Міністерство освіти і науки України
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Б. Ю. Жураковський, В. А. Нікітін

Технології Інтернету речей

Лабораторний практикум

Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для здобувачів ступеня бакалавра
за спеціальностями 121 «Інженерія програмного забезпечення»,
123 «Комп'ютерна інженерія», 126 «Інформаційні системи та технології»

Електронне мережеве навчальне видання

Київ
КПІ ім. Ігоря Сікорського
2024

УДК 004.738(075.8)
Ж91

Автори: *Жураковський Богдан Юрійович*, д-р техн. наук, професор
Нікітін Валерій Андрійович

Рецензенти: *Коршун Н. В.*, д-р техн. наук, професор,
професор кафедри інформаційної та кібернетичної безпеки
ім. професора Володимира Бурячка Київського столичного
університету імені Бориса Грінченка

Макаренко А. О., д-р техн. наук, професор,
професор кафедри мобільних та відеоінформаційних технологій
Державного університету інформаційно-комунікаційних технологій

Відповідальний
редактор *Батрак Є. О.*, канд. техн. наук, доц.

Гриф надано Методичною радою КПІ ім. Ігоря Сікорського
(протокол № 4 від 01.02.2024 р.)
за поданням вченої ради факультету/навчально-наукового інституту
(протокол № 7 від 29.01.2024 р.)

Жураковський Б. Ю.

Ж91 Технології Інтернету речей [Електронний ресурс] : лаб. практикум : навч. посіб. для здобувачів ступеня бакалавра за спец. 121 «Інженерія програмного забезпечення», 123 «Комп'ютерна інженерія», 126 «Інформаційні системи та технології» / Б. Ю. Жураковський, В. А. Нікітін; КПІ ім. Ігоря Сікорського. – Електрон. текст. дані (1 файл). – Київ : КПІ ім. Ігоря Сікорського, 2024. – 113 с.

Посібник призначений для організації виконання комп'ютерних лабораторних практикумів студентами та ознайомлення із процесом моделювання «розумних» систем. Призначений для студентів, які навчаються за спеціальностями: 121 «Інженерія програмного забезпечення», 123 «Комп'ютерна інженерія», 126 «Інформаційні системи та технології» денної та заочної форм навчання.

УДК 004.738(075.8)

Реєстр. № НП 23/24-297. Обсяг 4,8 авт. арк.

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
проспект Берестейський, 37, м. Київ, 03056
https://kpi.ua
Свідоцтво про внесення до Державного реєстру видавців, виготовлювачів
і розповсюджувачів видавничої продукції ДК № 5354 від 25.05.2017 р.

ЗМІСТ

Вступ

Одним з напрямків діяльності бакалаврів за спеціальностями 121 «Інженерія програмного забезпечення», 123 «Комп’ютерна інженерія», 126 «Інформаційні системи та технології» є знання технологій, які використовуються для побудови Internet of Things (IoT) систем. Масштаб таким систем може суттєво відрізнятися, враховуючи те, що цей напрям активно розвивається та застосовується у більшій кількості напрямів. Це може бути автоматизація домашніх процесів, таких як керування освітлювальною системою, забезпечення неперервного контролю безпеки, виконання необхідних робіт для підтримки життєдіяльності господарства. До більш масивної системи можна віднести розумні міста, які мають ефективну інтеграцію фізичних, цифрових та людських систем в цифровому середовищі заради підтримки сталого розвитку.

Інтернет речей тісно пов’язаний із кіберфізичним простором, оскільки вони взаємодіють у формуванні спільної технологічної екосистеми, де фізичний світ поєднується з цифровим простором. IoT створює мережу підключених пристроїв, які обмінюються даними через Інтернет, забезпечуючи зв'язок між об'єктами. Кіберфізичний простір, у свою чергу, поєднує фізичні об'єкти з віртуальними системами, що дозволяє контролювати і керувати реальними процесами через цифрові моделі.

Ці дві концепції взаємодіють, використовуючи дані, зібрані та оброблені IoT-пристроями, для оптимізації керування фізичними процесами у кіберфізичному просторі. Наприклад, IoT-датчики можуть збирати дані про стан обладнання об'єкту реального світу, а кіберфізичні системи можуть аналізувати ці дані та надавати рекомендації щодо оптимізації роботи об'єкта.

Тому, завданням даного навчального посібника, відповідно до робочої навчальної програми курсу «Технології Інтернету речей», є розгляд та вивчення основних технологій, протоколів та підходів щодо проектування

відповідних систем.

Загальний порядок виконання лабораторних робіт:

- ознайомитися з теоретичними відомостями до відповідної лабораторної роботи;

- сумлінно виконати отримане завдання;

- скласти протокол лабораторної роботи;

- здати та захистити лабораторну роботу.

Вимоги до оформлення протоколів лабораторних робіт

Протокол лабораторної роботи охайно оформлюється у вигляді текстового файлу.

На титульному аркуші зверху з вирівнюванням по центру вказують назву університету, факультету та кафедри. Нижче під ними розміщують номер, назву лабораторної роботи; ще нижче, з вирівнюванням вправо, вказують прізвище автора роботи, групу та прізвище викладача, який буде здійснювати перевірку роботи.

На наступних листах вказують мету роботи та розміщують результати відповідної роботи.

Для захисту лабораторної роботи необхідно:

- надати протокол з результатами роботи;

- продемонструвати роботу програмного продукту;

- відповісти на запитання викладача.

Лабораторна робота №1

Тема: Інтерфейси обміну даними між IoT пристроями. RS232

Мета роботи — ознайомити студентів з інтерфейсами обміну даними, розглянути протокол RS-232 і навчитися взаємодіяти із IoT пристроями з його використанням.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Обмін даними відіграє важливу роль в Інтернету речей. Враховуючи велику кількість виробників, стандартизація та уніфікація дозволяє забезпечити сумісність між пристроями та системами. Різноманіття таких інтерфейсів залежить від різних чинників [1]:

1. енергоефективність

Деякі пристрої IoT працюють на батарейках або інших джерелах живлення з обмеженою енергією. Використання спеціалізованих низькоенергетичних інтерфейсів може допомогти зберігати енергію та продовжувати роботу пристроїв на тривалий час.

2. дальність зв’язку

IoT може включати пристрої, розташовані на різних відстанях один від одного. Різні інтерфейси можуть забезпечити підтримку зв'язку на різних відстанях, включаючи локальні мережі, мережі на великій відстані та безпроводові технології.

3. пропускна здатність

Деякі пристрої IoT потребують великої пропускної здатності для передачі великої кількості даних, наприклад, відео, зображень або потокового аудіо. Різні інтерфейси можуть забезпечити високу швидкість передачі даних.

4. специфічні вимоги застосувань

Певні IoT рішення можуть вимагати специфічних характеристик комунікації, таких як низька затримка, висока надійність або підтримка багатьох підключених пристроїв.

Найбільш популярні інтерфейси обміну даних:

1) RS-232

Цей стандарт був розроблений для організації зв'язку між комп'ютерами та іншими електронними пристроями. Він забезпечує дуплексне з’єднання [8].

Інтерфейс RS-232 передбачає використання різних ліній для передачі інформації. Основні лінії включають:

_TX (Transmit)

Для передачі даних від передавача до приймача;

_RX (Receive)

Для отримання даних від приймача до передавача;

_RTS (Request to Send)

Ця лінія вказує, що передавач готовий до відправки даних;

_CTS (Clear to Send)

Ця лінія вказує, що приймач готовий прийняти дані від передавача;

_DTR (Data Terminal Ready)

Ця лінія вказує, що пристрій готовий приймати або передавати дані;

_DSR (Data Set Ready)

Ця лінія вказує, що зовнішній пристрій готовий для комунікації;

_RI (Ring Indicator)

Ця лінія вказує на входження сигналу дзвінка (для модемів);

_GND (Ground)

Лінія заземлення для забезпечення електричної стійкості системи

Інтерфейс RS-232 дозволяє передавати дані на відстань до кількох метрів, залежно від швидкості передачі та якості кабелю. Однак цей стандарт має деякі обмеження, зокрема низьку швидкість передачі даних порівняно з сучасними стандартами, такими як USB та Ethernet.

2) RS-485

Призначений для передачі даних на більші відстані (до 1200 метрів) та в більш широких мережах. RS-485 є послідовним інтерфейсом передачі даних із збалансованими лініями передачі, що дозволяє досягнути високої надійності комунікації на відстанях до кількох кілометрів.

3) IEEE 1284

Стандартний інтерфейс паралельного порту отримав свою первинну назву через американську компанію Centronics – виробника принтерів. Перші версії цього стандарту були орієнтовані виключно на принтери, мали на увазі передачу даних лише в один бік (від комп'ютера до принтера) і мали невисоку швидкість передачі (150-300 Кбайт/с). Такі швидкості неприйнятні для сучасних друкуючих пристроїв. Крім того, для роботи з деякими пристроями необхідна двостороння передача даних. Тому деякі компанії (Xircom, Intel, Hewlett Packard, Microsoft) запропонували декілька модифікацій швидкісних паралельних інтерфейсів: EPP (Enhanced Parallel Port) – до 2 Мбайт/с, ECP (Extended Capabilities Port) – до 4 Мбайт/с і ін. На основі цих розробок в 1994 році Інститутом інженерів по електроніці і електротехніці був прийнятий стандарт IEEE 1284-1994, нині повсюдно використовуваний в персональних комп'ютерах як стандартний паралельний інтерфейс.

4) UART

UART, або універсальний асинхронний приймач-передавач, є одним із

найбільш використовуваних протоколів зв’язку між пристроями.

При правильній конфігурації UART може працювати з багатьма різними типами послідовних протоколів, які включають передачу та отримання послідовних даних. У послідовному зв’язку дані передаються побітово за допомогою однієї лінії або дроту. У двосторонньому зв'язку ми використовуємо два дроти для успішної послідовної передачі даних (рис. 1.1). Залежно від програми та вимог до системи, для послідовного зв’язку потрібно менше схем і проводів, що зменшує вартість реалізації [7].

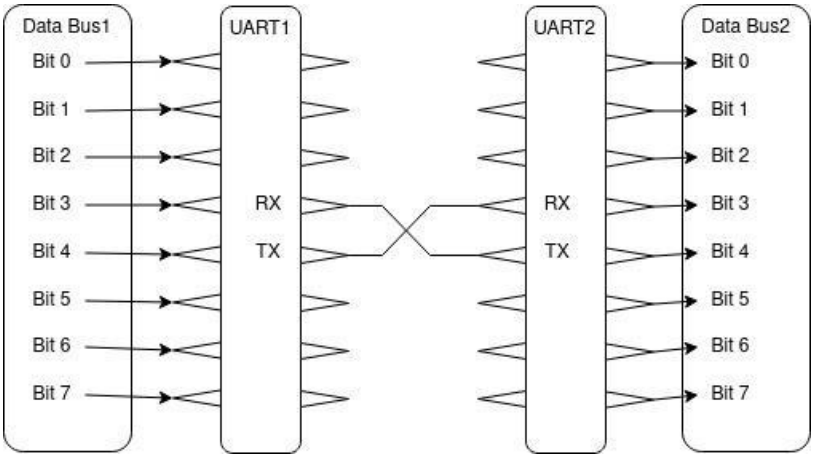

Рисунок 1.1 — Схема взаємодії UART з шиною даних

5) USB

USB - це абревіатура від Universal Serial Bus, а він став стандартом у зв'язку, з'єднанні та живленні для різних комп'ютерних пристроїв. Він широко використовується у багатьох пристроях, починаючи від клавіатур і мишей до сканерів, принтерів, флеш-накопичувачів, телефонів, планшетів, зовнішніх жорстких дисків, навіть телевізорів. Цей стандарт став популярним завдяки своїй універсальності та можливості підключення різних пристроїв до комп'ютера без необхідності використання спеціальних роз'ємів для кожного пристрою [2].

Він представляє собою інтерфейс plug-and-play, що дозволяє підключати пристрої до комп'ютера та автоматично виявляти їх. У більшості випадків комп'ютер самостійно ідентифікує та налаштовує підключений пристрій.

Проте іноді може знадобитися встановлення драйверів для операційної системи, щоб забезпечити належну роботу пристрою. USB виконує функцію зв'язку та передачі даних між підключеними пристроями, що робить його дуже універсальним та зручним для використання.

USB може передавати живлення пристроям. Наприклад, в смартфонах USB-порт використовується як для передачі даних, так і для зарядки акумуляторів. Це забезпечує зручність, оскільки один порт може виконувати різні функції.

Однак, не лише смартфони використовують цю можливість. Сучасні ноутбуки та деякі пристрої тепер можуть заряджати свої батареї через порт USB Type-C, навіть без використання окремих адаптерів живлення. Це робить їх користування ще зручнішим, оскільки можна використовувати той самий кабель для заряджання декількох пристроїв.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 1

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;
- мета роботи;
- постановка завдання і алгоритмічного вирішення;
- відповіді на завдання у текстовому форматі та графічними зображеннями за необхідності.

Завдання на лабораторну роботу

Завдання 1.1. Взаємодія через послідовний порт з використанням мови програмування Python3

1.1.1. Створити новий PTY пристрій для прослуховування (рис. 1.2) [5].

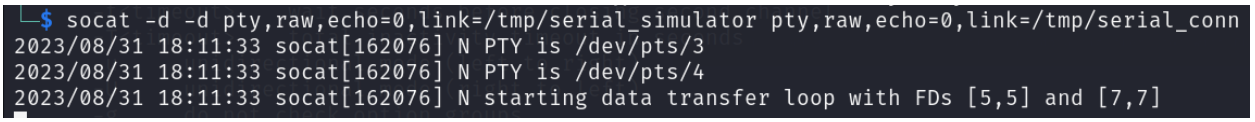

Рисунок 1.2 — Створення віртуального з’єднання

pty - створює псевдотермінал (pty). Інший процес може підключитись, використовуючи її як послідовну лінію або термінал;

raw - встановлює raw режим, таким чином пропускаючи вхідні та вихідні дані майже необробленими;

echo — включає або виключає local echo;

link - генерує symbolic link, яке вказує на псевдотермінал (pty). Це може допомогти вирішити проблему, коли pty генеруються з більш-менш непередбачуваними іменами, що ускладнює прямий доступ до автоматично згенерованих socat pty.

1.1.2. Створити скрипт-симулятор (рис. 1.3).

```
└─$ cat simulator.py
#!/usr/bin/python3

import sys
import logging

import serial

DEFAULT_ADDR = '/tmp/serial_simulator'

logging.basicConfig(level=logging.INFO)

addr = sys.argv[1] if len(sys.argv) > 1 else DEFAULT_ADDR

conn = serial.serial_for_url(addr)
logging.info(f'Ready to receive requests on {addr}')
while True:
    request = conn.readline()
    logging.info('REQ: %r', request)
    request = request.strip().decode().lower()
    reply = 'Test-conn,24C,305682,1.05A\n' if request == '*idn?' else 'NACK\n'
    reply = reply.encode()
    logging.info('REP: %r', reply)
    conn.write(reply)
```

Рисунок 1.3 — Скрипт симулятору

1.1.3. Створити скрипт з клієнтською частиною (рис. 1.4).

```
└─$ cat client.py
#!/usr/bin/python3

import sys
import serial
DEFAULT_ADDR = '/tmp/serial_conn'
DEFAULT_CMD = '*IDN?'
args = len(sys.argv) - 1
if args == 0:
    addr, cmd = DEFAULT_ADDR, DEFAULT_CMD
elif args == 1:
    addr, cmd = DEFAULT_ADDR, sys.argv[1]
else:
    addr, cmd = sys.argv[1:3]

cmd += '\n'
s = serial.serial_for_url(addr)
s.write(cmd.encode())
print(s.readline())
```

Рисунок 1.4 — Скрипт клієнтської частини

1.1.4. Запустити simulator.py.

1.1.5. Запустити client.py та отримати результат роботи.

Завдання 1.2. Взаємодія через послідовний порт з використанням вбудованих засобів Linux

1.2.1. Перезапустити команду socat;

1.2.2. Перезапустити скрипт simulator.py;

1.2.3. Відкрити нову вкладку терміналу та поставити на прослуховування /tmp/serial_conn.

1.2.4. Відкрити нову вкладку терміналу та надіслати запит до /tmp/serial_conn з використанням команди echo.

1.2.5. Пересвідчитись, що результат як на скриншоті (рис. 1.5).

Рисунок 1.5 — Отримання відповіді на запит

При використанні реального пристрою, рекомендується використання утиліти minicom [6].

Додаткові завдання

Завдання 1.3. Налаштувати HTTP-over-serial. Написати скрипти за необхідності.

Завдання 1.4. Написати утиліту (скрипт) для передачі файлів з використанням serial. За бажанням, можна додати перевірку на цілісність з використанням CRC32.

Контрольні запитання

1. Що таке послідовний порт?
2. Які є лінії передачі в RS-232?
3. Які інтерфейси є для передачі даних?
4. Які особливості має RS-232?
5. Яке використання RS-232 в IoT?

Лабораторна робота №2

Тема: Вивчення концепції Internet of Things (IoT) із використанням Cisco Packet Tracer

Мета роботи — ознайомити студентів з концепцією Internet of Things та технологіями, що використовуються в IoT, а також навчити створювати прості IoT-проекти за допомогою Cisco Packet Tracer.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Що таке IoT?

Інтернет речей (IoT) - це мережа фізичних пристроїв, здатних збирати та обмінюватися даними через Інтернет без прямого втручання людини. Ці пристрої можуть бути вбудовані в різні об'єкти, такі як домашні прилади, автомобілі, будівлі, медичні прилади та інші предмети повсякденного життя.

Принциповою складовою IoT є сенсори, які збирають дані з фізичних об'єктів, та вбудовані системи, які обробляють та передають ці дані через Інтернет. Також важливою складовою є програмне забезпечення та хмарні сервіси, які дозволяють збирати та аналізувати дані з IoT пристроїв [2].

Платформи Інтернету речей, рішення та послуги, які надають такі гравці, як AT & T, Inc., пропонують такі рішення, як послуги розгортання, рішення для транспортних засобів, інтегровані рішення, розумні міста та інші. Cisco Systems, Inc. забезпечує підключення до мережі, контроль даних і обмін даними, керування підключенням і хмарні обчислення. Siemens AG надає промислові периферійні, цифрові двійники та інші рішення.

Мережеве обладнання для підключення пристроїв IoT до мережі

Для підключення пристроїв IoT до мережі можна використовувати Access Point (рис. 2.1). Це бездротовий мережевий пристрій, що

використовується для створення мережі Wi-Fi. Access Point зазвичай підключається до мережі за допомогою кабелю Ethernet та має вбудовану антену, що дозволяє передавати та отримувати дані від інших пристроїв, що підключені до бездротової мережі. Access Point може мати різні параметри, такі як ім'я мережі (SSID), канал, тип шифрування, обмеження швидкості та інші.

Рисунок 2.1 - Вигляд різних пристроїв Cisco Access Point

У Cisco Packet Tracer Access Point можна налаштовувати як самостійний пристрій або як частину більшої мережі. Він може бути підключений до маршрутизатора або іншого комутатора в мережі, що дозволяє підключати до бездротової мережі різні пристрої, такі як комп'ютери, смартфони, планшети, принтери та інші [9].

Також можна використовувати Home Gateway – це пристрій, який використовується в мережах з доступом до Інтернету в домашніх умовах. Він є частиною домашньої мережі та забезпечує з'єднання між локальною мережею (LAN) та широкосмуговим доступом до Інтернету.

Home Gateway можна налаштовувати для підключення до різних типів Інтернет-підключень, таких як кабельний, DSL або оптоволоконний. Він також має можливість налаштування бездротової мережі Wi-Fi, що дозволяє підключатися до Інтернету з різних пристроїв, таких як комп'ютери, смартфони, планшети тощо.

Home Gateway також може функціонувати як маршрутизатор, що дозволяє передавати пакети даних між різними мережами (рис. 2.2). Він має

вбудовану функцію брандмауера, що забезпечує безпеку мережі, і може налаштовуватися для різних типів VPN з'єднань.

Рисунок 2.2 - Зовнішній вигляд різних моделей Cisco Home Gateway

Також можна налаштувати мережу для роботи з пристроями IoT, використовуючи звичайний роутер, який підтримує можливість створення власного HTTP серверу.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 2

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;
- мета роботи;
- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 2.1. Створення простої мережі використовуючи switch, сервер, ПК, Access Point та пристрої IoT.

2.1.1 На рис. 2.3 зображена побудована мережа, яка складається з 2 пристроїв IoT: лампи та детектора руху. Вони будуть підключені до Access Point PT(в Cisco Packet Tracer розташований у вкладці Network Devices – Wireless Devices) – це пристрій, який використовується в мережах з доступом до Інтернету в домашніх умовах. Він є частиною домашньої мережі та забезпечує з'єднання між локальною мережею (LAN) та широкосмуговим доступом до Інтернету. До нього будуть підключені IoT пристрої. Для керування IoT системою будемо використовувати PC.

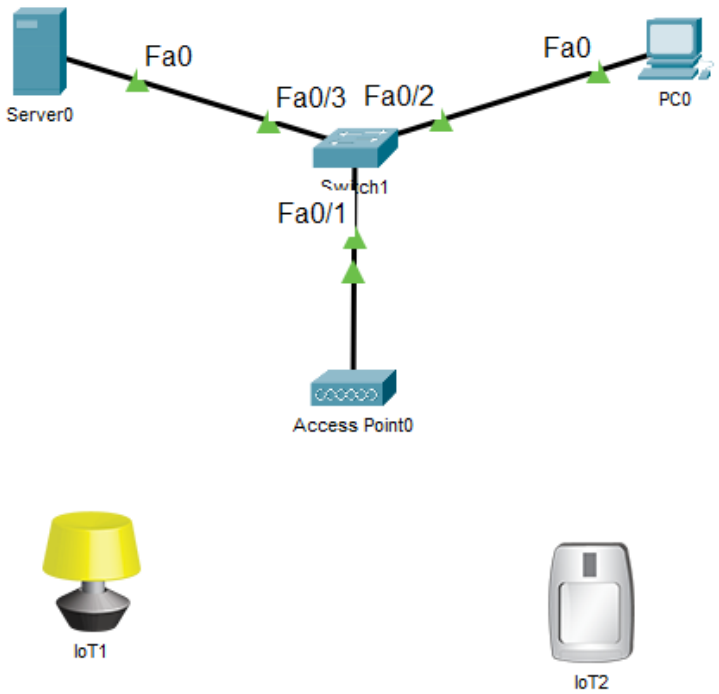

Рисунок 2.3 - Схема мережі для подальшого налаштування екосистеми IoT

2.1.2 Налаштуємо IP адреси в мережі. Встановимо IP-адресу Server0 192.168.0.1 та маску підмережі 255.255.255.0 – рис. 2.4.

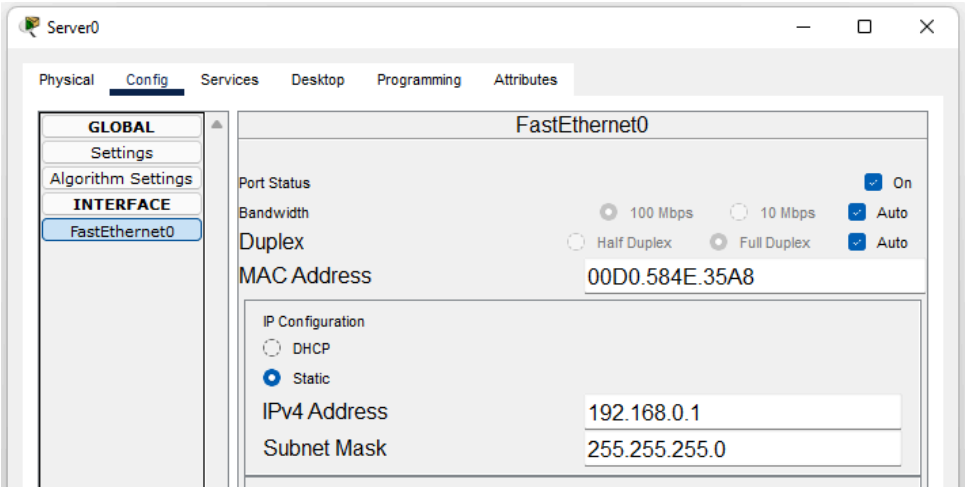

Рисунок 2.4 - Вікно налаштувань Server0 із встановленою IP-адресою 192.168.0.1

2.1.3 Перейдемо у вкладку Services, розділ DHCP. Увімкнемо DHCP встановивши перемикач у положення On. Це дозволить іншим пристроям мережі автоматично отримувати IP адреси. Встановимо Start IP Address починаючи з 192.168.0.2, бо адреса 192.168.0.1 вже зарезервована сервером. Отриманий результат на рис. 2.5.

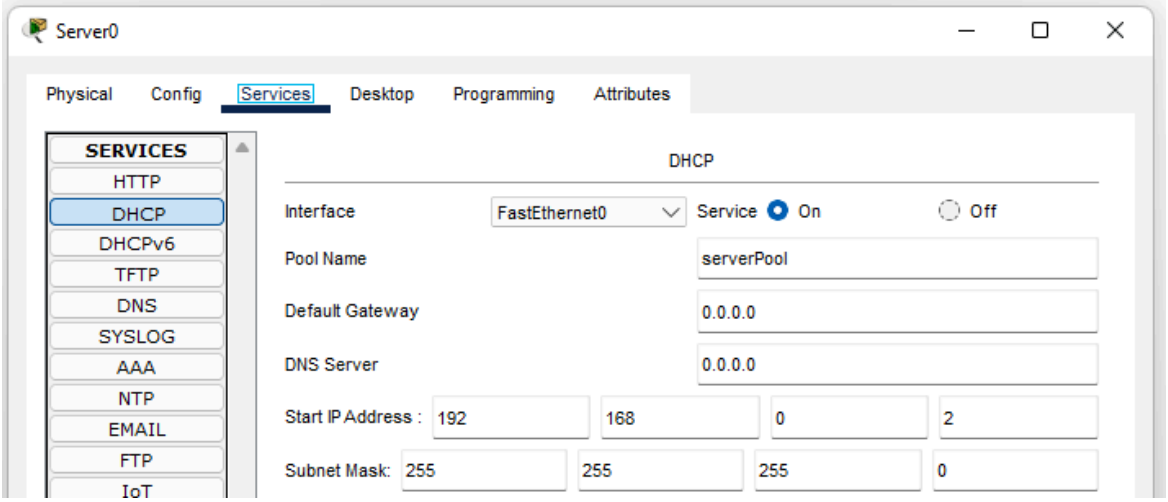

Рисунок 2.5 - Вікно налаштувань протоколу DHCP на Server0

2.1.4 Перевіримо, чи PC0 автоматично отримає IP адресу. Встановимо перемикач у положення DHCP. На рис. 2.6 можна побачити, що PC0 отримав ip адресу 192.168.0.2. Автоматичне отримання IP адрес буде зручним при додаванні більшої кількості пристроїв в мережу.

