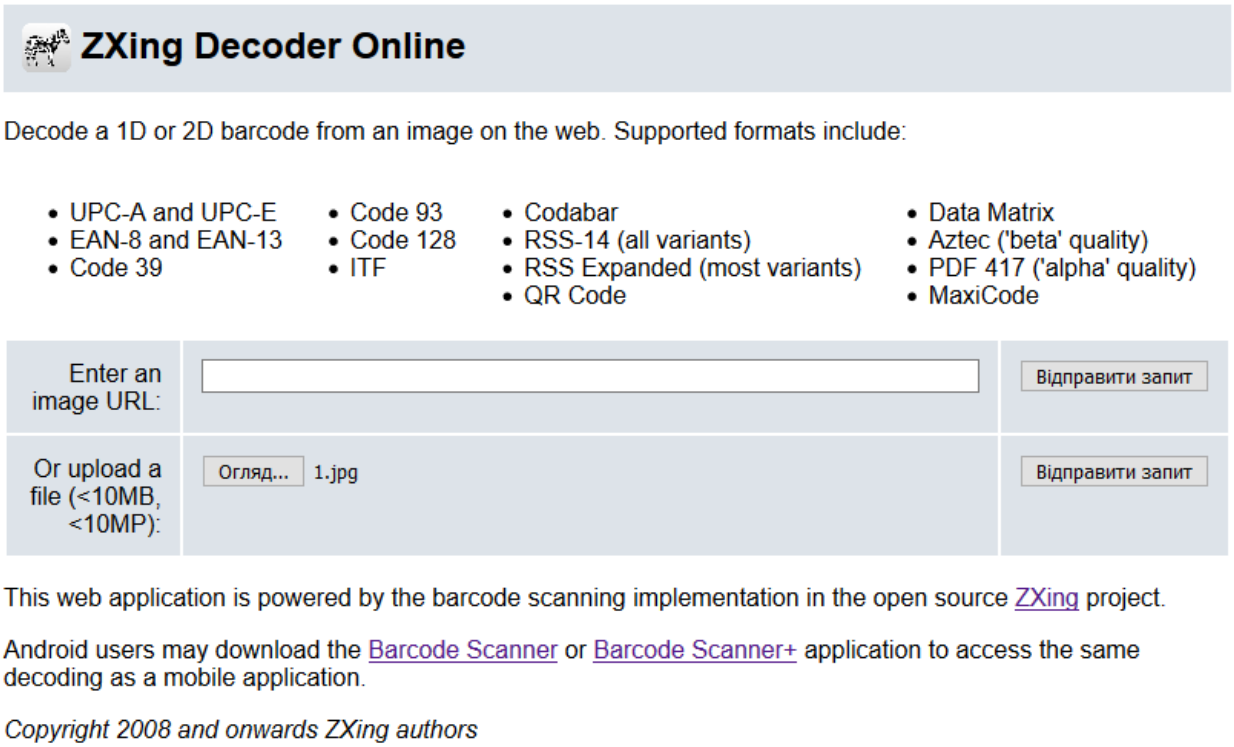


Рисунок 3.1 – Zxing Decoder Online

Натисніть «Відправити запит», при вдалій обробці запиту буде повернений результат з кодом, подібно до наведеного на рис. 3.2.

Decode Succeeded

Raw text	90494406
Raw bytes	(Not applicable)
Barcode format	EAN_8
Parsed Result Type	PRODUCT
Parsed Result	90494406

Рисунок 3.2 - Повернений результат з кодом

3.3.1.2 Робота з Монітором мережі FireFox.

Активуйте в браузері FireFox інструмент «Монітор мережі», або аналогічний в інших браузерах. Робота з Монітором мережі описана за посиланням. Виставите опцію «Вимкнути кеш» з панелі інструментів Монітору мережі. Відкрийте в браузері сторінку https://zxing.org . Подивіться список мережних запитів, які були зроблені при завантаженні сторінки, зокрема зверніть увагу на наступні значення:

- скільки запитів було зроблено;
- які причини та типи запитів;
- які були повернуті результати (стани), скористуйтеся даним ресурсом Вікіпедії для визначення стану.

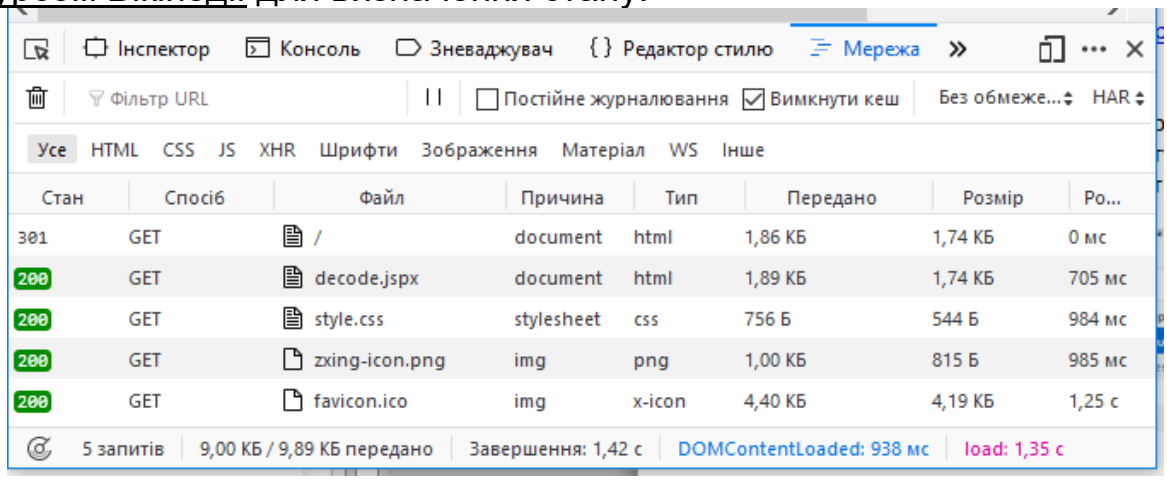


Рисунок 3.3 - Інструмент «Монітор мережі» в браузері FireFox

3.3.1.3 Аналіз заголовків 1-го мережного запиту та відповіді.

Активуйте панель Подробиць мережних запитів , відкрийте панель «Заголовки» для першого запиту в списку. Ознайомтеся з призначенням використаних в запиті заголовків з ресурсів Інтернет, наприклад з Вікіпедії , сторінкою із MDN або іншими аналогічними.

Заповніть перші два поля таблиці, наведеної нижче, для пояснення заголовків **в запиті**. Для зручності розшифровки можете скористатися посиланнями в колонці Заголовок. Інші поля можете розібрати за бажанням.

Таблиця 3.1 - Пояснення значень заголовків запиту

Заголовок	Значення	Пояснення значень заголовків
Accept-Encoding	gzip, deflate, br	
Accept-Language	uk-UA,uk;q=0.8,en-US;q=0.5,en;q=0.3	
Cache-Control	no-cache	
Connection	keep-alive	
Host	zxing.org	
Pragma	no-cache	
Upgrade-Insecure-Requests	1	
User-Agent	Mozilla/5.0 (Windows NT 10.0; …) Gecko/20100101 Firefox/64.0	

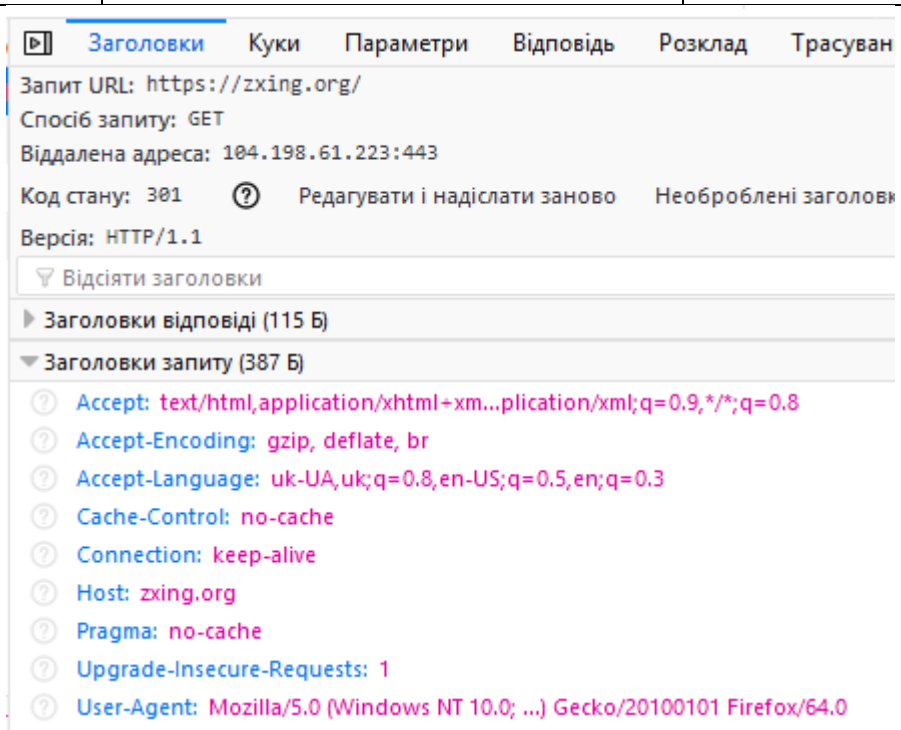


Рисунок 3.4 – Приклад заголовків

Заповніть поля таблиці для заголовку Location для пояснення **відповіді** . Для зручності розшифровки можете скористатися посиланнями в колонці Заголовок.

Таблиця 3.2 - Пояснення значень заголовків відповіді

Заголовок	Значення	Пояснення значень заголовків
Location	/w/decode.jspx	

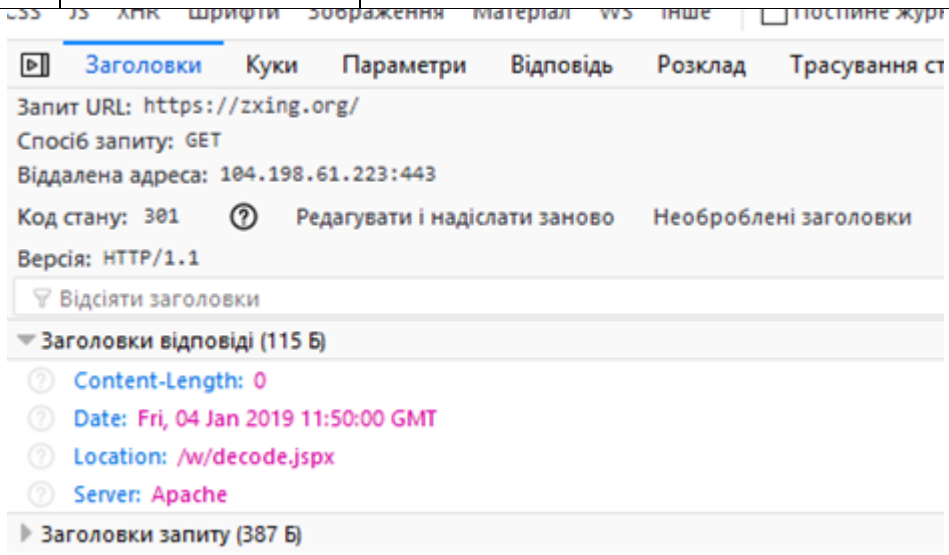


Рисунок 3.5 – Приклад значень заголовків відповіді

Перегляньте вміст відповіді у вкладці «Відповідь» на перший запит. Зробіть висновок щодо результату запиту. Впишіть результати в перший рядок таблиці 3.

Таблиця 3.3 - Пояснення до запитів та відповідей

№	Запит (метод, URL)	Пояснення до запиту, ініціатор	Відповідь (стан – пояснення)	Пояснення до відповіді
1	GET https://zxing.org/	Запит на документ, ініціював користувач в рядку браузеру	301 - ???	
2				
3				
4				
5				

3.3.1.4 Аналіз заголовків 2-го мережного запиту та відповіді.

Проаналізуйте другий запит та відповідь на нього. Зверніть увагу на зовнішній вигляд у вікні «Перегляд» у порівнянні з виглядом завантаженої сторінки. Чим вони відрізняються?

Відкрийте корисне навантаження відповіді на панелі «Відповідь» (у форматі HTML). Для того, щоб зручніше проаналізувати вміст завантаженої сторінки, скопіюйте його, відкрийте на іншій закладці браузеру посилання та помістіть скопійований текст в поле «Option 1: Copy-paste your HTML document here» після чого натисніть на кнопку «format HTML». У вікні «Formatted HTML:» з'явиться відформатований код сторінки в форматі HTML.

Проаналізуйте HTML код.

Зверніть увагу на тег **link** в заголовку HTML (допомога по призначенню тегу тут), та img (допомога по призначенню тегу тут).

Заповніть 2-й рядок таблиці 3.

3.3.1.5 Аналіз заголовків 3,4 мережного запиту та відповіді.

Проаналізуйте 3-й та 4-й запити та відповіді. За якої причини з`явилися ці запити? Заповніть 3-й та 4-й рядок таблиці 3.

3.3.1.6 Аналіз заголовків 5 мережного запиту та відповіді.

Для розуміння 5-го запиту, прочитайте цю статтю https://uk.wikipedia.org/wiki/Favicon . Заповніть 5-й рядок таблиці 3.

3.3.1.7 Аналіз форми в HTML .

Проаналізуйте частину HTML що відповідає за 2-гу форму (отримання коду по зображенню, що відправляється). Про форми в HTML та тег form, можна прочитати за цим посиланням.

3.3.1.8 Аналіз заголовків та змісту POST

Використовуючи кнопку «Очистити» що знаходиться на панелі інструментів Монітору мережі, очистіть список запитів. Завантажте будь-який файл з зображенням штрих-коду, як робили це в п.1. Натисніть «Відправити запит». При вдалій обробці запиту буде повернений результат з кодом.

Використовуючи подробиці мережного запиту POST продивіться та проаналізуйте Заголовки. Зробіть копію екранів з заголовками та впишіть пояснення в таблиці 4. Для пришвидшення пошуку призначення заголовків користуйтесь посиланнями, що прив'язані до назв заголовків. Про типи MIME можна прочитати тут.

Таблиця 3.4 - Пояснення значень заголовків запиту

Заголовок	Значення	Пояснення значень заголовків
Content-Length		
Content-Type		

Використовуючи подробиці мережного запиту POST продивіться та проаналізуйте Параметри.

Зробіть копію екрану корисного навантаження запиту.

3.3.1.9 Аналіз відповіді на POST

Відкрийте корисне навантаження відповіді на панелі «Відповідь» (у форматі HTML). Для того, щоб зручніше проаналізувати вміст завантаженої сторінки, скопіюйте його, відкрийте на іншій закладці браузеру посилання https://www.freeformatter.com/html-formatter.html та помістіть скопійований текст в поле «Option 1: Copy-paste your HTML document here» після чого натисніть на кнопку «format HTML». У вікні «Formatted HTML:» з'явиться відформатований код сторінки в форматі HTML.

Знайдіть місце в документі HTML, де виводиться значення штрих-коду. Зробіть копію екрану і виділіть це місце.

3.3.2 Реалізація клієнта HTTP в Node-RED

В даній частині практичної роботи необхідно в Node-RED розробити програму для визначення штрих-коду по зображенню, використовуючи сервіси https://zxing.org . Для розуміння цієї частини практичної знадобляться результати, отримані в частині 2.1.

3.3.2.1 Підготовчі роботи.

На диску C:\ створіть директорію «Temp», якщо вона ще не існує. У цій директорії створіть директорію «barcods», в якій будуть розміщуватися усі необхідні файли для практичної роботи. В директорії C:\Temp\barcods створіть директорію "raw" для розміщення вихідних файлів зображень. Скопіюйте в папку «C:\Temp\barcods» та «C:\Temp\barcods\raw» файл з зображенням штрих-коду, що використовувався в попередній частині, наприклад взятий звідси.

3.3.2.2 Читання файлу.

Завантажте Node-RED.

Зробіть статус усіх потоків неактивним.

Створіть новий потік з назвою «Laba2_WEBAPI». Цей потік буде використовуватися для даної частини практичної роботи.

З довідника ознайомтеся з вбудованими функціями роботи з файлами. З розділу “Storage” палітри вузлів вставте вузол “File in”. Налаштуйте його таким чином, щоб він зчитував дані з файлу з штрих-кодом та виводив його вміст на панель відлагодження (Debug). Зверніть увагу, що виведення в Debug треба робити як “complete msg object” а не як “msg.payload”.

Зробіть розгортання потоку, та з використанням панелі відлагодження подивіться що зміст файлу виводиться як масив байт (рис. 3.6).

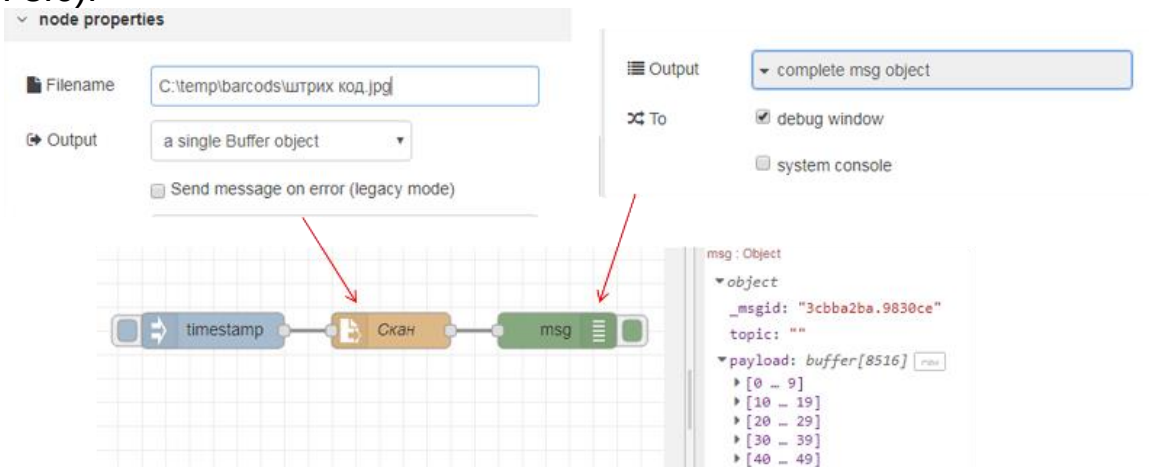


Рисунок 3.6 – Реалізація потіку та відображення результатів

3.3.2.3 Формування запиту.

Для роботи в Node-RED HTTP-клієнту використовується вузол «HTTP request». Ознайомтеся з його роботою з довідника.

Для формування запиту необхідно сформувати на вході вузла «HTTP request» його заголовки та тіло. З попередньої частини практичної роботи видно, що для отримання штрих-коду по зображенню необхідно сформувати запит з методом POST, який буде багато-частинним, тобто складатися з розділів, що відокремлюються межами. У даному випадку розділ є тільки один, який включає вміст файлу зображення. Для пришвидшення виконання практичної роботи імпортуйте в потік підготовлений код для вузлу «Формувати запит», файл імпорту завантажте звідси. Уважно ознайомтеся з його вмістом.

Модифікуйте програму, як це показано на рис. 3.7. Зробіть розгортання, активуйте ініціювання повідомлення в Inject і подивіться на вміст об’єкту на панелі відлагодження. Порівняйте результат з заголовками та вмістом запиту POST з п.1.8 попередньої частини («Знайомство з роботою HTTP»). Що відрізняється і чому?

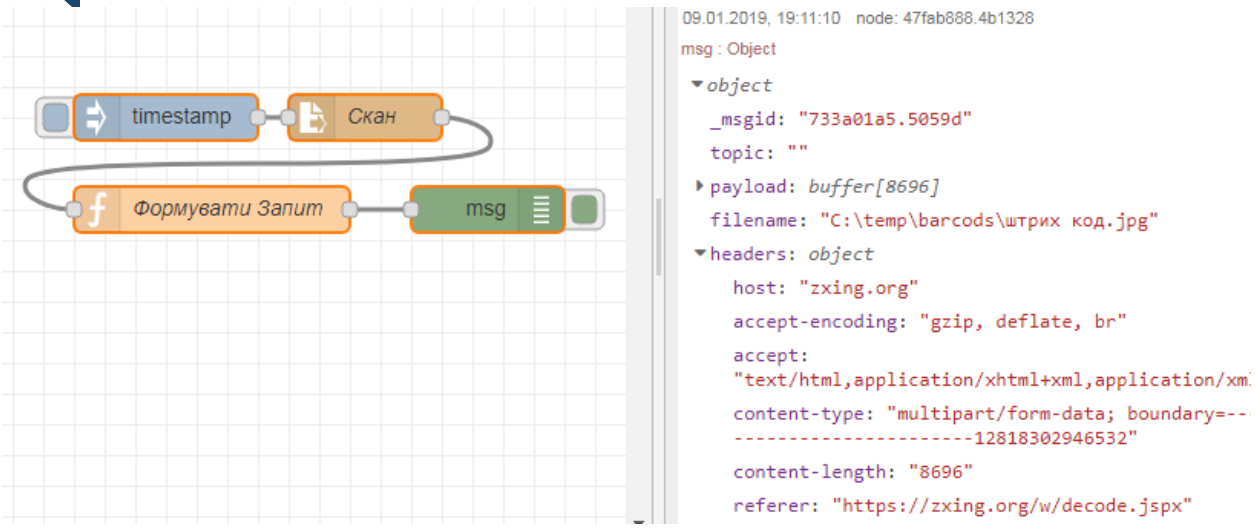


Рисунок 3.7 – Модифікована програма і результати виконання

3.3.2.4 Відправлення запиту.

З розділу «function» палітри вузлів вставте вузол «HTTP request». Модифікуйте програму, як це показано на рис. 3.8. Зверніть увагу, що для кращого тестування бажано вивести як вхідне так і вихідне повідомлення «HTTP request».

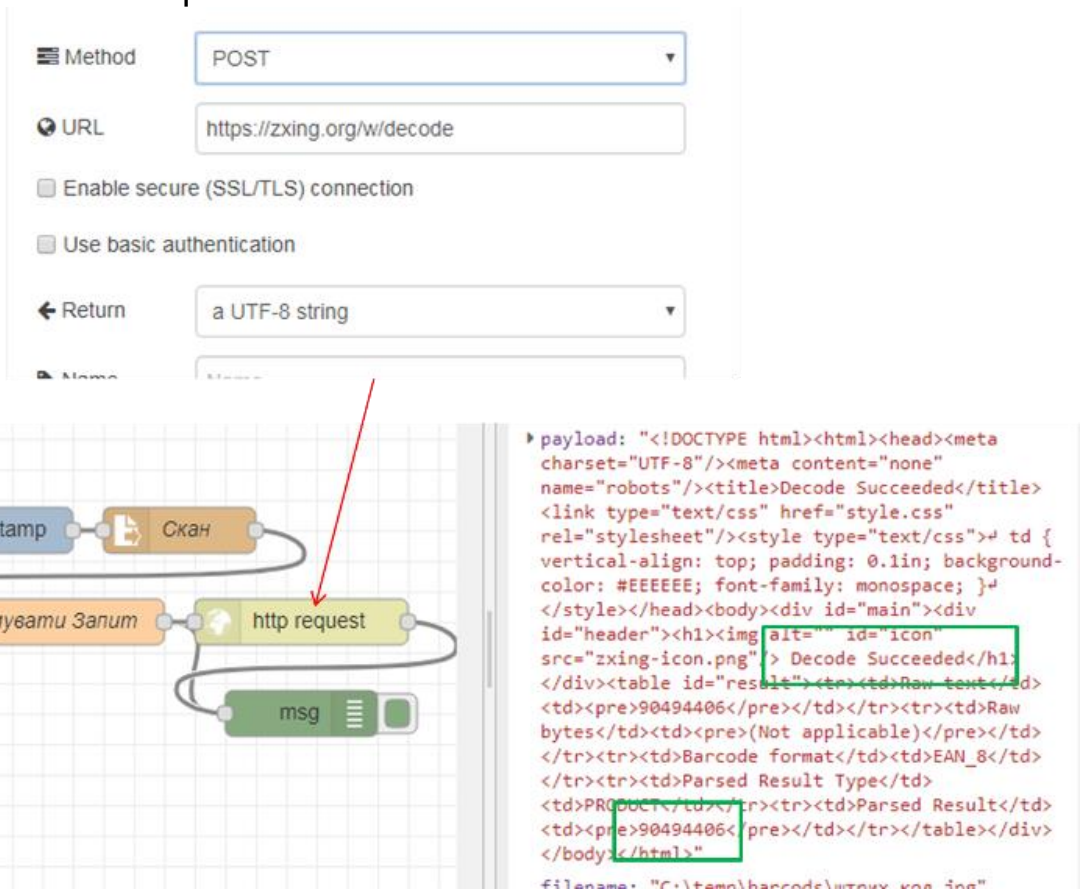


Рисунок 3.8 – Використання вузлу «HTTP request»

Зробіть розгортання, активуйте Inject, через кілька секунд на панелі відлагодження з’явиться інформація про об’єкт-відповідь. Порівняйте

результат з заголовками та вмістом відповіді POST з п.9 попередньої частини («Знайомство з роботою HTTP»). Результат виконання вважається позитивним, якщо у відповіді буде напис «Decode Succeeded». Знайдіть значення штрих-коду.

3.3.2.5 Розбирання запиту по частинам.

Для витягування потрібних елементів з HTML-контенту в Node-RED використовується вузол «HTML». Ознайомтеся з його роботою з довідника.

З розділу function палітри вузлів вставте «HTML». Модифікуйте програму, як це показано на рис. 3.9. Селектор вибирає елемент HTML з ID=”result” у якого вибирає усі-елементи нащадки з тегом «pre».

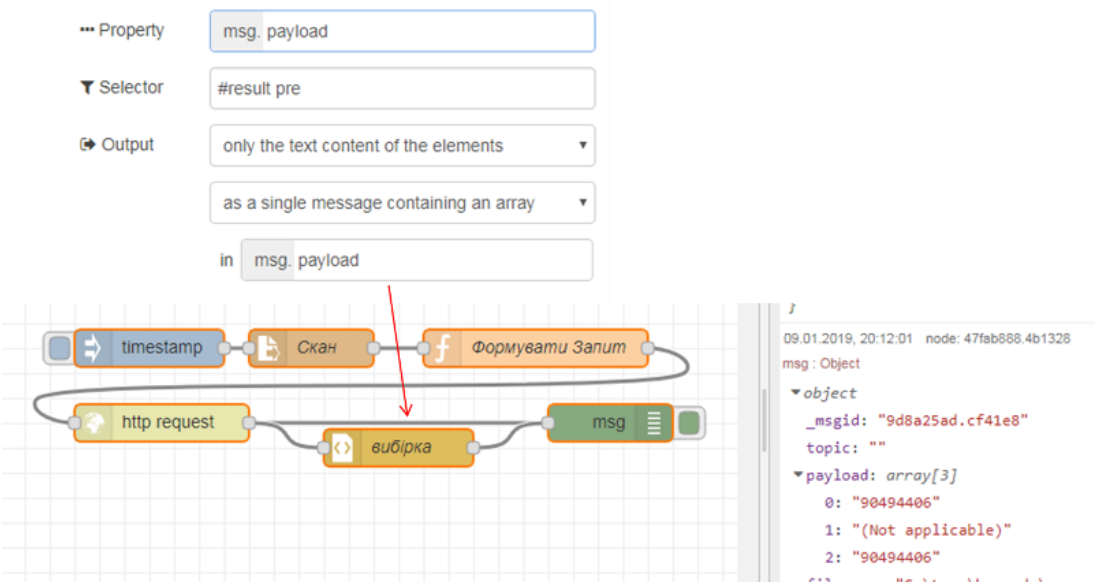


Рисунок 3.9 – Модифікована програма

Зробіть розгортання, активуйте Inject, через кілька секунд на панелі відлагодження з’явиться інформація про масив вибірки.

Додайте вузол-функцію для присвоєння значення msg.ID рівним номеру штрих-коду при позитивній відповіді, та “ERROR” при помилці обробки (рис. 3.10). Зробіть розгортання, активуйте Inject та перевірте роботу програми.

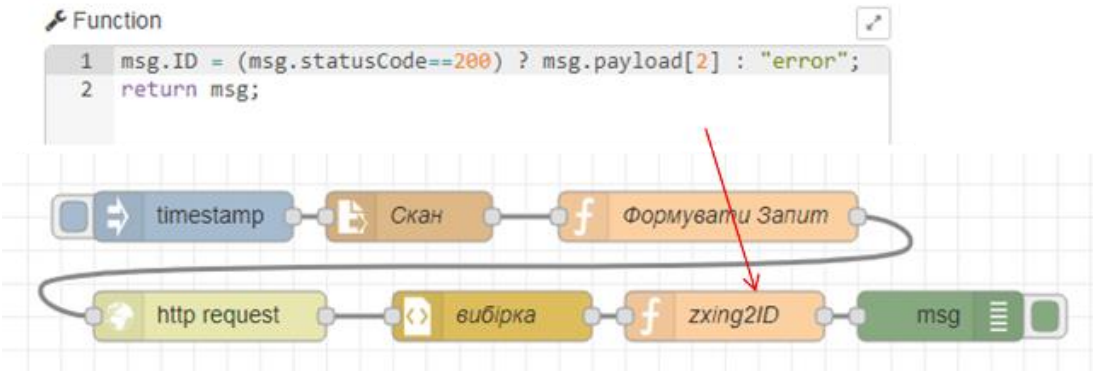


Рисунок 3.10 – Додавання в програму функції zxing2ID

3.3.3 Використання відкритого WEB API

Ознайомтеся з принципами роботи WEB-API та REST (даються в лекційному матеріалі).

Багато застосунків в Інтернеті мають відкритий API інтерфейс для доступу до різних ресурсів як сервісів чи даних. Більшість з них є платними і надаються за підпискою. Деякі з них мають можливість обмеженого користування на певний період чи на певну продуктивність.

Дана частина практичної роботи призначена для знайомства з тестовими утилітами та для самостійної побудови WEB-клієнта для доступу через API з Node-RED.

3.3.3.1 Знайомство з сервісами IPAPI.

Веб-застосунок IPAPI дає можливість визначити деталі місця розташування за IP адресою. Це можна зробити через сторінку Веб-інтерфейсу, або через відкритий API-інтерфейс (обмеження на безкоштовне використання до 1000 запитів/день). Повний опис API доступний за посиланням.

Зайдіть на сторінку за посиланням https://ipapi.co. Ознайомтеся зі змістом сторінки. Зверніть увагу на ту інформацію, яка надається по IP-адресі, а також на значення Вашої білої адреси IP, вірніше від якої Ваш пристрій спілкується в Інтернеті. Слід розуміти, що у більшості випадків видима IP-адреса – це одна з адрес провайдера, що надає послуги Інтернету, тому координати будуть саме цього провайдера.

Подивіться на приклад запиту і відповіді в форматі JSON.

3.3.3.2 Робота з онлайн утилітами для API-тестування

Для тестування API Ви можете користуватися будь якою утилітою, наприклад https://reqbin.com/ (у відео використовується https://apitester.com який застарів)

Відкрийте сторінку однієї з утиліт наприклад https://reqbin.com/

Для перевірки роботи API https://www.myip.com введіть в поле адреси https://api.myip.com у метод – «GET», і натисніть «SEND» або "TEST" в залежності від вибраного програми.

Проаналізуйте відповідь

Повторіть те саме з адресою http://ip-api.com/json/8.8.8.8/ .

Повторіть те саме зі своєю білою IP-адресою, або просто відправивши http://ip-api.com/json/. Порівняйте отримані результати з тими, що показані на сторінці https://ip-api.com

3.3.3.3 Створення клієнту для IPAPI в Node-RED

Запустіть Node-RED, створіть нову вкладку з назвою IPAPI.

Самостійно реалізуйте програму, яка буде використовуючи вузол “Http request” для витягування інформації про білий IP, використовуючи сервіси https://www.myip.com.

Аналогічно зробіть для сервісів http://ip-api.com/json/. Зверніть увагу, що **IP-API може повернути негативну відповідь** - це може бути спричинено обмеженим використанням безкоштовного сервісу.

Зверніть увагу, що **IPAPI може повернути негативну відповідь. Подивіться поля заголовків** https://reqbin.com/ в запиті, і заповніть їх аналогічно у вузлі headers

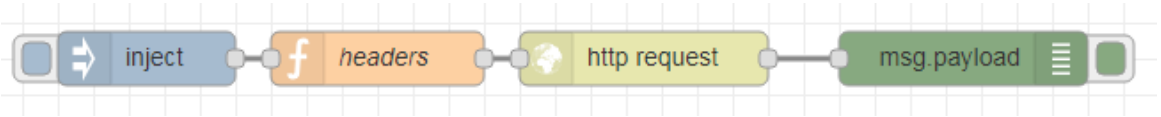


Рисунок 3.11 – Реалізація завдання

Зробіть копію фрагменту екрану виводу у вікні відлагодження і збережіть для звіту.

Наступні частини не є обов’язковими для виконання!

3.3.4 Використання WEB-сокетів для з’єднання мобільного застосунку та Node-RED

В даній частині практичної роботи використовується безкоштовна версія застосунку Wireless Barcode Scanner для пристроїв з Android. Даний застосунок дає можливість відсканувати і розпізнати штрих-код і передати його по наступним мережним протоколам:

- Bluetooth,
- TCP/IP,
- UDP
- HTTP
- WebSocket

В даній частині в якості віддаленого сканера пропонується використати пристрій з Android, який буде передавати сканований штрих-код по WebSocket через WiFi. Мобільний пристрій і ПК повинні бути в одній мережі.

3.3.4.1 Налаштування дозволів використання вхідних портів на ПК

Типово, вхідні порти ПК заблоковані брандмауером Windows. Тому необхідно налаштувати брандмауер Windows для можливості використання вхідного порту з мережі, а саме:

- відключити правила, що забороняють доступ до порту 1880 та роботи Node.js
- добавити правило, що дозволяє доступ до порту 1880

Відкрийте налаштування брандмауера Windows через «Панель керування»-«Адміністрування» - «брандмауер Windows». Відключіть усі правил, що забороняють доступ JavaScript Node.js, добавте та активуйте правило для вхідних підключень по порту 1880 (рис. 14)

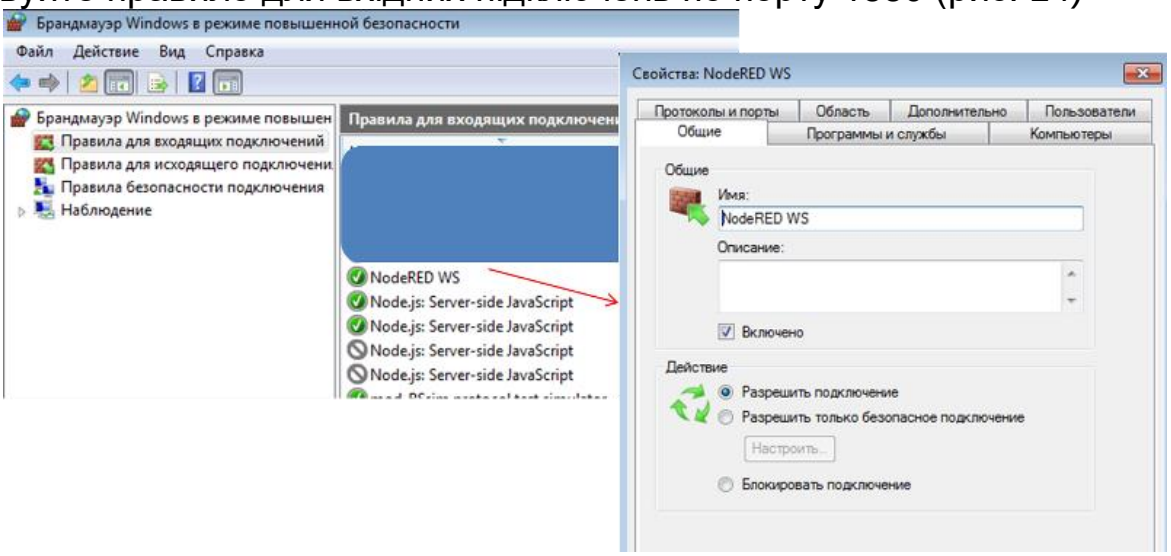


Рисунок 3.14 – Налаштування правил для доступу JavaScript Node.js

3.3.4.2 Конфігурація WebSocket server в застосунку Node-RED

Відкрийте програму, що була створена в частині 2.2 «Реалізація клієнта HTTP в Node-RED» (Потік «Laba2_WEBAPI»). Вставте вузол “Websocket In”, що буде очікувати з’єднання та отримувати по ньому дані (рис. 3.15).

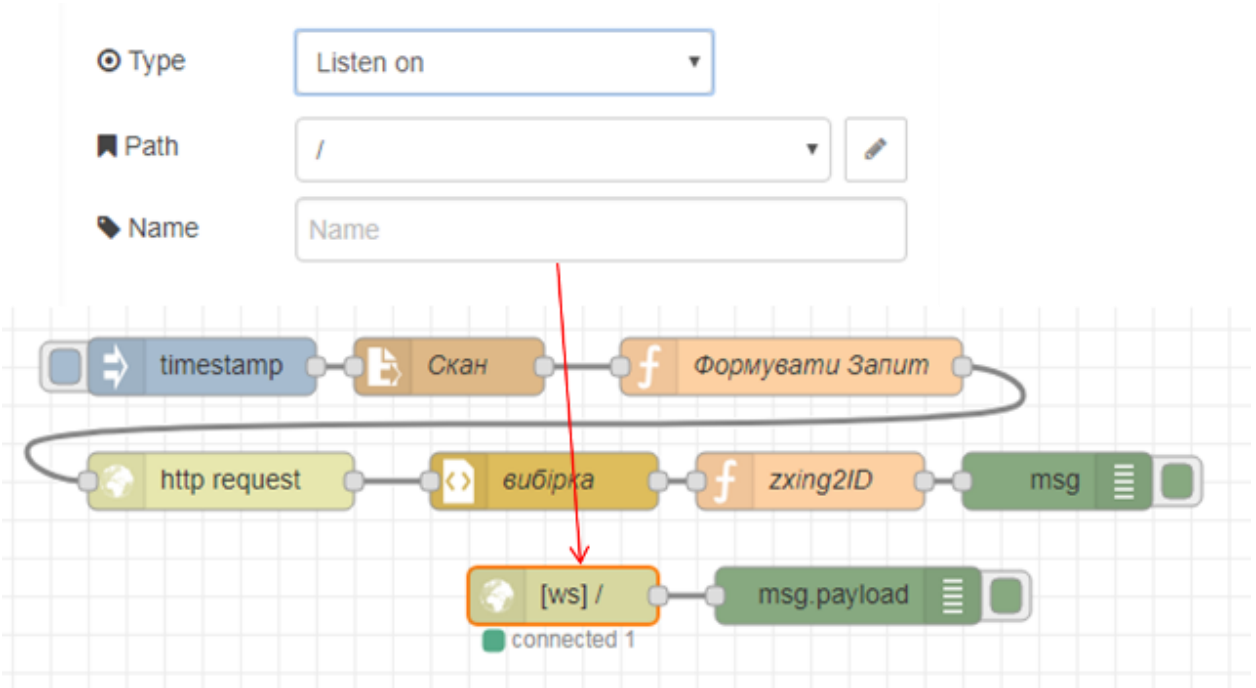


Рисунок 3.15 – Додавання вузлу Websocket In

3.3.4.3 Встановлення та налаштування мобільного застосунку

Встановіть на мобільний пристрій безкоштовну версію Wireless Barcode Scanner .

Визначте IP-адресу ПК, через яку буде проводитися доступ до нього з мережі (можна подивитися у властивостях мережної карти).

Запустіть застосунок Wireless Barcode Scanner. Відкрийте конфігурацію через пункт меню «setting». Виставіть наступні налаштування в розділі Data link:

- Data transfer – вкл.
- Connection type – WebSocket client
- Configuration – Host=xx.xx.xx.xx, Port=1880

де xx.xx.xx.xx – номер IP ПК з Node-RED

Вийдіть з конфігурації застосунку в режим сканування. Якщо з'єднання відбулося, внизу екрана з'явиться зелена літера "**i**", якщо ні – червона «**!**» .

3.3.4.4 Перевірка роботи

У Wireless Barcode Scanner в робочому режимі відскануйте штрих-код шляхом натискання кнопки "Scan".

На панелі відлагодження Node-RED повинен з'явитися буфер відсканованих символів, що закінчуються двома символами – кінця і переведення рядку , тобто 0xd та 0xa (рис. 16).

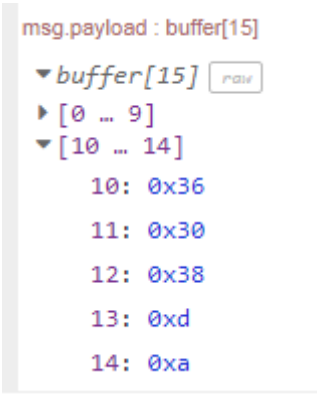


Рисунок 3.16 – буфер відсканованих символів

3.3.4.5 Визначення ID

Додайте функцію яка буде перетворювати отриманий буфер в символьний рядок ID (рис. 3.17).

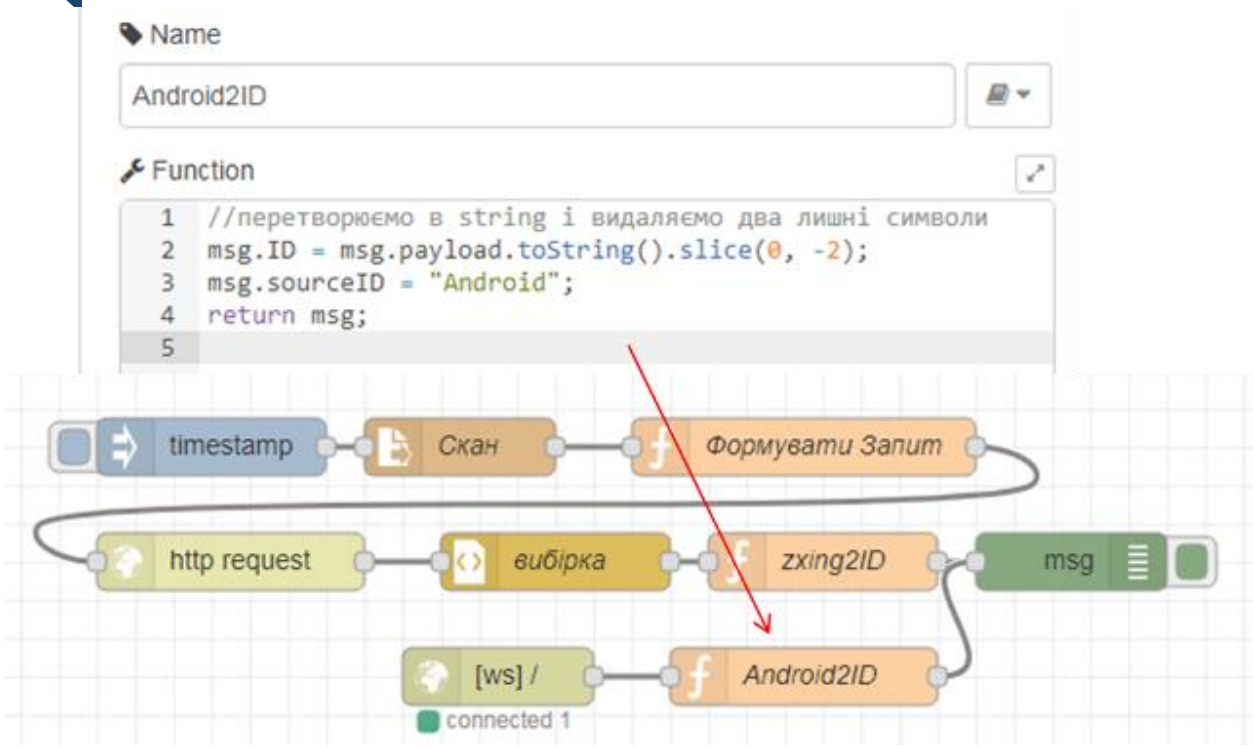


Рисунок 3.17 – Додавання функції Android2ID

Перевірте роботу, проконтролюйте щоб ID в повідомленні співпадав з ID на штрих-коді.

3.3.5 Доробка застосунку в Node-RED для фіксації штрих-кодів в базі даних та виведення по ним інформації

У даній частині практичної роботи необхідно доробити розроблену програму таким чином, щоб вона виконувала наступні функції:

1. ведення бази даних (текстовий файл) по продуктам за зчитуваними штрих-кодами; в БД записується:
   - країна походження продукту: визначається автоматично за номером штрих-коду EAN-13
   - джерело надходження в базу номеру штрих-коду: ім’я файлу зображення штрих-коду або напис «Android»
   - дата та час надходження в базу номеру штрих-коду
   - найменування продукту: вноситься вручну
   - опис продукту: вноситься вручну
   - посилання на PDF-файл документації: вноситься вручну
2. перевірка наявності в директорії нових файлів з розширенням *.jpg і *.png, автоматичне їх сканування і перейменування (для уникнення повторної обробки)
3. внесення в БД тільки тих штрих-кодів, які в ній відсутні
4. при скануванні існуючого в БД штрих-коду на веб-інтерфейс повинна бути виведена інформація за вказаним продуктом включно з PDF-файлом документації
5. помилка читання повинна бути зафіксована в журналі подій

3.3.5.1 Формування бази країн по EAN-коду

Завантажте та інсталюйте Notepad++, який Вам знадобиться для перегляду та коректної правки текстових файлів.

Відкрийте потік LAbA2_WEBAPI. Імпортуйте в потік частину програми з цього посилання. Вона матиме вигляд, як на рис. 3.18.

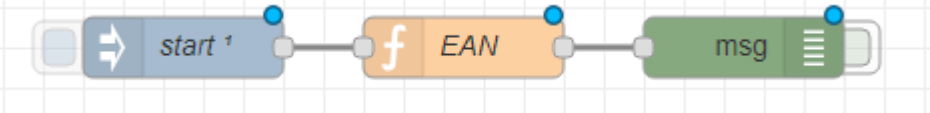


Рисунок 3.18 – Потік імпортованої програми

Зробіть розгортання проекту. Активуйте відображення повідомлень з Msg (воно за замовчуванням в прихованому режимі). При старті системи, або при активації кнопкою елементу Inject “Start” у вікно налагодження буде виведено масив, в якому індекс відповідає за номер країни, а значення – за назву. Про деталі відповідності номеру країні можна прочитати за цим посиланням.

Подивіться на вміст функції EAN. При старті значення кодів також будуть записуватися в контекст потоку, у його властивість «arEAN», значення якої можна буде прочитати у будь якому вузлі цього потоку.

3.3.5.2 Модифікація структури запису по продукту та виведення помилок сканування

Імпортуйте в потік вузол-функцію «IDparse» з цього посилання. Модифікуйте програму щоб вона виглядала так, як на рис. 3.19.

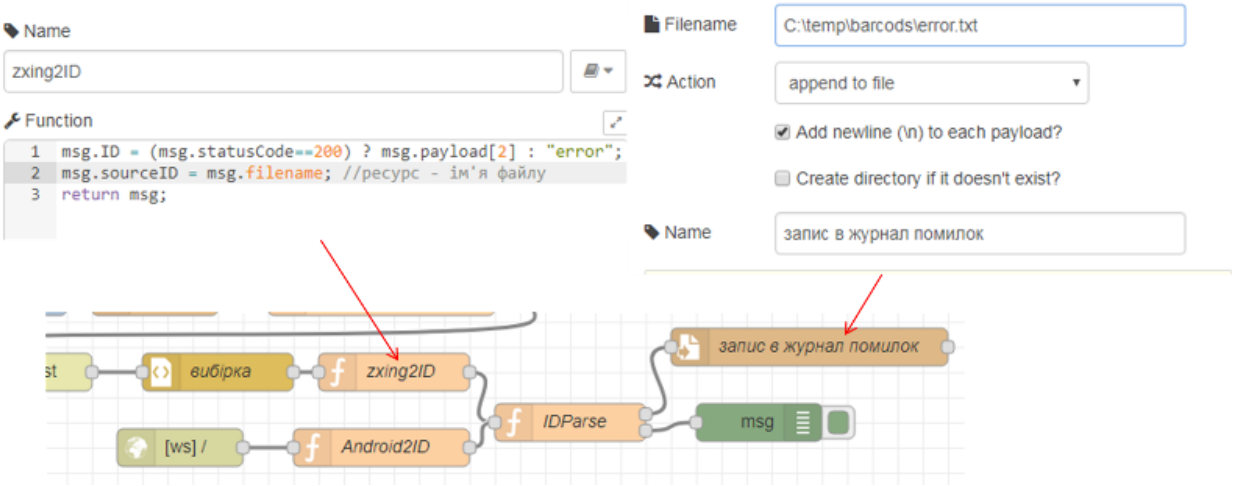


Рисунок 3.19 – Модифікація програми

Зробіть розгортання програми, активуйте повідомлення вприскуванням Inject. Тепер після обробки запиту на панелі відлагодження повинен з’явитися об’єкт з властивістю prodinfo, в якій будуть поля та значення, що описані в переліку функцій.