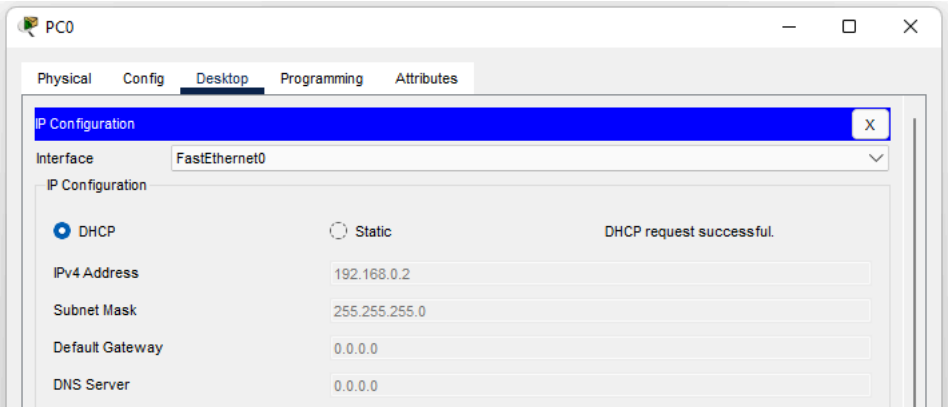

Рисунок 2.6 - Результат автоматичного отримання IP адреси PC0

2.1.5 В налаштуваннях серверу, перейдемо у вкладку Services, розділ IoT та встановимо перемикач в положення On. Це дозволить використовувати сервер для реєстрацій та входу на сторінку управління IoT пристроями за протоколами HTTP та HTTPS з PC0 або інших пристроїв підключених до мережі. Отриманий результат на рис. 2.7.

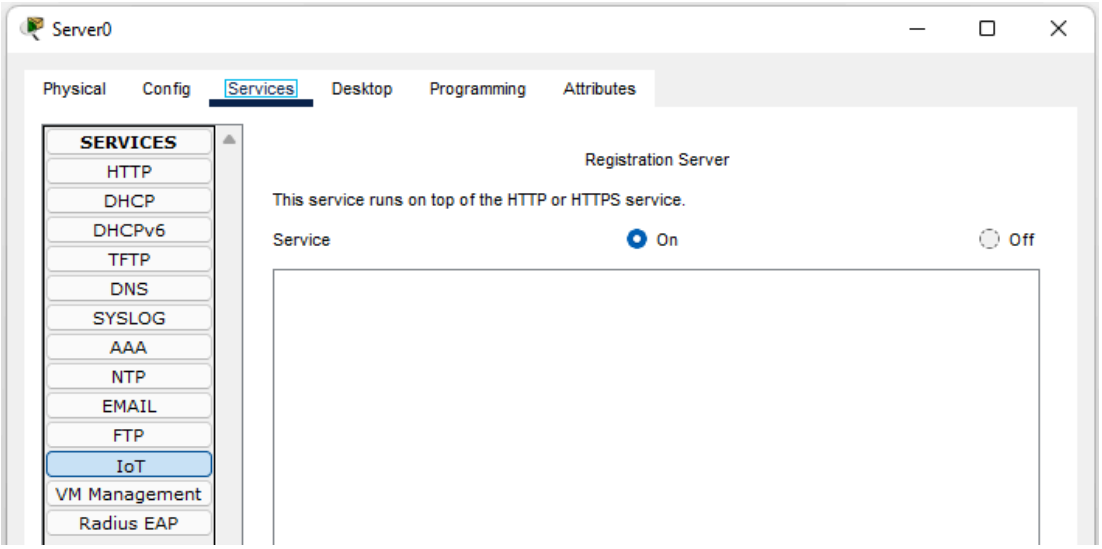

Рисунок 2.7 – Увімкнений перемикач сервісу реєстрацій та входу для управління IoT пристроями на Server0

2.1.6 Тепер ми можемо перейти до управління та налаштування пристроїв IoT з PC0. Для цього перейдемо в браузер та введемо IP-адресу сервера (192.168.0.1). Завантажиться сторінка входу в акаунт – рис. 2.8.

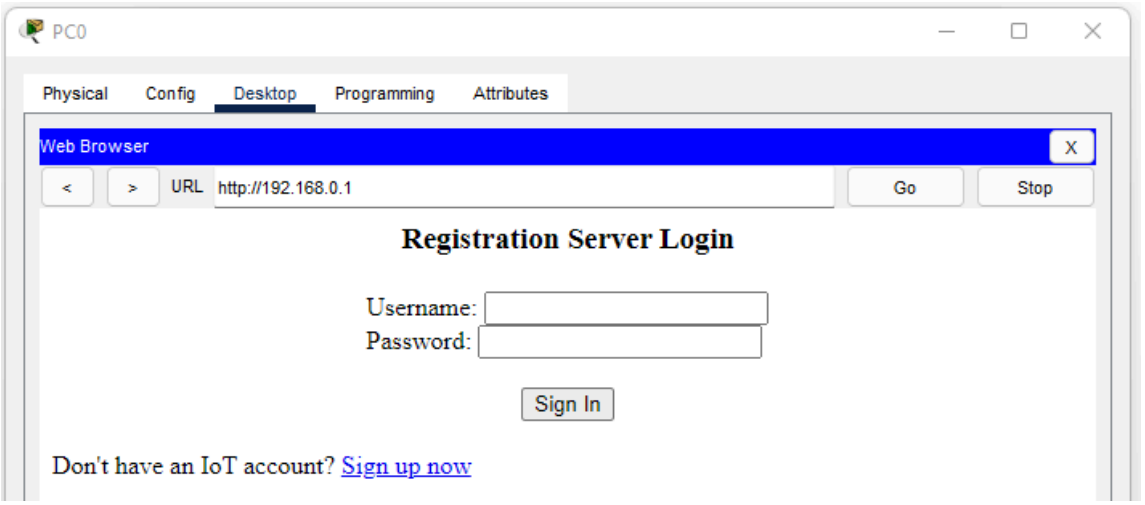

Рисунок 2.8 - Сторінка входу в акаунт керування пристроями IoT з PC0

2.1.7 Через те, що ми не маємо аккаунта на Server0, натиснемо на посилання “Sign up now” внизу відкритої сторінки. Створимо новий акаунт. Задаємо username admin та пароль admin – рис. 2.9.

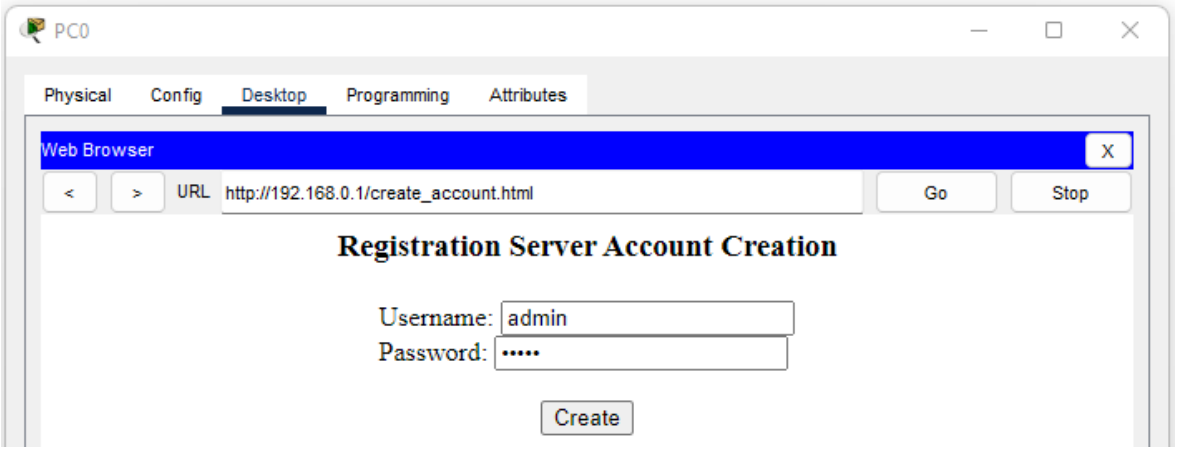

Рисунок 2.9 – Вигляд сторінки реєстрації акаунту для керування пристроями IoT

2.1.8 Натиснемо кнопку Create для створення нового акаунту. Ми перейдемо на сторінку управління пристроями IoT, однак вона буде порожньою, через те, що не було додано жодного пристрою. Щоб їх підключити, необхідно спочатку налаштувати Access Point. Натиснемо на

нього лівою кнопкою миші, перейдемо у вкладку Config, розділ Port1, задаємо назву нашої мережі WiFi (SSID) IoT-WiFi. Отриманий результат на рис. 2.10.

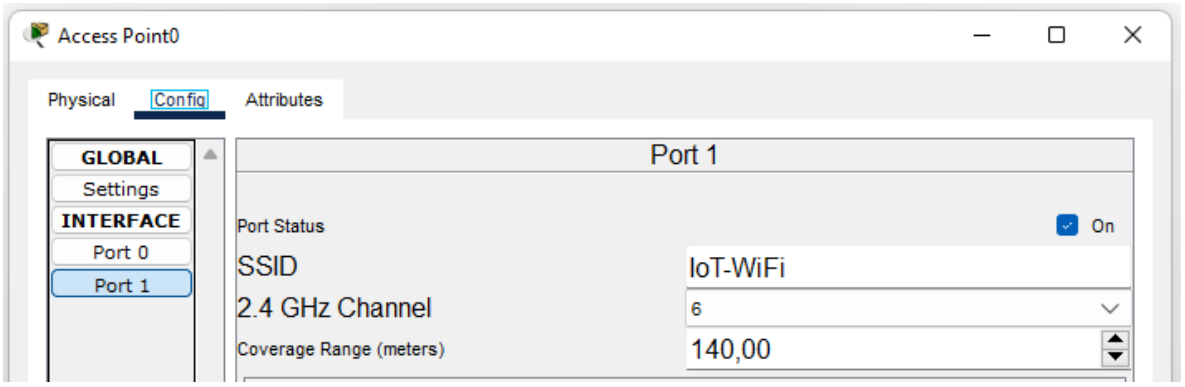

Рисунок 2.10 - Результат встановлення назви мережі IoT-WiFi на Access Point

2.1.9 Зайдемо в налаштуванні лампи. Для цього натиснемо на неї лівою кнопкою миші та перейдемо у вкладку Config, розділ Wireless0. Вкажемо SSID мережі IoT-WiFi. Лампа автоматично підключиться до Access Point. Отриманий результат на рис. 2.11.

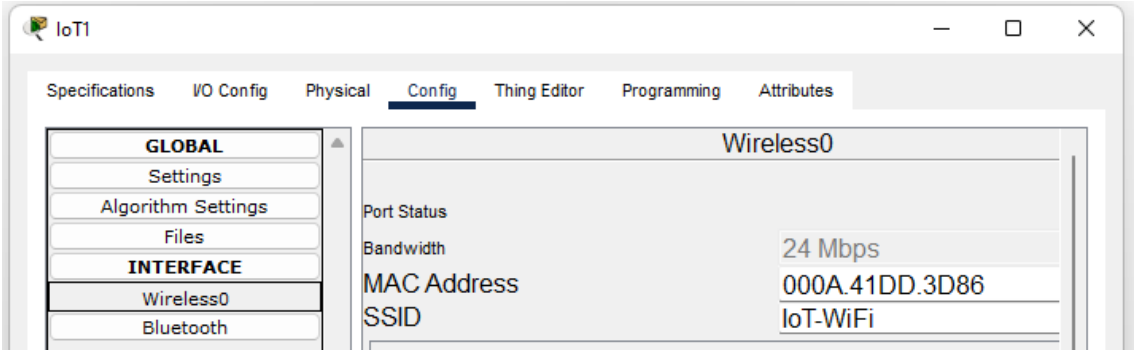

Рисунок 2.11 – Результат встановлення підключення лампи до Access Point через WiFi

2.1.10 Перейдемо у розділ Settings, Встановимо відображене ім’я Lamp, встановимо перемикач Gateway/DNS IPv4 у положення DHCP та задаємо підключення до віддаленого серверу (Remote Server). Введемо IP-адресу сервера (192.168.0.1) та введемо дані створеного аккаунту (ім’я користувача та пароль). Натиснемо кнопку Connect. Отриманий результат на рис. 2.12.

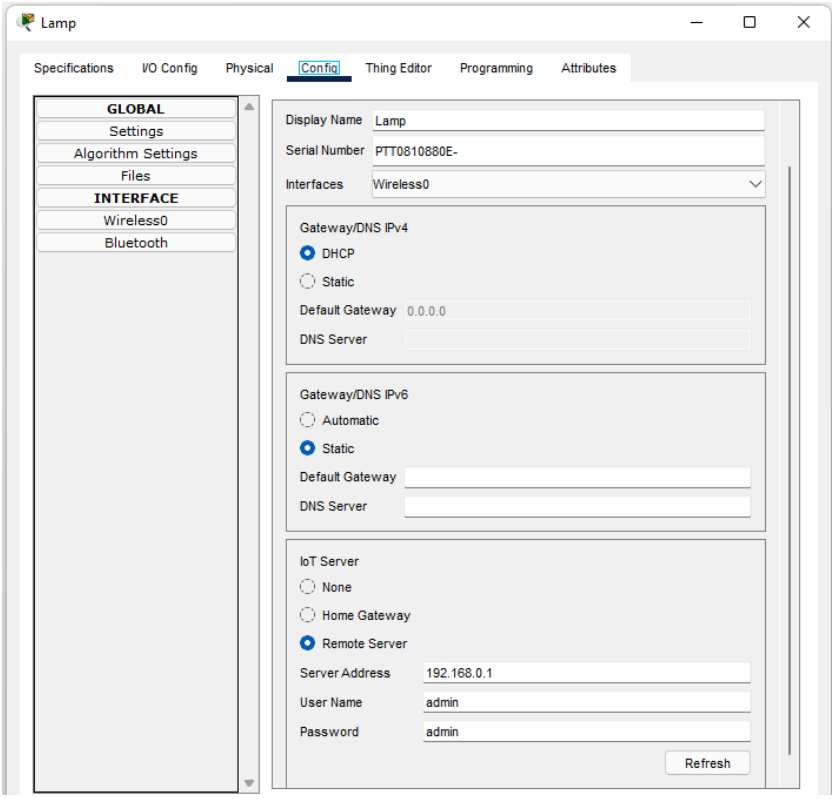

Рисунок 2.12 - Результат налаштування підключення лампи до серверу

2.1.11 Так само налаштуємо і датчик руху. Підключимо його до мережі WiFi та у вкладці Settings задаємо йому відображене ім’я Motion Detector. Так само встановимо перемикач Gateway/DNS IPv4 у положення DHCP та задаємо підключення до віддаленого серверу (Remote Server). Введемо IP-адресу сервера (192.168.0.1) та введемо дані створеного аккаунту. Натиснемо кнопку Connect. Отриманий результат на рис. 2.13.

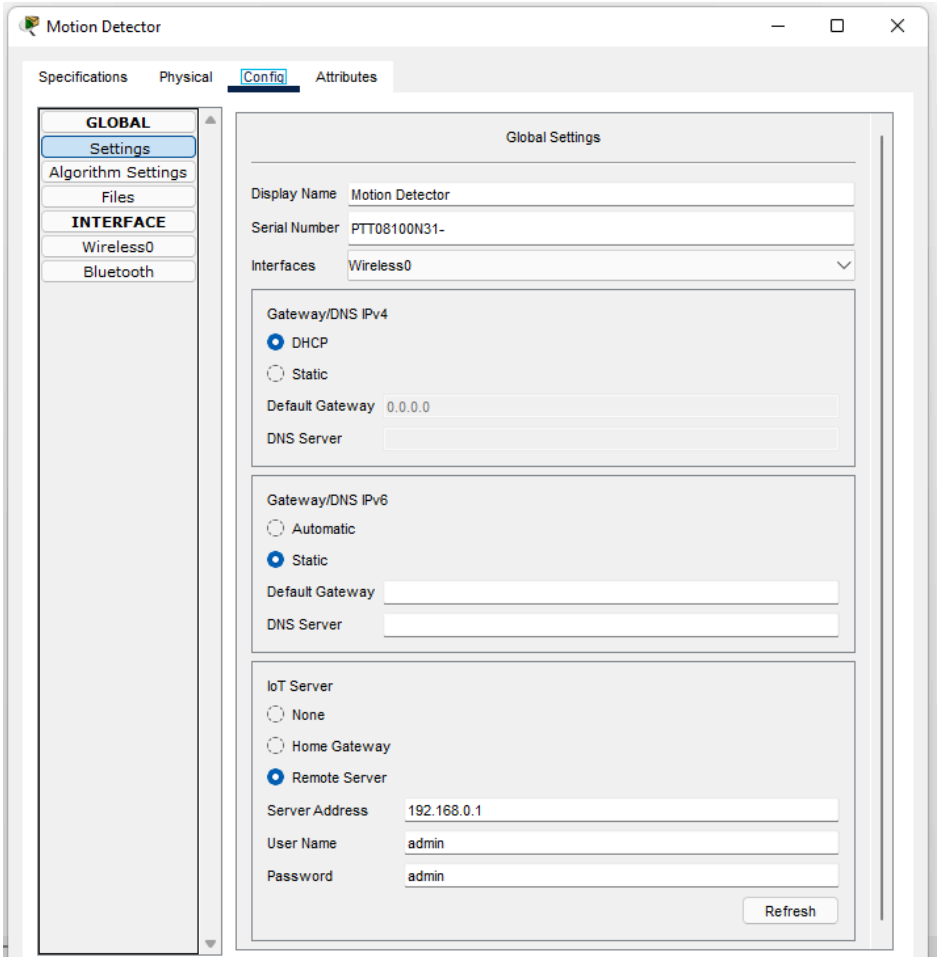

Рисунок 2.13 - Результат налаштування підключення датчика руху до серверу

2.1.12 Тепер, при вході на сторінку керування пристроями IoT з браузера на PC0, можна побачити, що було підключено два нових пристрої. Якими можна дистанційно керувати. Можна включати лампу встановивши перемикач у положення On або Dim. Датчик руху спрацює, при проведенні біля нього курсору мишки із натиснутою клавішею Alt. Іконка статусу на сторінці керування при цьому зміниться. Отриманий результат на рис. 2.14.

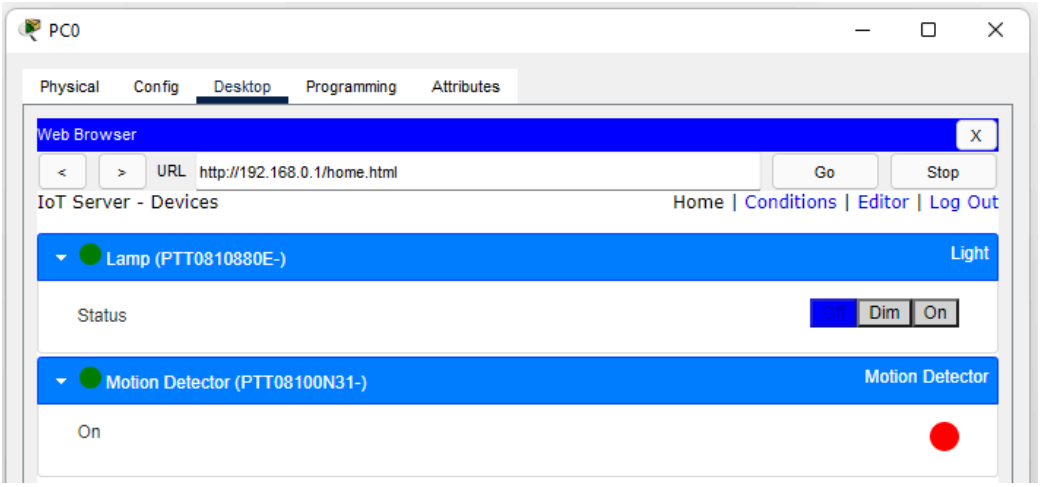

Рисунок 2.14 – Результат підключення двох IoT пристроїв з можливістю їх дистанційного керування

Завдання 2.2. Створення бездротової мережі з використанням Home Gateway DC100 та підключенням до мережі за допомогою смартфону.

2.2.1 Побудуємо макет житлового будинку, використовуючи інструмент малювання прямокутників (гаряча клавіша R). Додамо декілька IoT пристроїв в мережу та встановимо Home Gateway(знаходиться у вкладці Network Devices – Wireless Devices). Через те, що за замовчуванням назва адреси встановлена як HomeGateway всі пристрої автоматично під’єдналися до мережі. Але в подальшому ми змінимо назву мережі та задамо пароль для доступу до неї. Отриманий макет будинку на рис. 2.15.

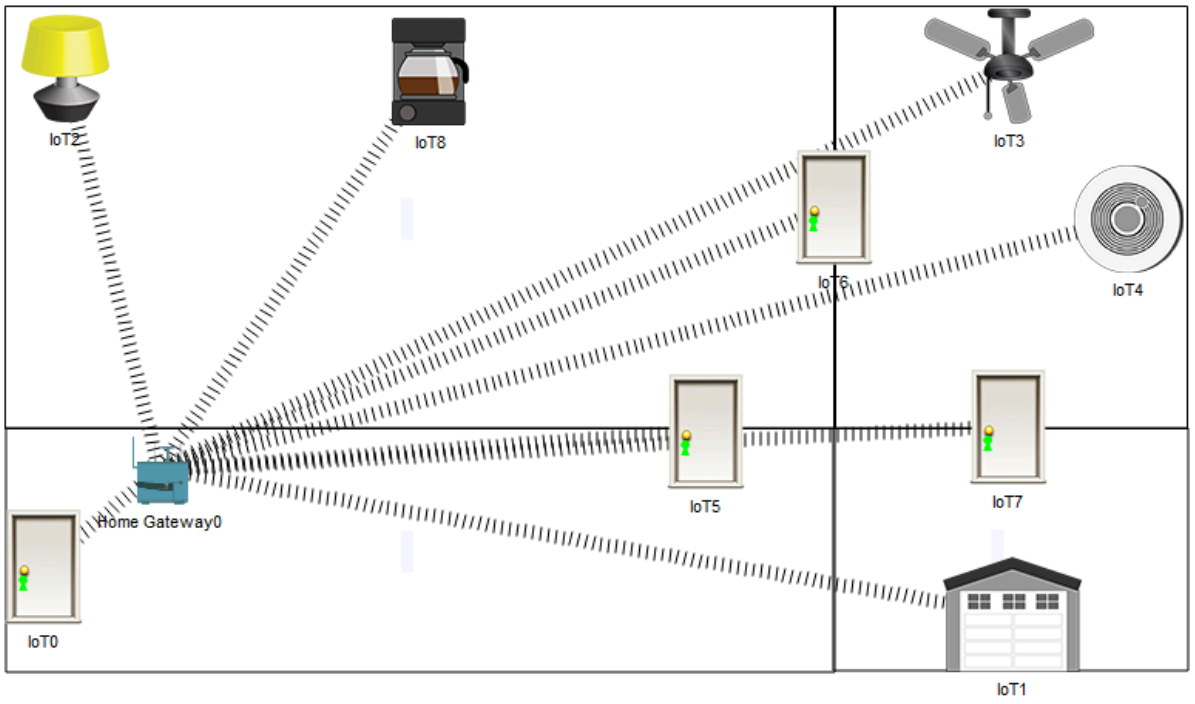

Рисунок 2.15 – Макет будинку з IoT пристроями

2.2.2 Зайдемо в налаштування Home Gateway, перейдемо у вкладку Config, розділ Wireless та змінимо назву мережі зі стандартної на Home-WiFi та встановимо пароль “password”, тип авторизації WPA2-PSK. Усі пристрої від’єднаються від мережі. Отриманий результат на рис. 2.16.

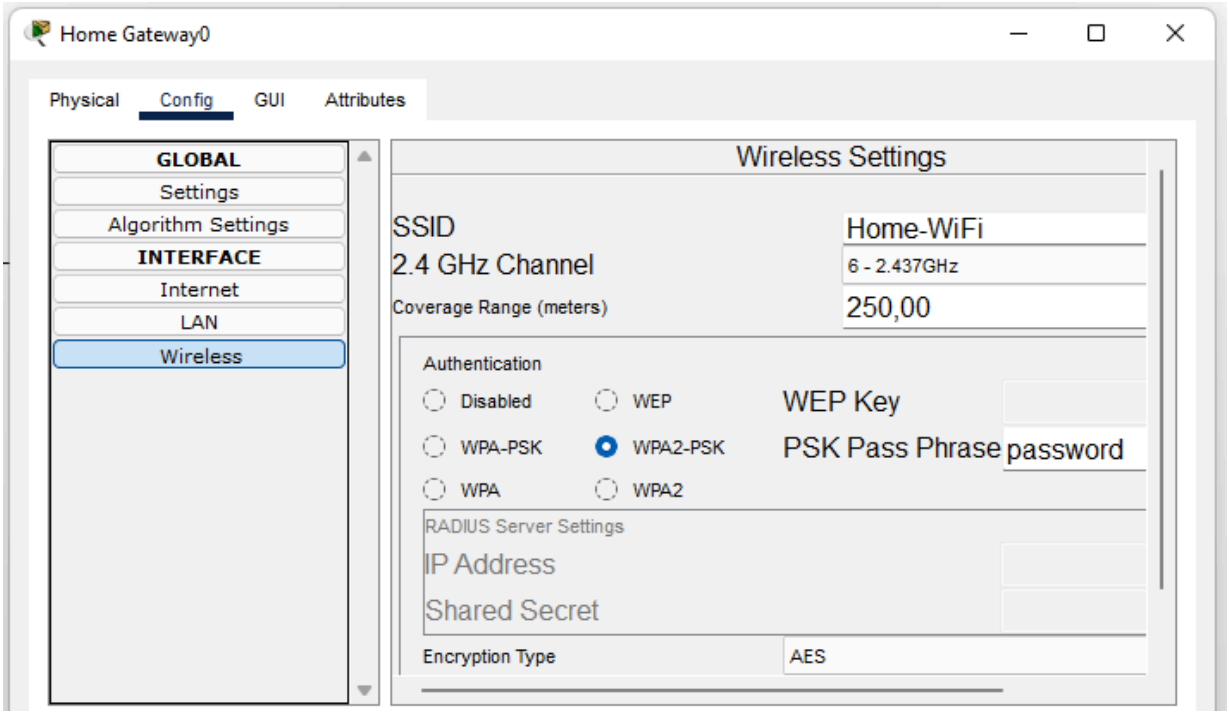

Рисунок 2.16 – Результат налаштування Home Gateway

2.2.3 Потрібно встановити ip адресу Home Gateway, він буде виступати в якості серверу. Зайдемо у вкладку Config, розділ LAN, вкажемо адресу 192.168.0.1 замість стандартної – рис. 2.17.

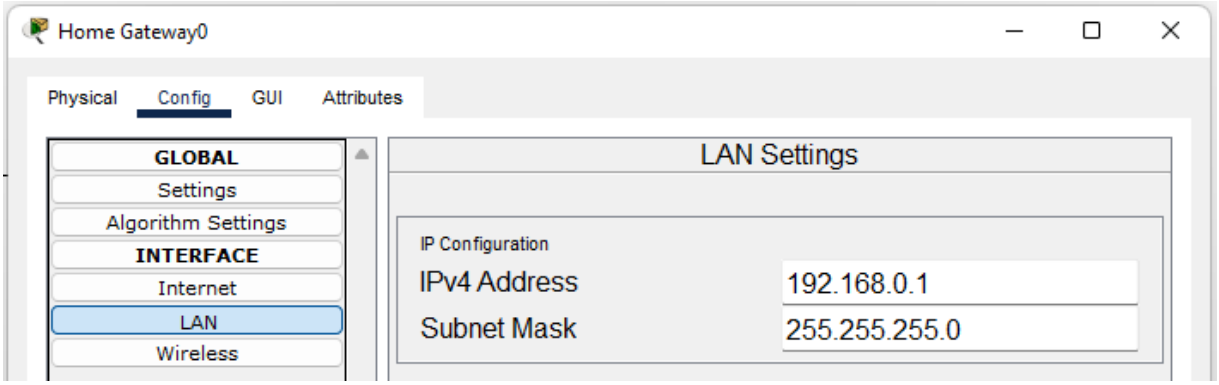

Рисунок 2.17 - Результат встановлення IP адреси для Home Gateway

2.2.4 Під’єднаємо всі пристрої до мережі WiFi. На кожному з пристроїв перейдемо у вкладку Config, розділ Wireless0, введемо назву WiFi (SSID) Home-WiFi, виберемо тип аутентифікації WPA2-PSK та введемо пароль password. Пристрій має автоматично підключитися до мережі. Отриманий результат на рис. 2.18.

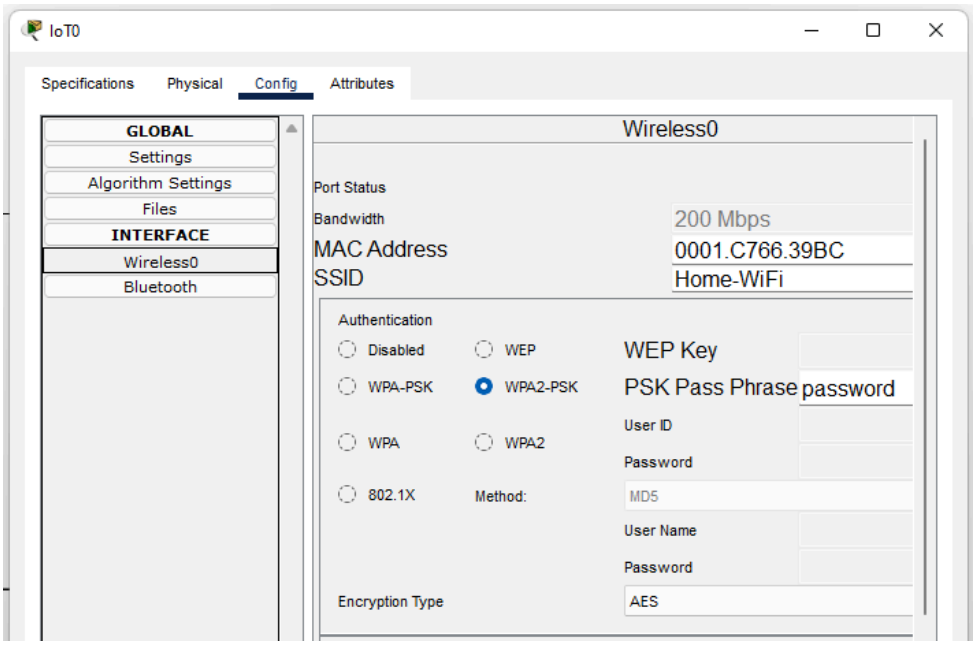

Рисунок 2.18 - Результат налаштування підключення пристрою IoT до мережі

WiFi

2.2.5 Також необхідно обрати на всіх пристроях тип підключення до серверу Home Gateway (вкладка Config, розділ Settings) – рис. 2.19.

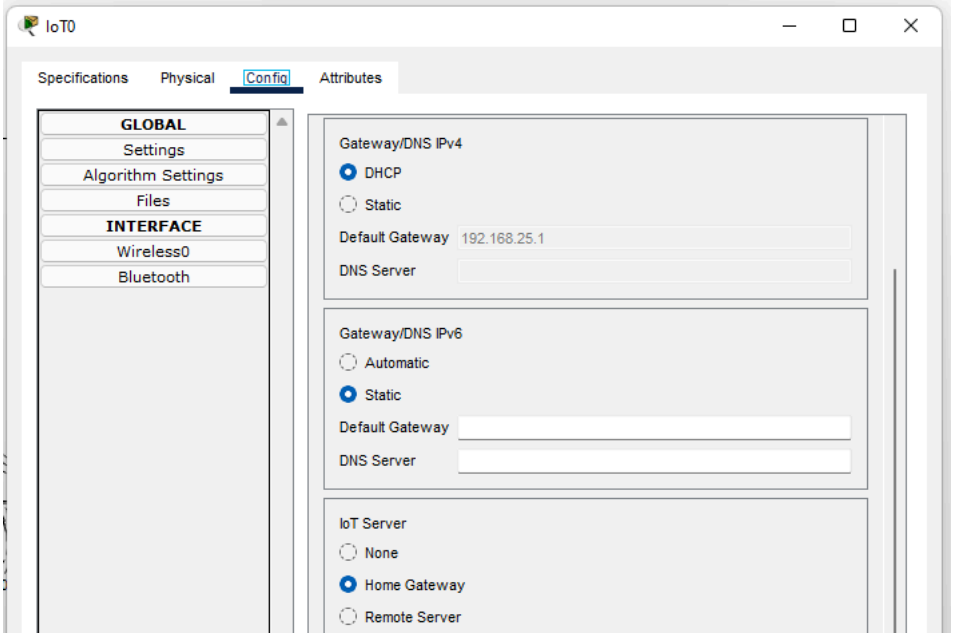

Рисунок 2.19 - Тип підключення до серверу встановлений Home Gateway

2.2.6 Після підключення всіх пристроїв до мережі, додамо ще в мережу смартфон.Знього буде здійснюватися керування IoT системою. Отриманий результат на рис. 2.20.

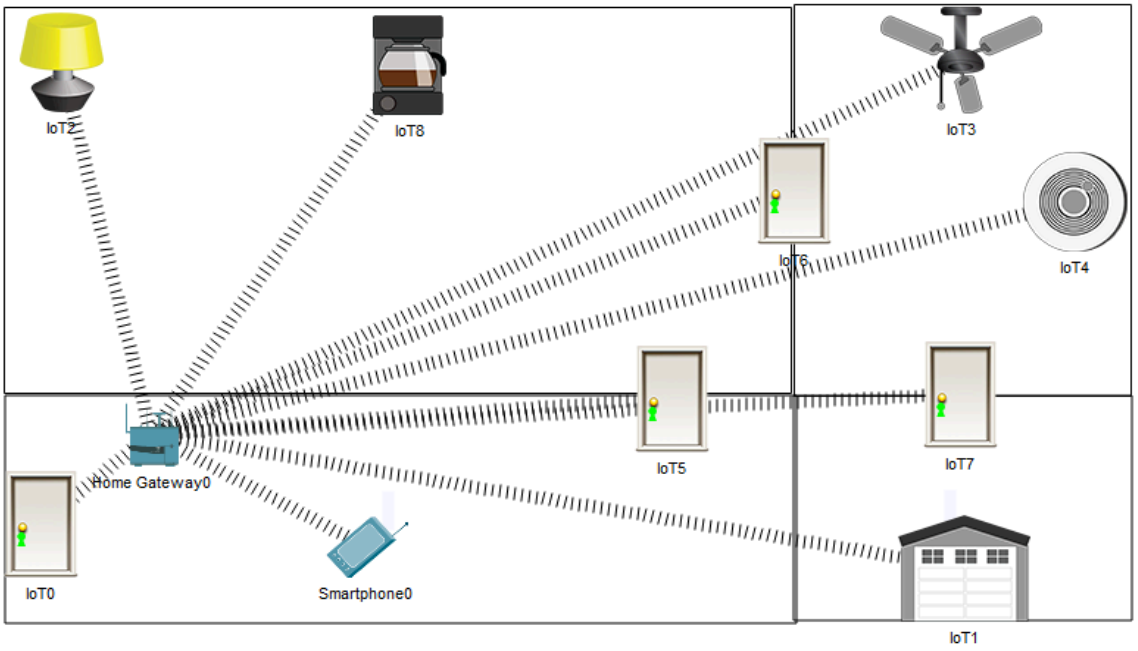

Рисунок 2.20 - Мережа зі смартфоном для подальшого його налаштування

2.2.7 Підключимо його до мережі так само як і інші пристрої. Для налаштування смартфону достатньо просто зайти в розділ Wireless та ввести назву та пароль від мережі – рис. 2.21.

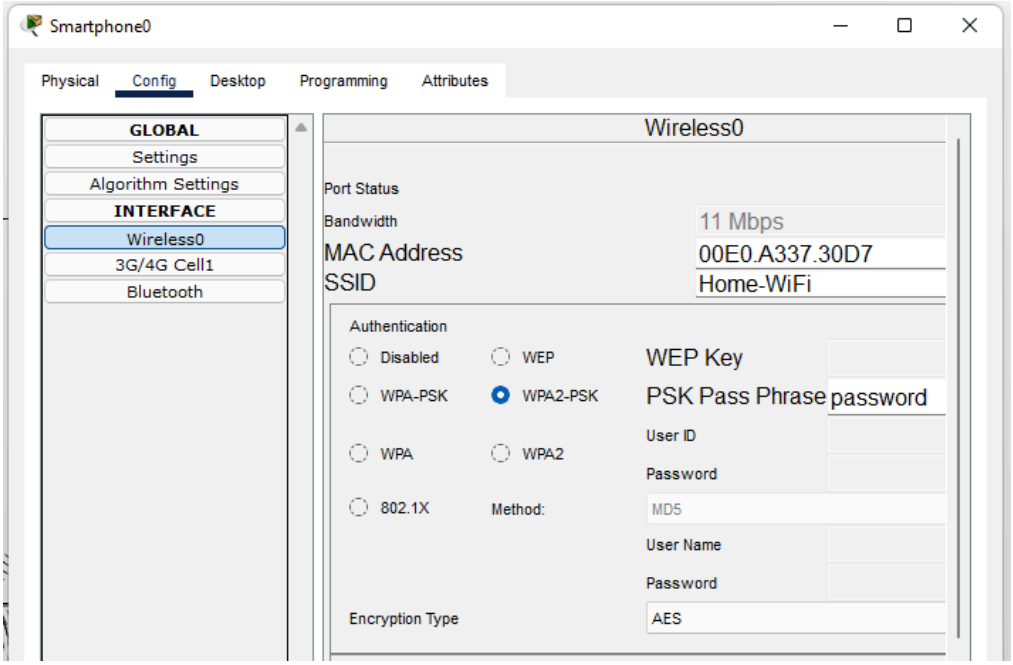

Рисунок 2.21 - Результат підключення смартфону до мережі

2.2.8 Зайдемо в список програм смартфона. Для цього натиснемо на нього лівою кнопкою миші та перейдемо у вкладку Desktop, зайдемо у програму IP Configuaration та автоматично отримуємо IP-адресу за допомогою DHCP, потім перейдемо в браузер і введемо IP-адресу IoT серверу (налаштовану адресу Home Gateway), та логін і пароль, які за замовчуванням admin, admin. Вигляд сторінки входу на рис. 2.22.

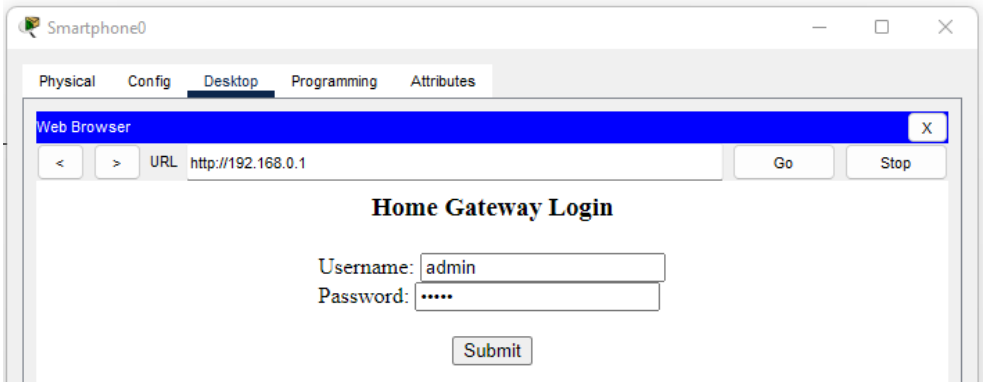

Рисунок 2.22. Вигляд сторінки входу з телефону до керування пристроями IoT

2.2.9 Натиснемо кнопку Submit та побачимо список підключених IoT пристроїв до Home Gateway. Кожним з них можна керувати (рис. 2.23).

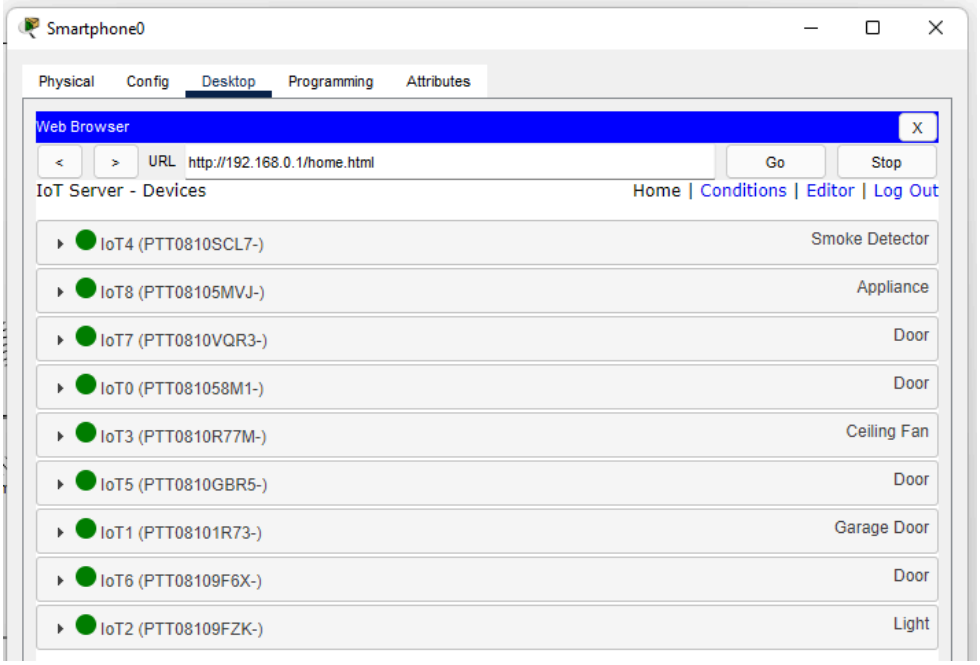

Рисунок 2.23 - Список підключених IoT пристроїв до мережі

Контрольні запитання

1) Що таке Internet of Things (IoT)?

2) Де використовуються мережі IoT?

3) Які мережеві пристрої необхідні для налаштування IoT системи?

4) Cisco Access Point – що це, та навіщо він потрібен?

5) Як налаштувати мережу з IoT пристроїв використовуючи Home Gateway?

.

Лабораторна робота № 3

Тема: Додавання IoT пристроїв до розумного будинку

Мета роботи — ознайомитись з побудовою «розумного» будинку з використанням дротових та бездротових пристроїв.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Побудова "розумного" будинку потребує інтеграції різних технологій та систем для забезпечення автоматизації, комфорту та енергоефективності. Потрібно враховувати такі ключові аспекти при плануванні та реалізації "розумного" будинку [11]:

1. **Планування систем**

Починається з визначення потреб користувача та вибору систем, які будуть інтегровані. Це можуть бути системи безпеки, енергозберігаючі системи, системи автоматизації освітлення, терморегулювання, управління побутовою технікою та інші.

2. **Комунікаційна інфраструктура**

Важливо створити мережу зв'язку, яка дозволить пристроям спілкуватися між собою. Це може бути бездротова мережа Wi-Fi, Bluetooth, Z-Wave, Zigbee або інші протоколи, що дозволяють зв'язок між пристроями.

3. **Датчики та актуатори**

Датчики вимірюють різні параметри (температуру, вологість, рух і т. п.), а актуатори виконують певні дії на основі цих даних (наприклад, ввімкнення/вимкнення освітлення).

4. **Управління та інтеграція**

Система управління координує роботу всіх пристроїв та забезпечує їх взаємодію. Це може бути центральна система керування або розумні хаби, що дозволяють об'єднувати пристрої різних виробників.

5. **Забезпечення безпеки**

З огляду на підвищений рівень підключених пристроїв, важливо враховувати заходи безпеки, такі як захист від кібератак, шифрування даних і застосування надійних паролів.

6. **Зручність для користувача**

Інтерфейси користувача, такі як мобільні додатки або голосові асистенти, роблять керування будинком зручним і доступним з різних пристроїв та місць.

7. **Енергоефективність**

Використання "розумних" систем може допомогти зменшити витрати на енергію, контролюючи освітлення, температуру, водопостачання та інші ресурси.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 3

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;

- мета роботи;

- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 3.1. Підготувати проект «розумного» будинку

3.1.1 Приклад зображено на рис. 3.1.

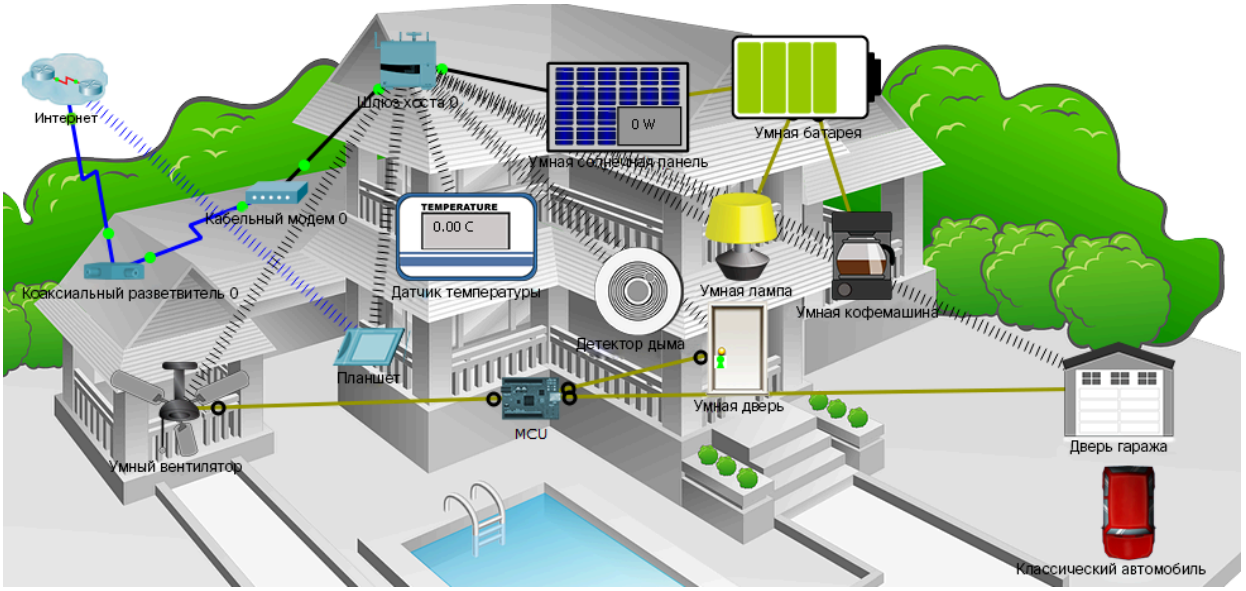

Рисунок 3.1 — Приклад проекту «розумного» будинку

3.1.2 Збережіть мережеві налаштування, які знаходяться на домашньому шлюзі. Для цього натисніть на піктограму **Home Gateway (Домашній шлюз),** щоб відкрити вікно **Home Gateway (Домашній шлюз).** Перейдіть на вкладку **Config (Конфігурація),** а потім у лівій панелі клацніть **LAN (Локальна мережа)**, щоб переглянути налаштування локальної мережі домашнього шлюзу (рис. 3.2).

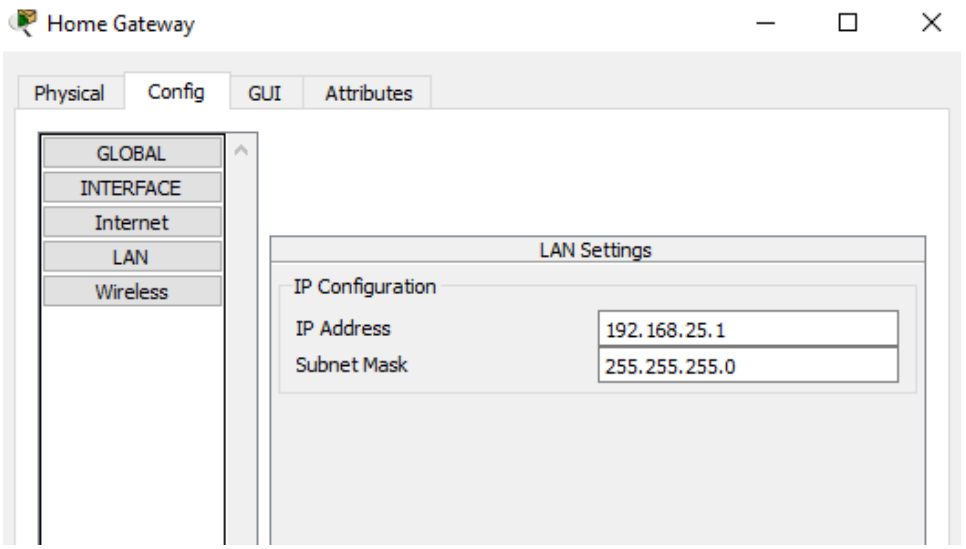

Рисунок 3.2 — Мережеві налаштування на домашньому шлюзі

3.1.3 Запишіть ідентифікатор SSID домашньої мережі та кодову фразу «WPA2-PSK» для використання у майбутньому (рис. 3.3).

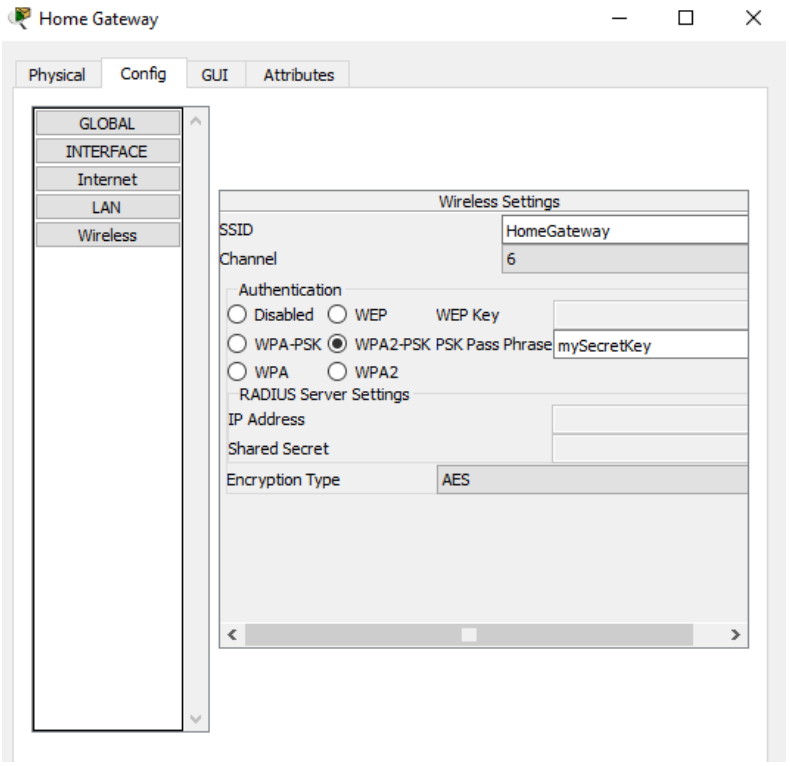

Рисунок 3.3 — Бездротова конфігурація на домашньому шлюзі

Завдання 3.2. Додати дротовий IoT пристрій до мережі «розумного» будинку

3.2.1 У полі **Device-Specific Selection (Вибір конкретного пристрою)** клацніть піктограму **Lawn Sprinkler (Установка для поливу),** після чого клацніть у місці робочого простору, де необхідно розмістити **Lawn Sprinkler (установку для поливу).**

3.2.2 У полі **Device-Type Selection (Вибір типу пристрою)** клацніть піктограму **[Connections] (Підключення)** (вона виглядає як блискавка). Клацніть піктограму типу роз'єму **Copper Straight Through (Мідний прямий)** у полі **Device-Specific Selection (Вибір конкретного пристрою).** Далі клацніть піктограму **Sprinkler (Розбризкувач)** та підключіть один кінець кабелю до інтерфейсу FastEthernet0 розбризкувача. Тепер клацніть піктограму **Home Gateway (Домашній шлюз)** та підключіть інший кінець кабелю до вільного інтерфейсу Ethernet (рис. 3.4).

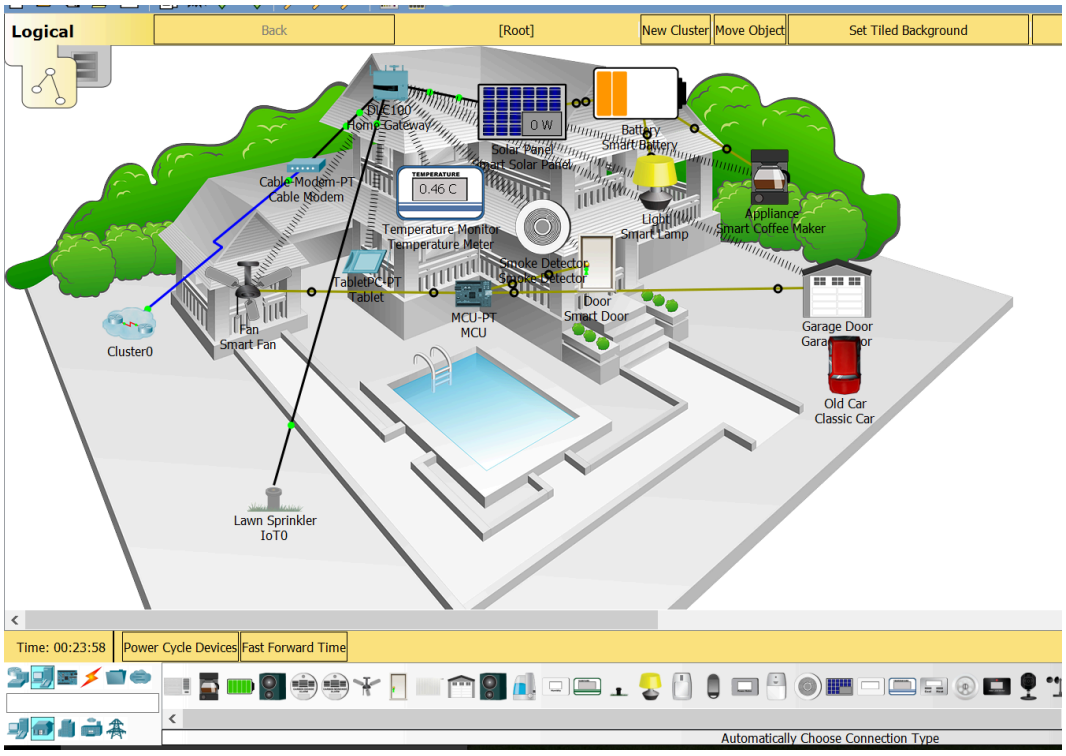

Рисунок 3.4 — Приклад дротового підключення

3.2.3 Натисніть піктограму **Lawn Sprinkler (Установка для поливу)** у робочому просторі, щоб відкрити вікно пристрою. Зауважте, що зараз установка для поливу має універсальне ім'я IoT0.