3.3.5.3 Запис в базу даних (текстовий файл)

Модифікуйте програму так, як показано на рис. 3.20.

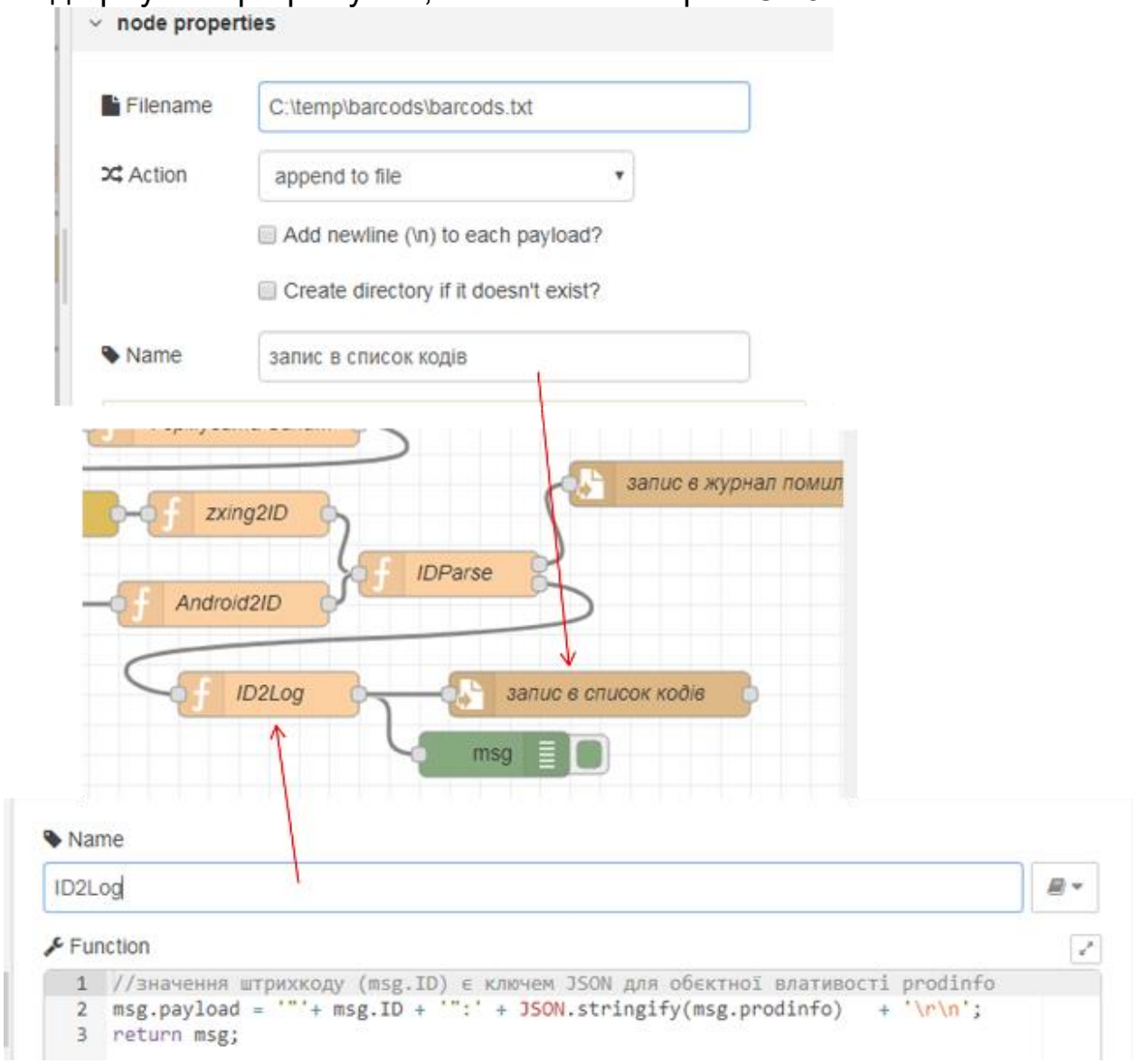


Рисунок 3.20 - Запис в базу даних

Розгорніть та запустіть на виконання, активуйте перерахунок потоку. В файл barcods.txt повинен записатися об’єкт з записами prodinfo.

Примітка. Якщо в файлі некоректно відображається кирилиця (характерно для Windows 7 x86) необхідно зробити наступні кроки:

- **встановити в Node-RED модуль contrib-iconv (рис. 3.21)**
- **модифікувати програму, як показано на рис. 3.22**

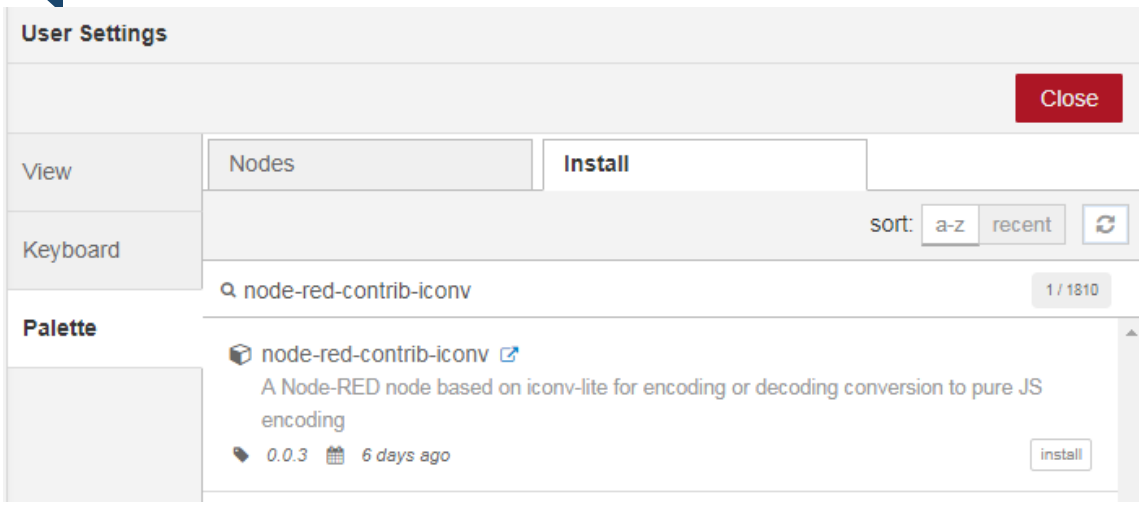


Рисунок 3.21 – Встановлення в Node-RED модуль contrib-iconv

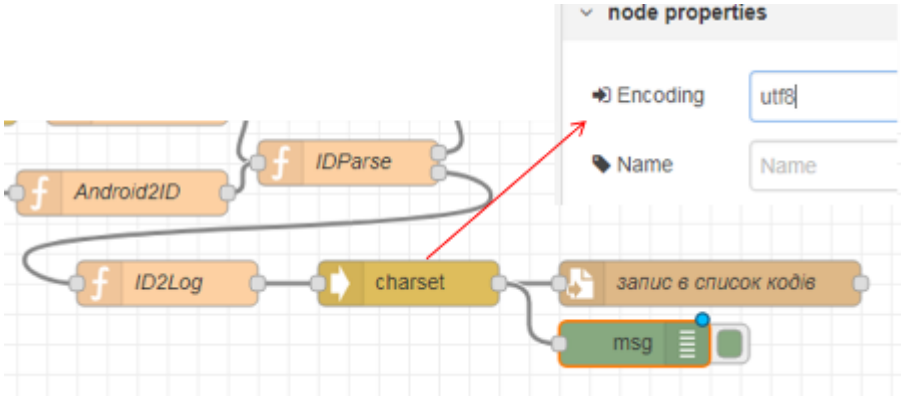


Рисунок 3.22 – Модифікована програма

3.3.5.4 Перевірка наявності в директорії файлів з розширенням *.jpg і *.png

Завантажте файли зі штрих-кодами в директорію «C:\Temp\barcods\raw» вони будуть використовуватися як файли з зображеннями для сканування. **Для пришвидшення сканування можете обробити файли так, щоб на них залишилися тільки зображення штрих-кодів.**

Інсталюйте модуль «node-red-contrib-fs-ops», в ньому знаходяться два вузли, які знадобляться для практичної роботи “dir” і “move”. Познайомтеся з принципами їх роботи з довідника.

Перевірте роботу алгоритму вибірки файлів за фільтрами *.jpg та *.png за допомогою фрагмента програми, показаного на рис. 3.23.

Рисунок 3.23 - Алгоритм вибірки файлів за фільтрами *.jpg та *.png

Зробіть розгортання, перекопіюйте декілька файлів з директорії «C:\Temp\barcods\raw» в «C:\Temp\barcods», запустіть фрагмент на виконання і подивіться на вміст панелі відлагодження.

3.3.5.5 Пошук графічних файлів в директорії

Імпортуйте вузол-функцію “find” з цього посилання. Цей вузол об’єднує масиви назв «*.jpg» та «*.png» і шукає в них перший файл, в якого назва не містить послідовність літер «cmplt» (ця послідовність використовуватиметься для маркування оброблених файлів). Модифікуйте фрагмент програми, як це показано на рис. 3.24. Перевірте його роботу.

Рисунок 3.24 - Пошук графічних файлів в директорії

3.3.5.6 Автоматичне сканування і перейменування знайдених файлів

Модифікуйте програму, щоб вона мала вигляд як на рис. 3.25. Зверніть увагу, що назву файлу з вузлу «Скан» треба видалити, тільки в такому разі назва файлу буде братися з властивості повідомлення “msg.filename”. Перевірте як працює програма.

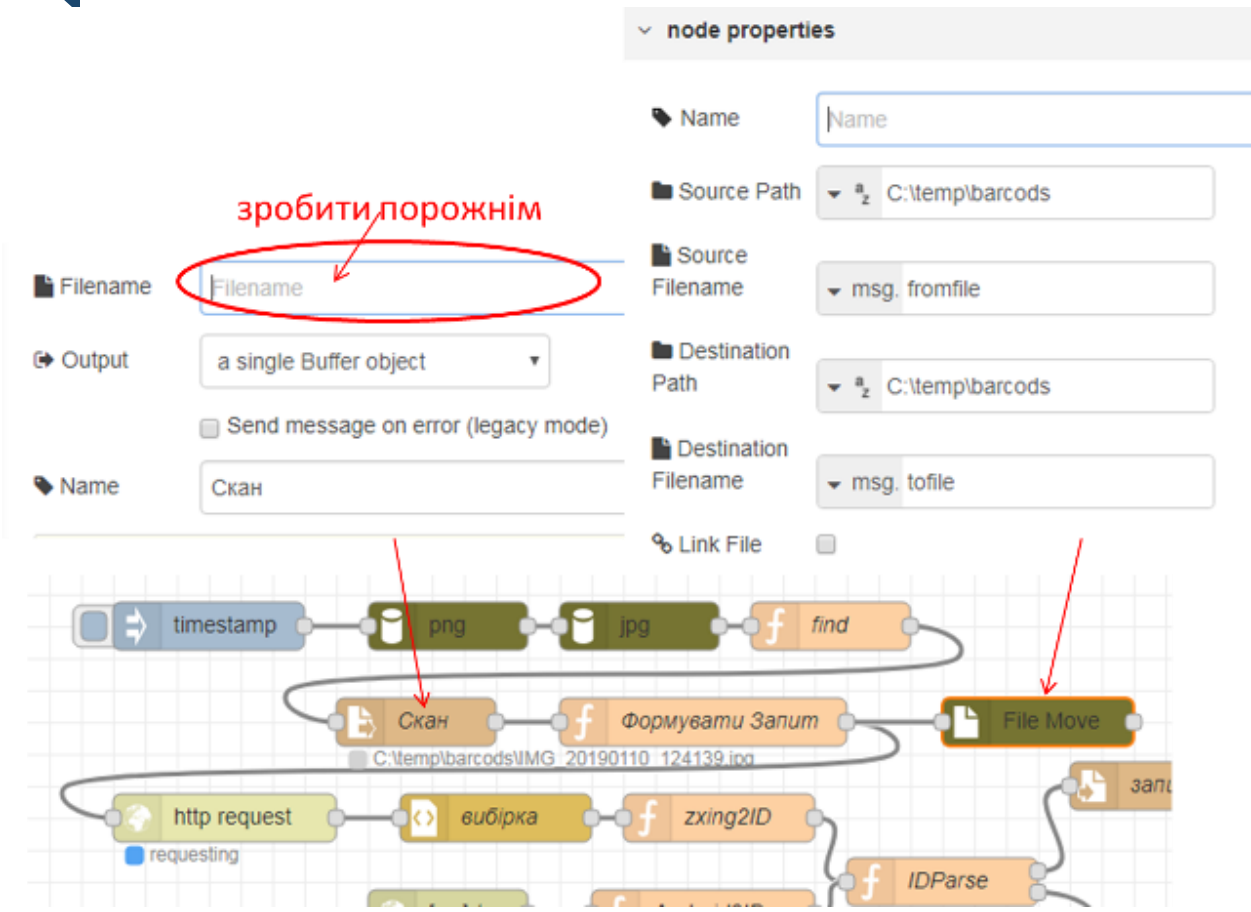


Рисунок 3.25 – Потік автоматичного сканування і перейменування знайдених файлів

Модифікуйте програму так, щоб вузол «Timestamp» генерував повідомлення кожні 2 секунди.

Таким чином кожні 2 секунди буде скануватися один із графічних файлів і перейменовуватися з префіксом «cmplt_».

3.3.5.7 Синхронізація сканування та періодичності обробки файлів

Видаліть усі графічні та текстові файли з директорії «C:\Temp\barcods» . Скопіюйте усі графічні файли з директорії «C:\Temp\barcods\raw» в «C:\Temp\barcods». Прослідкуйте з якою частотою обробляються запити у вузлі «http request» та “File move”. Node-RED дозволяє розсинхронізовувати потоки таким чином, що на вхід вузла може прийти декілька повідомлень, які він буде обробляти поступово, буферизуючи їх. Однак в деяких випадках така обробка небажана. Тому в даному пункті необхідно зробити синхронізацію: пошук нових файлів проводити тільки у випадку, коли не відбувається очікування відповіді на запит «http request». Контроль стану вузла можна проводити за допомогою вузла “Status”.

Модифікуйте програму так, щоб відбувалася синхронізація між знаходженням нового файлу і відсутності очікування відповіді від серверу (рис. 3.26). Також модифікуйте функцію «find» (рис. 3.27).

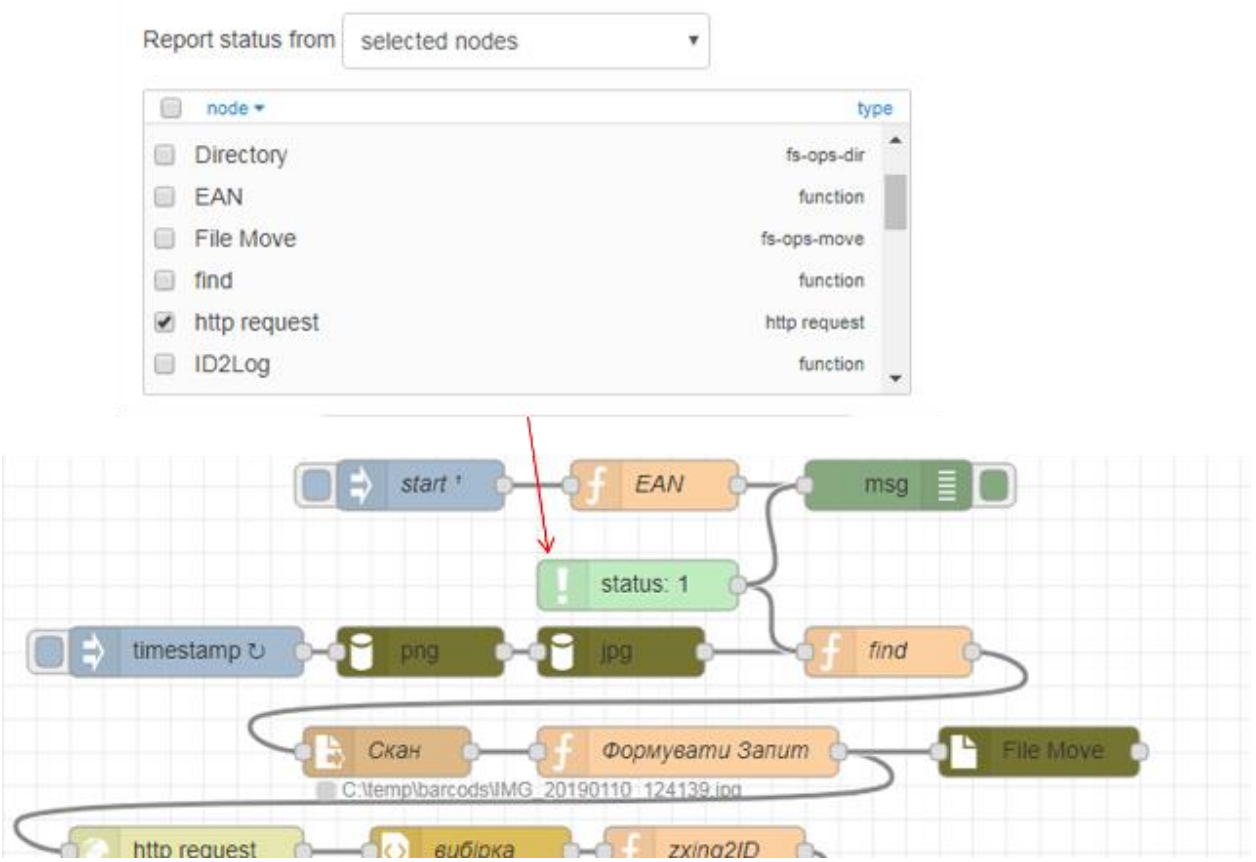


Рисунок 3.26 – Синхронізація знайдених файлів

```
//отримання значення з контексту "Занятість", або ініціалізація
var busy =  context.get ("busy") || false;
//якщо отримали повідомлення від вузла Status
if (msg.status!==undefined) {
    //якщо значення тексту статусу "httpin.status.requesting"
    if (msg.status.text ==="httpin.status.requesting")
    busy = true;// виставляємо "Занятість"
    else busy = false;//якщо інше значення тексту статусу
    context.set ("busy", busy);//записати значення в контекст вузла
}
//зчитувати назви файлів тільки коли не Занятий і повідомлення прийшло від Directory
if (!busy &&  msg.files !== undefined) {
    var path = "C:\\temp\\barcods\\";//шлях
    //обєднуємо масиви назв файлів *.png і *.jpg в один
    var files = msg.files[0].concat(msg.files[1]);
    msg.files = undefined;//видаляємо старі масиви
    //перебираємо назви файлів в масиві
    for (var i=0; i < files.length; i++){
        if (files[i].indexOf('cmplt')==-1) {//якщо в імені НЕ є 'cmplt'
            msg.filename = path + files[i];//записуємо в filename повне ім'я з шляхом
            msg.fromfile = files[i];//в fromfile ім'я файлу
            msg.tofile =  "cmplt_" + files[i];//в tofile ім'я для перейменування
            return msg;
        }
    }
}
```

Рисунок 3.27 – Код модифікованої функції find

3.3.5.8 Внесення в БД тільки тих штрих-кодів, які в ній відсутні

Імпортуйте вузол-функцію «Наявна в БД?» з цього посилання. Модифікуйте програму, як це показано на рис. 3.28. Вузол “Directory” перевіряє наявність файлу «barcods.txt» в директорії. Якщо файлу не існує, вузол «switch» направляє повідомлення тим саме шляхом, як і до модифікації програми. Якщо файл є, повідомлення ініціює вузол «Читання», який зчитує зміст БД. Вузол-функція «Наявна в БД» перевіряє, чи є запис в БД, і якщо ні направляє повідомлення стандартним шляхом. Якщо запис в базі існує, то на другому (не підключеному) виході формується повідомлення з об’єктом, зчитаним з БД. Цей вихід буде використовуватися в подальшому.

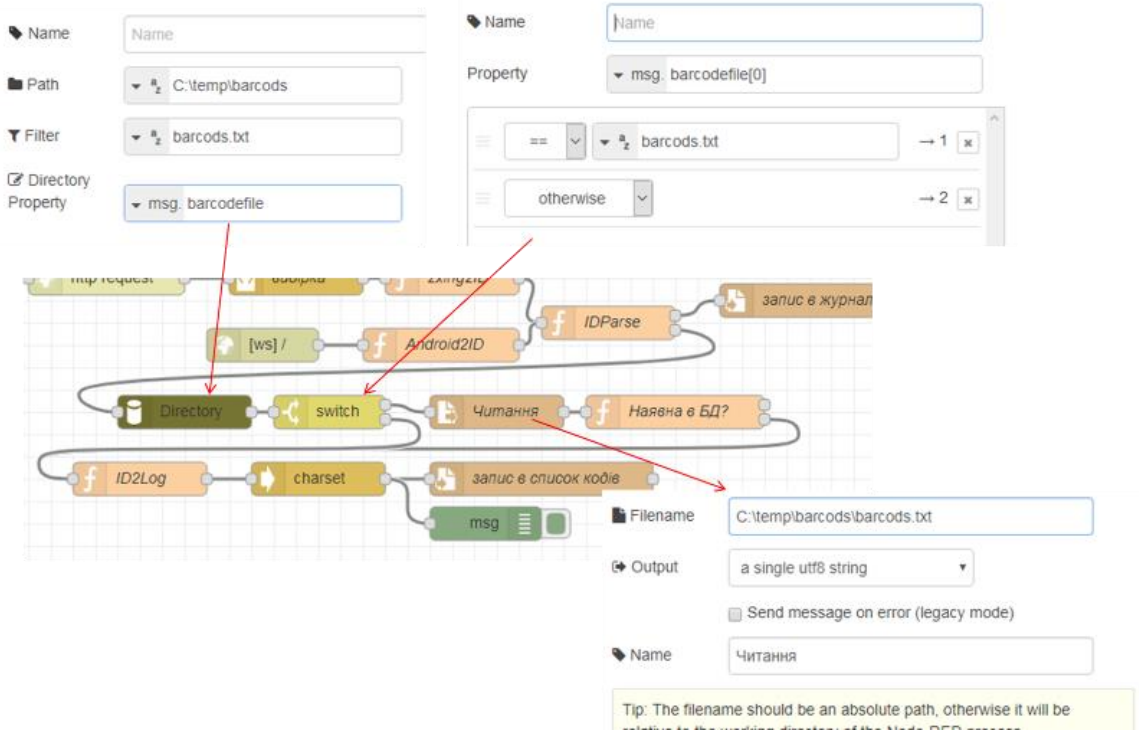


Рисунок 3.28 – Модифікація програми

Видаліть усі графічні та текстові файли з директорії «C:\Temp\barcods» . Скопіюйте усі графічні файли з директорії «C:\Temp\barcods\raw» в «C:\Temp\barcods». Зробіть розгортання потоку і запустіть на виконання. Після обробки всіх файлів, скопіюйте ще раз один з них, в БД не повинен з’явитися новий запис.

3.3.5.9 Виведення інформації на ВЕБ-інтерфейс при скануванні існуючого в БД штрих-коду

Імпортуйте вузли для ВЕБ-інтерфейсу з цього посилання. Модифікуйте програму як показано на рис. 3.29. Зробіть розгортання.

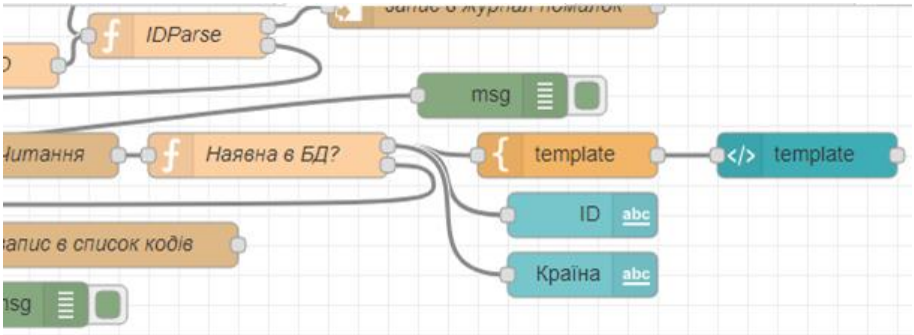


Рисунок 3.29 - Виведення інформації на ВЕБ-інтерфейс при скануванні існуючого в БД штрих-коду

Відкрийте Веб-інтерфейс на закладці «Штрих-коди». Скопіюйте один із файлів зображень штрих-коду що є в БД в директорію «C:\Temp\barcods» або зробіть його скан з мобільного застосунку. На Веб-інтерфейсі повинен з’явитися його код, та країна походження. Повторіть те саме з іншим файлом.

3.3.5.10 Ручна модифікація БД та виведення додаткової інформації на ВЕБ-інтерфейс при скануванні існуючого в БД штрих-коду

Для двох або більше штрих-кодів із файлу «barcods.txt» знайдіть назви, опис та посилання на файл PDF з Інтернету. Заповніть ці поля в БД, використовуючи Notepad++. Наприклад:

"3606480071058":{"Country":"Франція","sourceID":"C:\\temp\\barcods\\IMG_20190110_124139.jpg","sourcedoc":"http://elprivod.nmu.org.ua/ua/student/techdoc/pch/atv12_%D1%80%D1%83%D0%BA_%D0%BF%D0%BE%D0%BB%D1%8C%D0%B7.pdf","name":"ATV12H037M2","descr":"variable speed drive ATV12 - 0.37kW - 0.55hp - 200..240V - 1ph - with heat sink","inputdate":"2019-1-11 23:34:54"}

"3389110146608":{"Country":"Франція","sourceID":"C:\\temp\\barcods\\IMG_20190110_124417.jpg","sourcedoc":"https://www.alliedelec.com/m/d/68afa7d7064e3bef2fc703440bf8f780.pdf","name":"XS630B1PAL2","descr":"inductive sensor XS6 M30 - L62mm - brass - Sn15mm - 12..48VDC - cable 2m","inputdate":"2019-1-11 23:35:24"}

Збережіть зміни. Скопіюйте файл з зображенням або відскануйте його мобільним застосунком Wireless Barcode Scanner. Дочекайтеся коли його буде оброблено, після чого на Веб-інтерфейсі повинен з’явитися документ, який було вказано в БД. Зробіть те саме з іншим файлом.

3.3.5.11 Самостійні завдання (необов’язкове для виконання)

1. Виведіть на Веб-інтерфейс всю інформацію про товар.
2. Наведена програма має певні вади. Зокрема, вона не класифікує штрих-коди по типу, тому невірно в ряді випадків визначає

країну походження. Зробіть модифікацію так, щоб країна походження визначалася тільки для кодів EAN-13. Можна також класифікувати типи штрих-кодів та обробляти їх окремо.

3.4 Питання для самоперевірки

1. Розкажіть про протокол MQTT.
2. Що таке брокер MQTT?
3. Навіщо в роботі використовувався брокер test.mosquitto.org
4. Що таке тема повідомлення?
5. Який синтаксис формування теми повідомлення?
6. Розкажіть про фільтр при підписці на повідомлення.
7. Розкажіть про спеціальні теми брокеру.
8. Які вузли Node-RED і як налаштовуються для роботи з MQTT?
9. Як засоби використовувалися в практичній роботі для мобільних застосунків при роботі з MQTT?
10. Розкажіть про основи роботи протоколу HTTP.
11. Які інструменти використовувалися в практичній роботі для роботи з HTTP?
12. Розкажіть про призначення заголовків HTTP. Які заголовки використані в практичній роботі?
13. Які методи HTTP і для чого використовувалися в практичній роботі?
14. Які вузли і яким чином використовувати для побудови клієнта HTTP в Node-RED?

3.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40.* URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).
2. Node-RED українською : довідник з Node-RED українською мовою. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).
3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ: НУХТ, 2021. 170 с.
4. Лабораторна робота №3. Протоколи IoT: Використання WEB API та Web-сокетів. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab2_2.md (дата звернення: 09.07.2024).
5. Протокол HTTP та WEB API. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B5%D0%BA%D1%86/HTTP API.md (дата звернення: 09.07.2024).

ПРАКТИЧНА РОБОТА №4. ІНТЕГРУВАННЯ З ЗАСТОСУНКАМИ GOOGLE ТА СТВОРЕННЯ ТЕЛЕГРАМ-БОТА

4.1 Мета

Метою практичної роботи є інтеграція Node-RED з Google Sheets та Telegram.

4.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:

– створення та налаштування сервісного аккаунту Google (для доступу іншого сервісу);
– інтегрування з застосунками Google;
– створення Телеграм-бота;

4.3 Хід роботи

У цій практичній роботі необхідно забезпечити інтегрування Node-RED з хмарним застосунком Google Sheet (Електронна таблиця) та сервісом Telegram. Це дасть можливість забезпечити передачу даних з RaspberryPI на хмару для аналізу та взаємодію з віддаленим користувачем. Node-RED можна запускати як на ПК так і на RPI.

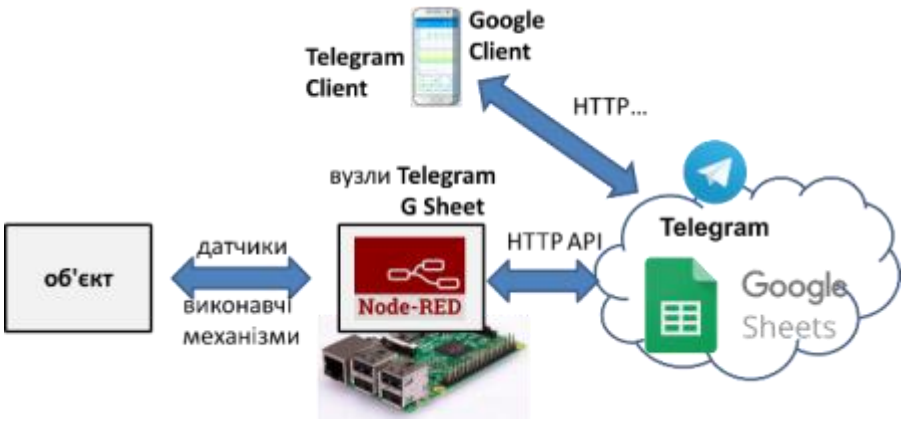


Рисунок 4.1 – Структура системи з використанням хмарних застосунків

4.3.1 Створення та налаштування сервісного аккаунту Google (для доступу іншого сервісу)

4.3.1.1 Створення облікового запису Google

Якщо у Вас немає облікового запису Google - створіть його на сайті. Це безкоштовно, потребується тільки поштова скринька і номер телефону.

4.3.1.2 Створення сервісного аккаунту Google

Зайдіть на сторінку налаштування API для сервісів, якими Ви користуєтеся на GoogleCloud https://console.cloud.google.com/apis.

При першому входженні Вам запропонують прийняти умови використання. Для того, щоб користуватися сервісами виставте опцію "Приймаю умови використання" після чого натисніть кнопку "Прийняти і продовжити"

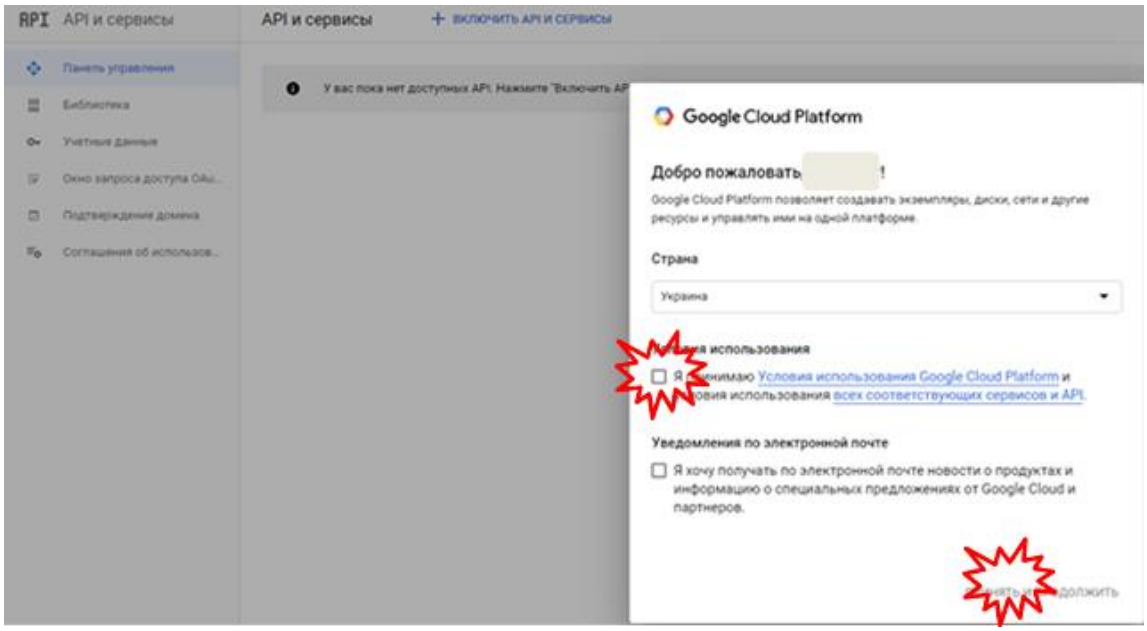


Рисунок 4.2 – Прийняття умов користування

4.3.1.3 Створення проекту

Натисніть "Створити проект"

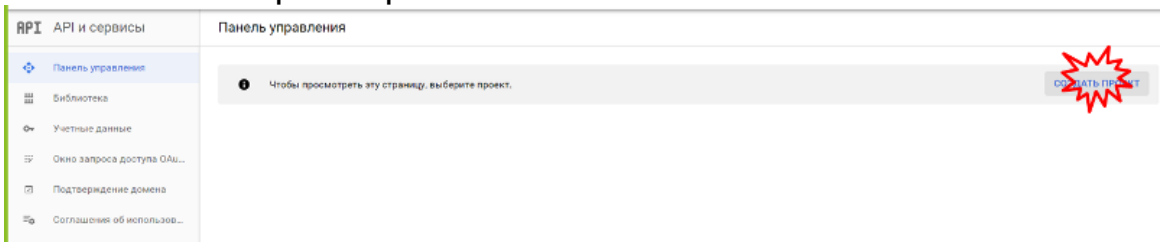


Рисунок 4.3 - Створення проекту

4.3.1.4 Налаштування проекту

Заповніть налаштування проекту, як показано на рисунку.

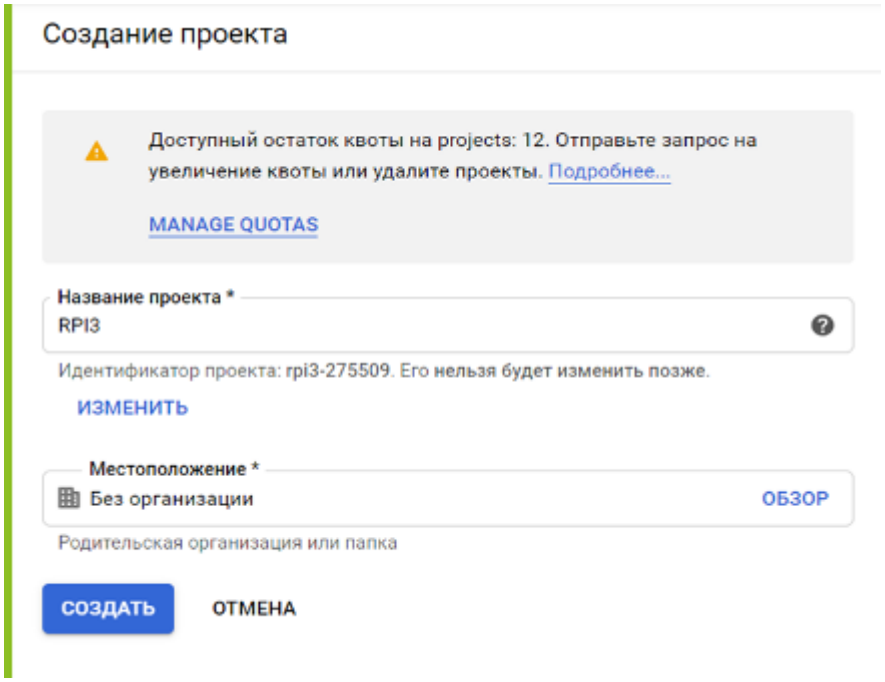


Рисунок 4.4 - Створення проекту (назва)

4.3.1.5 Створення сервісного аккаунту

Зайдіть в головне меню і перейдіть в розділ створення сервісних аккаунтів.

Натисніть "Створити сервісний аккаунт".

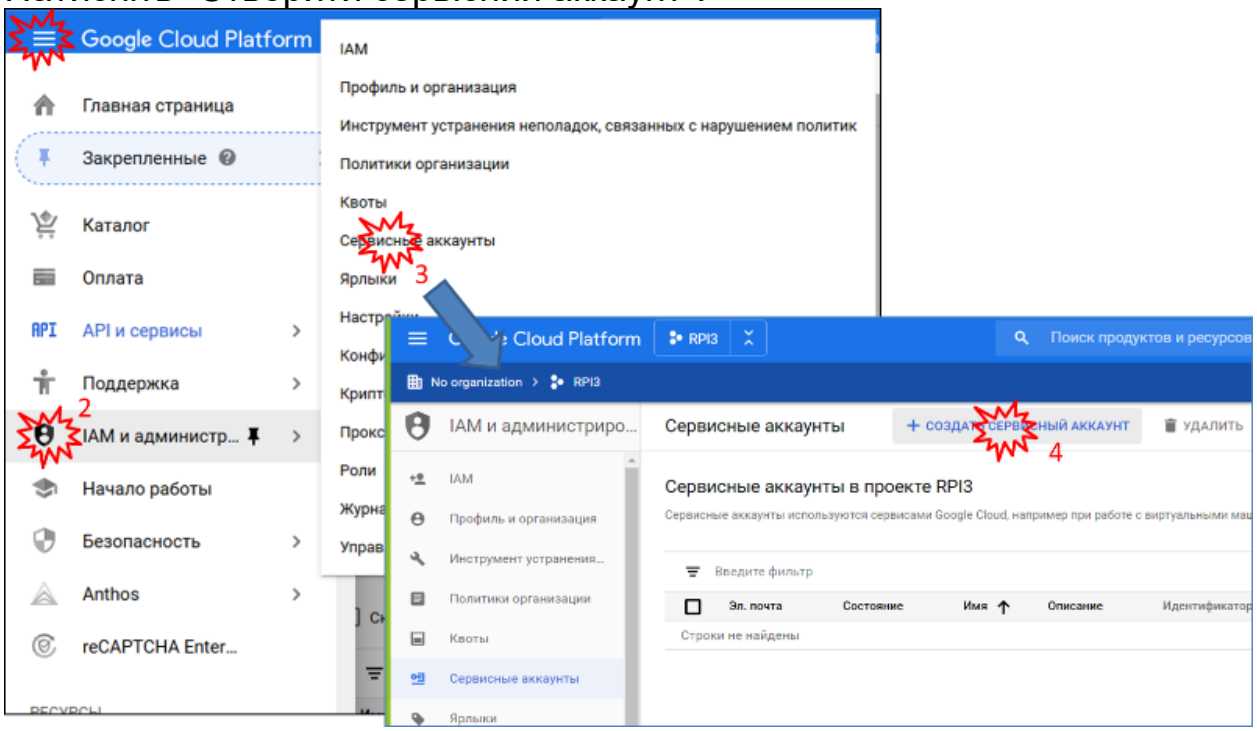


Рисунок 4.5 – Створення сервісного акаунту

На першому кроці означте назву

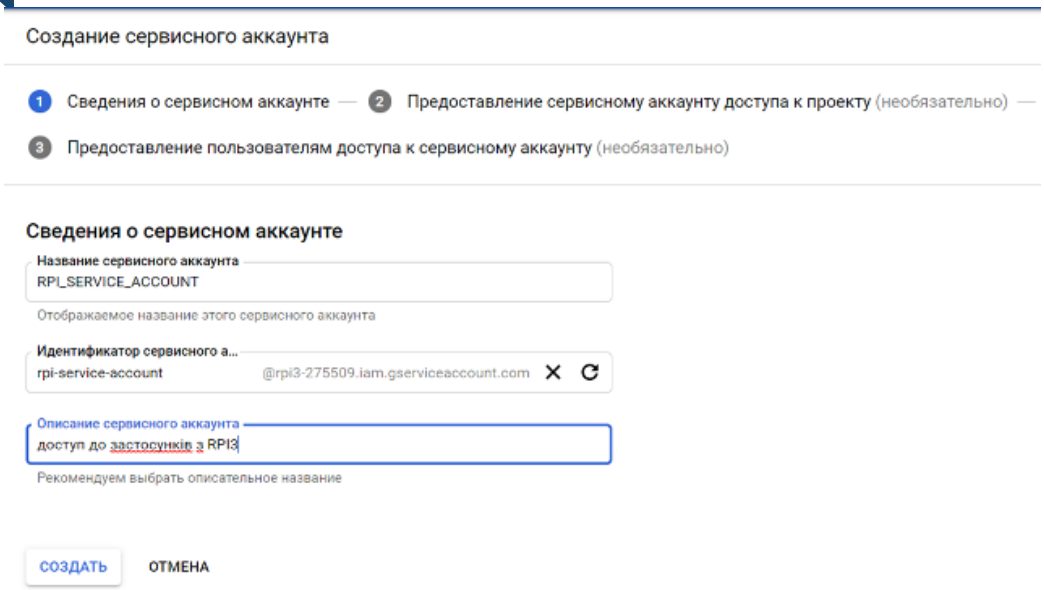


Рисунок 4.6 - Створення сервісного акаунту (назва)

На другому кроці можна поки нічого не вписувати, а просто натиснути "Продовжити".

На третьому кроці кроці виберіть "Створити ключ", після чого у новому вікні виберіть "Створити".

Система запропонує відкрити або зберегти файл JSON. Збережіть цей файл, він Вам потребується в майбутньому.

Після збереження, натисніть "Готово".

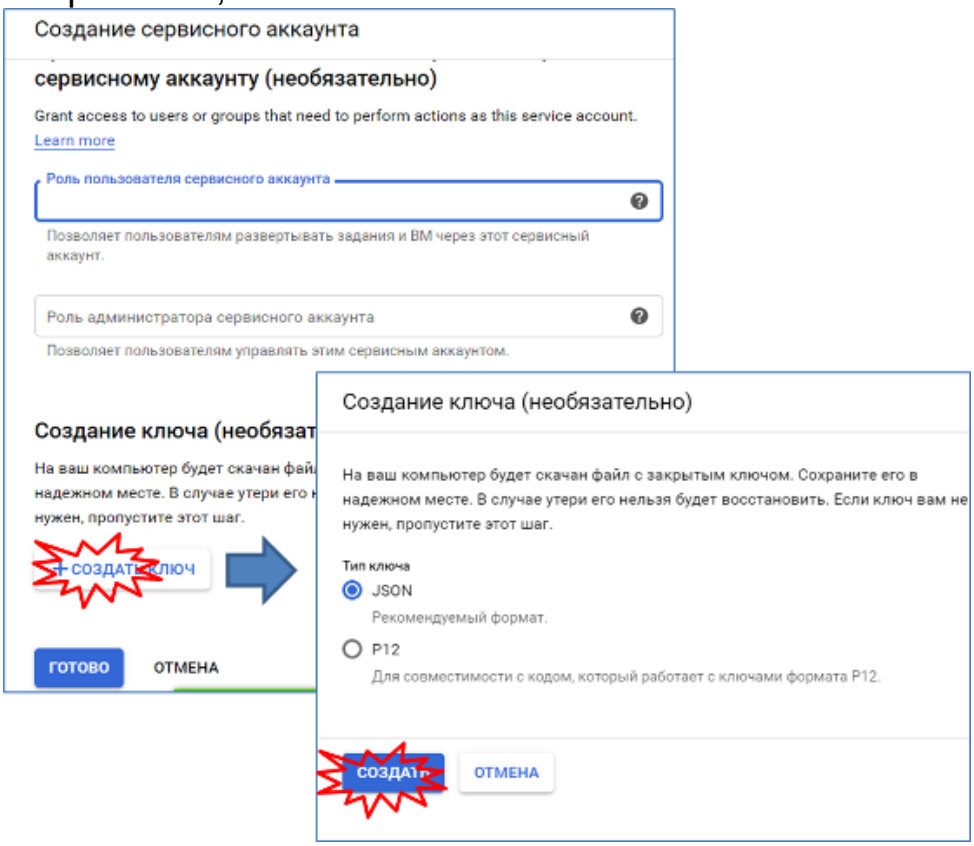


Рисунок 4.7 - Створення ключа сервісного акаунту

4.3.1.6 Надання доступу до сервісів Google Sheet

Перейдіть на сторінку з сервісами.

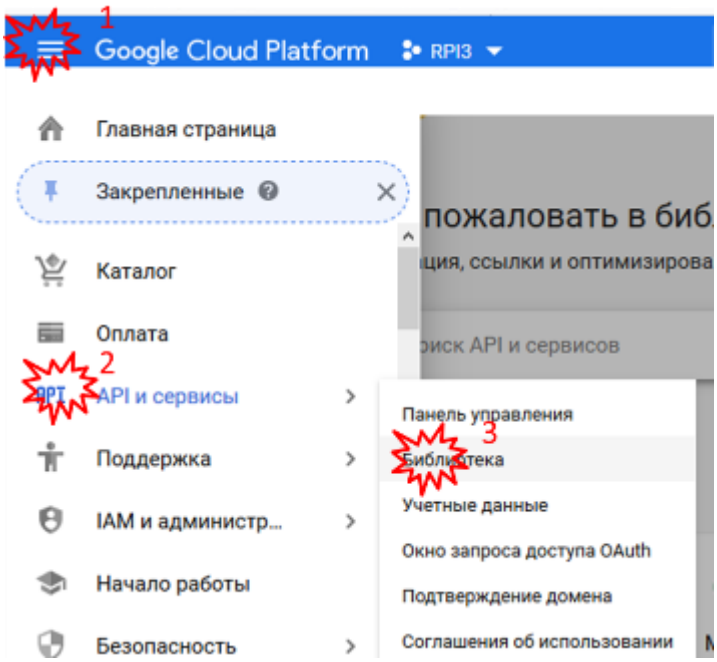


Рисунок 4.8 – Активація доступу до сервісів Гугл

Зайдіть в розділ G Suite.
Знайдіть Google Sheets.

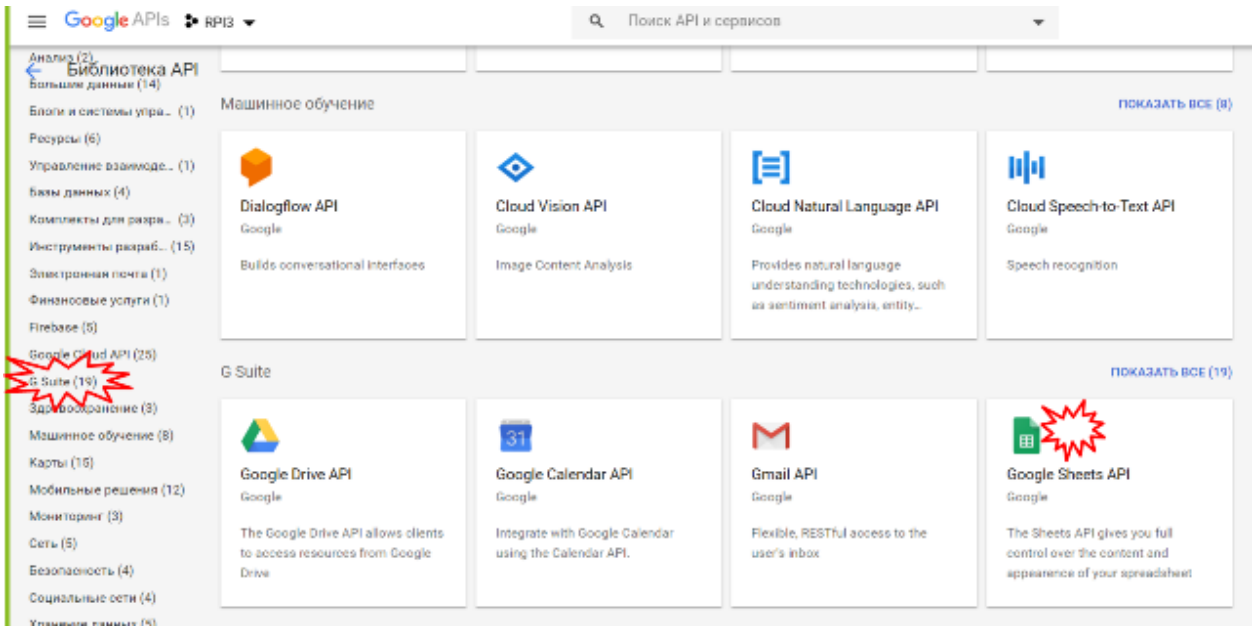


Рисунок 4.9 – Вибір хмарного сервісу

Натисніть "Включити" для активації доступу до цього сервісу.