3.2.4 Змініть назву з “IoT0” на “Sprinkler1”

3.2.5 Встановіть Home Gateway в якості IoT серверу (рис. 3.5)

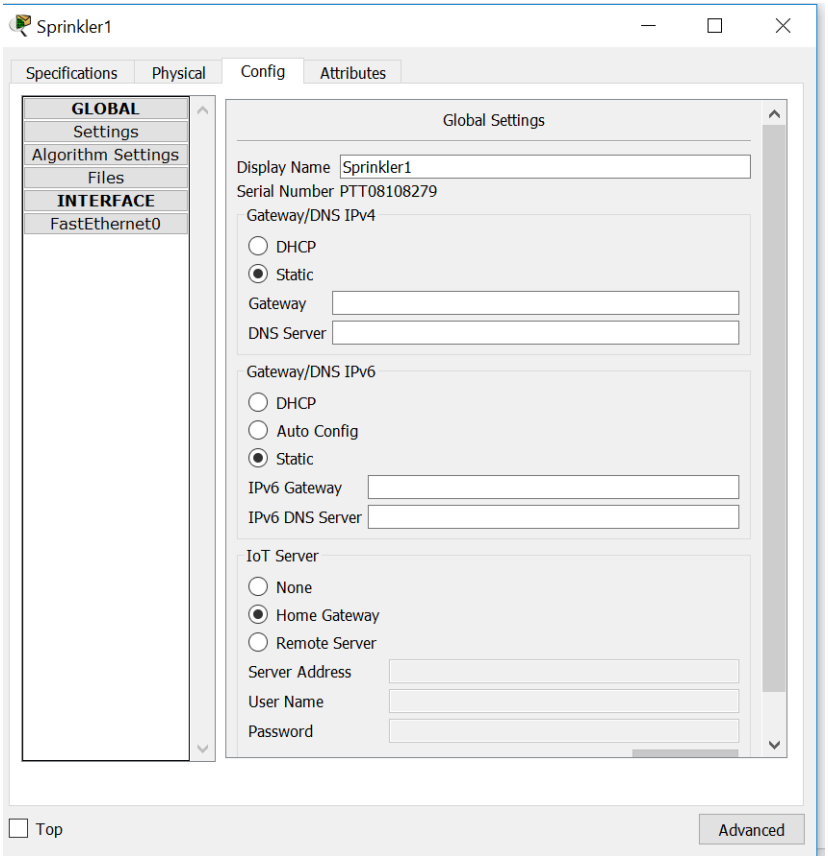

Рисунок 3.5 — Приклад налаштування установки для поливу

3.2.6 Клацніть FastEthernet0 та змініть значення параметра **IP Configuration (Конфігурація IP)** на DHCP (рис. 3.6).

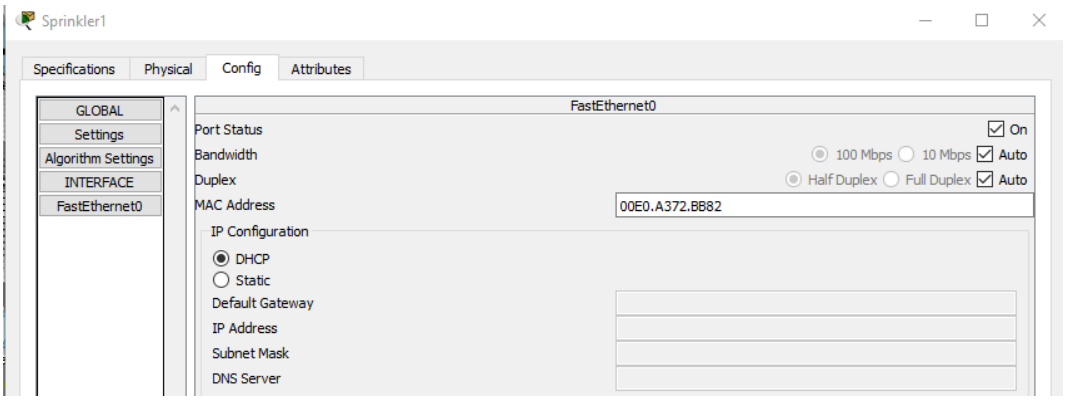

Рисунок 3.6 — Використання DHCP на установці для поливу

3.2.7 Переконайтеся, що установка для поливу є в мережі. Увійдіть до **Home Gateway (Домашній шлюз)** з вікна **Tablet (Планшет).** Пристрій

Sprinkler1 має бути вказаний у списку IoT Server - Devices (Сервер IoT — пристрої) (рис. 3.7).

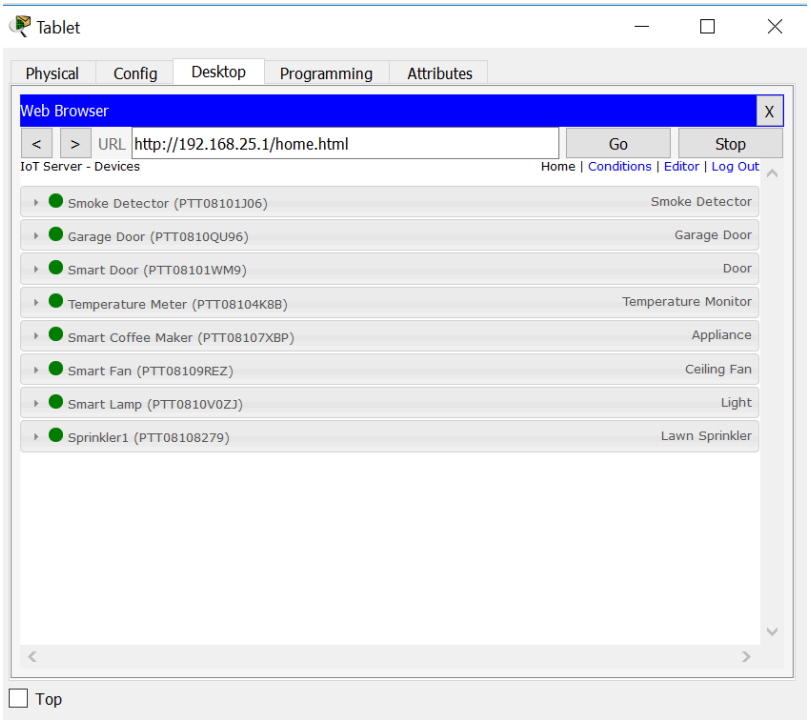

Рисунок 3.7 — Установка для поливу у списку на IoT Server

Завдання 3.3. Додавання до мережі бездротових пристроїв

3.3.1 У полі **Device-Specific Selection (Вибір конкретного пристрою)** клацніть піктограму **Wind Detector (Датчик вітру),** а потім клацніть у місці робочого простору, де потрібно розмістити цей датчик вітру.

3.3.2 Додайте бездротовий модуль до датчика вітру. Натисніть на піктограму **Wind Detector (Датчик вітру)** у робочому просторі, щоб відкрити вікно IoT. У нижньому правому куті вікна пристрою IoT натисніть кнопку **Advanced (Додатково).** Зауважте, що у верхній частині вікна з'являються додаткові вкладки. Перейдіть на вкладку **I/O Config (Налаштування вводу-виводу)**. У списку **Network Adapter (Мережний адаптер)** виберіть PT-IOT-NM-1W (бездротовий адаптер).

3.3.3 Налаштуйте на датчику вітру підключення до бездротової мережі. Клацніть вкладку **Config (Налаштування).** У полі Display Name

(Відображуване ім'я) введіть Wind_Detector та змініть значення поля IoT Server (сервер IoT) на Home Gateway (Домашній шлюз) (рис. 3.8).

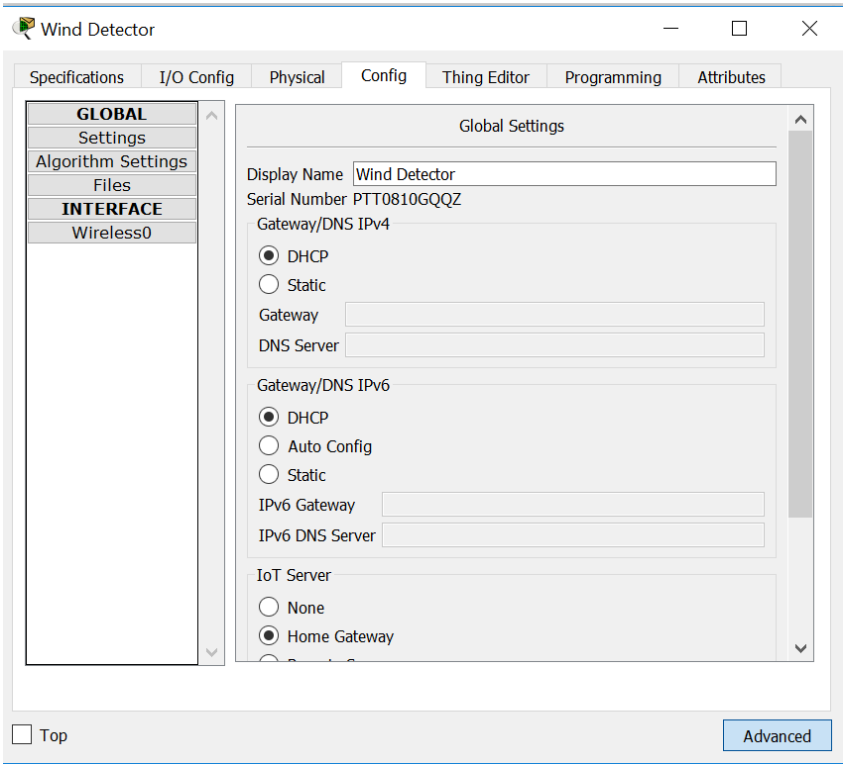

Рисунок 3.8 — Приклад налаштування датчику вітру

3.3.4 Потім клацніть Wireless0 у лівій панелі. Змініть тип автентифікації на WPA2-PSK і введіть mySecretKey в поле PSK Pass Phrase (Кодова фраза PSK). Саме ці налаштування бездротового зв'язку, задані на домашньому шлюзі. Потрібно встановити бездротове підключення між датчиком вітру та домашнім шлюзом.

3.3.5 Перевірте, що датчик вітру з’явився у списку пристроїв на IoT Server.

Контрольні запитання

1) Що таке актуатори?

2) Яка різниця між датчиками та актуаторами?

3) Наведіть приклади та сфери застосування датчиків та актуаторів

4) Яким чином "розумні" системи можуть підвищувати енергоефективність?

5) В яких випадках доцільно використовувати дротове з'єднання замість бездротового?

Лабораторна робота №4

Тема: Підключення пристроїв IoT та моніторинг їх роботи

Мета роботи — отримати навички підключення IoT пристроїв до мережі та проводити їх моніторинг з використанням пристроїв користувача.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Концепція Інтернету речей (IoT) відкриває безліч можливостей для підключення різноманітних пристроїв до Інтернету з метою збору даних, автоматизації процесів та покращення якості життя. До пристроїв IoT можна віднести датчики, домашні пристрої, медичні пристрої, транспортні засоби та багато інших речей, які обов’язково мають вихід в Інтернет [2].

Підключення пристроїв IoT складається з:

1. Протоколи зв'язку

Протоколи, такі як Wi-Fi, Bluetooth, Zigbee, Z-Wave, LoRaWAN, NB-IoT тощо, використовуються для забезпечення зв'язку між пристроями IoT та мережею Інтернету [3].

2. Хмарні платформи

Вони надають середовище для збору, збереження та обробки даних, зібраних від пристроїв IoT. Такі платформи дозволяють моніторити та керувати пристроями віддалено через Інтернет.

3. Безпека

Забезпечення безпеки важливе для захисту пристроїв IoT від кібератак та несанкціонованого доступу. Застосування шифрування, аутентифікація, оновлення програмного забезпечення є ключовими аспектами безпеки в IoT.

4. API та протоколи взаємодії

Використання стандартних API та протоколів (наприклад, MQTT, CoAP, HTTP) дозволяє різним пристроям спілкуватися та обмінюватися даними між собою та з серверами.

Моніторинг роботи пристроїв IoT:

1. Відстеження даних

Збір та аналіз даних, що надходять від пристроїв IoT, дозволяє виявляти аномалії, прогнозувати проблеми та оптимізувати роботу системи в цілому.

2. Віддалений моніторинг

Застосування спеціальних програмних рішень або платформ дозволяє віддалено спостерігати за станом пристроїв IoT, виявляти проблеми та вживати відповідних заходів без присутності фізично на місці.

3. Управління даними та аналітика

Застосування аналітики даних дозволяє отримувати цінний інсайт із зібраних даних, що полегшує прийняття рішень щодо покращення роботи системи IoT.

4. Масштабованість та оптимізація

Моніторинг дозволяє виявляти обсяги роботи пристроїв та оптимізувати їх функціонування для забезпечення ефективності та масштабованості.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 4

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;

- мета роботи;

- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 4.1. Додавання до мережі домашнього шлюзу

4.1.1 Клацніть піктограму **Wireless Devices (Безпроводові пристрої)** у полі **Device-Type Selection (Вибір типу пристрою)**. Натисніть піктограму **Home Gateway (Домашній шлюз),** а потім натисніть логічний робочий простір, щоб додати пристрій (рис. 4.1).

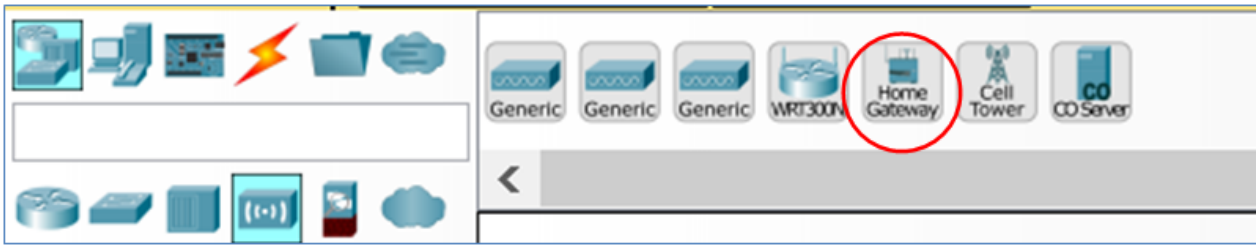

Рисунок 4.1 — Розташування елементу Home Gateway

4.1.2 Клацніть піктограму з'єднувача **Copper Straight-Through (Медний прямий)** у полі **Device-Type Selection (Вибір типу пристрою),** а потім натисніть домашній шлюз, щоб підключити один кінець кабелю. Далі клацніть **Cable Modem (Кабельний модем)**, щоб підключити інший кінець кабелю до порту Internet (Інтернет) (рис. 4.2).

Рисунок 4.2 — Розташування елементу Copper Straight-Through

4.1.3 Через декілька секунд на обох кінцях кабелю повинні загорітися зелені індикатори, вказуючи на те, що підключення встановлено (рис. 4.3).

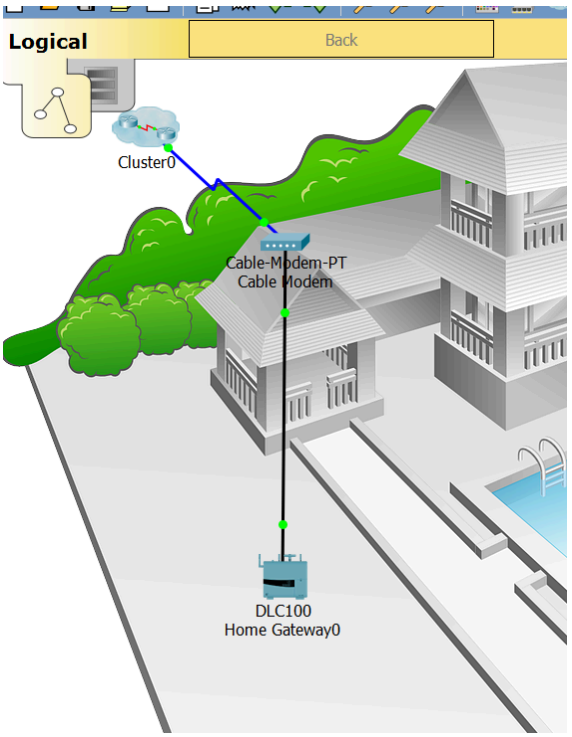

Рисунок 4.3 — Зелені індикатори після встановлення підключення

Завдання 4.2. Додавання до бездротової мережі IoT пристроїв

4.2.1 Клацніть піктограму **Home Devices (Домашні пристрої)** у полі **Device-Type Selection (Вибір типу пристрою)** та додайте у робочий простір пристрої **Fan (Вентилятор), Door (Двері)** та **Lamp (Лампа)** (рис. 4.4)**.**

Рисунок 4.4 — Розташування елементів Вентилятор, Двері та Лампа

4.2.2 Клацніть піктограму **Fan (Вентилятор)** у робочому просторі, щоб відкрити вкладку **Config (Конфігурація),** а потім натисніть кнопку **Advanced (Додатково)** у нижньому правому куті вікна. Зауважте, що вкладки у верхній частині вікна конфігурації змінилися. З'явилися додаткові вкладки.

4.2.3 Перейдіть на вкладку **I/O Config (Налаштування вводу-виводу)** і в полі **Network Adapter (Мережевий адаптер)** змініть тип на безпровідний адаптер PT-IOT-NM-1W (рис. 4.5).

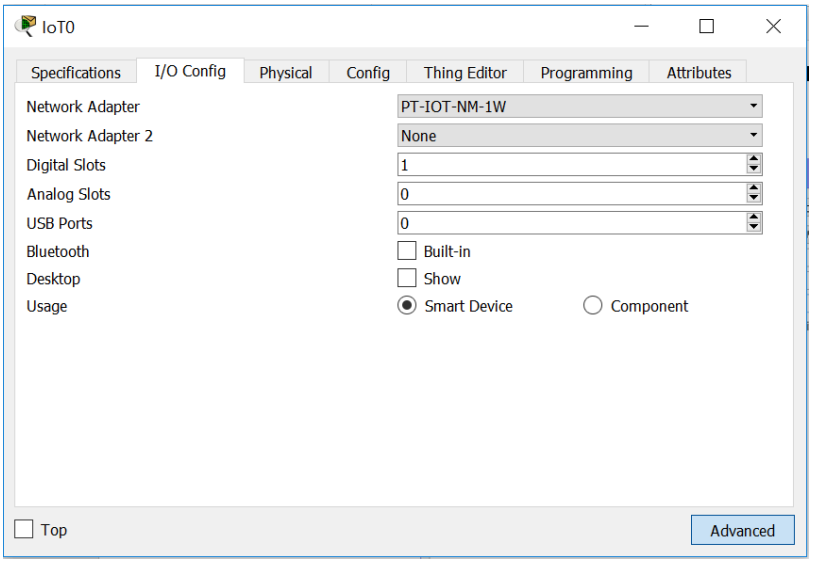

Рисунок 4.5 — Налаштування бездротового адаптера

4.2.4 Клацніть вкладку **Config (Налаштування).** У полі **Display Name (Ім'я, що відображається)** введіть **Ceiling Fan (Стельовий вентилятор)** (рис. 4.6)**.**

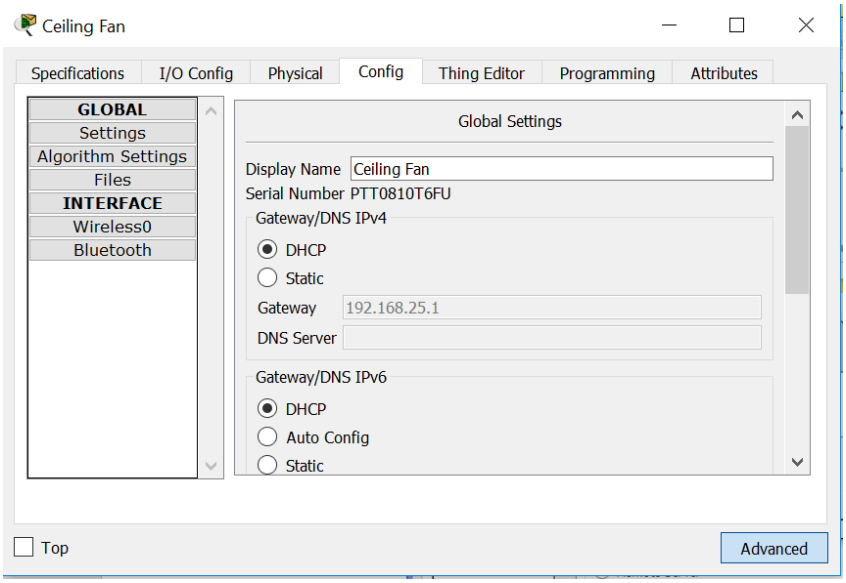

Рисунок 4.6 — Зміна ім’я для вентилятору

4.2.5 Перейдіть на вкладку **Config (Конфігурація),** клацніть інтерфейс Wireless0 у лівій панелі.

4.2.6 У параметрах конфігурації мережа HomeGateway повинна бути у списку, наведеному в полі SSID. Переконайтеся, що в параметрах **IP Configuration (Конфігурація IP)** встановлено прапорець DHCP, вказано IP-адресу 192.168.25.100 та стандартний шлюз 192.168.25.1 (рис. 4.7). Це означає, що вентилятор підключений до мережі та отримує конфігураційні дані IP-адреси від домашнього шлюзу.

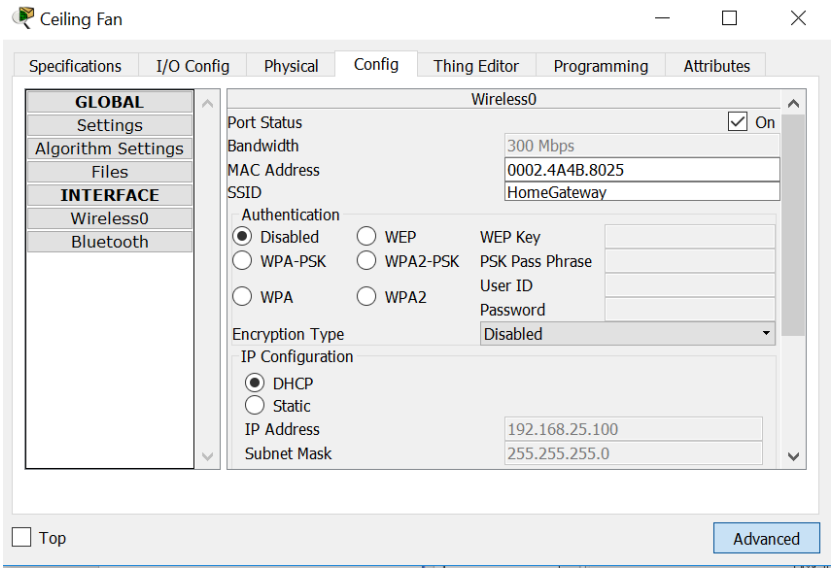

Рисунок 4.7 — Мережеві налаштування вентилятору

4.2.7 Підключіть пристрої **Door (Двері)** та **Lamp (Лампа)** до бездротової мережі, виконавши ті самі кроки, що й для вентилятора.

Завдання 4.3. Додавання до мережі бездротового планшета

4.3.1 Клацніть піктограму **End Devices (Кінцеві пристрої)** у полі **Device-Type Selection (Вибір типу пристрою)** та додайте до робочого простору **Wireless Tablet (Безпровідний планшет)** (рис. 4.8)**.**

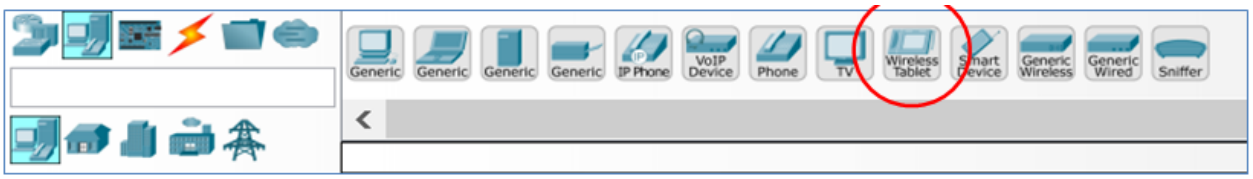

Рисунок 4.8 — Розташування елемента Wireless Tablet

4.3.2 Натисніть піктограму **Tablet (Планшет),** щоб відкрити вікно налаштування планшета.

4.3.3 Перейдіть на вкладку **Config (Конфігурація)** та клацніть інтерфейс Wireless0. У полі SSID змініть значення **Default (За замовчуванням)** на HomeGateway. Після зміни ідентифікатора мережі SSID планшет повинен протягом декількох секунд отримати IP-адресу через DHCP (рис. 4.9).

Рисунок 4.9 — Мережеві налаштування планшету

4.3.4 Перейдіть на вкладку **Desktop (Робочий стіл)** і клацніть піктограму **Web Browser (Веб-браузер)**, щоб відкрити браузер. Введіть 192.168.25.1 (адреса домашнього шлюзу) у полі URL і натисніть **Go (Перейти).**

4.3.5 На сторінці **Home Gateway Login (Вхід до домашнього шлюзу)** введіть ***admin*** як ім'я користувача та ***admin*** як пароль і натисніть **Submit (Надіслати),** щоб підключитися до сервера домашнього шлюзу. Зверніть

увагу, що у списку **IoT Server - Devices (Сервер IoT - пристрої)** домашнього шлюзу немає пристроїв (рис. 4.10).

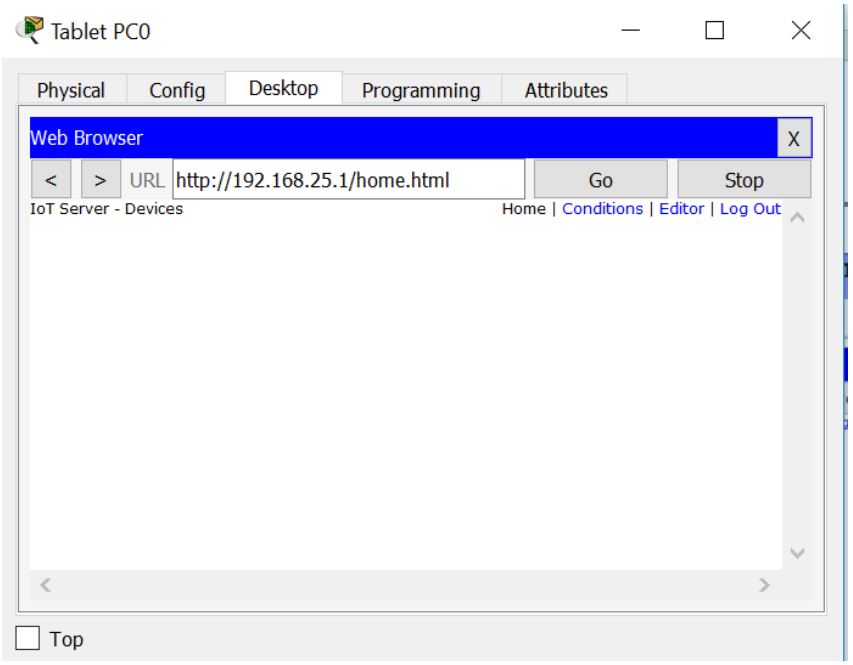

Рисунок 4.10 — Список IoT пристроїв на IoT Server

4.3.6 Клацніть піктограму **Fan (Вентилятор)** у робочому просторі, відкрийте вкладку **Config (Конфігурація)** та виберіть **Settings (Параметри)** у лівій панелі. У списку параметрів **IoT Server (сервер IoT)** натисніть кнопку **Home Gateway (Домашній шлюз)** (рис. 4.11)**.**

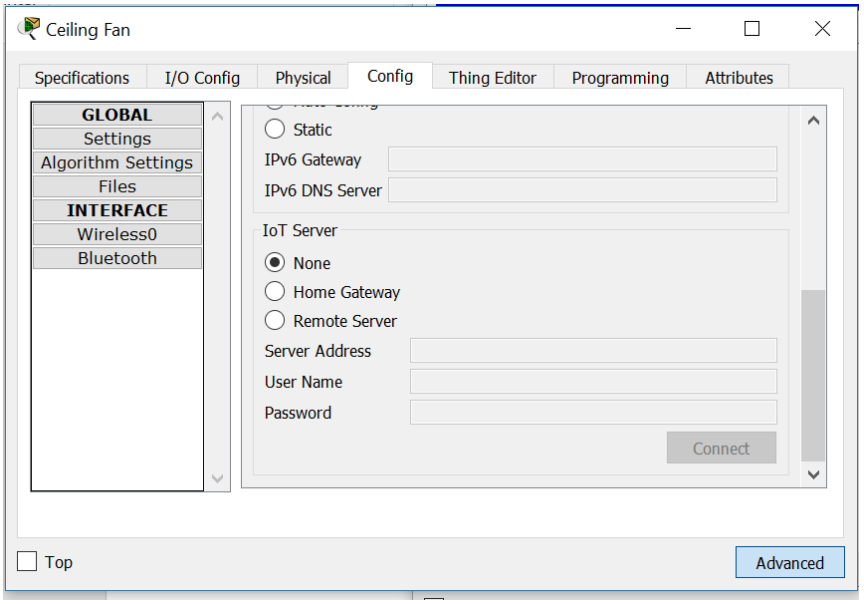

Рисунок 4.11 — Налаштування IoT Server на Вентиляторі

4.3.7 Підключіть всі інші пристрої до IoT Server.