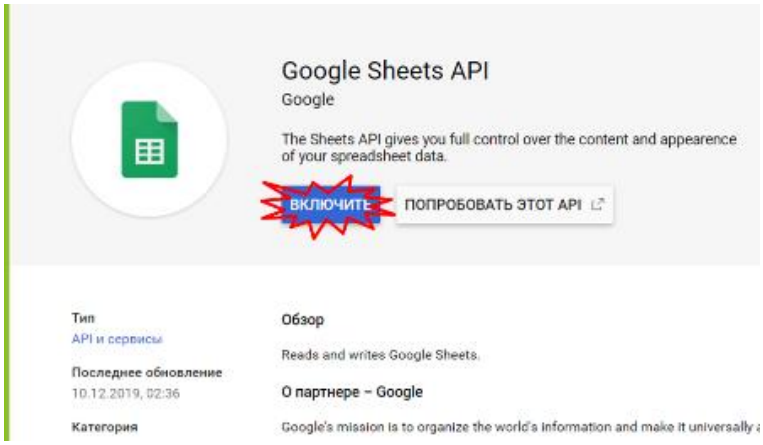


Рисунок 4.10 – Включення активації сервісу

4.3.2. Інтегрування з застосунками Google

У даній частині практичної роботи необхідно забезпечити запис значень з Node-RED в електронну таблицю, що обслуговується хмарним електронним застосунком Google Sheet.

4.3.2.1 Встановлення бібліотеки node-red-contrib-google-sheets

- запустіть Node-RED
- встановіть бібліотеку node-red-contrib-google-sheets

4.3.2.2 Створення таблиці Google Sheet

Google Sheet - це хмарний застосунок від Google для роботи з електронними таблицями. За функціональністю і принципами роботи він схожий на Microsoft Excel. Усі створені таблиці зберігаються на Гугл Диску (Google Drive).

Зайдіть на головну сторінку Google і зайдіть в застосунок Google Sheet(Таблиці).

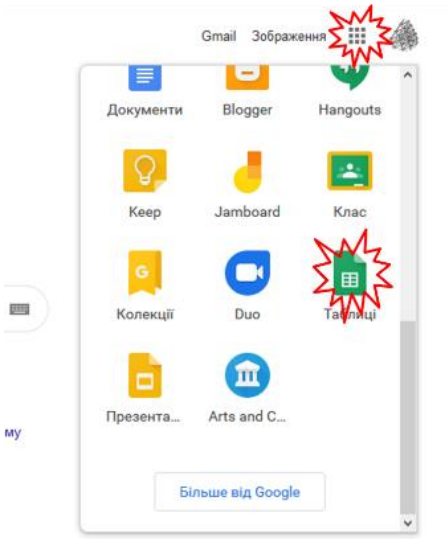


Рисунок 4.11 – Створення таблиці Google Sheet

Альтернативно можна відразу перейти на сторінку

У новому вікні натисніть кнопку "+" (створити) щоб створити нову електронну таблицю

Змініть назву документу на якусь більш прийнятну, наприклад RPIData

4.3.2.3 Створення фрагменту для наповнення буферу

У даному пункті необхідно створити фрагмент, який буде вміщувати буфер останніх 60-ти значень імітованих змінних rad та val а також дати та часу їх зміни. Такий буфер можна організувати різним чином, однак для спрощення були використані властивості масивів в JavaSript як черг та стеків. Кожне нове обчислення записується на верх масиву. Таким чином, спочатку "буфер-масив" буде наповнюватися аж до 60-го елементу. Коли елементів стане більше ніж 60 (тобто 61), нижні елементи виймаються, і масив "зсувається" вниз. І так кожного разу при виклику функції. Для збереження даних масиву між викликами використовується контекст потоку.

Про всяк випадок, зроблена також перевірка на переповнення масиву: коли кількість елементів повинна бути 60, а вона все одно більша - масив обрізається до 60-ти елементів. Така ситуація не повинна відбуватися, але бажано передбачати такі випадки.

- Запустіть Node-RED, якщо він не запущений
- Створіть новий потік з назвою clouds.
- Додайте туди фрагмент, який наведений на рисунку, код функції наведений під рис. 3.12.

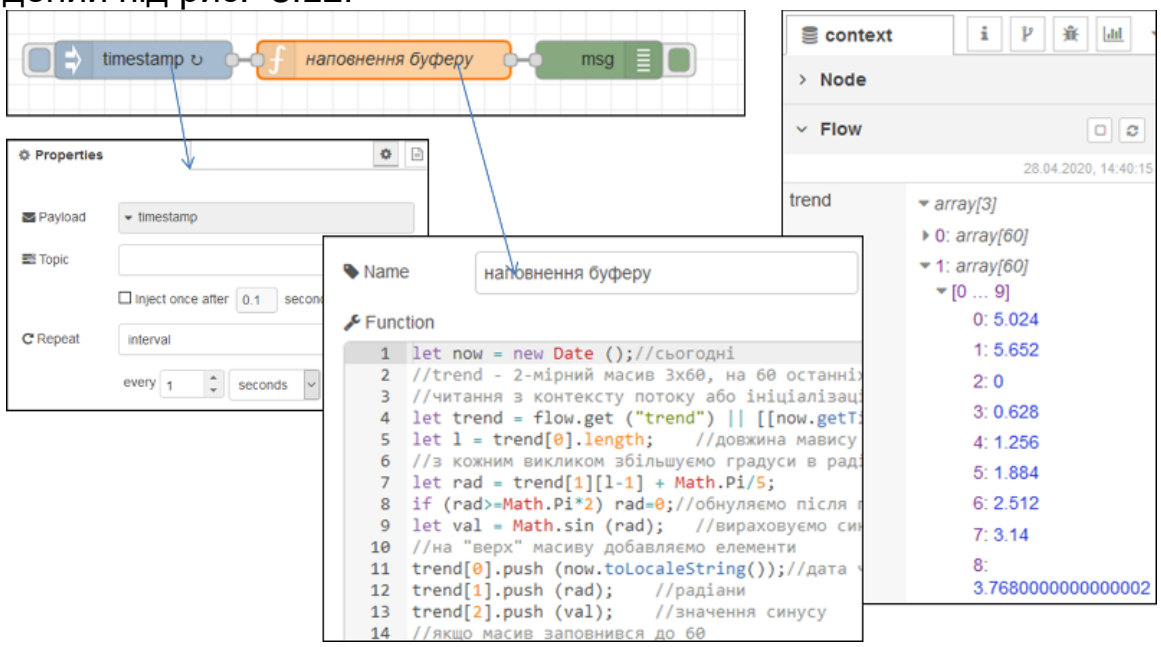


Рисунок 4.12 – Фрагмент програми для запису даних в масив

let now = new Date ();//сьогодні
//trend - 2-мірний масив 3x60, на 60 останніх значень

```
//читання з контексту потоку або ініціалізація масиву-буферу
let trend = flow.get ("trend") || [[now.toLocaleString()],[0],[0]];
let l = trend[0].length;    //довжина мавису
//з кожним викликом збільшуємо градуси в радіанах
let rad = trend[1][l-1] + Math.PI/5;
if (rad>=Math.PI*2) rad=0;//обнуляємо після повного кола
let val = Math.sin (rad);   //вираховуємо синус
//на "верх" масиву добавляємо елементи
trend[0].push (now.toLocaleString());//дата час
trend[1].push (rad);    //радіани
trend[2].push (val);    //значення синусу
//якщо масив заповнився до 60
if (trend[0].length >60){
    trend[0].shift ();//вилучаємо перший (найстаріший) елемент
    trend[1].shift ();
    trend[2].shift ();
    //після цього масиви повинні зменшитися на 1 елемент (60)
    //і зсунутися вниз
    //у випадку, якщо раптом елементів більше 60
    //наприклад були добавлені випадково стороннім кодом
    //зробити кількість елементів =60
    trend[0].length = 60;//
    trend[1].length = 60;
    trend[2].length = 60;
}
flow.set ("trend", trend);//записати в контекст потоку
return msg;
```

Зробіть розгортання потоку

Проаналізуйте контекст потоку, як він наповнюється з кожною секундою, він повинен мати вигляд як на рис. 4.12 (не забудьте натискати кнопку оновлення)

4.3.2.4 Створення фрагменту запису в електронну таблицю значень буферу

Користуючись рекомендаціями щодо роботи з базовими функціями читання та запису Google Sheet :

Надайте доступ до Гугл таблиці створеному раніше сервісному аккаунту.

Додайте наступний фрагмент потоку.

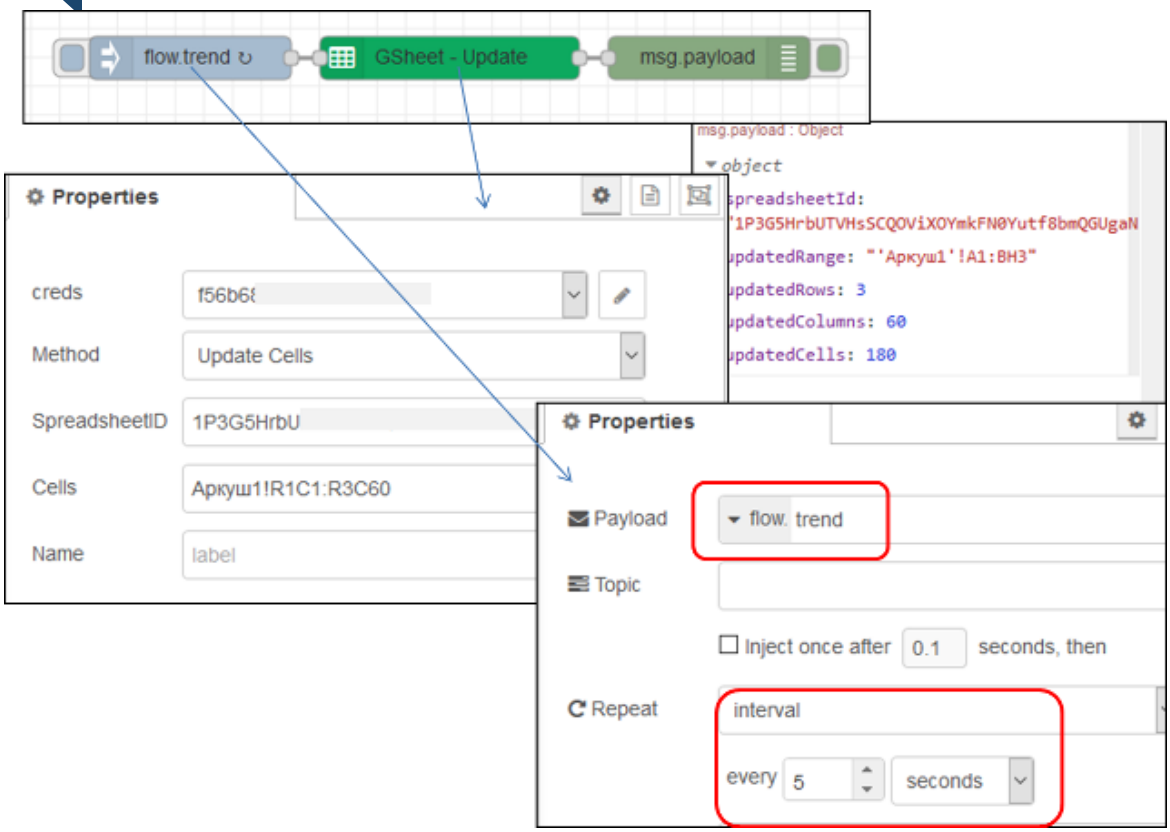


Рисунок 4.13 – Фрагмент запису в електронну таблицю значень буферу

Зробіть розгортання потоку

Перейдіть до Гугл таблиці там повинні відображатися дані з буфера і кожні 5 секунд оновлюватися.

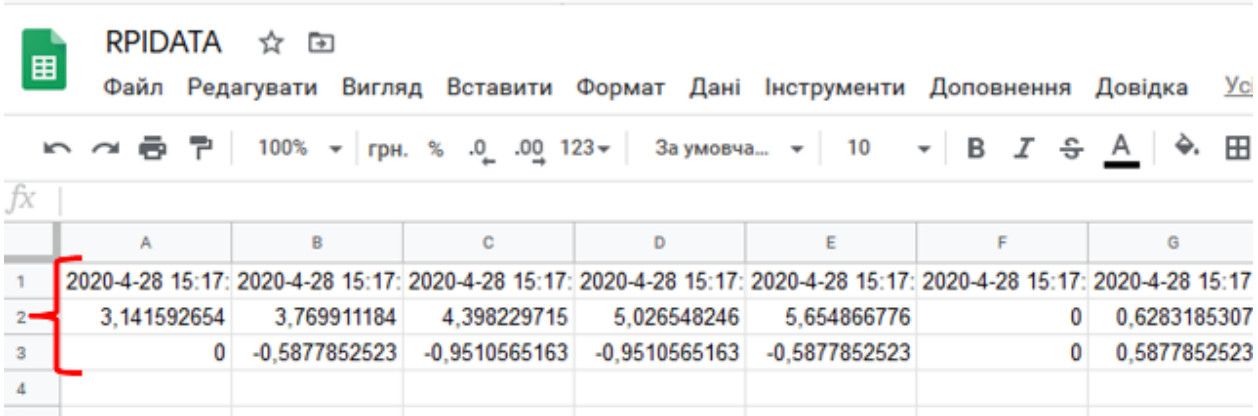


Рисунок 4.14 – Таблиця з записаними даними

4.3.2.5 Створення діаграми

Виділіть три рядки з даними і створіть по ним діаграму залежності значень змінних від часу

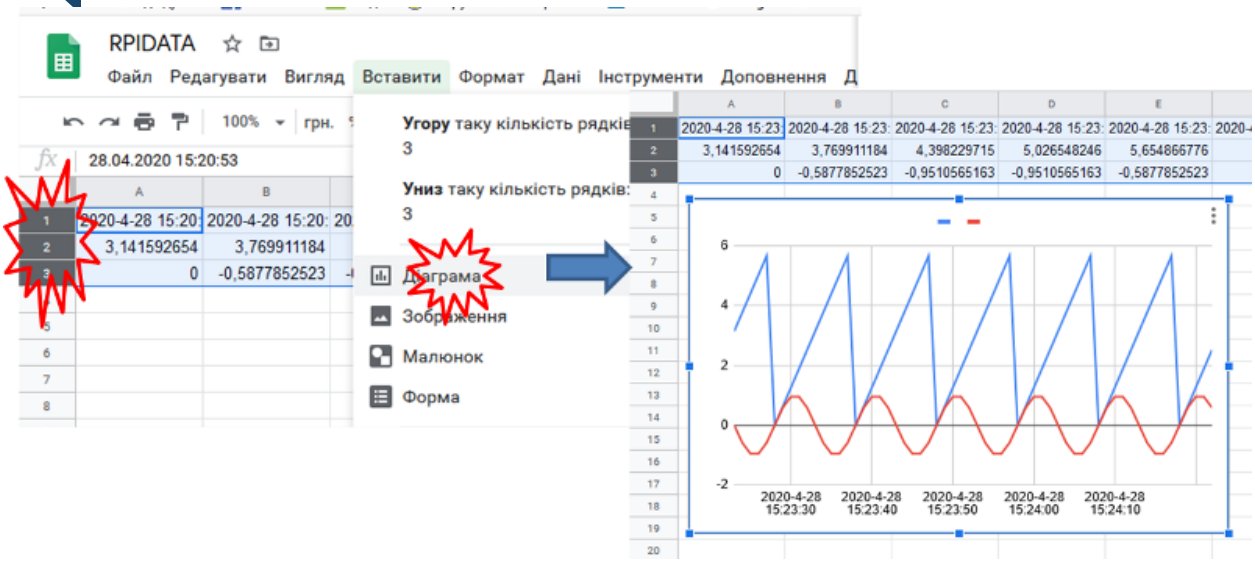
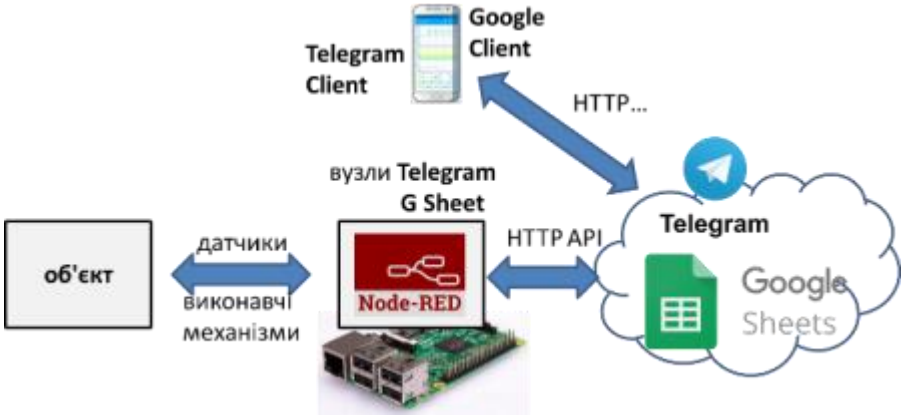


Рисунок 4.15 – Відображення діаграми за записаними в Google Sheet даними

4.3.3 Створення Телеграм-бота.

У цій частині практичної роботи необхідно реалізувати Телеграм-бота, який буде в онлайн режимі забезпечувати зв'язок користувача з RPI.



4.3.3.1. Створення облікового запису Telegram

Якщо у Вас немає облікового запису Telegram - завантажте клієнтський застосунок і створіть обліковий запис за посиланням. Це безкоштовно, потребується тільки номер телефону.

4.3.3.2. Реєстрація нового телеграм-бота.

Знайдіть в телеграмі і додайте до своїх контактів @BotFather - це бот, який створює ботів і введіть команду /start з'явиться вікно з

доступними командами

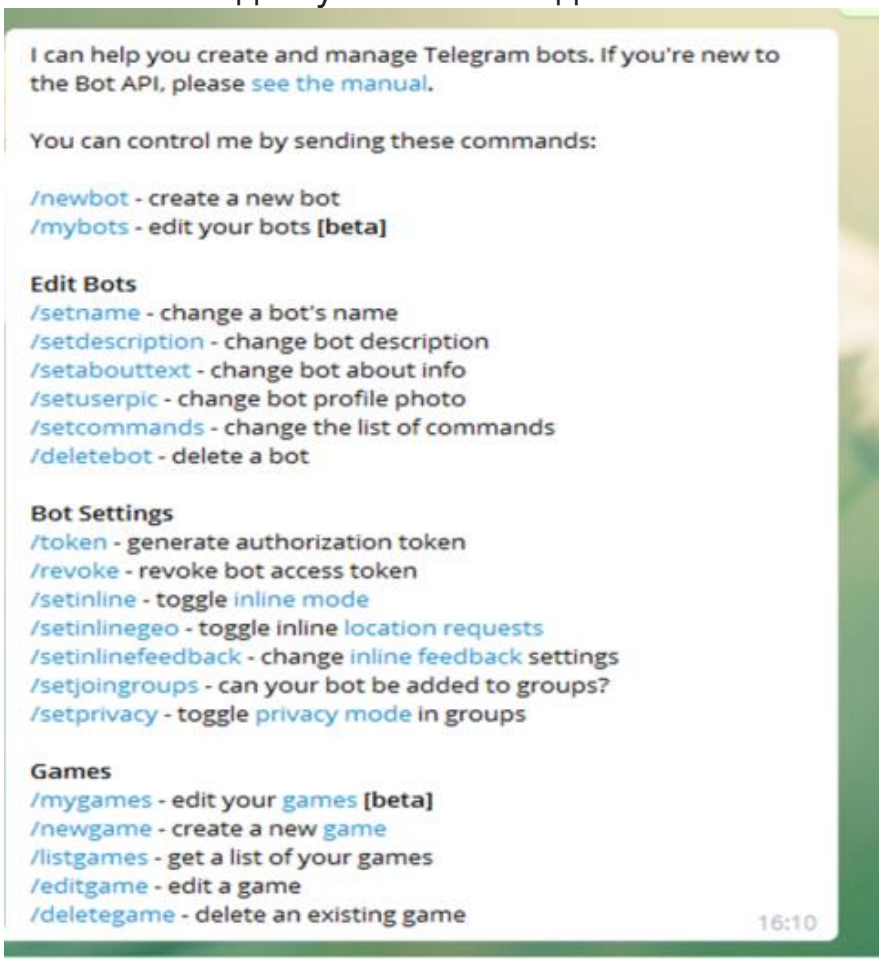


Рисунок 4.16 - Доступні команди створення телеграм-бота

Напишіть (або натисніть) команду /newbot для створення нового бота.

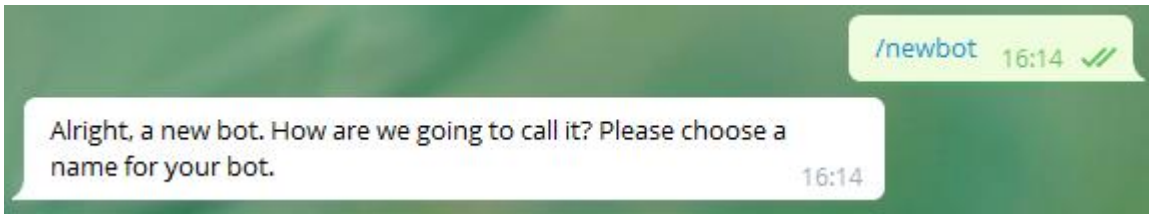


Рисунок 4.17 - Команда створення телеграм-бота

Додайте йому ім'я, наприклад RPI Ivanenko Ivana

Рисунок 4.18 - Надання імені телеграм-боту

Тепер треба ввести ім'я користувача для бота, який повинен закінчуватися на Bot , наприклад

RPI_IvanenkoIvana_Bot За допомогою цього нікнейма можна знайти й додати вашого бота до своїх контактів.

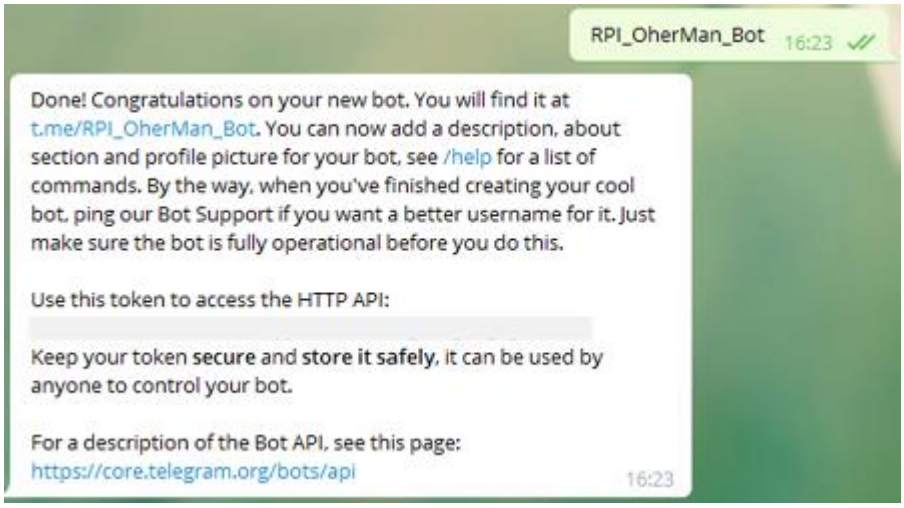


Рисунок 4.19 - Надання імені користувача телеграм-боту

Збережіть токен, який видав бот, бо він нам знадобиться згодом для керування.

Додайте бота до своїх контактів t.me/<username>.

За допомогою RPI_IvanenkoIvana_Bot ви зможете завжди відредагувати своїх ботів.

4.3.3.3 Встановлення бібліотеки в Node-RED

Запустіть Node-RED, встановіть бібліотеку node-red-contrib-telegrambot

4.3.3.4 Створення першого варіанту бота

Ознайомтеся з описом бібліотеки node-red-contrib-telegrambot створіть в Node-RED новий потік з іменем bot створіть потік, як показано на рис. 4.20; з правилами заповнення конфігураційного вузла ознайомтеся в описі

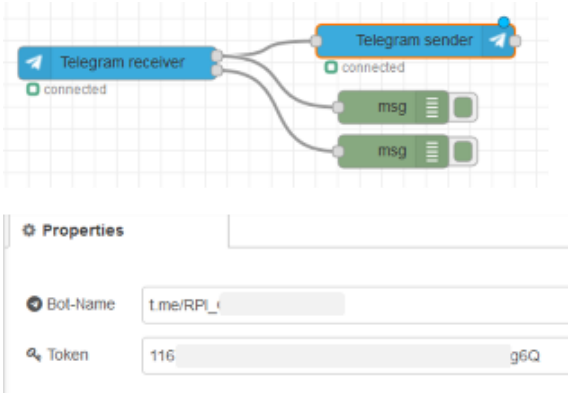


Рисунок 4.20 - Перший варіант бота в Node-RED

4.3.3.5 Перший запуск бота

Зроблений потік робить ехо-відповіді на будь яке повідомлення в приватному чаті.

Зробіть розгортання потоку.

Зайдіть в Telegram додайте контакт свого бота, якщо ще не добавили.

Зайдіть в чат і натисніть кнопку Розпочати або команду /start.

Якщо бот працює, Вам повинна повернутися та сама команда.

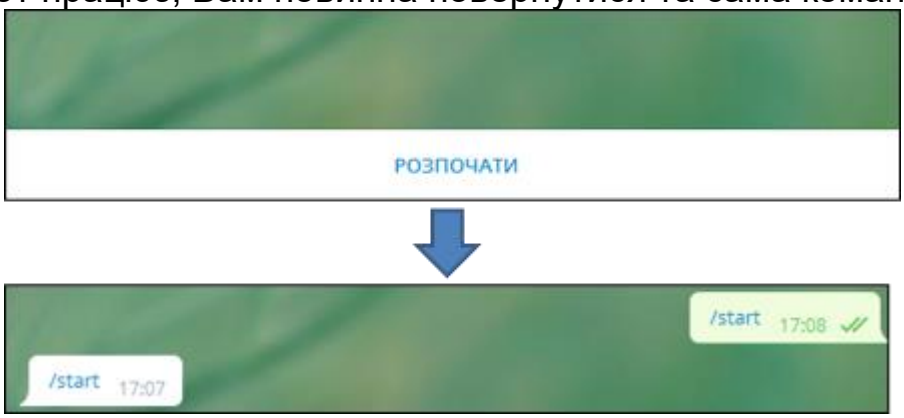


Рисунок 4.21 - Перший запуск бота – перевірка

4.3.3.6 Аналіз отриманого повідомлення

У вікні налагодження перегляньте структуру повідомлення (рис.4.22).

Визначте chatId він вам знадобиться для відправки повідомлення до вказаного чату.

Рисунок 4.22 - Аналіз отриманого повідомлення

4.3.3.7. Формування стартового повідомлення допомоги

У подальшому бот буде використовуватися в діалозі тільки з використанням команд. Перша команда - це /start має виводити список доступних команд.

Видаліть попередній фрагмент потоку для exo-відповіді, створіть новий потік.

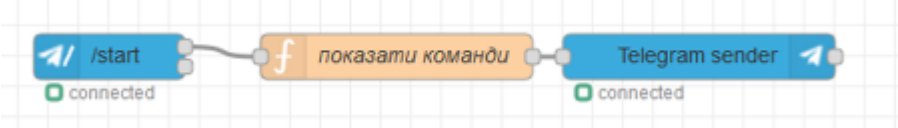


Рисунок 4.23 - Формування стартового повідомлення допомоги

Функція "показати команди має наступний вигляд"

```
let txtmsg = "/start - показати базові команди\r\n";
txtmsg += "/sp - виставлення уставок\r\n";
txtmsg += "/pv - отримання плинних значень\r\n";
txtmsg += "/alm - список активних тривог\r\n";
txtmsg += "/rprt - сформувати звіт\r\n";
msg.payload.content = txtmsg;
return msg;
```

Зробіть розгортання потоку, у чаті наберіть команду /start, повинно вийти повідомлення з командами

Натисніть на команду /start - дія повинна повторитися, що показує, що команди в списку можна натискати

4.3.3.8. Відправка тестового повідомлення

Модернізуйте потік так, щоб можна було відправити текстове повідомлення у вказаний чат. Ідентифікатор chatId необхідно було визначити в пункті 4.3.2.5.

Зверніть увагу, що в inject повинен бути формат корисного навантаження {}JSON.

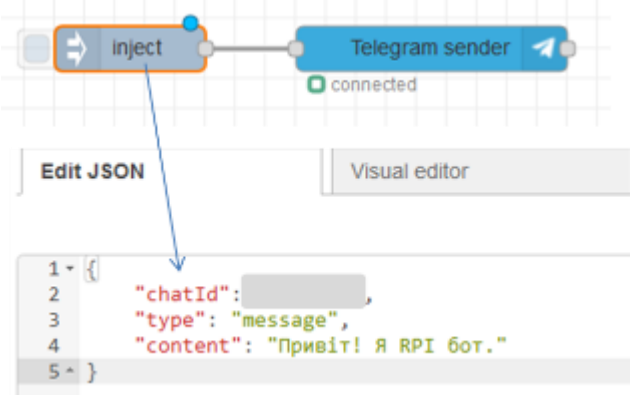


Рисунок 4.24 - Відправка тестового повідомлення

Згенеруйте відправку повідомлення в чат і проконтролюйте його отримання

4.3.3.9. Генерування тривог

У цьому пункті необхідно зробити формування повідомлення в Телеграм, коли значення змінних rad та val що імітувалися в попередній частині практичної роботи будуть вище заданих значень.

Модифікуйте функцію наповнення буферу з попередньої частини практичної роботи, вставивши фрагмент між flow.set ("trend", trend); та return

```
flow.set ("trend", trend);//записати в контекст потоку
//-------------------
let rtdb = { //глобальна змінна
    trend: trend,
    rad: rad,
    val: val
}
global.set ("rtdb", rtdb);
//--------------------
return msg;
```

Цей фрагмент записує буфер, та значення змінних в глобальний контекст.

Додайте до потоку bot наступний фрагмент:

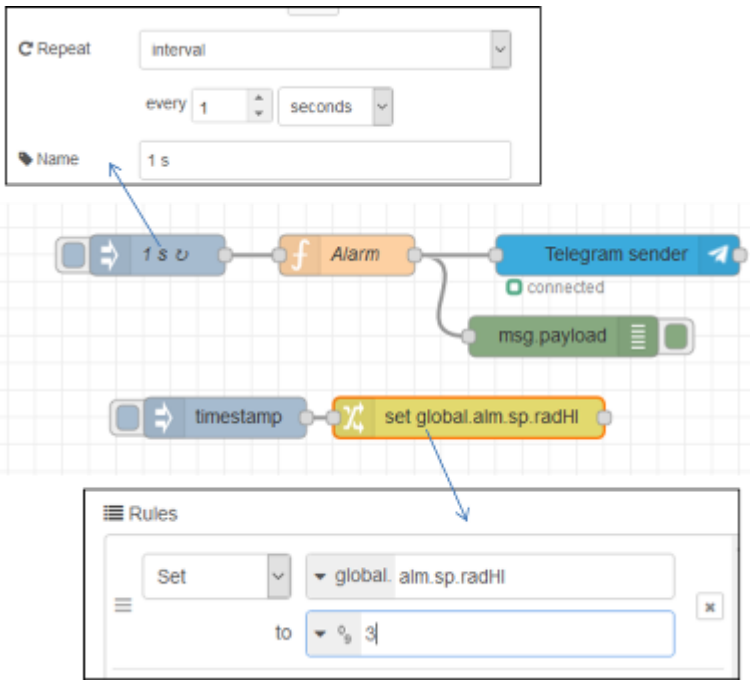


Рисунок 4.25 - Генерування тривог - програма

Функція Alarm має наступний зміст

```
let rtdb = global.get ("rtdb") || {};
let alm = global.get ("alm") || { //система тривог
   state:{radHI:false, valHI:false}, //стан тривог
   sp: {radHI:10.0, valHI:2.0}     //уставки тривог
   };

let almmsg = "";
//умова тривоги активна?
let alm_radHI = rtdb.rad > alm.sp.radHI;
let alm_valHI = rtdb.val > alm.sp.valHI;
//якщо тривога тільки активувалася
if (alm_radHI && !alm.state.radHI) {
   almmsg += "Радіани дорвінюють " + rtdb.rad + " , що вище зданого значення " + alm.sp.radHI + "\r\n";
}
if (alm_valHI && !alm.state.valHI) {
   almmsg += "Синус дорвінює " + rtdb.val + " , що вище зданого значення " + alm.sp.valHI + "\r\n"
}
//запис у стан тривоги
alm.state.radHI = alm_radHI;
alm.state.valHI = alm_valHI;
//збереження станів тривоги в глобальному контексті
global.set ("alm", alm);
//якщо хоча виникла хоча б одна тривога - відправка повідомлень
if (almmsg.length > 1) {
   msg.payload = {chatId : #######, //тут має бути ваш ідентифікатор
            type : 'message',
            content : almmsg}
   return msg;
}
```

Вузол change потрібен для того щоб змінювати уставку для однієї з змінних, наприклад для rad. Ця уставка аварійно високого значення зберігається в глобальному контексті як alm.sp.radHI. Значення rad змінюється від 0 до 6.28 а val від -1 до 1. Таким чином, щоб згенерувати тривогу про високе значення, необхідно його вказати в цих межах.

Використовуючи фрагмент з вузлом timestamp та change сформуйте значення alm.sp.radHI рівним 3.

У результаті з певним періодом повинні генеруватися в чаті тривоги про перевищення значення.

Верніть alm.sp.radHI в значення вище 6.28 , щоб тривоги не генерувалися.

4.3.3.10. Виставлення уставок

У цьому пункті необхідно зробити зміну уставок з чатую Для цього використовується команда /sp яка формує клавіатуру з 3-ма кнопками:

- /rad - вибору уставки верхнього рівня для змінної rad , відправиться відповідна команда
- /val - вибору уставки верхнього рівня для змінної val, відправиться відповідна команда
- відміна - відмова від вибору

Далі при виборі змінної формується відповідна команда, яка обробляється окремим обробником, що просить ввести значення цих змінних уставок.

Додайте до потоку bot наступний фрагмент:

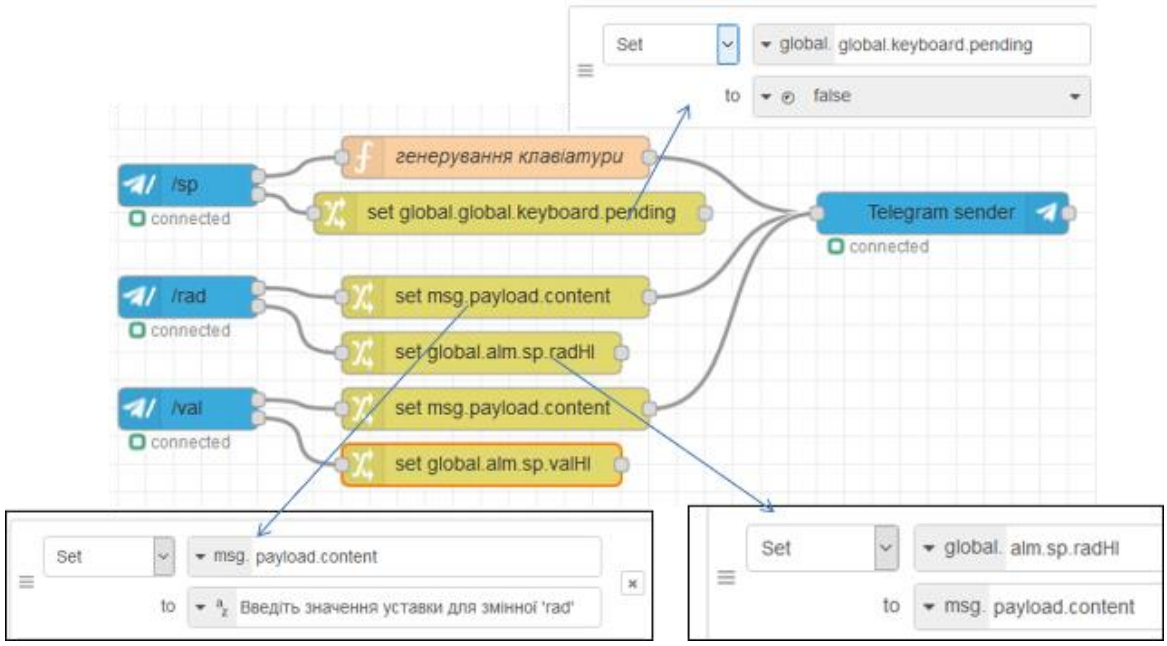


Рисунок 4.26 - Виставлення уставок - програма

Означте код функції генерування клавіатури, який наведений нижче.

```
context.global.keyboard = {pending : true};
let opts = {
    reply_to_message_id: msg.payload.messageId,
    reply_markup: JSON.stringify({
        keyboard: [
            ['/rad'],
            ['/val'],
            ['відміна']],
                'resize_keyboard' : true,
                'one_time_keyboard' : true
    })
};
msg.payload.content = 'Виберіть змінну';
```

```
msg.payload.options = opts;
return [msg];
```

Зробіть розгортання потоку.
Введіть команду /sp

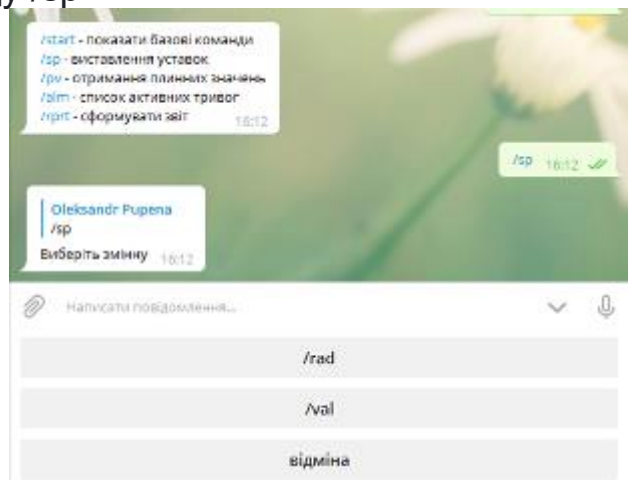


Рисунок 4.27 - Виставлення уставок – перевірка

Натисніть кнопку /rad.

Введіть значення 3.

Проконтролюйте, що тривоги будуть спрацьовувати, поверніть задане значення до 10.

Аналогічним чином змініть значення уставки для змінної /val (наприклад 0.5).

4.3.3.11. Отримання плинних значень

Реалізуйте самостійно команди:

- видачі активних значень змінних у форматі "назва - значення" у новому рядку;
- видачі списку активних тривог у форматі "назва тривоги" у новому рядку, якщо активних тривог немає - видати повідомлення "Немає активних тривог";
- команда на формування звіту може бути використана в курсовому проекті.

4.3.3.12. Тестування чат бота в приватному чаті одногрупника

Попросіть свого товариша протестувати Вашого чат-бота
Відправте ім'я чат-бота для перевірки викладачем

4.3.3.13. Тестування чат бота в загальному чаті (не обов'язкове для виконання)

Попросіть у викладача запрошення на бот-чат групу.
Відлагодьте ваш бот, щоб він правильно працював у групі.

4.4 Питання для самоперевірки

1. Назвіть основні причини, чому стандарти кібербезпеки в сфері IT не підходять для систем АСКТП.
2. Дайте визначення поняттю "кібератака".
3. Яка мета та причини кібератак?
4. Розкажіть про внутрішні та зовнішні загрози кібератак.
5. Що таке вектор атак. Назвіть приклади векторів атак?
6. Наведіть приклади вразливостей для кібератак в системі.
7. Навіщо потрібні політики безпеки?
8. Наведіть приклади вхідних вразливих точок для підприємства з інтегрованою системою керування, що має впроваджений зв'язок з Інтернет.
9. Що передбачає ідентифікація та контроль активів і персоналу?
10. Що передбачають архітектурні підходи до захисту?
11. Що таке зона безпеки і тракт?
12. Що таке демілітаризована зона?
13. Що передбачає захист фізичного доступу?
14. Назвіть мережні захисні компоненти.
15. Що передбачає керування доступом?
16. Що таке зміцнення конфігурацій?
17. Що дає шифрування повідомлень?
18. Розкажіть про стратегію захисту вглибину.
19. Прокоментуйте "13 хороших практик" для кіберзахисту АСКТП.

4.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40.* URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).
2. Node-RED українською : довідник з Node-RED українською мовою. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).
3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ : НУХТ, 2021. 170 с.
4. Лабораторна робота №4. Інтегрування з застосунками Google. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%

D0%B0%D0%B1%D0%BE%D1%80/lab5_1.md (дата звернення: 09.07.2024).

5. Лабораторна робота №5. Створення Телеграм-бота. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab5_2.md (дата звернення: 09.07.2024).

6. Використання хмарних сервісів для IoT. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B5%D0%BA%D1%86/cloud.md (дата звернення: 09.07.2024).

ПРАКТИЧНА РОБОТА №5. РОБОТА З OPC UA

5.1 Мета

Дослідити можливості та функціональність OPC UA для промислових застосувань. Навчитися встановлювати і налаштовувати OPC UA сервери та клієнти, аналізувати дані, використовувати методи для взаємодії з сервером, працювати з історичними даними та діагностикою серверів, а також підключатися до онлайн-серверів. Вивчити основи роботи з OPC UA клієнтами в середовищі Node-RED.

5.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:
- встановлення тестових утиліт для роботи з OPC;
- використання клієнта для тестування OPC UA Сервера;
- тестові клієнти для мобільних застосунків;
- основи роботи з клієнтськими запитами OPC UA в Node-RED.

5.3 Хід роботи

5.3.1 Встановлення тестових утиліт для роботи з OPC

5.3.1.1 Завантаження та встановлення тестових застосунків

Завантажте та встановіть OPC UA C++ Demo Server (під Winidows). Завантаження потребує реєстрації, але це цілком безкоштовно.
Завантажте та встановіть тестовий OPC UA Client UaExpert.

5.3.1.2 Перший запуск

Запустіть "OPC UA C++ Demo Server" та погодьтеся на внесення порту до списку дозволених брандмауером.
У консольному вікні серверу знайдіть запис з URL кінцевої точки.

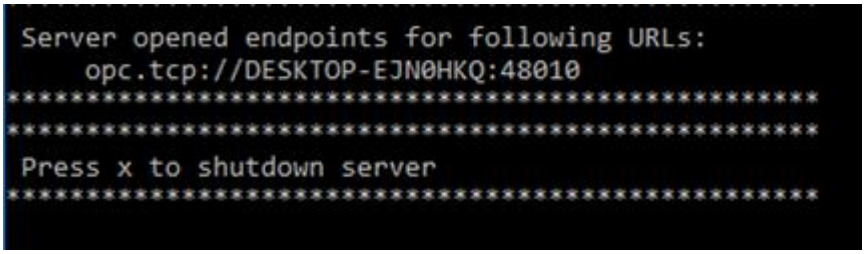


Рисунок 5.1 – Запис з URL кінцевої точки

Запустіть UaExpert, натисніть "+" (Add server).

Додайте тестовий OPC UA Server через меню Custom Discovery, шляхом копіювання URL, який наведений у консольному вікні серверу.

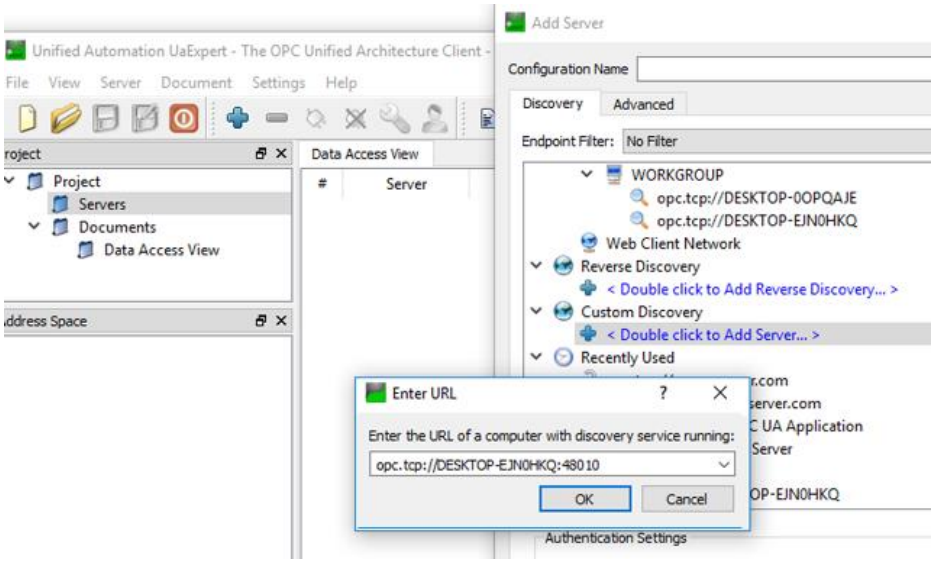


Рисунок 5.2 – Додавання тестового OPC UA Server

Виберіть підключення без захисту (None) і натисніть Ok.

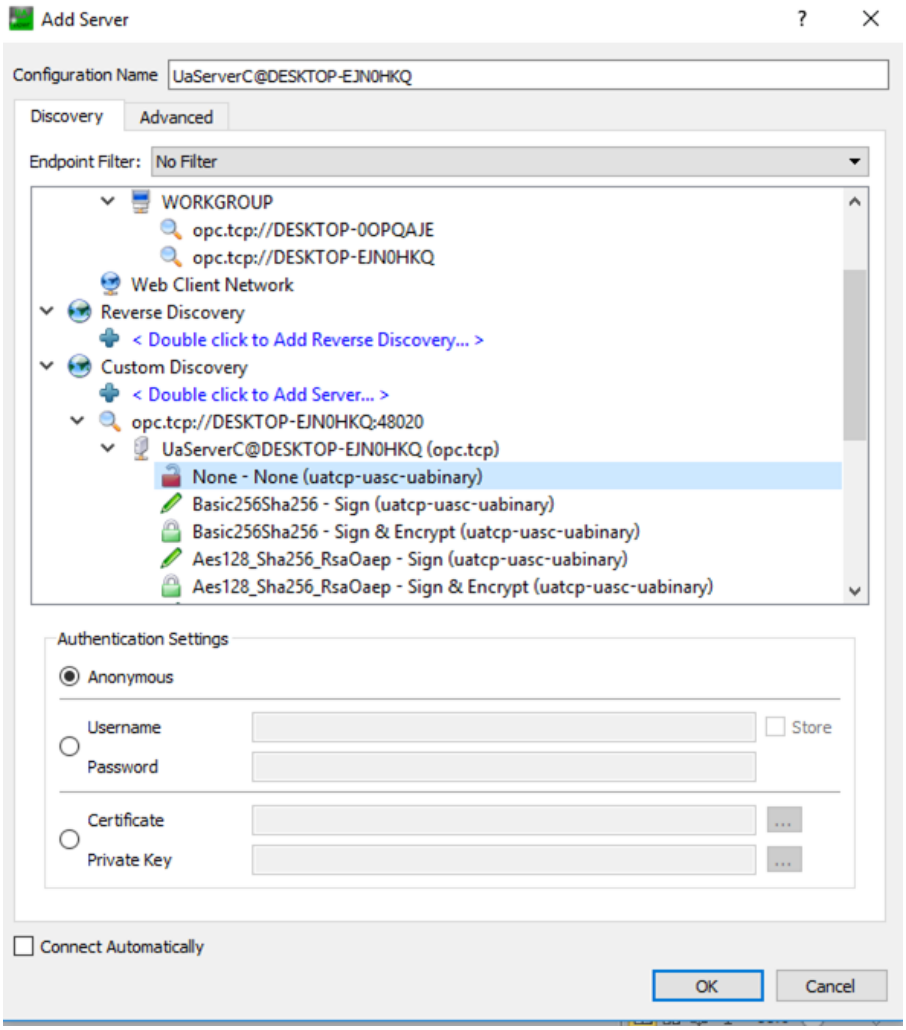


Рисунок 5.3 – Дерево тестового серверу

Для доданого серверу натисніть Connect.

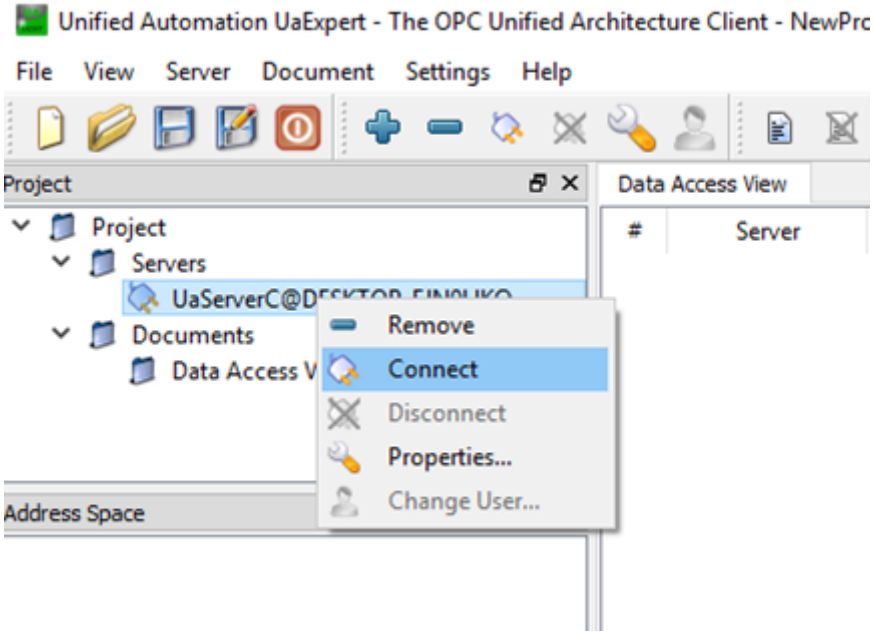


Рисунок 5.4 – Виконання зв'язку серверу

У вікні що з'явиться, виберіть Trust Server Certificate, після чого натисніть Continue.