4.3.8 Перевірте, що всі пристрої є у списку на IoT Server (рис. 4.12).

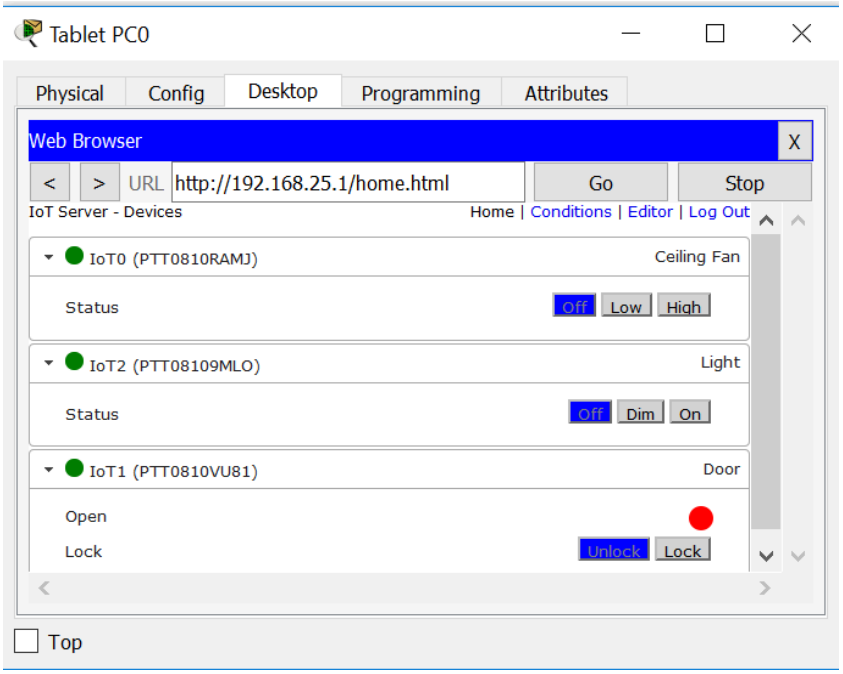

Рисунок 4.12 — Відображення IoT пристроїв на IoT Server

Контрольні запитання

1) Які протоколи існують для забезпечення безпеки трафіку в бездротових мережах?

2) Чому можуть виникати колізійні ситуації при передачі даних у бездротових мережах?

2) Що таке CSMA/CD та як це вирішує колізійні ситуації?

3) Які існують бездротові технології передачі даних?

4) Для чого використовуються мережеві адаптери?

5) Яка задача IoT Server?

Лабораторна робота № 5

Тема: Побудова моделі «розумної» кімнати за допомогою Cisco Packet Tracer

Мета роботи — отримати навички проектування та моделювання «розумних» приміщень із використанням Cisco Packet Tracer.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Моделювання "розумної" кімнати в контексті Internet of Things (IoT) передбачає інтеграцію різноманітних сенсорів, пристроїв та систем для забезпечення зручності, безпеки та ефективності управління простором. Основна мета - створити оточення, яке може реагувати на потреби користувача та оптимізувати використання ресурсів [2].

Основні етапи моделювання "розумної" кімнати:

1. **Сенсори та датчики:**

- Встановлення різних типів сенсорів, таких як датчики температури, вологості, руху, освітлення та інших.
- Інтеграція цих сенсорів в мережу IoT для збору та передачі даних.

2. **З'єднання та мережа:**

- Використання пристроїв IoT та протоколів зв'язку для створення бездротової мережі [3].
- Забезпечення безпеки мережі для захисту від несанкціонованого доступу.

3. **Управління освітленням та теплотою:**

- Застосування системи "розумного" освітлення, яка реагує на наявність людей та враховує природне освітлення.

- Встановлення терморегуляторів, які регулюють температуру відповідно до установок користувача та умов.

4. **Автоматизація засобів безпеки:**

- Використання відеоспостереження та рухових датчиків для забезпечення безпеки в кімнаті.
- Повідомлення користувача в разі виявлення небезпеки або неправомірного доступу.

5. **Управління аудіо- та відеосистемами:**

- Використання системи "розумного" аудіо та відео, яка може адаптуватися до вибору користувача та умов оточення.

6. **Системи автоматизованого управління:**

- Розробка інтелектуальної системи керування, яка може навчатися від реакцій користувача та оптимізувати параметри для забезпечення максимального комфорту та ефективності.

7. **Інтерфейс користувача:**

- Розробка зручного інтерфейсу для взаємодії з системою, можливо, використовуючи мобільний додаток або голосовий інтерфейс.

8. **Віддалене керування:**

- Забезпечення можливості віддаленого керування системою через Інтернет, щоб користувач міг моніторити та керувати "розумною" кімнатою навіть здалеку.

Моделювання "розумної" кімнати здійснюється в контексті поєднання різноманітних технологій IoT, від датчиків і пристроїв до хмарових платформ та штучного інтелекту. Такий підхід дозволяє створити інтелектуальне середовище, яке може адаптуватися до потреб користувача та оптимізувати використання ресурсів [2].

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 5

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов'язково повинен містити наступну інформацію:

- назва лабораторної роботи;

- мета роботи;

- постановка завдання і алгоритм його розв'язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 5.1. Спроектувати «розумну кімнату» з використанням таких пристроїв, як домашній шлюз (Home Gateway), вентилятор та вікно

5.1.1 Перейдіть до розділу "Мережеві пристрої", виберіть підкатегорію "Бездротові пристрої", там ви знайдете "Домашній шлюз (Home Gateway)". Просто перетягніть його на робочу область (рис. 5.1).

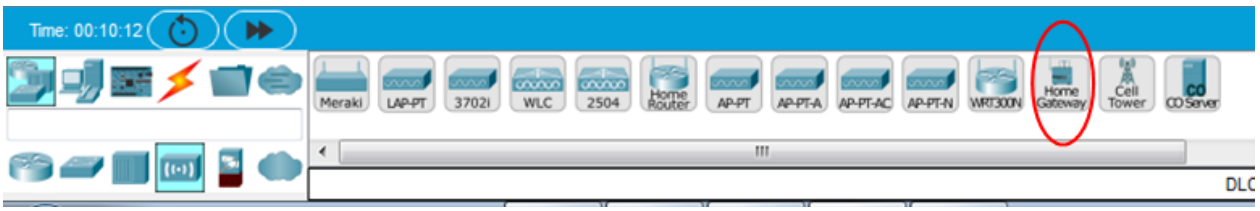

Рисунок 5.1 — Розташування елементу Домашній шлюз

5.1.2 У вкладці "Кінцеві пристрої" перейдіть до підрозділу "Будинок" та виберіть "Вентилятор". Перетягніть його на робочу область (рис. 5.2).

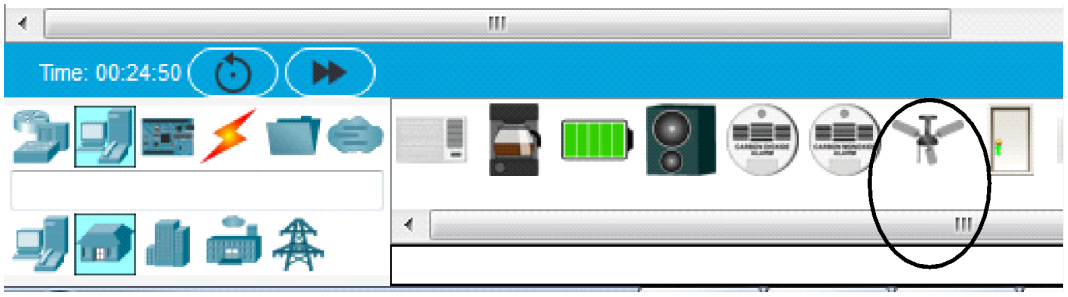

Рисунок 5.2 — Розташування елементу Вентилятор

5.1.3 На піктограмі "Вентилятора" натисніть лівою кнопкою миші. Коли відкриється вікно властивостей, перейдіть до вкладки "Config". Там знайдете опцію для зміни імені пристрою. Змініть його на "Ceiling fan", як показано на рисунку 5.3.

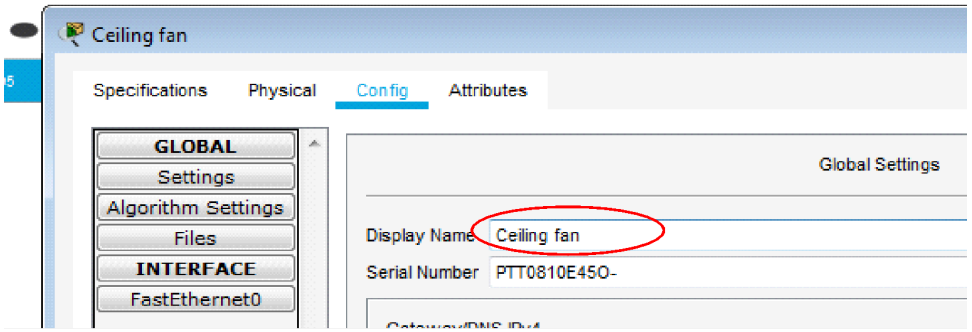

Рисунок 5.3 — Зміна назви елементу Вентилятор

5.1.4 В кінці списку пристроїв підрозділу "Будинок" виберіть елемент "Вікно" (Window). Перетягніть його на робоче поле (рис. 5.4).

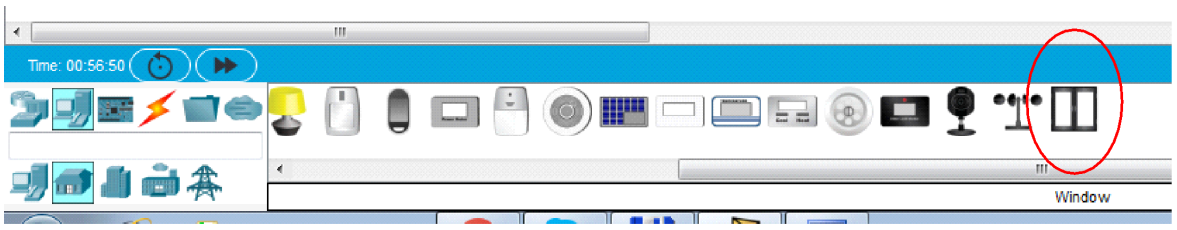

Рисунок 5.4 — Розташування елементу Вікно

5.1.5 Змініть назву на «Window» використовуючи аналогічні кроки, як для вентилятора (рис. 5.5).

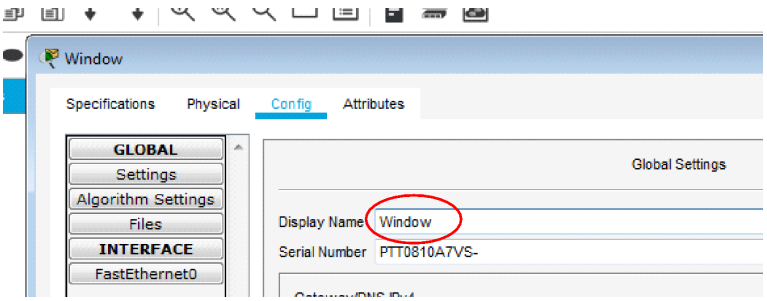

Рисунок 5.5 — Зміна назви елементу Вікно

5.1.6 Просто обведіть прямокутником кожен елемент кімнати по черзі, виконавши дії, зображені на рисунку 5.6.

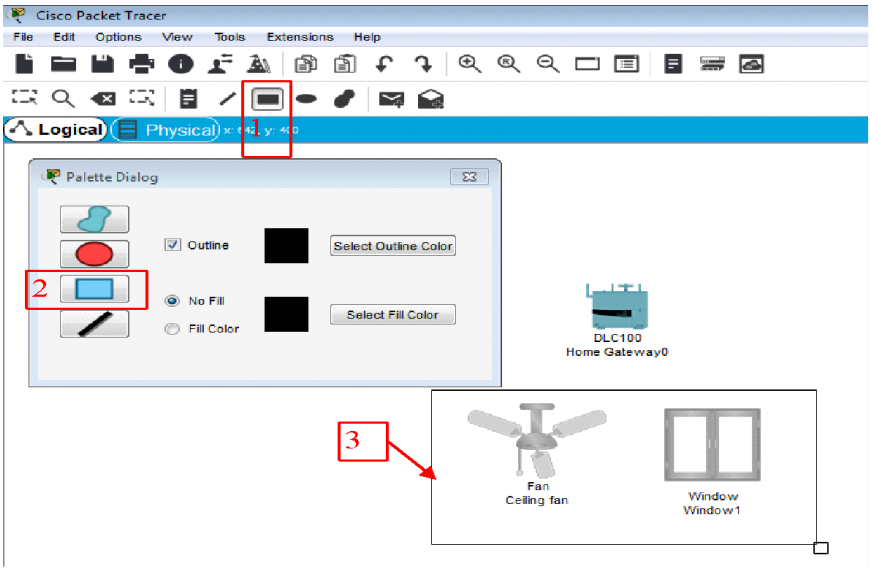

Рисунок 5.6 — Обводка елементів кімнати

5.1.7 Зробіть мітку назви кімнати (рис. 5.7).

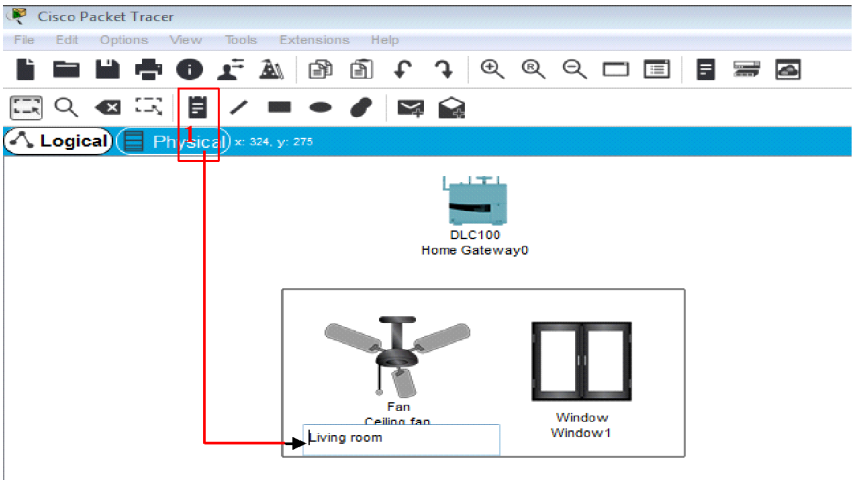

Рисунок 5.7 — Нанесення назви кімнати

5.1.8 Налаштуємо бездротову мережу, щоб під'єднати елементи до Домашнього шлюза. Натисніть на елемент вентилятор. У нижньому правому куту повинна бути кнопка "Розширено" (Advanced). Клікніть по ній. Встановіть мережевий адаптер, як показано на рисунку 5.8.

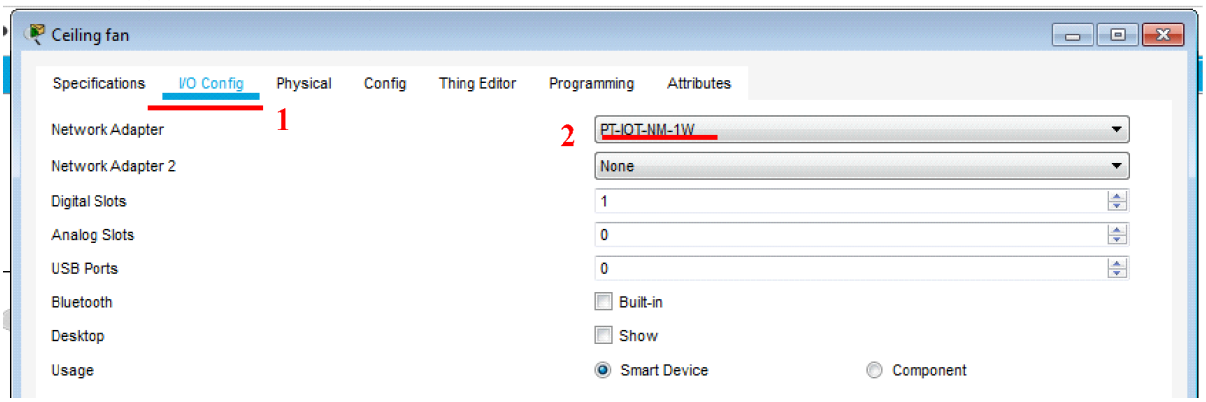

Рисунок 5.8 — Налаштування мережевого адаптера на Вентиляторі

5.1.9 Зробіть те ж саме для елемента «Вікно» (рис. 5.9).

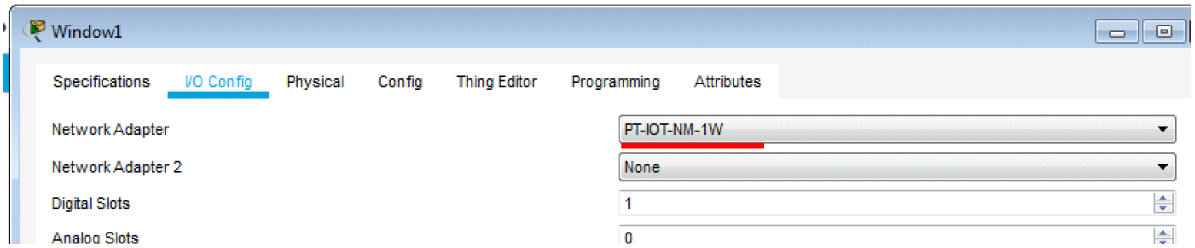

Рисунок 5.9 — Налаштування мережевого адаптера на Вікні

5.1.10 У результаті отримаємо підключення, як показано на рис. 5.10.

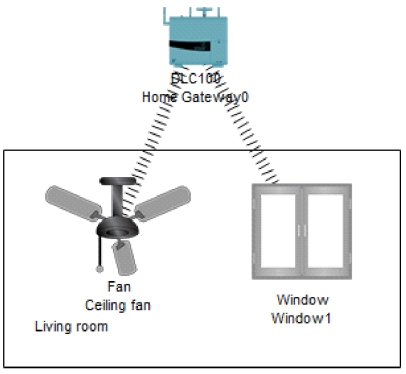

Рисунок 5.10 — Результат налаштування мережевих адапторів

5.1.11 Оберемо смартфон для здійснення управління “розумними” пристроями (рис. 5.11).

Рисунок 5.11 — Розташування елементу Смартфон

5.1.12 Розміщуємо смартфон в робочому полі, клікаємо на іконці та налаштовуємо підключення до Домашнього шлюзу (Home Gateway), виконуючи дії 1-3 (рис. 5.12).

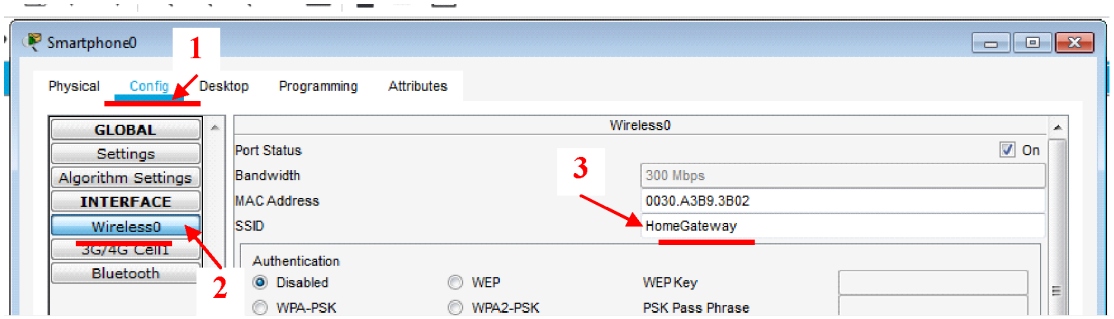

Рисунок 5.12 — Конфігурування параметрів мережі на Смартфоні

5.1.13 Відкриваємо вкладку IoT Monitor на смартфоні (рис. 5.13).

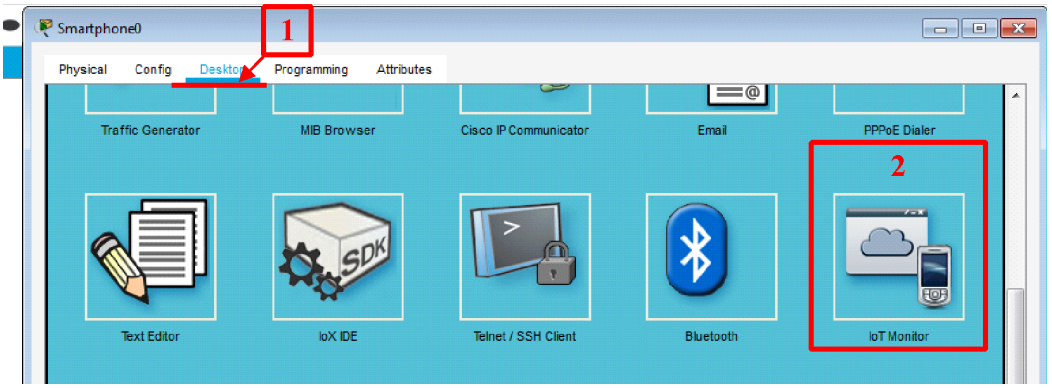

Рисунок 5.13 — Відкриття IoT Monitor через вкладку Desktop на смартфоні

5.1.14 IP-адреси Монітора смартфона та шлюзу повинні співпадати (рис. 5.14 та 5.15).

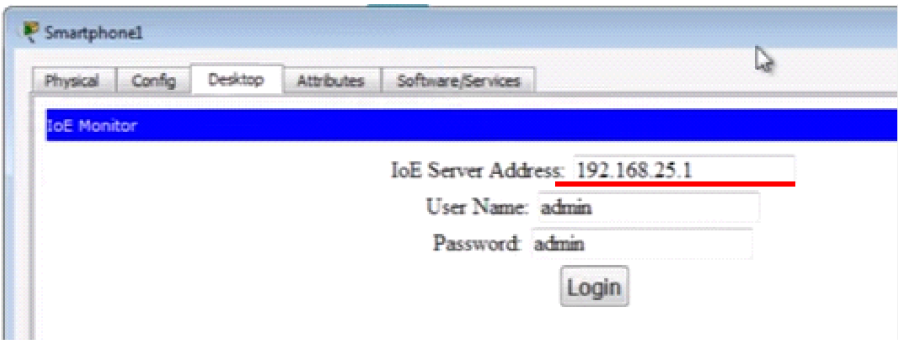

Рисунок 5.14 — IP-адреса в IoT Monitor на смартфоні

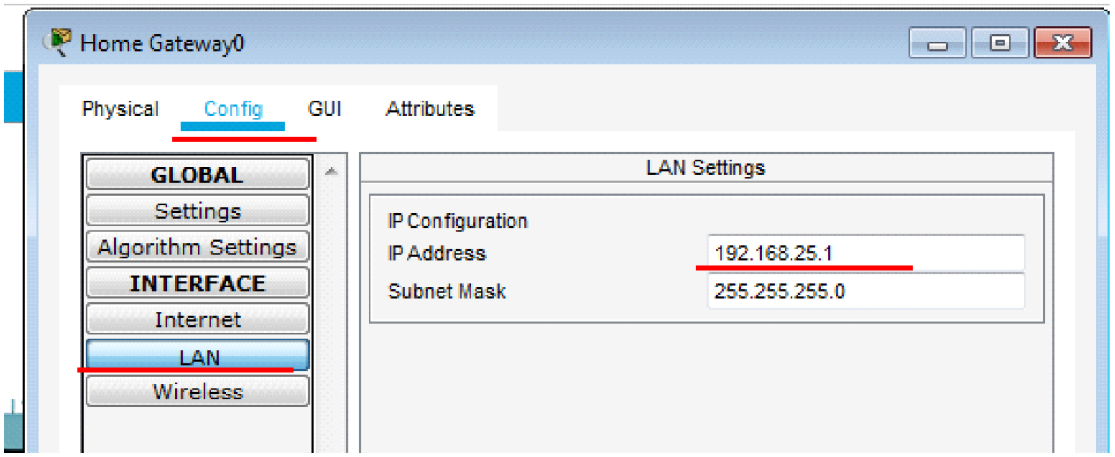

Рисунок 5.15 — Налаштована IP-адреса на Домашньому шлюзі

5.1.15 Знайдіть елемент “Вентилятор” (fan) на робочому полі та клікніть на нього. У вкладці Config прокручуємо повзунок до самого кінця та натискаємо на кнопку Домашній шлюз (Home Gateway) (рис. 5.16).

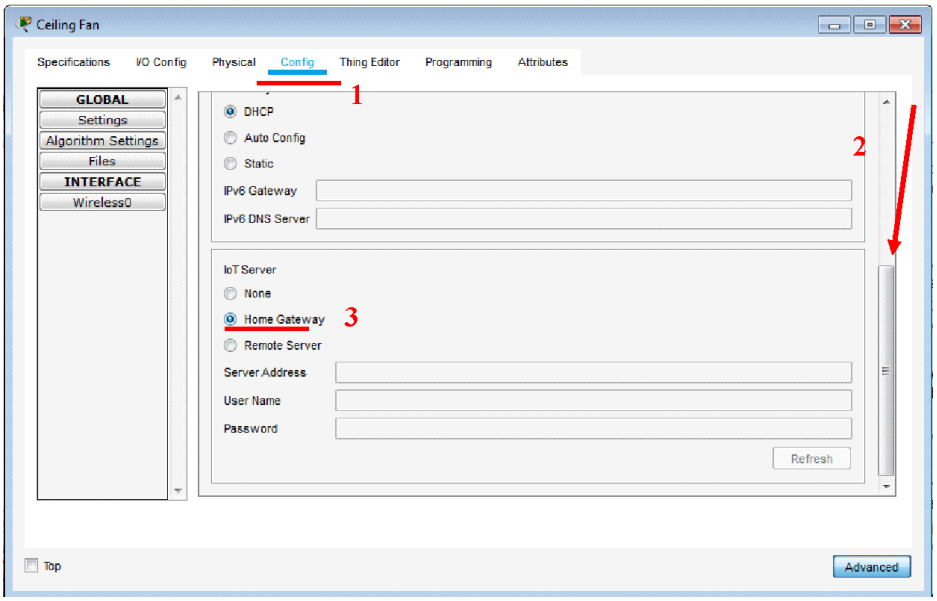

Рисунок 5.16 — Налаштування IoT Server на Вентиляторі

5.1.16 Те ж саме робимо для «Вікна» (Window). Заходимо в IoT Monitor на смартфоні, щоб перевірити підключення та можливість керування (рис. 5.17).

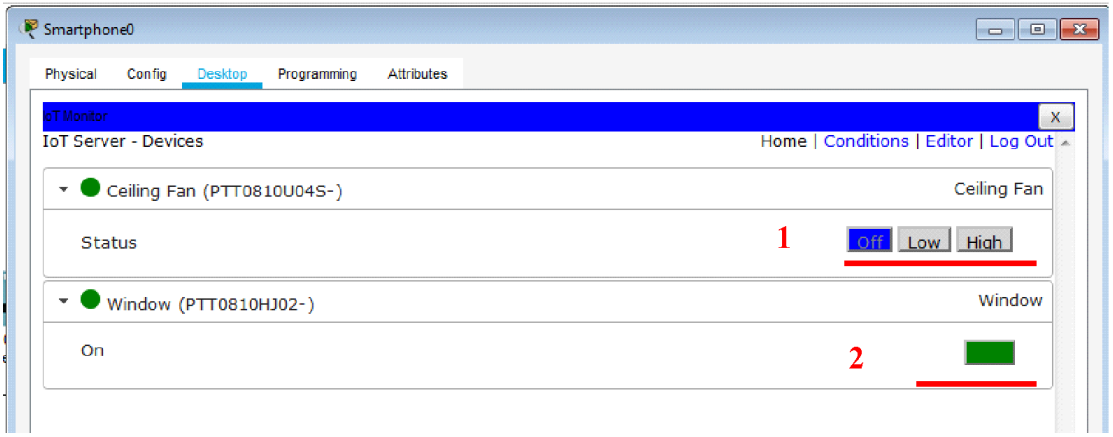

Рисунок 5.17 — Вікно IoT Monitor після підключення пристроїв

5.1.17 Пристроями тепер можна керувати, натискючи на їх підключення. На рис. 5.18 зображено приклад зміни руху вентилятора та відкривання/закривання вікна.

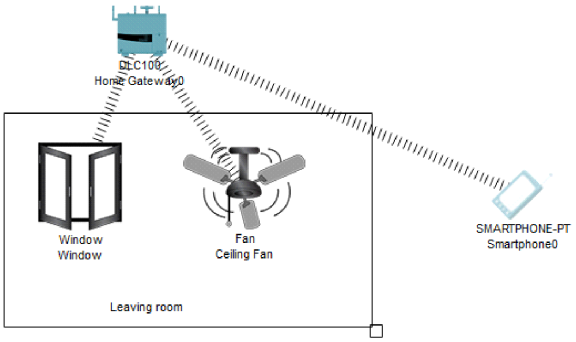

Рисунок 5.18 – Керування пристроями через смартфон

Завдання 5.2. Моделювання регулювання температури в IoT кімнаті

5.2.1 Додаємо термостат до готової кімнати (рис. 5.19).

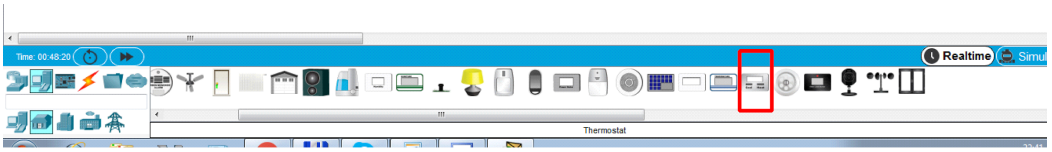

Рисунок 5.19 — Розташування Термостату

5.2.2 В якості дротового з’єднання використаємо мідний кабель (рис. 5.20).

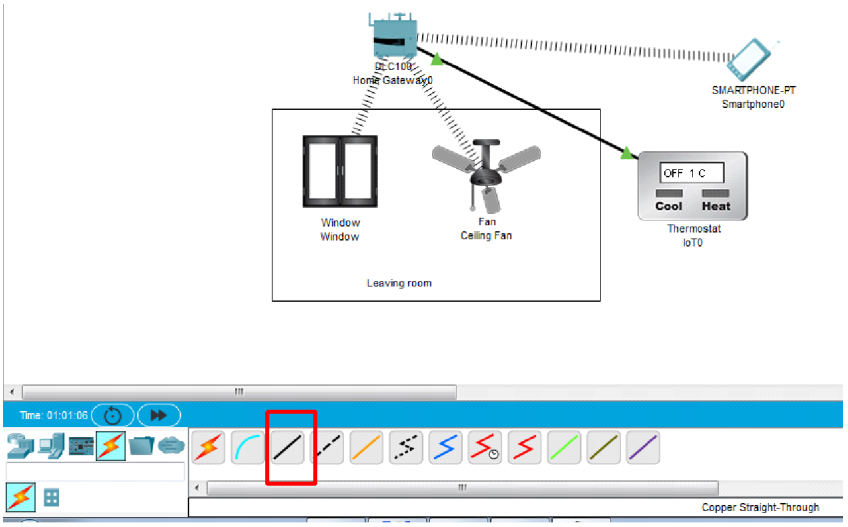

Рисунок 5.20 — Дротове з’єднання між Домашнім шлюзом та Термостатом

5.2.3 Налаштовуємо конфігурацію керування термостатом на IoT Server. Натискаємо «Condition» (рис. 5.21).

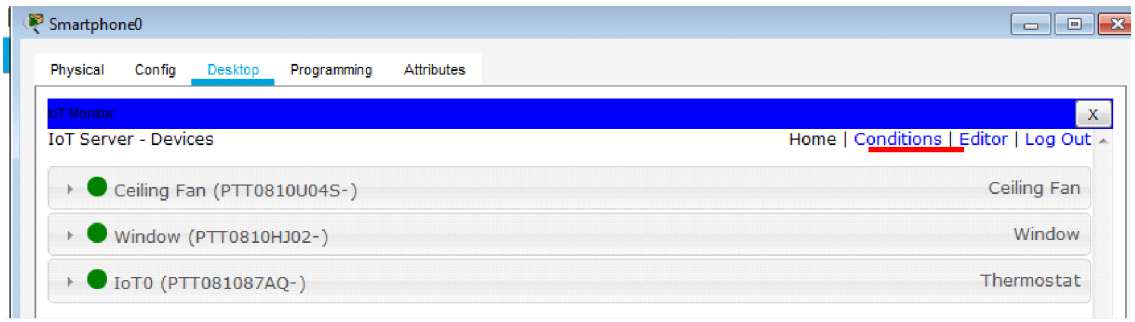

Рисунок 5.21 — Вкладка умов для керування пристроями

5.2.4 Заповнюємо послідовно всі поля, як на рис. 5.22. Це правило наштовує вентилятор на автоматичне включення при досягненні певного граничного значення.

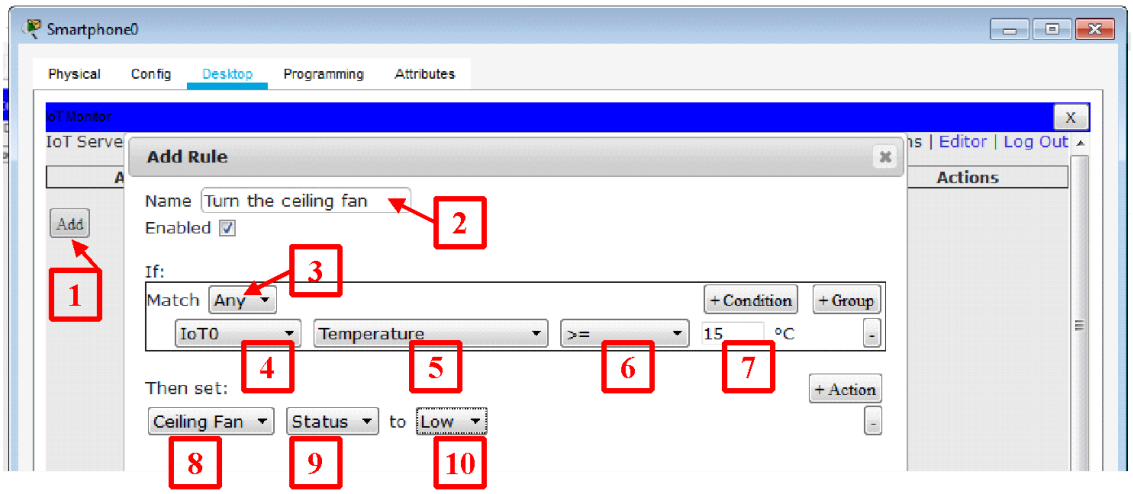

Рисунок 5.22 — Налаштування правила включення Вентилятора

5.2.5 Додаємо ще одне правило, щоб вентилятор так само автоматично переставав працювати (рис. 5.23).

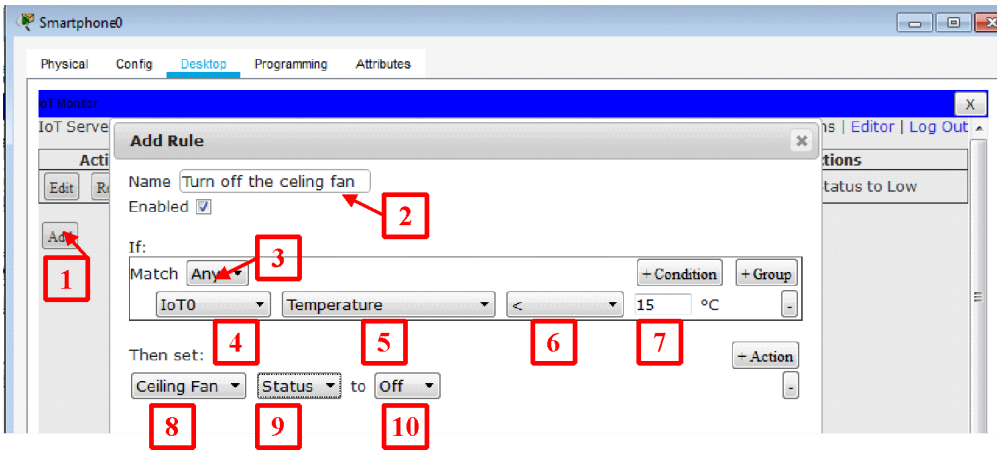

Рисунок 5.23 — Автоматичне відключення вентилятора при умові

5.2.6 В результаті повинні відображатись два правили у списку (рис. 5.24).

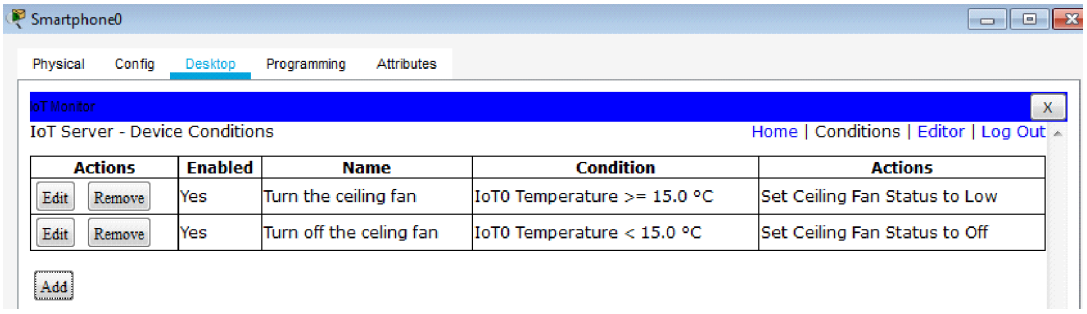

Рисунок 5.24 — Список правил для Вентилятора

5.2.7 Ці налаштування дозволять пристрою автоматично визначати необхідність того чи іншого стану в залежності від поточного значення температури.

Завдання 5.3. Моделювання IoT мережі гаража

5.3.1 Додаємо та підключаємо детектор диму (Smoky detector).

5.3.2 Додаємо та підключаємо пожежний розпилювач (Fire Sprinkler).

5.3.3 Додаємо та підключаємо сирену (Siren).

5.3.4 Додаємо червону машину.

5.3.5 Приклад зібраної схеми можна побачити на рис. 5.25.

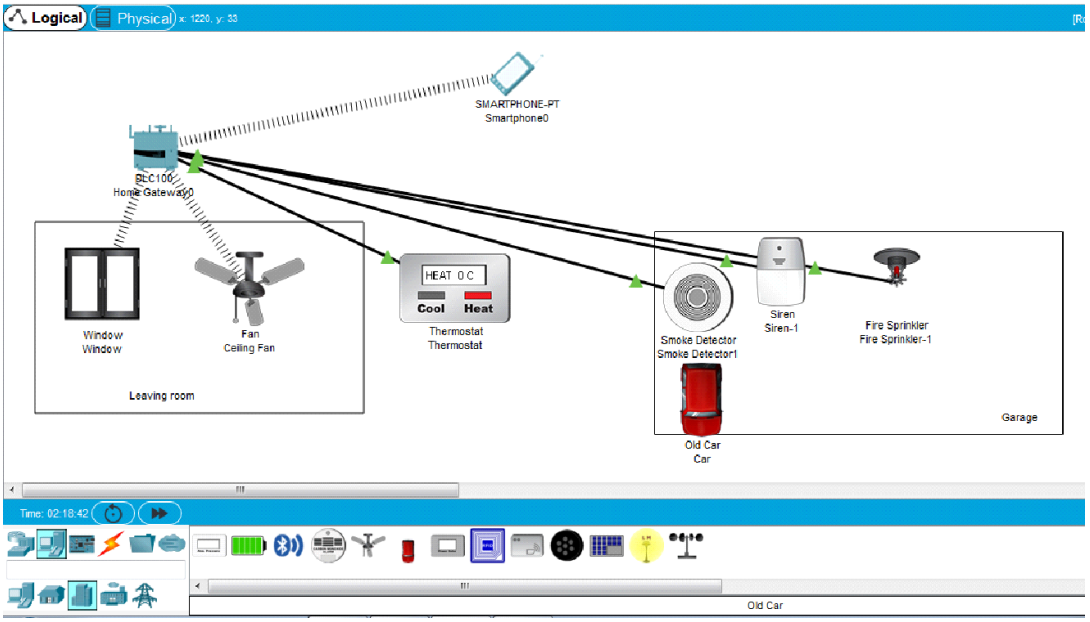

Рисунок 5.25 — Приклад схеми із IoT мережею гаража

5.3.6 Для детектору газів встановлюємо умову: якщо рівень газів на перевищує 10, то вмикається аварійна сигналізація.

5.3.7 Додаємо правило роботи для пожежного розпилювача з використанням смартфона: якщо рівень газів на детекторі газів перевищує 10.5, то вмикається пожежний розпилювач.

5.3.8 На рис. 5.26 та 5.27 наведено приклади правил для спрацьовування сигналізації та пожежного розпилювача.

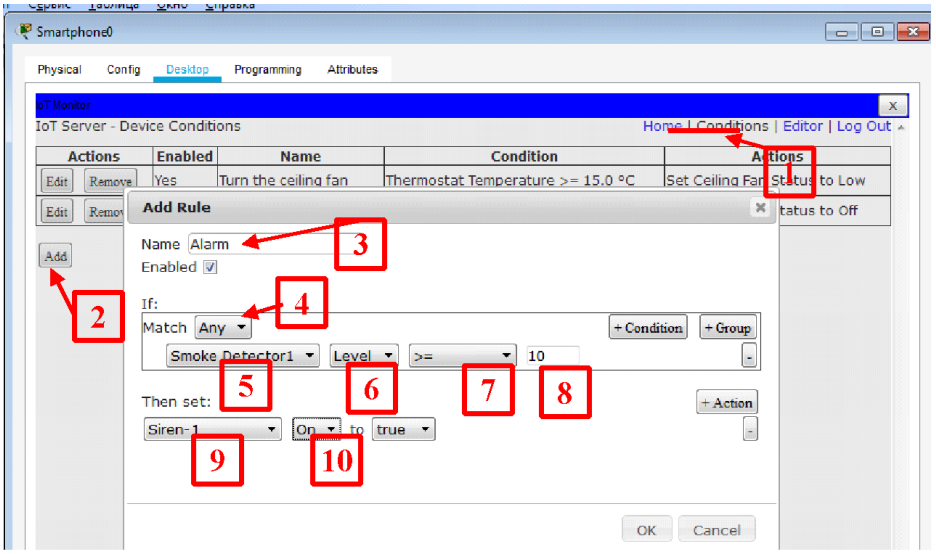

Рисунок 5.26 — Правило для спрацьовування сигналізації

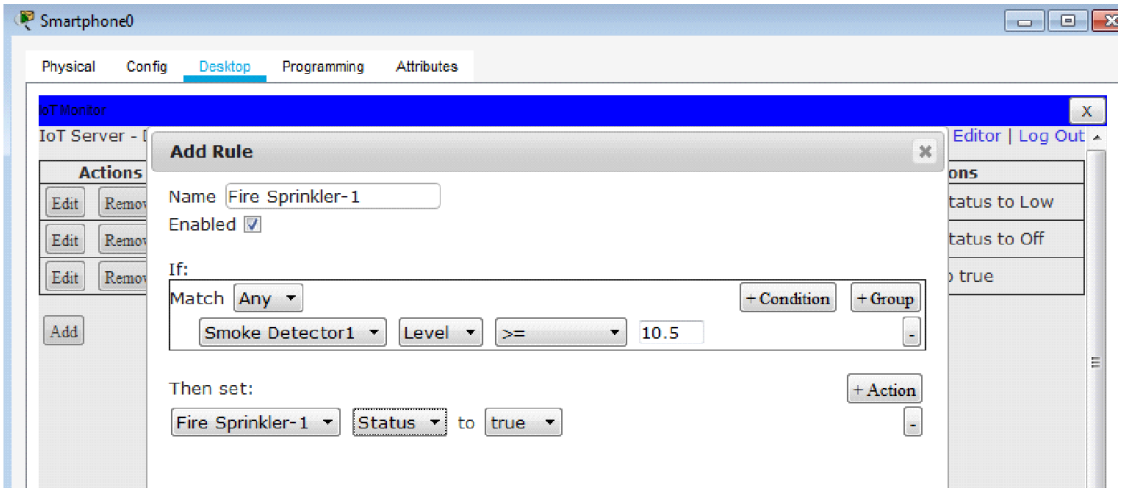

Рисунок 5.27 — Правило для спрацьовування пожежного розпилювача

5.3.9 Для перевірки коректності розробленої схеми, увімкнить машину та почекайте граничне значення рівня газів.

Завдання 5.4. Підготовка технічного рішення для “розумної” житлової кімнати

5.4.1 Спроектувати «розумну кімнату», яка включає такі елементи (рис. 5.28):

- Вікно (Window);

- Термостат (Thermostat);

- Батарея обігріву (Furnace);

- Кондиціонер (Air Conditioner);

- Двері (Door).

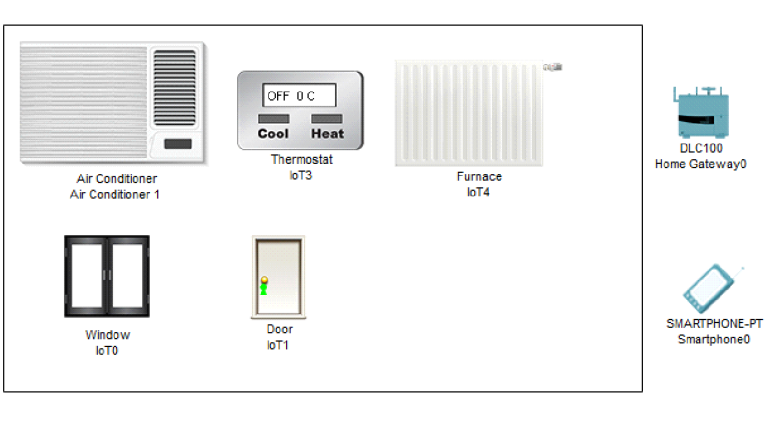

Рисунок 5.28 — Пристрої, які повинні бути використанні при проектуванні житлової IoT кімнати

5.4.2 Управління пристроями здійснюйте через Домашній шлюз (Home Gateway) за допомогою смартфону (Smartphone).

5.4.3 Опишіть логіку з’єднання пристроїв.

5.4.4 Опишіть правила управління пристроями.

Контрольні запитання

1) Які протоколи та технології можуть використовувати IoT пристрої для комунікації?

2) Як «розумна» кімната може спростити життя мешканців?

3) Як би ви облаштували своє приміщення?

4) Які IoT пристрої можна було б розташувати в офісному приміщенні?

5) Опишіть процес налаштування правил в Cisco Packet Tracer (наприклад, для включення вентилятору).

Лабораторна робота № 6

Тема: Реалізація IoT-проекту

Мета роботи — ознайомитись з візуальним програмуванням мікроконтролера та реалізувати IoT-проект з використанням Cisco Packet Tracer .

ТЕОРЕТИЧНІ ВІДОМОСТІ

Візуальне програмування мікроконтролерів є підходом до створення програм для вбудованих систем, де програми розробляються за допомогою графічного інтерфейсу замість традиційного текстового коду. Цей підхід спрощує розробку для тих, хто не є професійним програмістом, але водночас може бути корисним для досвідчених розробників для прискорення процесу розробки [10].

Основні характеристики візуального програмування:

1. **Блочне програмування (Block Programming)**

У візуальному програмуванні використовуються блоки або елементи, які представляють різні функції або операції. Користувач може перетягувати ці блоки та з'єднувати їх, щоб створювати програмні алгоритми. Кожен блок зазвичай відповідає певній команді або операції.

2. **Графічний Інтерфейс**

Користувачі працюють з графічним інтерфейсом, де вони можуть взаємодіяти з програмою, перетягуючи, з'єднуючи та налаштовуючи блоки. Це дозволяє легше розуміти логіку програми і спрощує введення команд.

3. **Візуальне Представлення Алгоритму**

Візуальне програмування надає можливість дивитися на програмний код у вигляді графічного алгоритму, що може полегшити розуміння та аналіз програми.

4. **Компіляція в Текстовий Код**

Часто візуальні програми конвертуються в текстовий код під час компіляції. Це дозволяє використовувати візуальне програмування для швидкої ініціалізації та відлагодження програм, а також перейти до текстового програмування для більшої гнучкості та розширюваності.

5. **Підтримка Мікроконтролерів**

Платформи візуального програмування можуть бути спеціально розроблені для конкретних мікроконтролерів чи мікропроцесорів, щоб забезпечити оптимальну підтримку для їх функцій та фіч.

6. **Приклади візуальних мов програмування:**

- Scratch для Arduino: графічний інтерфейс для програмування Arduino.
- Blockly: бібліотека для створення візуальних редакторів програм.
- LabVIEW: програмне забезпечення для візуального програмування, часто використовується в інженерії та автоматизації.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 6

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;
- мета роботи;
- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 6.1. Підключення та візуальне програмування мікроконтроллера

6.1.1 На рисунку 6.1 зображено приклад схеми, яку потрібно зібрати. Правильно під’єднайте контакти.

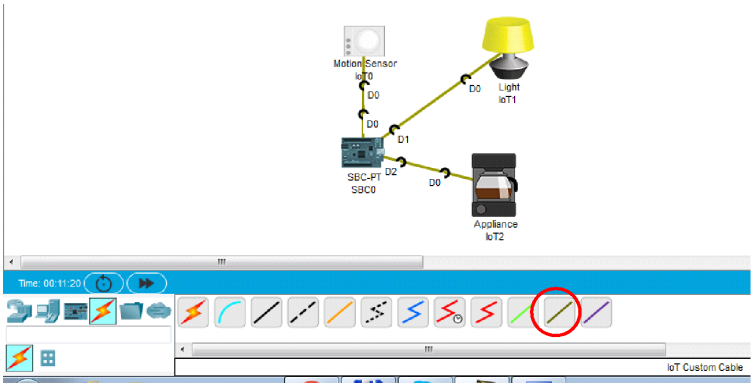

Рисунок 6.1 — Приклад схеми із SBC мікроконтроллером

6.1.2 Тепер потрібно запрограмувати SBC. Для цього, відкрийте властивості Програмування на SBC0. Клікніть на «Blink Python» та створіть новий проект. Оберіть вид програмування, як зображено на рис. 6.2.

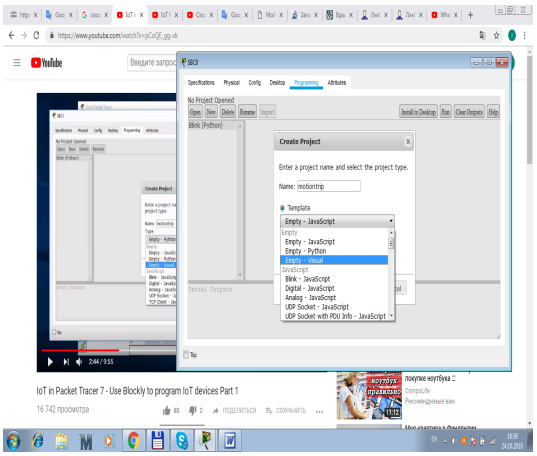

Рисунок 6.2 — Вибір мови програмування

6.1.3 Зробіть подвійний клік на «main.visual», а потім Program та ознайомтесь з елементами візуального програмування у цьому середовищі (рис. 6.3).

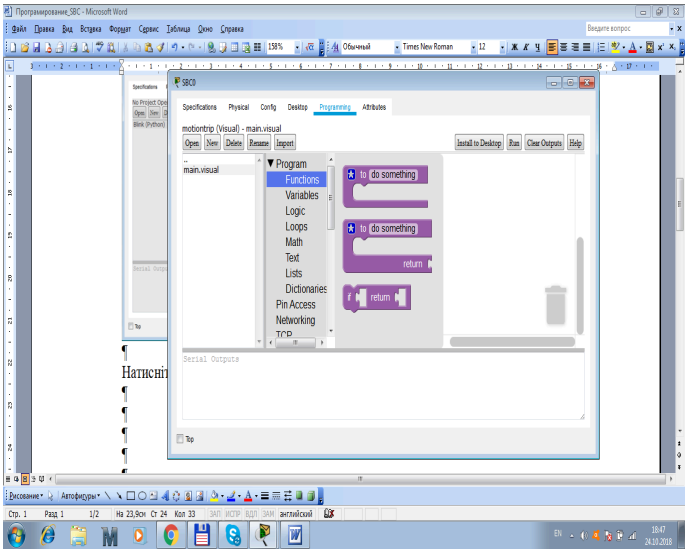

Рисунок 6.3 — Елементи візуального програмування

6.1.4 Оберіть «Змінні» - Variables та перетягніть елемент на робочу область. Змініть назву змінної (рис. 6.4).