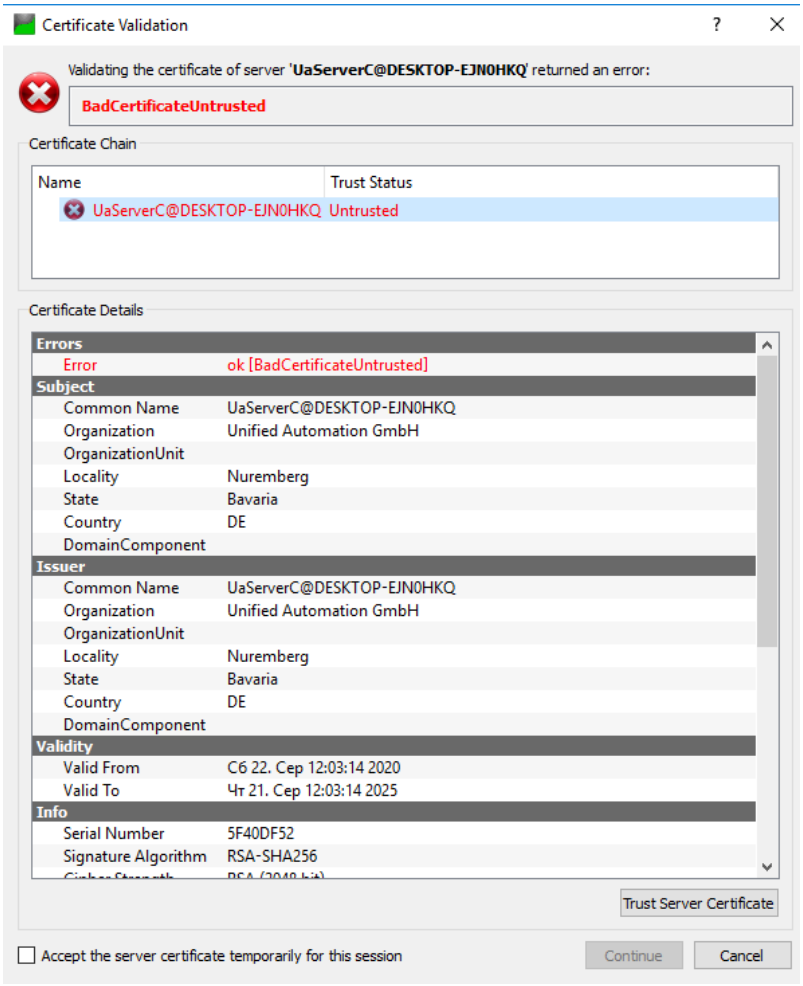


Рисунок 5.5 – Валідація сертифікату тестового серверу

5.3.1.3 Перегляд сертифікатів

Зайдіть в меню Settings->Manage Certifacates, передивіться сертифікат.

5.3.2 Використання клієнта для тестування OPC UA Сервера

5.3.2.1.Аналіз об'єктів-папок

Використовуючи UaExpert проаналізуйте зміст об'єктів першого (Root) та другого рівня (Objects, Types, Views), проаналізуйте:

- перелік та зміст атрибутів: NodeID, NodeClass;
- перелік та зміст посилань (References).

Проаналізуйте подібним чином ще одну з папок, вкладену в Objects.

5.3.2.2 Аналіз Variable

Відкрийте для перегляду об'єкт BuildingAutomation.AirConditioner_1

Знайдіть DataVariable Temperature проаналізуйте атрибути Value та DataType.

Проаналізуйте атрибути для властивостей AirConditioner_1.Temperature.EURange та AirConditioner_1.Temperature.EngineeringUnits

Проаналізуйте атрибути Value та DataTуредля наступних об'єктів:

- Demo.Static.Arrays.AnalogMeasurement
- Demo.Static.Arrays.Structure
- Demo.Static.Matrix.Float

5.3.2.3 Використання Data Access View

Якщо не відкрито вікно Data Access View , створіть новий документ такого типу.

Перетягніть AirConditioner_1 на вікно Data Access View , повинні помістится усі об'єкти, що знаходяться всередині.

Подвійним кліком по полю Value в Data Access View змініть значення TemperatureSetPoint рівною 75.

Через контекстне меню в Data Access View зайдіть в налаштування Subscription Settings, виставіть періодичність публікації рівною 4 секундам (4000). Перевірте, що значення змінюються не раніше ніж через 4 секунди.

Виставіть періодичність публікації рівною 1 секунді.

Через контекстне меню Temperature в Data Access View зайдіть в налаштування Monitoring Item Settings значення зони нечутливості 1 градус.

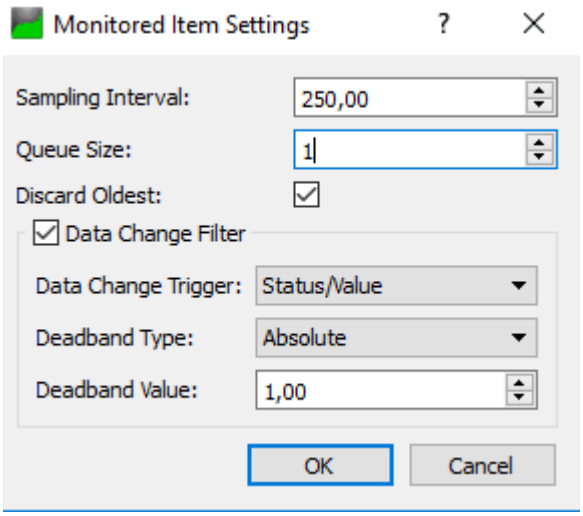


Рисунок 5.6 – Контекстне меню налаштування перегляду параметру

Змініть налаштування уставки TemperatureSetPoint рівною 70. Тепер значення температури повинно оновлюватися тільки при зміні на 1 градус.

Зробіть копію екрану для звіту.

5.3.2.4 Використання методів

Через контекстне меню метода AirConditioner_1.Stop виберіть Call і зупиніть установку кондиціонування.

Подивіться значення AirConditioner_1.StateCondition.Message та AirConditioner_1.State

Через контекстне меню метода AirConditioner_1.StartWithSetpoint виберіть Call і запустіть установку кондиціонування з уставкою температури = 25 градусів, і вологістю - 55%.

5.3.2.5 Використання History Trend View

Подивіться значення атрибуту Historizing для Temperature

Створіть документ History Trend View (Document->Add->History Trend View).

Використовуючи метод StartLogging запустіть реєстрацію в трендовий архів.

Подивіться значення атрибуту Historizing для Temperature . Тепер воно має бути TRUE, що значить, що дані для цієї змінної пишуться в історію.

Перетягніть AirConditioner_1.Temperature у вікно Configuration документа History Trend View

Виберіть Cyclic Update і натисніть Start:

- на закладці Numeric Value Ви побачите тренд

– перейдіть на закладку з іменем змінної та перегляньте як змінюється значення

Змініть Update Interval на 5 секунд і перезапустіть оновлення. Зверніть увагу, що через 5 секунд підтягуються усі точки для графіку.

Зробіть копію екрану для звіту.

5.3.2.6 Використання Server Diagnostics View

Створіть документ Server Diagnostics View.

Передивіться доступну інформацію у вікнах даного документу.

5.3.2.7 Підключення до серверу в Інтернеті

Підключіть UaExpert до сервера за адресою opc.tcp://opcuaserver.com:48010

Подивіться на структуру адресного простору, там повинні бути такі самі об'єкти.

5.3.3 Тестові клієнти для мобільних застосунків

Необов'язкове завдання

5.3.3.1 Встановлення клієнтського застосунку

Встановіть застосунок HMI з драйвером зв'язку OPC UA Client для Android або iOS на мобільний телефон або планшет.

Один з прикладів - Suppanel HMI. Відеоурок доступний за посиланням

https://play.google.com/store/apps/details?id=com.suppanel.suppanel&hl=uk

5.3.3.2 Розробка HMI

Налаштуйте зв'язок з тесовим OPC UA сервером opc.tcp://opcuaserver.com:48010

Розробіть примітив для відображення плинної температури одного з кондиціонерів.

5.3.4 Основи роботи з клієнтськими запитами OPC UA в Node-RED

5.3.4.1 Завантаження та встановлення пакунку node-red-contrib-opcua

Встановіть модуль node-red-contrib-opcua. Для роботи Node-RED OPC UA Server рекомендується поставити версію 0.2.52 через npm:

npm install -g node-red-contrib-opcua@0.2.52

5.3.4.2 Читання значення змінної

Запустіть тестовий сервер UaCPPServer.
Запустіть тестовий клієнт UaExpert і підключіться до UaCPPServer.
Запустіть Node-RED та створіть новий проект. Створіть фрагмент коду, наведений нижче. Зверніть увагу, що кінцева точка буде інша. Значення NodeID: ns=3;s=AirConditioner_1.Temperature.

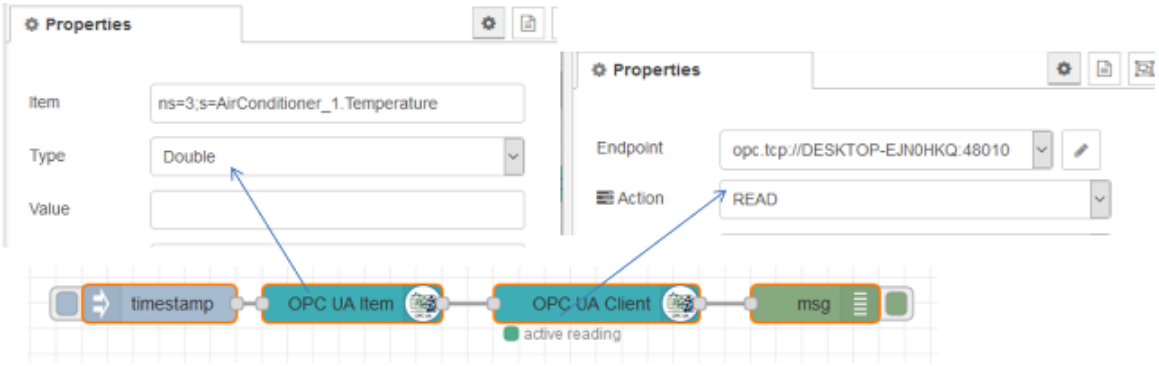


Рисунок 5.7 – Налаштування зчитування параметра

Зробіть розгортання проекту. Ініціюйте потік. Використовуючи UaExpert подивіться чи отримане значення коректне.

5.3.4.3 Читання значення масиву змінних

Скопіюйте фрагмент коду. Замініть NodeID: ns=2;s=Demo.Static.Arrays.Double
Перевірте результат.

5.3.4.4 Читання значення масиву структур

Скопіюйте фрагмент коду
Модифікуйте його відповідно до наведеного нижче фрагменту. NodeID: ns=2;s=Demo.Static.Arrays.Structure;datatype=ExtensionObject

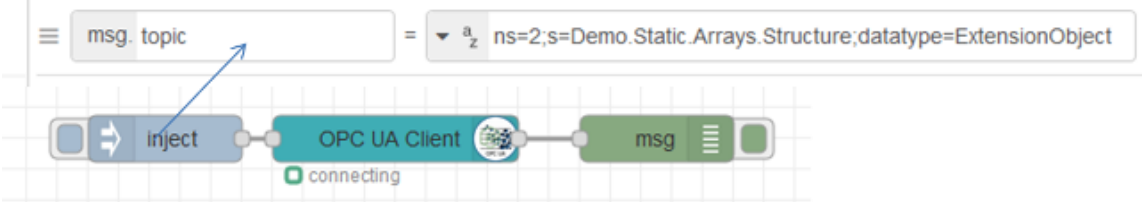


Рисунок 5.8 – Налаштування зчитування масиву структур

5.3.4.5 Читання кількох змінних за один запит

Реалізуйте та перевірте фрагмент програми для зчитування кількох змінних.

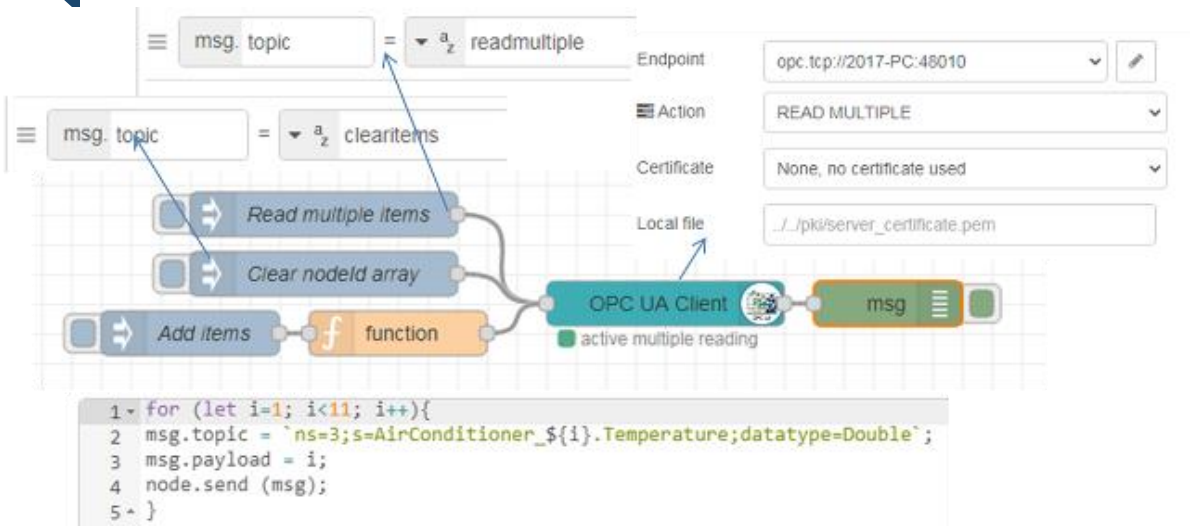


Рисунок 5.9 – Налаштування зчитування кількох змінних

5.3.4.6 Запис змінної

Реалізуйте наведений нижче фрагмент

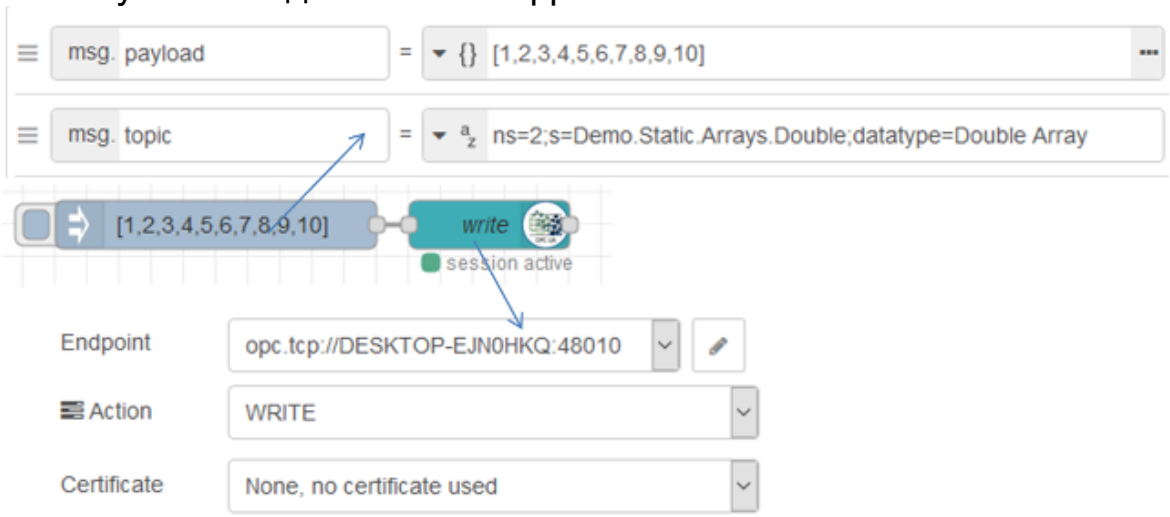


Рисунок 5.10 –

Зробіть розгортання та ініціюйте потік.

За допомогою клієнта UaExpert перевірте, що значення на сервері змінилися.

5.3.4.7 Запис кількох змінних за один запит

Імпортуйте код вузла Inject

[{"id":"1eeb5ff0.f9c0e","type":"inject","z":"dd08c8ba.629018","name":"","props":[{"p":"payload"},{"p":"topic","vt":"str"}],"repeat":"","crontab":"","once":false,"onceDelay":0.1,"topic":"ns=3;s=AirConditioner_1.HumiditySetpoint","payload":"[{\"datatype\":\"Double\",\"nodeId\":\"ns=3;s=AirConditioner_1.TemperatureSetPoint\",\"value\":55},{\"datatype\":\"Double Array\",\"nodeId\":\"ns=2;s=Demo.Static.Arrays.Double\",\"value\":[1,22]}]","payloadType":"json","x":150,"y":760,"wires":[["c202eff8.0d922"]]}]

• Реалізуйте наступний фрагмент програми. У вузлі Write Multiple повинен бути вибраний режим "Writ Multiple".

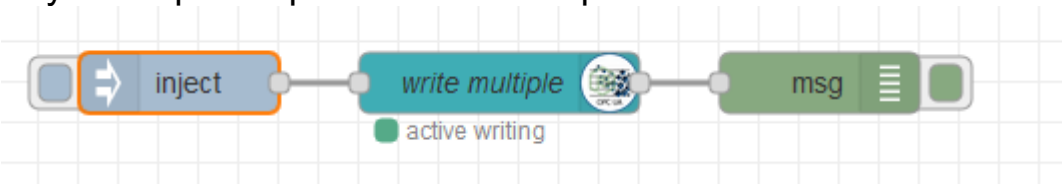


Рисунок 5.11 – Потік запису кількох змінних за один запит

5.3.4.8 Підписування на змінну

Реалізуйте та перевірте роботу наведеного нижче фрагменту.

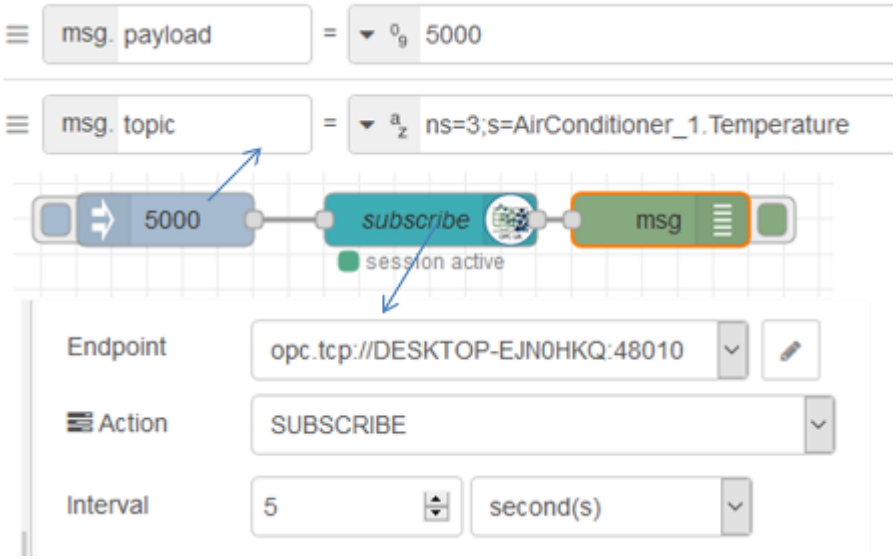


Рисунок 5.12 –

5.3.4.9 Підписування на кілька змінних

Самостійно реалізуйте фрагмент коду, який буде при старті Node-RED формувати підписку на змінні ns=3;s=AirConditioner_x.Temperature, де X - від 1 до 10

5.4 Питання для самоперевірки

1. Що таке OPC UA і яка його основна функція в промислових застосуваннях?
2. Як встановити та налаштувати OPC UA сервер на вашій системі?
3. Які інструменти можна використовувати для тестування OPC UA серверів?
4. Як здійснити базовий аналіз даних, доступних через OPC UA сервер?

5. Які методи можна використовувати для взаємодії з OPC UA сервером?
6. Як працювати з історичними даними на OPC UA сервері?
7. Які основні параметри діагностики OPC UA серверів?
8. Як підключитися до онлайн OPC UA серверів?
9. Які існують мобільні додатки-клієнти для роботи з OPC UA?
10. Як налаштувати OPC UA клієнт в середовищі Node-RED?

5.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40.* URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).
2. Node-RED українською : довідник з Node-RED українською мовою. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).
3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ : НУХТ, 2021. 170 с.
4. Лабораторна робота №6. Робота з OPC UA. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab_opcua.md (дата звернення: 09.07.2024).
5. OPC UA. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B5%D0%BA%D1%86/OPC_UA.md (дата звернення: 09.07.2024).
6. OPC UA Servers – Downloads. *Unified Automation.* URL: https://www.unified-automation.com/downloads/opc-ua-servers.html (дата звернення: 09.07.2024).

ПРАКТИЧНА РОБОТА №6. РОБОТА З ПЛАТФОРМОЮ UBIDOTS

Для виконання завдань практичного заняття слід підготувати дані для надсилання. Як об'єкт контролю буде використовуватися імітаційна модель кондиціювання, реалізована на OPC UA сервері.

6.1 Мета

Мета роботи полягає в набутті навичок використання платформи Ubidots для збирання, відправлення та візуалізації даних з пристроїв за допомогою Node-RED. Це включає налаштування OPC UA сервера та клієнта, реєстрацію на Ubidots, створення нових пристроїв та змінних, а також створення дашбордів і віджетів для відображення даних та генерації подій.

6.2 Завдання

В рамках практичної роботи необхідно виконати наступні завдання:

- інсталяція та перевірка тестового OPC UA сервера та клієнта
- імпорт та перевірка роботи потоку для збору даних
- реєстрація на платформі Ubidots
- створення та налаштування нового пристрою
- створення програми в Node-RED для відправки даних
- перегляд змінних на платформі
- створення та налаштування Dashboard
- додавання різних віджетів
- модифікація програми зміни завдання
- додавання віджетів для зміни завдання
- генерування події
- створення доступу

6.3 Хід роботи

6.3.1 Інсталяція та перевірка тестового OPC UA сервера та клієнта

Даний пункт необхідно виконувати, якщо не виконувалась практична робота по OPC.

Інсталюйте тестовий OPC UA C++ Demo Server , якщо ще не інстальовано.

Інсталюйте тестовий OPC UA Client UaExpert, якщо ще не інстальовано.

Запустіть OPC UA C++ Demo Server та OPC UA Client, з'єднайте їх та перевірте взаємодію.

Інсталюйте модуль node-red-contrib-opcua, якщо він ще не інстальовано.

6.3.2 Імпорт та перевірка роботи потоку для збору даних

На локальному ПК запустіть Node-RED.
Створіть новий проект.
Скачайте файл експорту потока збору даних та імпортуйте.

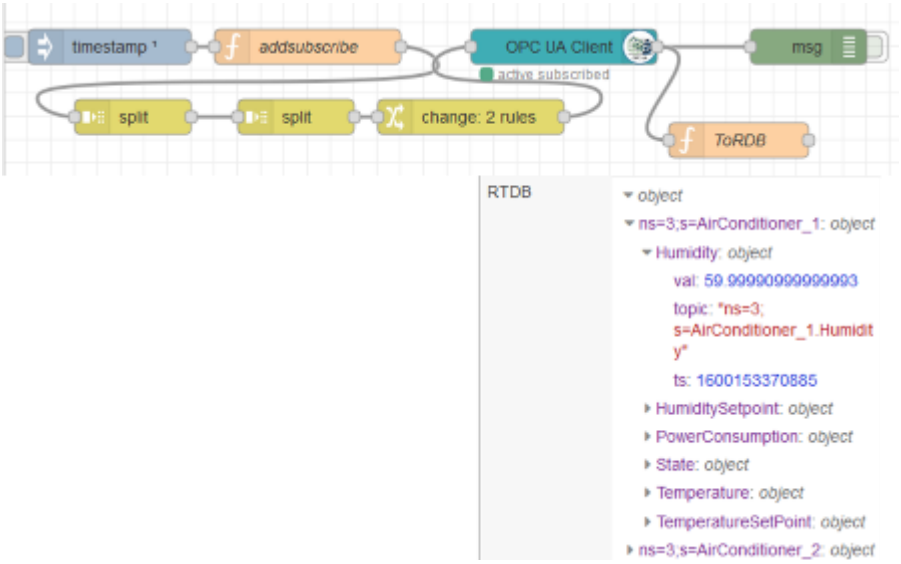


Рисунок 6.1 – Імпорт та перевірка роботи потоку для збору даних

Зробіть розгортання потоку, переглядаючи та змінюючи значення змінних за допомогою OPC UA Client UaExpert перевірте, що у глобальному контексті "RTDB" дані оновлюються
Перегляньте та проаналізуйте вміст потоку.