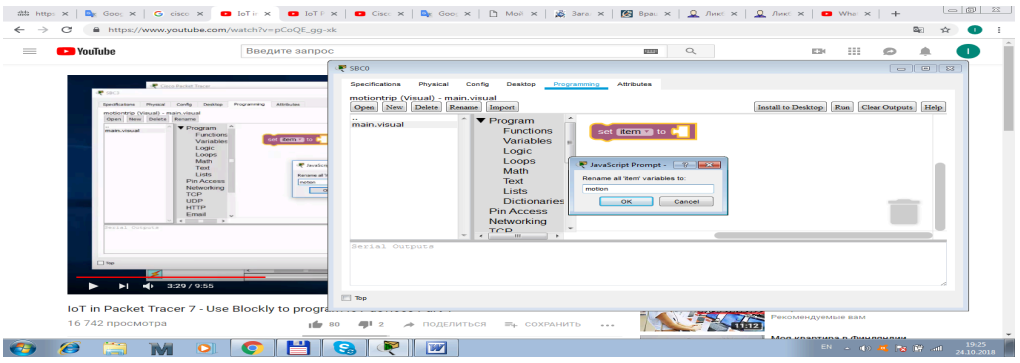

Рисунок 6.4 — Створення змінної та зміна її назви

6.1.5 Далі розташуйте улемент “0” з підрозділу “Math” і вставте його в чарунку (рис. 6.5). Результат має виглядати як на рис. 6.6.

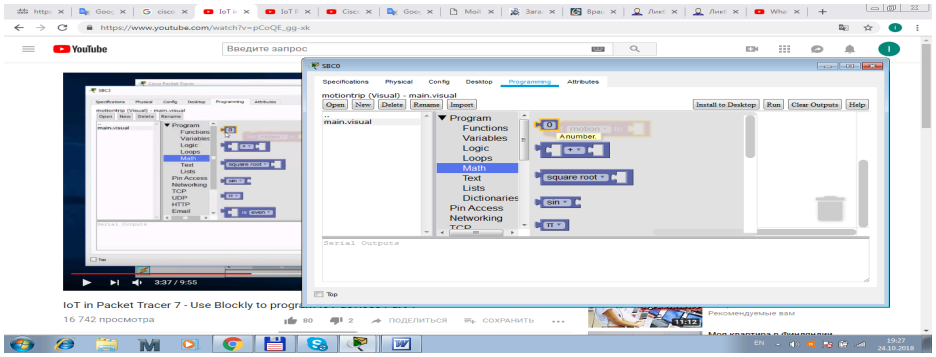

Рисунок 6.5 — Ініціалізація змінної

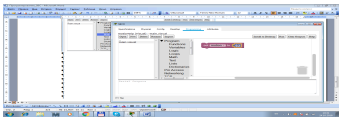

Рисунок 6.6 — Результат ініціалізації змінної

6.1.6 З підрозділу функцій розташовуємо і встановлюємо початок основної функції “Main” (рис. 6.7). Змінюємо назву з “do something” на “main”.

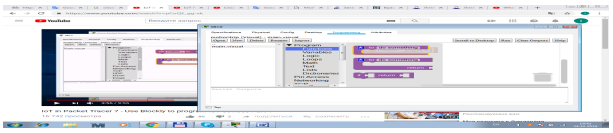

Рисунок 6.7 — Декларування головної функції main

6.1.7 З Pin Access дістаємо функцію pinMode для розподілу контактів (рис. 6.8).

Рисунок 6.8 — Створення екземплятру pinMode

6.1.8 Створюємо копії цього екземпляру (всього 3 штуки), налаштовуємо входи/виходи, і додаємо піни в тіло функції (рис. 6.9).

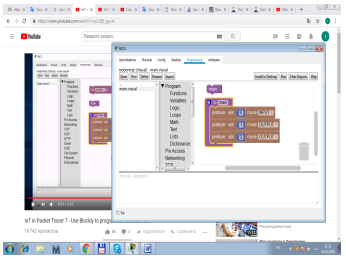

Рисунок 6.9 — Вигляд функції main після вставки пінів

6.1.9 Додаємо функцію циклу з підрозділу Loops та функцію true з підрозділу Logic. Включаємо їх до функції main (рис. 6.10).

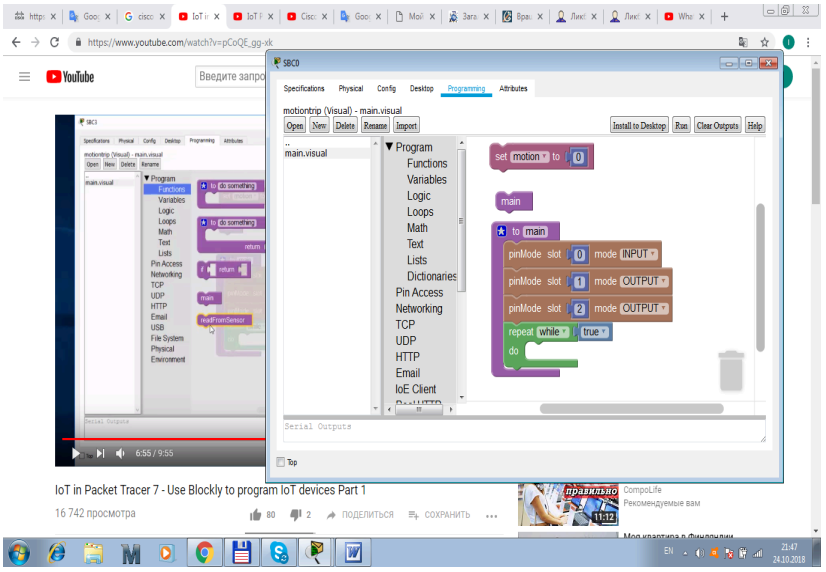

Рисунок 6.10 — Функція main після додавання циклу

6.1.10 Розташуємо функцію з розділу Functions, яка буде зчитувати сигнали з сенсорів. Змінюємо її назву, як показано на рис. 6.11.

Рисунок 6.11 — Створення функції readFromSensor

6.1.11 розташовуємо функції digitalRead та затримку з розділу Pin Access; також, нам знадобиться функція Print з розділу Text; окрім цього, в робочу область додамо функцію motion з розділу Variables. Підготуємо підфункцію, як показано на рис. 6.11.

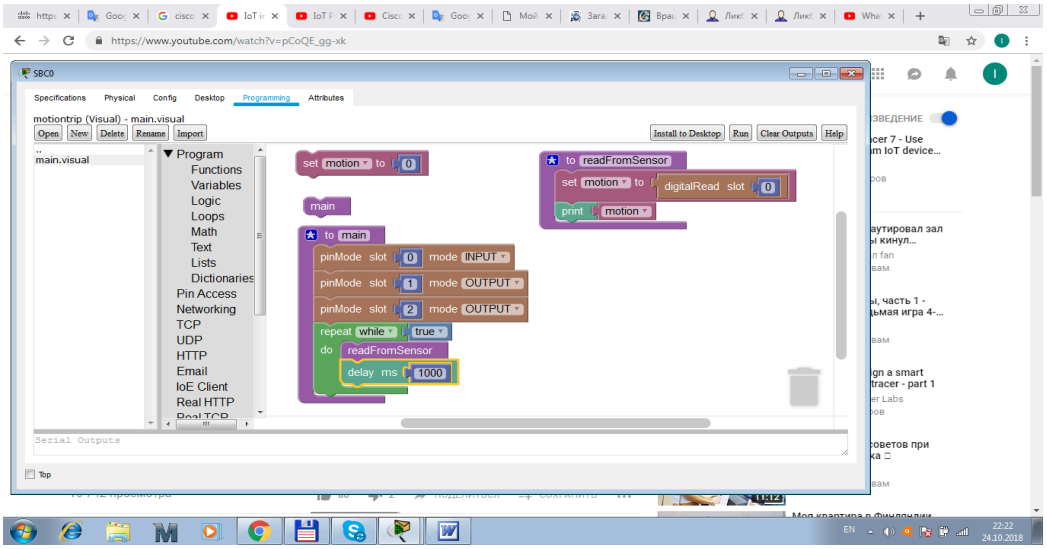

Рисунок 6.11 — Додавання читання даних від сенсору

6.1.12 Для зручності, треба розмістити вікна як на рис. 6.12.

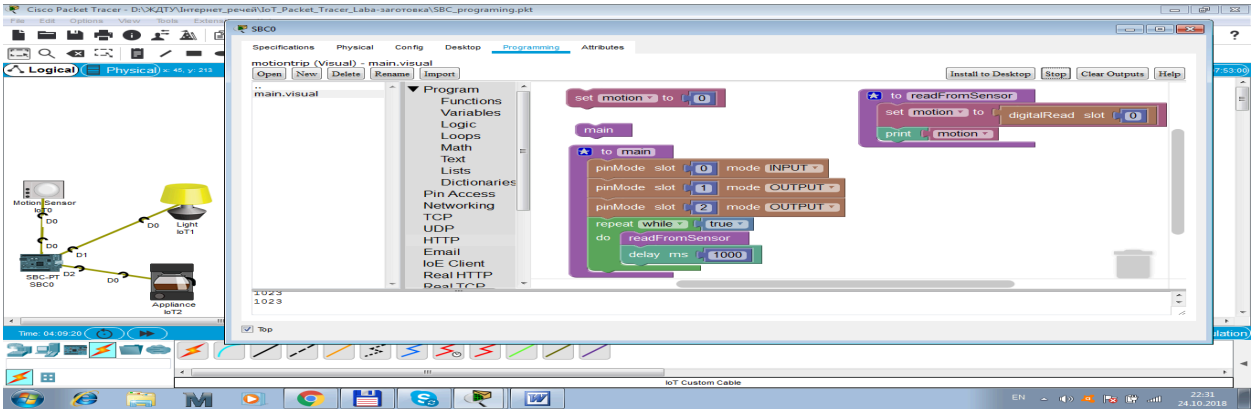

Рисунок 6.12 — Розташування вікон

6.1.13 Натискаємо кнопку “Run”, щоб почати виконання. У вікні виводу повинні виводитись нулі, але якщо провести курсором миші біля детектору руху та утримувати ALT, то повинен з’явитися код.

6.1.14 Створимо підпрограму для управління пристроями IoT. Формуємо функцію запису. Додаємо її до основної програми (рис. 6.13).

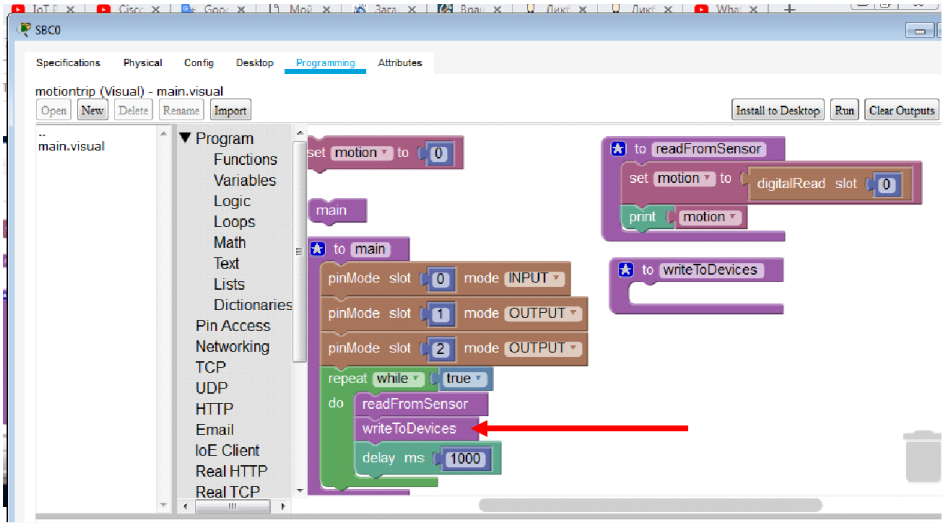

Рисунок 6.13 — Вставка звернення до функції writeToDevices в функції main

6.1.15 Далі необхідно забезпечити виконання умови: після читання сигналу з датчику (D0), потрібно запустити виконання певної дії. Нехай такою дією буде запис активного рівня в D1 та D2. Додаємо в робочу область функцію if-do-else та розташовуємо її в підпрограму запису. Перетягуємо функцію умови з підрозділу Logic та додаємо до тіла функції. Розташовуємо елементи «print» та «лапки» з підрозділу Text. Прописуємо відповідні повідомлення (рис. 6.14).

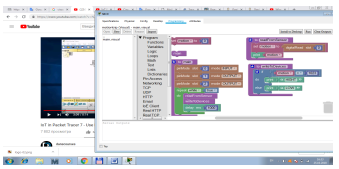

Рисунок 6.14 — Функція writeToDevices

6.1.16 Перевіримо роботу підпрограми. Для цього натиснемо кнопку Run та перевіряємо роботу детектора як в минулий раз (рис. 6.15).

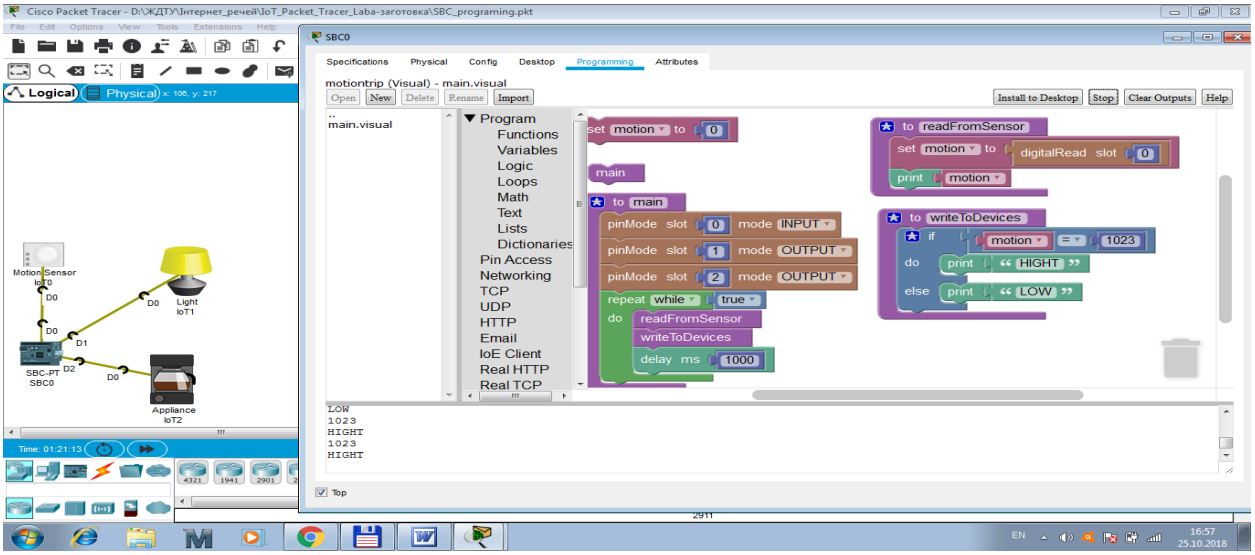

Рисунок 6.15 — Перевірка роботи детектора

6.1.17 Реалізуємо функцію керування пристроєм. Дістаємо функцію CustomWrite з розділу Pin Access. Приклад реалізації зображено на рис. 6.16.

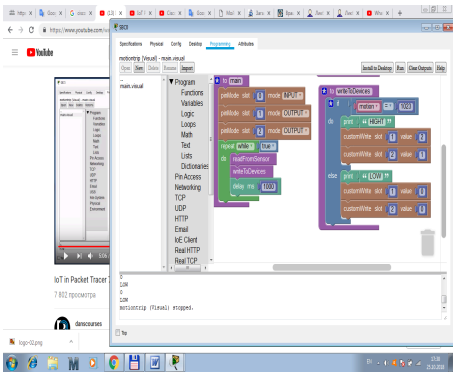

Рисунок 6.16 — Реалізація функції writeToDevices

6.1.18 Запускаємо програму на виконання. При спрацюванні датчика руху повинні вмикатися лампа та кавоварка. Після роботи вони вимикаються (рис. 6.17).

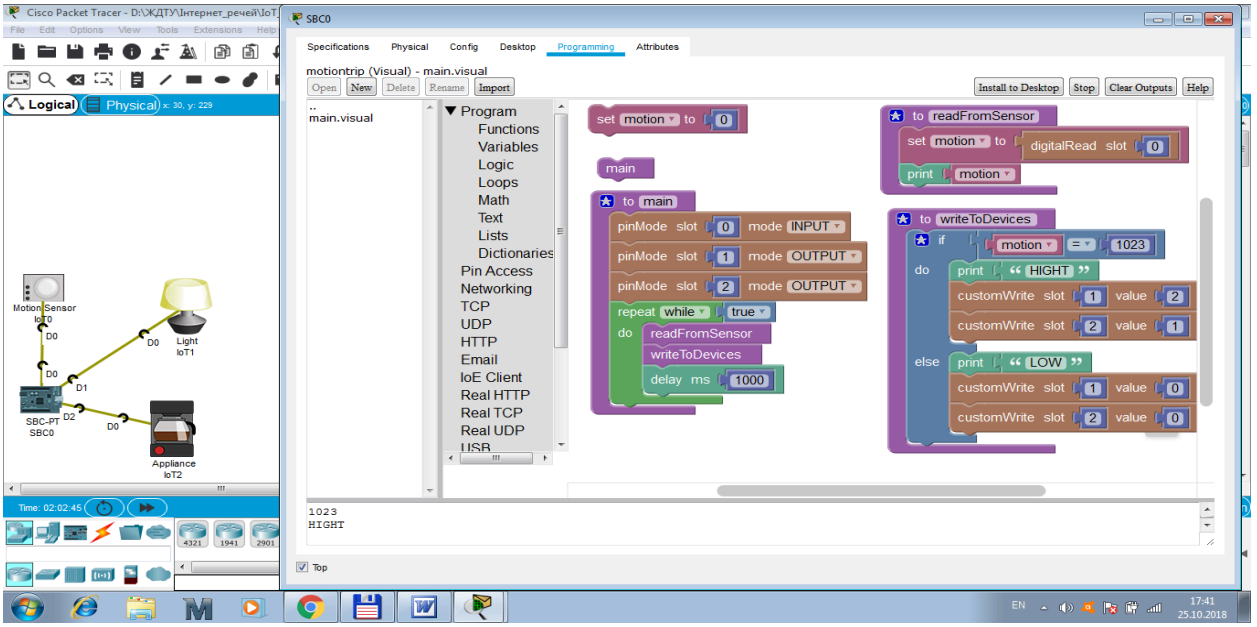

Рисунок 6.17 — Готова схема роботи

Завдання 6.2. Реалізація IoT-проекту з програмуванням та підключенням домашнього шлюзу до мережі

6.2.1 Перетягніть пристрій Home Gateway на робочу область (рис. 6.18).

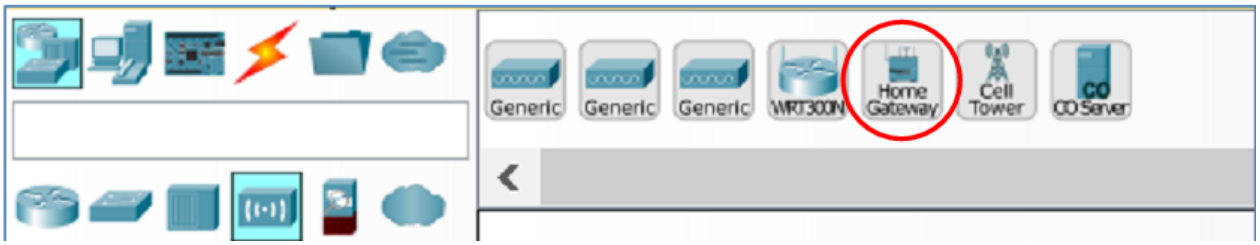

Рисунок 6.18 — Розташування елементу Home Gateway

6.2.2 Оберіть Copper Straight-Through у полі Тип виділення пристрою (рис. 6.19), а потім натисніть іконку Home Gateway для під’єднання. З’єднайте кабельний модем з портом Інтернету.

Рисунок 6.19 — Розташування елементу Copper Straight-Through

6.2.3 Через декілька секунд на кінцях ліній мають бути зелені позначки. Це підтвердження того, що ланки працюють (рис. 6.20).

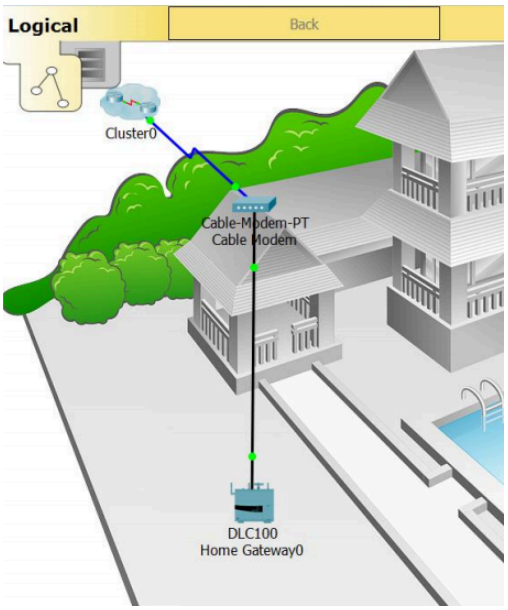

Рисунок 6.20 — З’єднання встановлено

6.2.4 Натисніть значок "Домашні пристрої" у полі "Вибір пристрою" та додайте "Вентилятор", "Двері" та “Лампа” до робочого простору (рис. 6.21).

Рисунок 6.21 — Розташування необхідних IoT пристроїв

6.2.5 Необхідно налаштувати бездротовий адаптер на вентиляторі. Для цього, клікніть по вентилятору і відкрийте вкладку Конфігурація. Після цього, клікніть по кнопці Додатково у нижньому правому куті вікна.

6.2.6 Зайдіть на вкладку Конфігурація вводу/виводу. Змініть тип мережевого адаптера на бездротовий зв'язок PT-IOT-NM-1W адаптер (рис. 6.22).

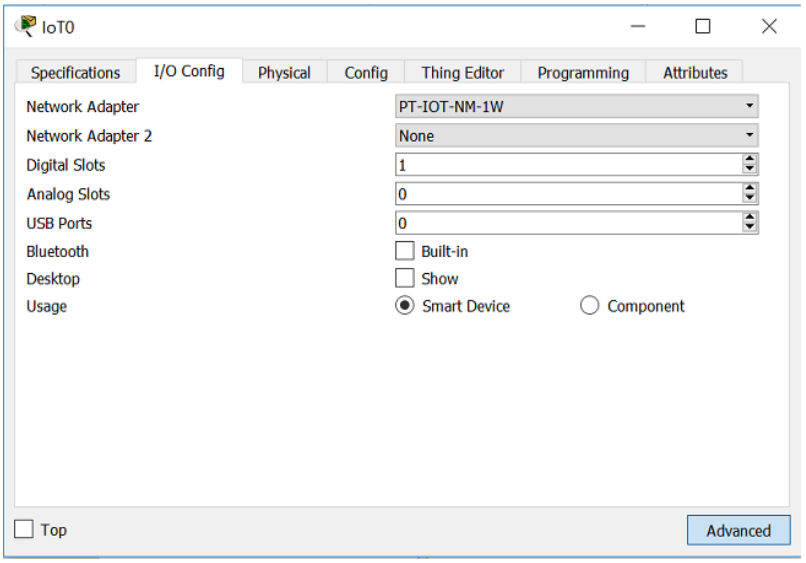

Рисунок 6.22 — Налаштування бездротового адаптера на вентиляторі

6.2.7 Перейдіть на вкладку Config. У полі Display Name введіть Ceiling Fan (рис. 6.23).

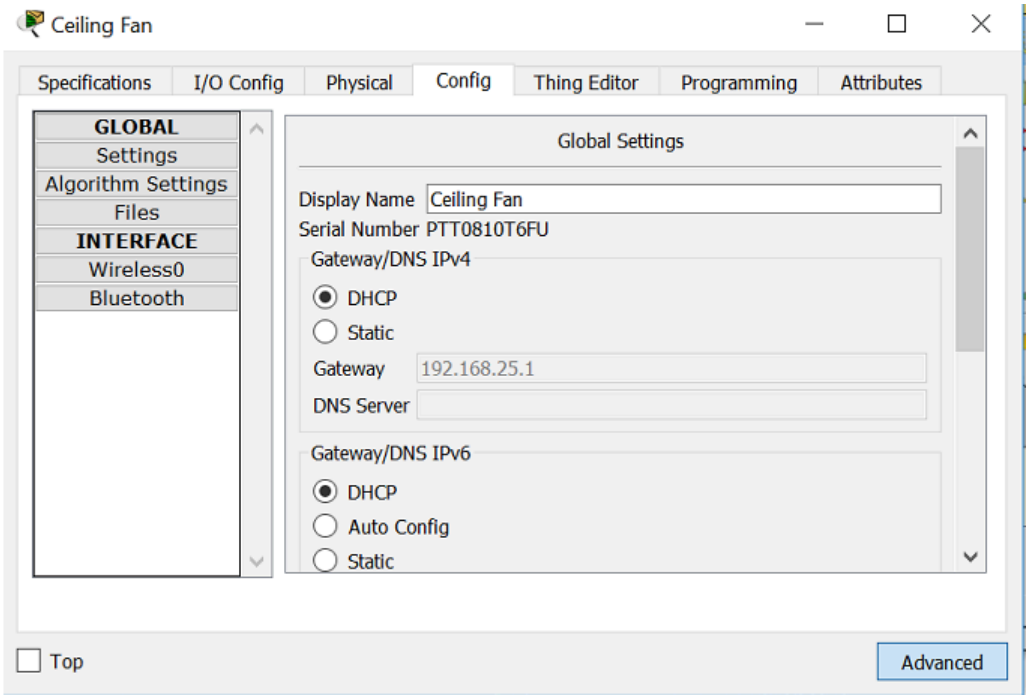

Рисунок 6.23 — Зміна ім’я на вентиляторі

6.2.8 Перебуваючи на вкладці Конфігурація, клацніть інтерфейс Wireless0 на лівій панелі. У налаштуваннях конфігурації мережа HomeGateway повинна бути вказана в полі SSID. Перевірте це.

6.2.9 DHCP вибрано в налаштуваннях IP-адреси, IP-адреса - 192.168.25.100 та Шлюз за замовчуванням - 192.168.25.1. Це вказує на те, що вентилятор підключено до мережі і є отримання IP конфігурації з домашнього шлюзу (рис. 6.24).