6.3.3 Реєстрація на платформі Ubidots

Обмеження безкоштовної ліцензії описано тут
Зайдіть на сайт https://ubidots.com/manufacturing/
Натисніть "Sing up" або перейдіть за посиланням
Натисніть "Take Me To Ubidots Stem"
Вкажіть реєстраційні дані та "Sign Up For Free"

6.3.4 Створення та налаштування нового пристрою

Зайдіть у розділ "Devices".
Натисніть "+", щоб створити новий пристрій.
У списку типів виберіть "Blank Devices".
Дайте ім'я пристрою "Conditioner_1".
Вкажіть розташування, яке відповідає поточному вашому за допомогою "set location".

6.3.5 Створення програми в Node-RED для відправки даних

На локальному ПК запустіть Node-RED, якщо він не запущений.

Встановіть бібліотеку ubidots-nodered після чого перезапустить Node-RED.

Створіть потік з ім'ям "ubidots".

Реалізуйте в потоці наведену нижче програму, зверніть увагу, що в полі Token потрібно вставити скопійований з однойменного поля в Devices. Оновлення виставите раз/хвилину, при частому оновленні швидко закінчиться денний ліміт точок. Обмеження безкоштовної ліцензії описано тут

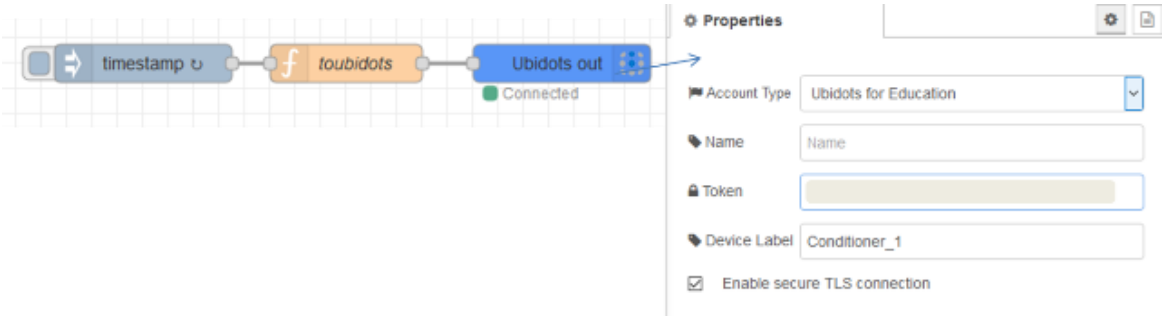


Рисунок 6.2 – Потік відправки даних в Ubidots

```
let cond = global.get ("RTDB.ns=3;s=AirConditioner_1");
let ob = {};
//https://ubidots.com/docs/hw/#introduction
for (const tag in cond){
    ob[tag] = {
        "value":cond[tag].val,
        "timestamp":cond[tag].ts,
        "context": { "key1": "value1", "key2": "value2"},
        "created_at": cond[tag].ts
    };
}
msg.payload = ob;
return msg;
```

Зробіть розгортання потоку.

6.3.6 Перегляд змінних на платформі

Перейдіть на платформу, відкрийте пристрій, оновіть сторінку, якщо вона була відкрита. Ви повинні побачити змінні у вигляді їх поточних значень.

Використовуйте кнопку "Toggle View" для зміни формату відображення.

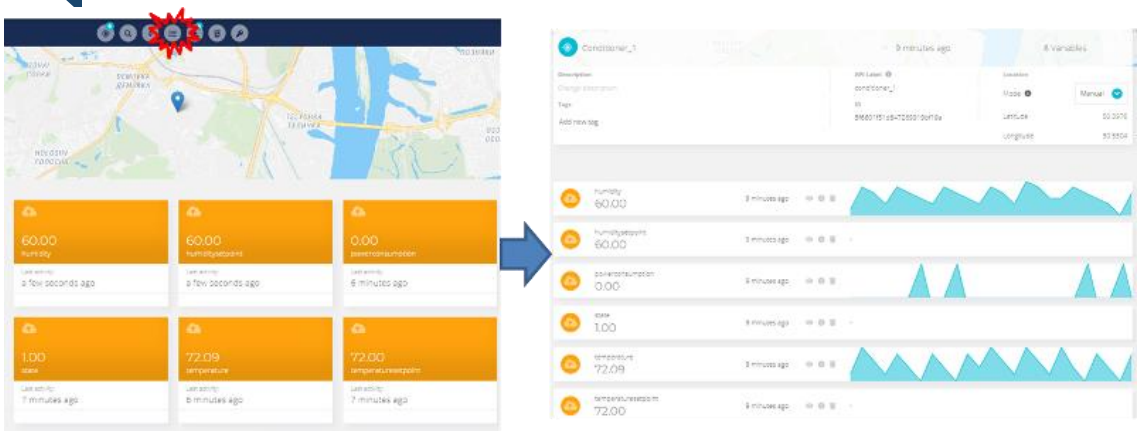


Рисунок 6.3 – Налаштування Ubidots

Виберіть змінну температуру для її детального аналізу

Рисунок 6.4 – Налаштування Ubidots

Встановіть властивості:

- масштаб "Allowed range" (0-100)
- одиниці виміру (°C)
- колір (клацнути по пензлику)

Налаштуйте кольори для інших змінних

6.3.7 Створення та налаштування Dashboard

Перейдіть в Dashboards.

Змініть ім'я Dashboard з назвою Demo Dashboard на Conditioner_1.

Налаштуйте, щоб на Dashboard відображалися останні 10 хвилин інформації.

6.3.8 Додавання різних віджетів

Самостійно додайте різні віджети на Dashboard з прив'язкою до різних змінних. Зверніть увагу, що деякі віджети вимагають однієї змінної, інші - кількох. Поки що додайте наступні віджети.

- HTML Canvas
- Manual input
- Slider
- Switch

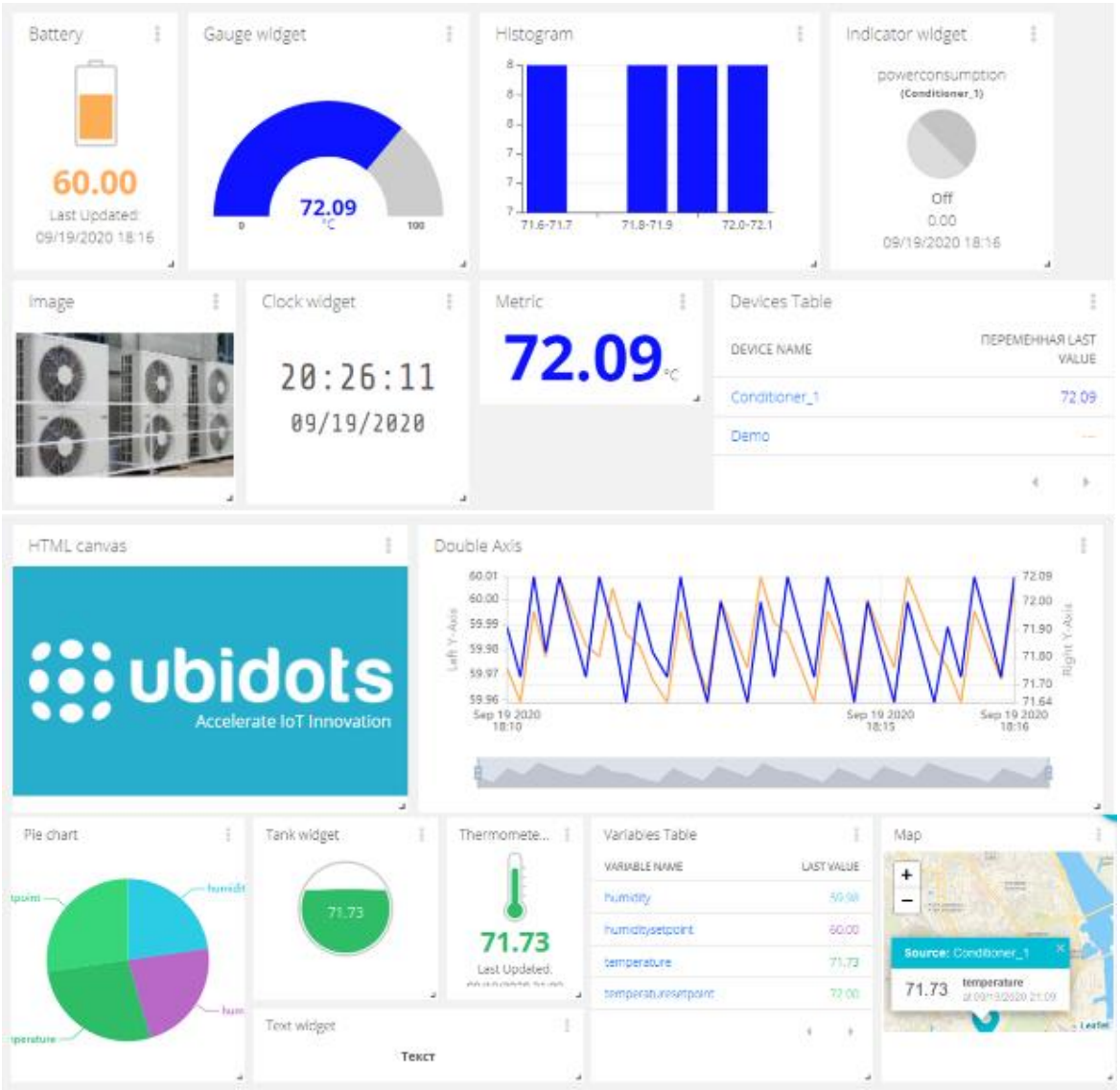


Рисунок 6.5 – Налаштування віджетів

Додавання віджету Canvas по прикладу Simple text with an Image

6.3.9 Модифікація програми зміни завдання

Додайте фрагмент програми з підпискою на зміну завдання з вологості та температури

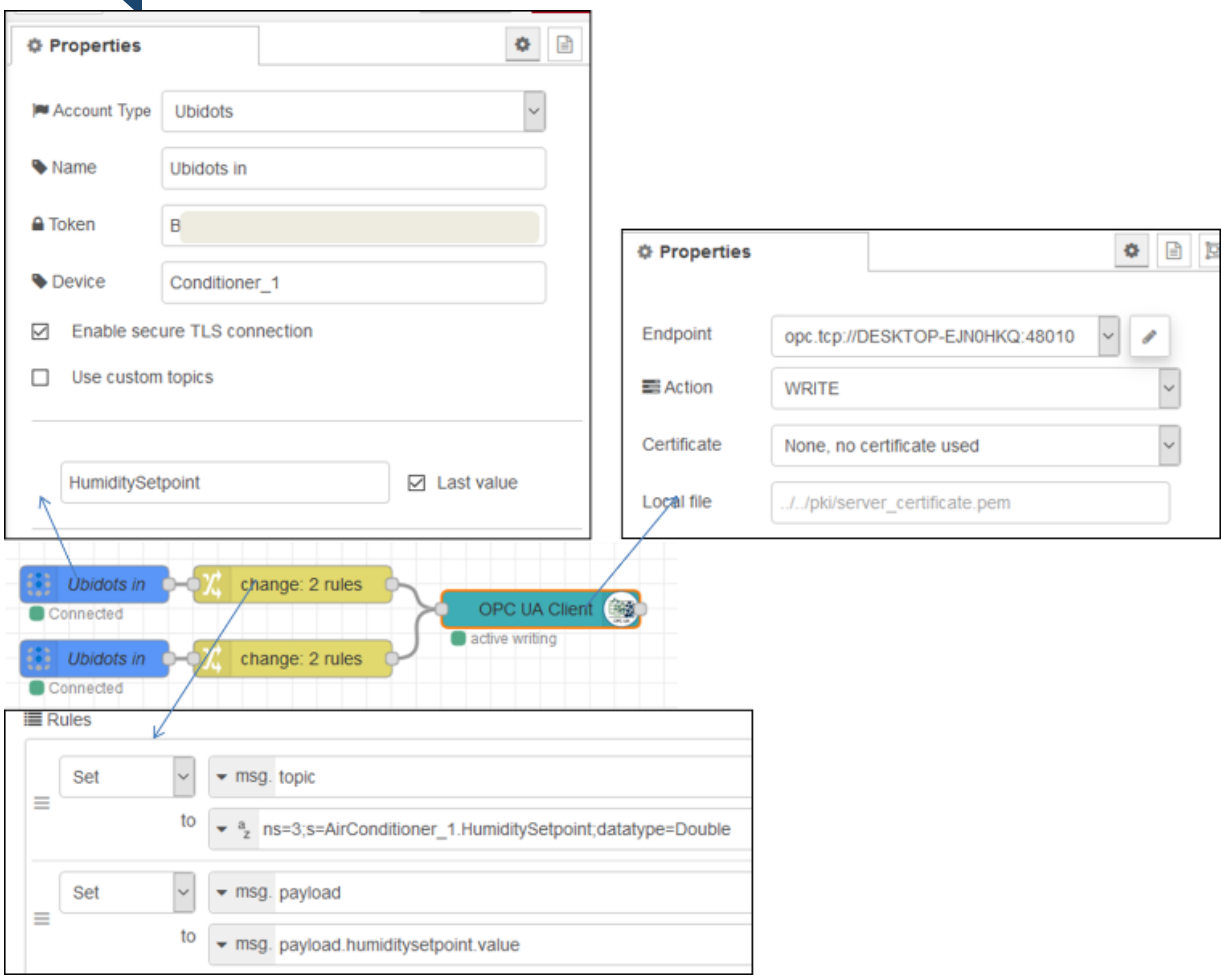


Рисунок 6.6 –

6.3.10 Додавання віджетів для зміни завдання

Додайте віджет Manual Input і налаштуйте його на зміну завдання заданої та поточної температури.

Змініть за допомогою цього віджету завдання на температуру та вологість.

Проведіть таку саму операцію з віджетом "Slider".

Використовуйте віджет "Switch" для формування завдання для 2-х вставок температури, наприклад 15 (ON) та 25 (OFF) градусів.

6.3.11 Генерування події

Перейдіть на сторінку налаштування подій Data->Events

Натисніть "Create Event"

Налаштуйте подію на надсилання поштового повідомлення, якщо температура перевищує 60 ºС.

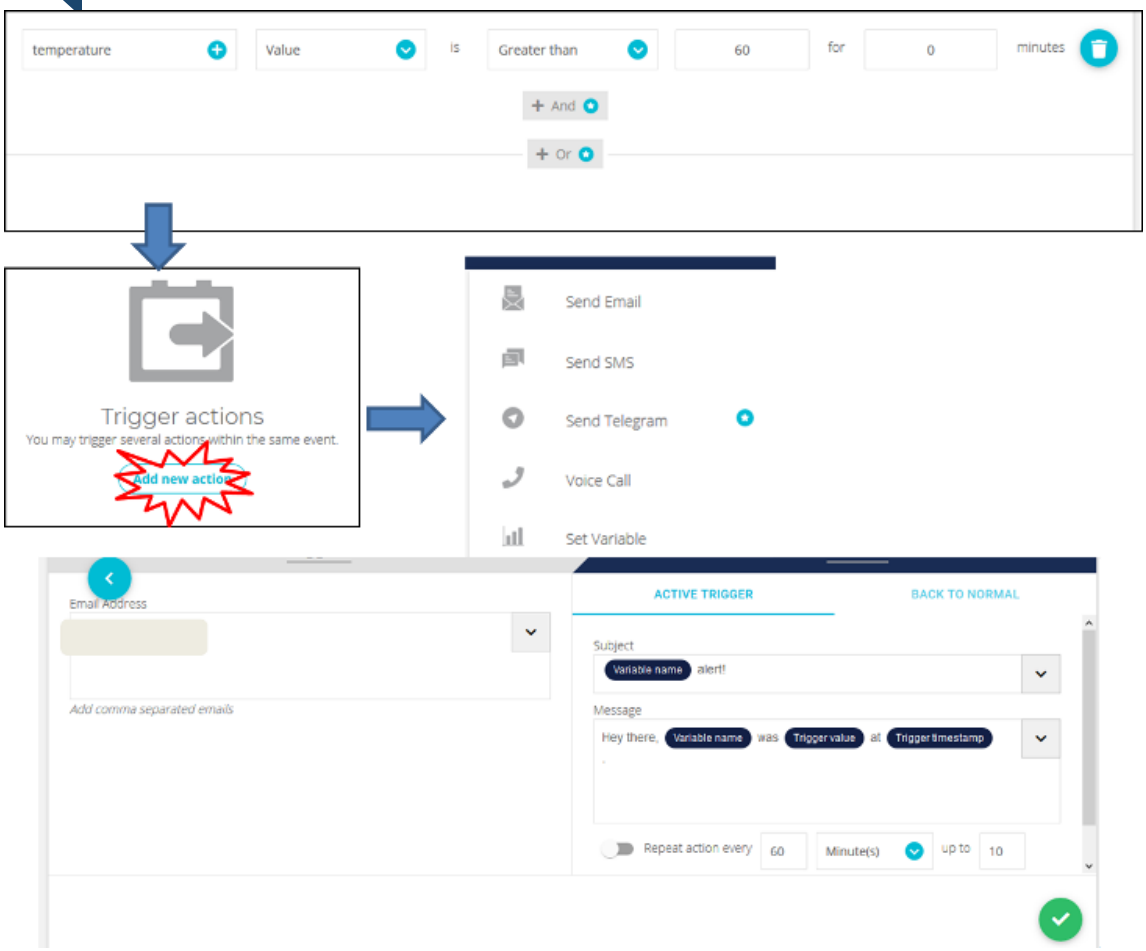


Рисунок 6.7 – Налаштування події при перевищенні температури 60 градусів

Встановіть завдання на 65 градусів, після перевищення 60 градусів має надіслати повідомлення на вказану пошту

6.3.12 Створення доступу

Перейдіть до налаштувань Dashboard і згенеруйте посилання для спільного доступу

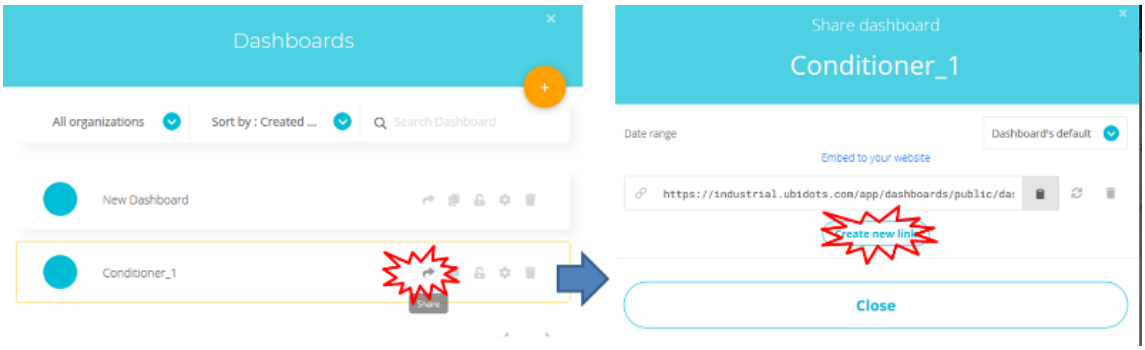


Рисунок 6.8 – Налаштування Dashboard і генерація посилання для спільного доступу

Перевірте, що посилання відкривається в іншому браузері.

Альтернативною платформою для виконання практичної роботи може бути https://thingsboard.io/

6.4 Питання для самоперевірки

1. Що таке OPC UA і як його налаштувати в рамках практичної роботи?
2. Які основні функції виконує Node-RED в цій практичній роботі?
3. Як зареєструватися на платформі Ubidots і створити новий пристрій?
4. Яким чином Node-RED взаємодіє з Ubidots для відправки даних?
5. Як створити і налаштувати змінні на платформі Ubidots?
6. Що таке дашборд на Ubidots і як його створити?
7. Як додавати віджети на дашборд та які їхні функції?
8. Яким чином можна модифікувати Node-RED програму для виконання різних завдань?
9. Що таке події на платформі Ubidots і як їх створювати?
10. Як створити доступні посилання для перегляду дашбордів на Ubidots?

6.5 Перелік рекомендованих джерел

1. Технології Індустрії 4.0. *TI40.* URL: https://pupenasan.github.io/TI40/ (дата звернення: 09.07.2024).
2. Node-RED українською : довідник з Node-RED українською мовою. URL: https://pupenasan.github.io/NodeREDGuidUKR/ (дата звернення: 09.07.2024).
3. Пупена О. М. Довідник з розроблення застосунків в середовищі NODE-RED : електронний довідник. Київ : НУХТ, 2021. 170 с.
4. Лабораторна робота №7. Робота з платформою Ubidots. *TI40.* URL: https://github.com/pupenasan/TI40/blob/master/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80/lab3_ubidots.md#%D0%BB%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0-%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0-7-%D1%80%D0%BE%D0%B1%D0%BE%D1%82%D0%B0-%D0%B7-%D0%BF%D0%BB%D0%B0%D1%82%D1%84%D0%BE%D1%80%D0%BC%D0%BE%D1%8E-ubidots (дата звернення: 09.07.2024).

ДОДАТОК А ФЛАГ QOS

Флаг QoS в байте 1 заслуговує особливої уваги, оскільки він є основою для різноманітної функціональності, яку підтримує MQTT. Флаги QoS містять наступні значення, що базуються на намірі та терміновості повідомлення:

0 = не більше одного разу: сервер спрацьовує і забуває. Повідомлення можуть бути втрачені або подвоєні.

1 = принаймні один раз: одержувач підтверджує доставку. Повідомлення можуть дублюватися, але доставка гарантована.

2 = рівно один раз: сервер забезпечує доставку. Повідомлення надходять точно один раз без втрати або подвоєння.

Давайте розглянемо, як використовувати різні рівні QoS у пристроях IoT та інших застосуваннях.

Де можна використовувати MQTT?

Оскільки застосунки IoT зараз впроваджуються в великих масштабах, MQTT став у центр уваги як відкритий, простий і масштабований спосіб розгортання розподіленого обчислення та функціональності IoT для більш широкої бази користувачів - як на споживчому, так і на промисловому ринках.

Як вже зазначено, MQTT - це легковаговий протокол обміну повідомленнями, побудований для ненадійних мереж та пристроїв з обмеженнями щодо джерела живлення та CPU. Однак це не означає, що зв'язок з потенційною втратою пакетів - єдина його застосовування. MQTT надає різні рівні обслуговування для різних типів інфраструктури IoT, від повторюваного збору даних до управління промисловими машинами:

Дані датчиків навколишнього середовища: як вже згадувалося, MQTT підтримує модель доставки повідомлень "не більше одного разу". У мережах з частковим покриттям території або великою затримкою це означає, що інформація може бути втрачена або подвоєна. У областях, де віддалені датчики записують і передають дані з визначеними інтервалами, це не є проблемою, оскільки нові показники надходять регулярно. Датчики в віддалених середовищах - зазвичай низькопотужні пристрої, що робить MQTT ідеальним рішенням для сенсорів IoT з відносно низьким пріоритетом передачі даних.

Дані про працездатність машин: для швидкої реакції на виникаючі проблеми та запобігання зупинкам. Наприклад, для вітроелектроустановки потрібна гарантована доставка поточних показників про працездатність місцевим командам ще до того, як ця інформація потрапить до центру обробки даних. У таких ситуаціях доставка повідомлень "принаймні один раз" гарантує, що відповідні фахівці своєчасно помітять необхідні повідомлення, навіть якщо вони надходять як дублікати. Це важливо для міжмашинного зв'язку з вищим пріоритетом.

Білінгові системи: є ще більш пріоритетні та точні повідомлення, які потрібно правильно обробляти. В бізнес-ситуаціях, де дублювання записів неприпустиме, включаючи білінгові системи, буде корисний флаг QoS передачі "рівно один раз". Це усуває подвоєння або втрату пакетів у системах виставлення рахунків або білінгу, скорочує кількість аномалій та непотрібних конфліктів щодо узгодження.

Навчально-методичне видання

Койфман Олексій Олександрович

АВТОМАТИЗАЦІЯ ПРОЦЕСІВ ВИРОБНИЦТВА НА БАЗІ ІНТЕРНЕТУ РЕЧЕЙ:

методичні вказівки до виконання практичних робіт

Самостійне електронне мережеве видання

Публікується в авторський редакції