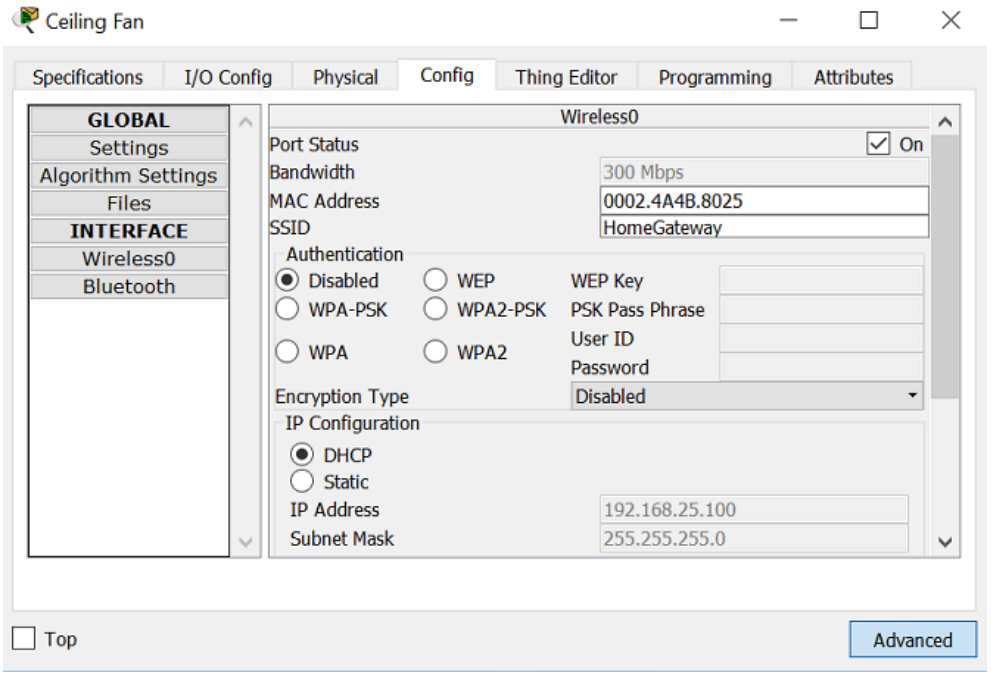

Рисунок 6.24 — Перевірка мережевої конфігурації на вентиляторі

6.2.10 Підключіть двері та лампу до бездротової мережі, виконуючи ті самі дії, які використовуються для вентилятора.

6.2.11 Підключимо пристрій кінцевого користувача. Для цього натиснемо на значок End Devices (Термін пристроїв) у полі Device type selection selection (Вибір пристрою) та додайте Wireless Tablet на робочу область (рис. 6.25).

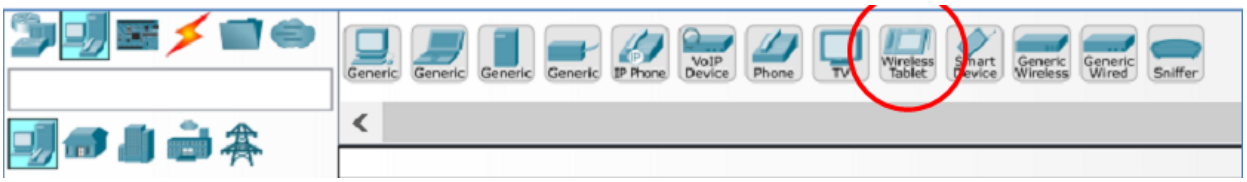

Рисунок 6.25 — Розташування елемента Wireless Tablet

6.2.12 Змінемо налаштування мережі планшету. Натисніть значок планшета, щоб відкрити вікно конфігурації планшета.

6.2.13 Перейдіть на вкладку Config і натисніть інтерфейс Wireless0. Пропишіть HomeGateway в ідентифікаторі SSID. Планшет повинен отримати IP-адресу по DHCP протоколу (рис. 6.26).

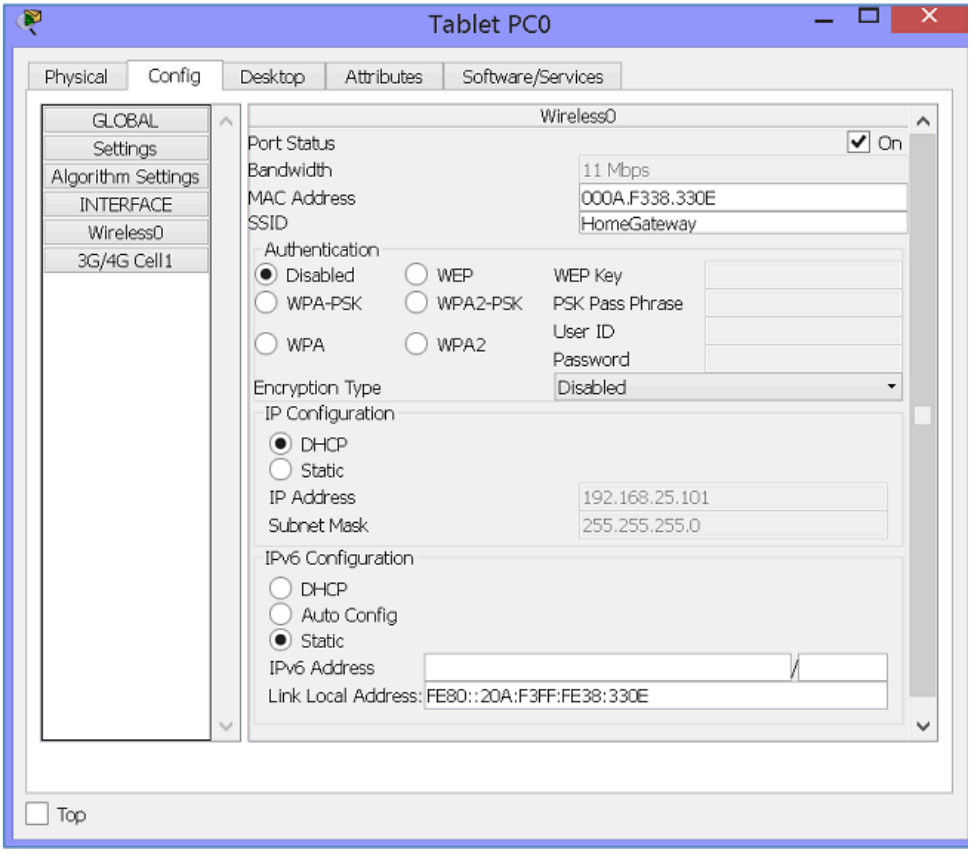

Рисунок 6.26 — Налаштування мережі на планшеті

6.2.14 Натисніть вкладку “Робочий стіл” і відкрийте веб-браузер.

6.2.15 Введіть 192.168.25.1 в полі URL-адреси та натисніть кнопку “Go”.

6.2.16 На головній сторінці домашнього шлюзу вводьте ім'я користувача - admin та пароль admin. Натисніть кнопку для входу.

6.2.17 Після входу, список повинен бути пустим (рис. 6.27).

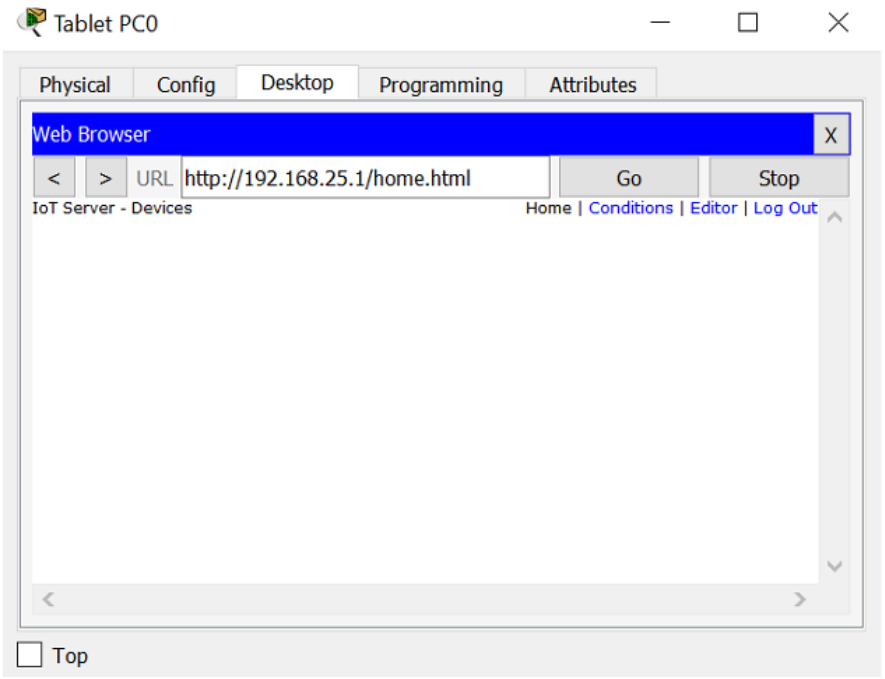

Рисунок 6.27 — Пустий список пристроїв на IoT Server

6.2.18 Налаштуємо вентилятор на те, щоб він підключався до IoT Server. Клікніть по вентилятору та перейдіть на вкладку Конфігурація. В списку параметрів сервера IoT, натисніть кнопку Home Gateway (рис. 6.28).

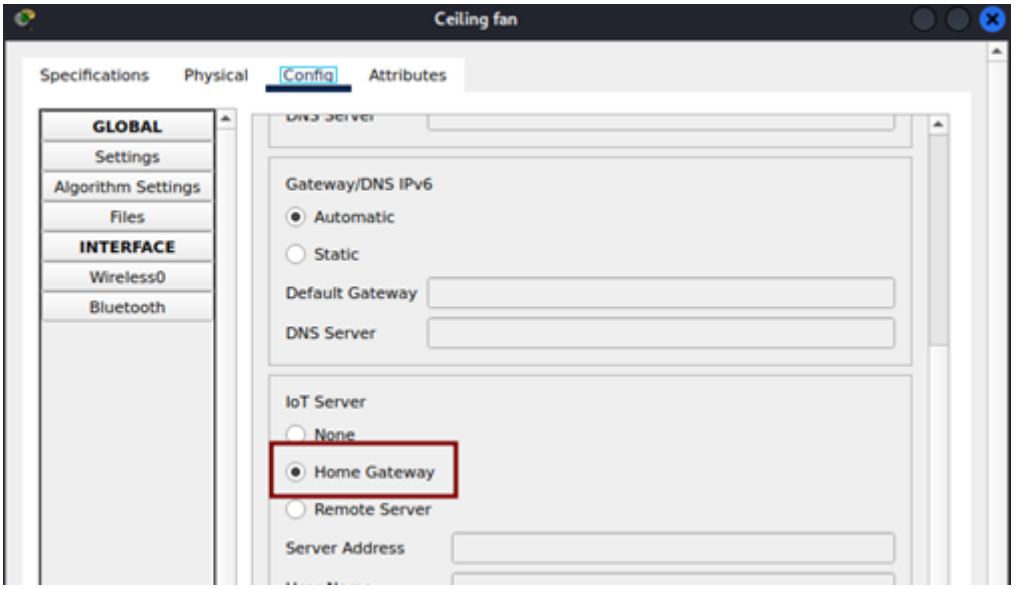

Рисунок 6.28 — Налаштування IoT Server на вентиляторі

6.2.19 Так само потрібно зареєструвати двері та лампу.

6.2.20 Клацніть значок планшета у робочому середовищі та відкрийте веб-браузер. Підключіться до домашнього шлюзу ввівши у вікні URL-адреси

192.168.25.1 та натисніть кнопку "Перейти". Введіть адміністратор як ім'я користувача та пароль і натисніть Надіслати.

6.2.21 Через кілька секунд всі три пристрої повинні бути перелічені в Home Gateway IoT Server – Devices (рис. 6.29).

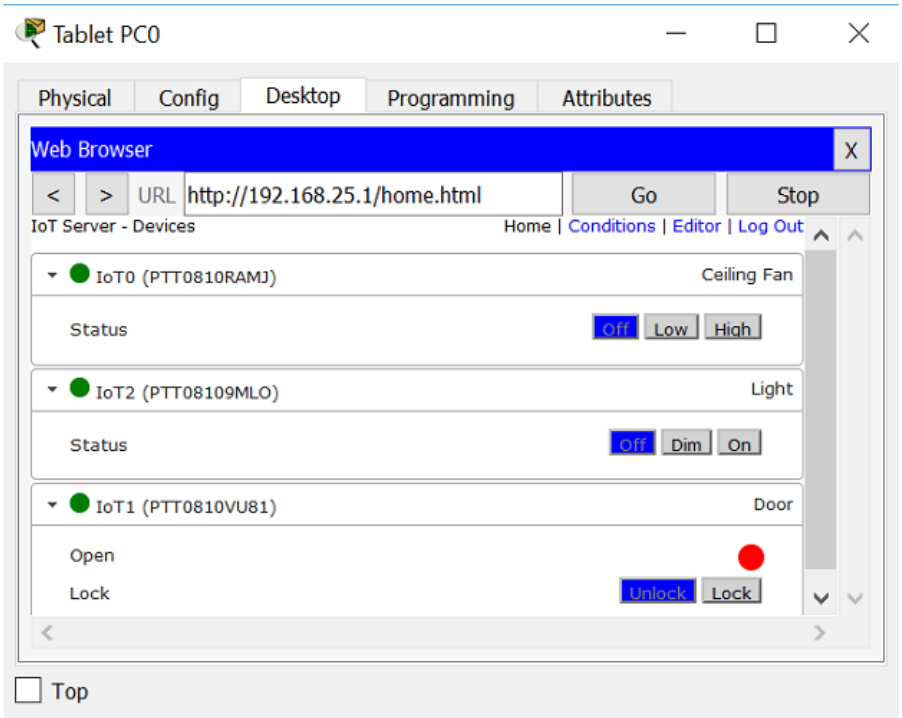

Рисунок 6.29 — IoT Server після реєстрації пристроїв

Завдання 6.3. Реалізація IoT-проекту з використанням SBC

6.3.1 Реалізуємо схему, яка зображена на рис. 6.30. Всі компоненти підключаємо через спеціальні кабелі IoT до SBC пристрою.

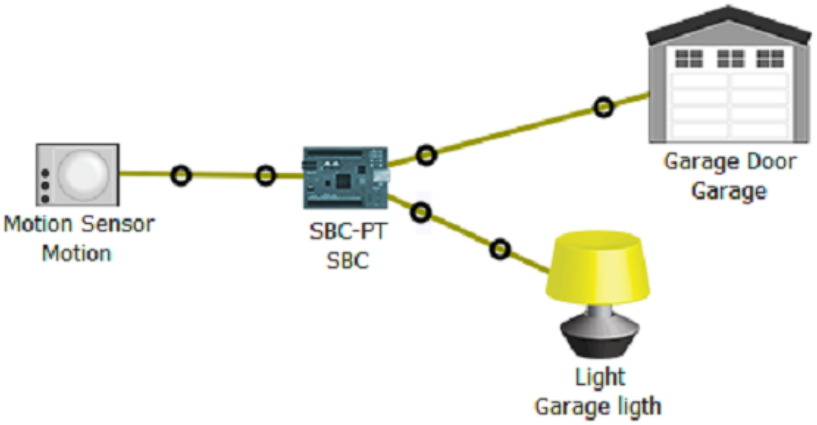

Рисунок 6.30 — Схема IoT-проекту

6.3.2 Приклад імітує відкриття гаражних дверей разом із включенням світла, коли Датчик руху виявив певний рух. У цьому випадку логіка симуляції не була реалізована сервером IoT, а індивідуальним програмним забезпеченням, що працює на платі SBC. Логіка програмного забезпечення дуже проста, сенсор був безпосередньо підключений до входу SBC. Коли об'єкти були виявлені, датчик подавав величину вхідного штифта SBC, Контролер потім порівнював його із заздалегідь встановленим пороговим значенням, і якщо логічна пропозиція була met, команду до вихідних штифтів було відправлено. Команда була користувацькою командою API що відкрив гаражні двері та включив вогні. Програма, що працює на SBC, зображена на рис. 6.31.

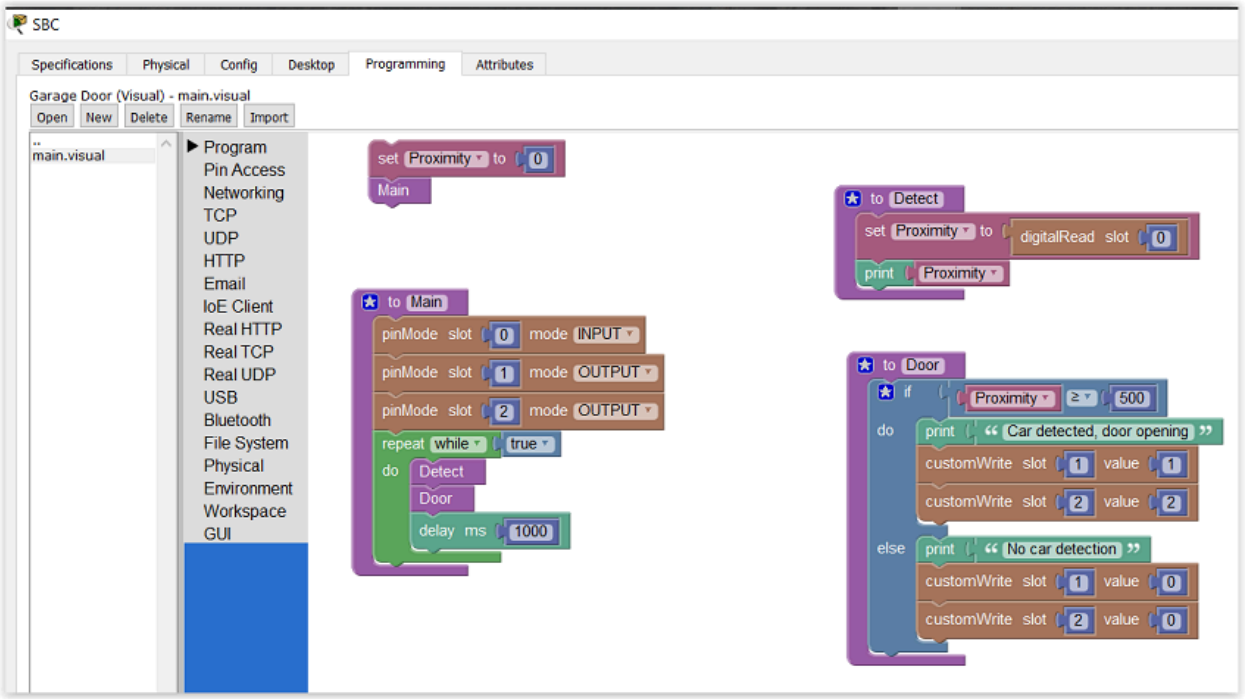

Рисунок 6.31 — Програма на SBC

Контрольні запитання

1) Які переваги та недоліки візуального програмування?

2) Чи підтримує Cisco Packet Tracer якісь мови програмування?

3) Для моделювання якого IoT-проекту ви використовували б Cisco

Packet Tracer?

Лабораторна робота № 7

Тема: Протоколи IoT. CoAP

Мета роботи — розглянути CoAP протокол, структуру пакету та його використання в IoT.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Що таке CoAP?

Протокол CoAP (Constrained Application Protocol) — це протокол з обмеженими можливостями для мереж з обмеженими ресурсами, таких як обладнання Інтернету речей (IoT) та інші вбудовані системи. CoAP був створений для забезпечення ефективного обміну даними та керування ресурсами в умовах, де доступність ресурсів, енергоефективність і мережева пропускна здатність обмежені.

Основні риси протоколу CoAP:

1. Легкість: CoAP розроблений як простий і легкий протокол, що дозволяє обмін даними на пристроях з обмеженими можливостями, таких як датчики, контролери та інші вбудовані системи.

2. Мінімальна вартість повідомлень: CoAP використовує мінімальну кількість байтів у заголовках повідомлень, завдяки чому зменшується накладні витрати на передачу даних.

3. Можливість спостереження (Observing): CoAP надає можливість спостерігати за змінами ресурсів на сервері. Це корисно для відслідковування стану об'єктів в режимі реального часу.

4. Методи запитів: CoAP використовує різні методи запитів, такі як GET, POST, PUT і DELETE, аналогічні методам HTTP. Це дозволяє взаємодіяти з ресурсами на сервері.

5. Кешування і проксі-сервери: Протокол дозволяє використовувати кешування, що підвищує ефективність передачі даних. Також, можна використовувати проксі-сервери для покращення маршрутизації.

6. Захист і безпека: CoAP надає механізми захисту даних та аутентифікації для забезпечення безпеки в обмежених мережах.

CoAP зазвичай використовується для передачі даних в мережах Інтернету речей, де пристрої мають обмежені ресурси, а ефективність передачі та обробки даних є критичними аспектами.

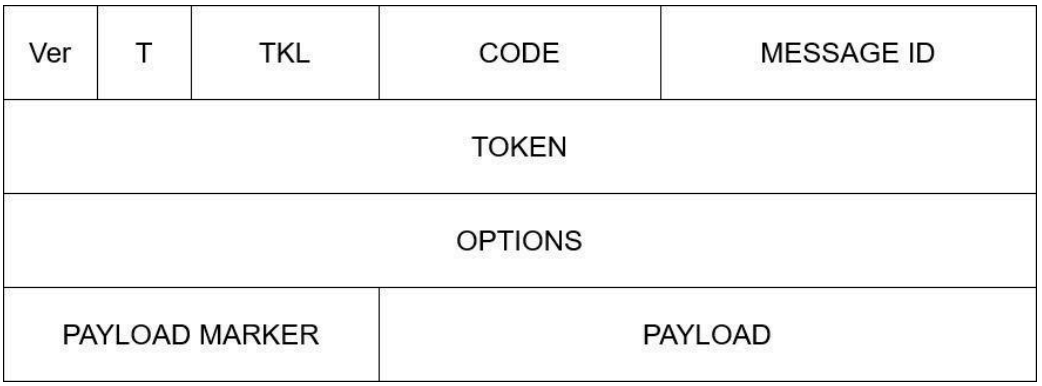

Рисунок 7.1 — Структура CoAP пакету

Пакети протоколу CoAP (Constrained Application Protocol) мають певну структуру, яка включає заголовок і опції. Основні елементи структури пакета:

1. **Заголовок (Header)**

Заголовок містить інформацію про тип повідомлення, код операції, ідентифікатор повідомлення, номер повідомлення та інші поля, які необхідні для обробки та ідентифікації повідомлення.

2. **Опції (Options)**

Опції передають додаткові дані, пов'язані з повідомленням. Вони використовуються для вказівок, таких як формат вмісту, URI шляхи, максимальний вік кешування, тощо. Опції можуть бути змінної довжини та включаються у заголовок повідомлення.

3. **Корисне навантаження (Payload)**

Це додаткові дані, які включаються до повідомлення. Корисне навантаження може бути відсутнім, або містити будь-яку корисну інформацію, таку як дані від сенсорів або відповіді на запит.

4. **Version**

Поле, яке вказує версію протоколу CoAP (4 біти).

5. **Type**

Тип повідомлення, який вказує на CON (підтверджуване), NON (непідтверджуване), ACK (підтвердження) або RST (скасування) (2 біта).

6. **TKL (Token Length)**

Довжина токена, який використовується для ідентифікації повідомлення (4 біти).

7. **Code**

Код операції, який вказує тип запиту або відповіді (8 біт).

8. **Message ID**

Ідентифікатор повідомлення, який використовується для відслідковування взаємодії (16 біт).

9. **Options**

Опції, які можуть включати додаткову інформацію, наприклад, URI-шлях або формат вмісту.

10. **Payload**

Корисне навантаження, яке містить дані повідомлення.

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 7

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний

матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва комп’ютерного практикуму;

- мета роботи;

- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 7.1. Налаштувати середовище

7.1.1 Приклад Dockerfile показано на рис. 7.2.

```
$ cat Dockerfile
FROM ubuntu:20.04
ARG DEBIAN_FRONTEND=noninteractive
RUN apt-get update && apt-get install -y vim iproute2 iputils-ping \
                                         net-tools netcat tcpdump
RUN apt-get install -y python3-pip

RUN apt-get install -y openssh-server
RUN echo 'root:12345' | chpasswd

RUN pip3 install aiocoap==0.4.7

ENTRYPOINT ["/usr/sbin/init"]
```

Рисунок 7.2 — Приклад Dockerfile

7.1.2 Приклад docker-compose.yml зображено на рис. 7.3.

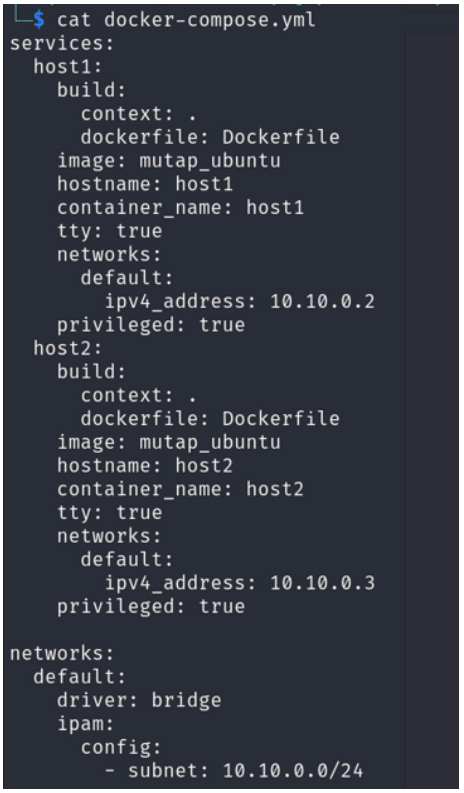

Рисунок 7.3 — Приклад docker-compose.yml

Завдання 7.2. Отримати .pcap з CoAP пакетами

7.2.1 Створюємо серверний додаток

```
#!/usr/bin/python3

import datetime
import logging

import asyncio

import aiocoap.resource as resource
import aiocoap



class BlockResource(resource.Resource):
    """Example resource which supports the GET and PUT methods. It sends large
    responses, which trigger blockwise transfer."""

    def __init__(self):
```

```
        super().__init__()
        self.set_content(b"This is the resource's default content. It is padded "
                b"with numbers to be large enough to trigger blockwise "
                b"transfer.\n")

    def set_content(self, content):
        self.content = content
        while len(self.content) <= 1024:
            self.content = self.content + b"0123456789\n"

    async def render_get(self, request):
        return aiocoap.Message(payload=self.content)

    async def render_put(self, request):
        print('PUT payload: %s' % request.payload)
        self.set_content(request.payload)
        return aiocoap.Message(code=aiocoap.CHANGED, payload=self.content)


class SeparateLargeResource(resource.Resource):
    """Example resource which supports the GET method. It uses asyncio.sleep to
    simulate a long-running operation, and thus forces the protocol to send
    empty ACK first. """

    def get_link_description(self):
        # Publish additional data in .well-known/core
        return dict(**super().get_link_description(), title="A large resource")

    async def render_get(self, request):
        await asyncio.sleep(3)

        payload = "Three rings for the elven kings under the sky, seven rings "\
                "for dwarven lords in their halls of stone, nine rings for "\
                "mortal men doomed to die, one ring for the dark lord on his "\
                "dark throne.".encode('ascii')
        return aiocoap.Message(payload=payload)
```

```
class TimeResource(resource.ObservableResource):
    """Example resource that can be observed. The `notify` method keeps
    scheduling itself, and calles `update_state` to trigger sending
    notifications."""

    def __init__(self):
        super().__init__()

        self.handle = None

    def notify(self):
        self.updated_state()
        self.reschedule()

    def reschedule(self):
        self.handle = asyncio.get_event_loop().call_later(5, self.notify)

    def update_observation_count(self, count):
        if count and self.handle is None:
            print("Starting the clock")
            self.reschedule()
        if count == 0 and self.handle:
            print("Stopping the clock")
            self.handle.cancel()
            self.handle = None

    async def render_get(self, request):
        payload = datetime.datetime.now().\
                strftime("%Y-%m-%d %H:%M").encode('ascii')
        return aiocoap.Message(payload=payload)

class WhoAmI(resource.Resource):
    async def render_get(self, request):
        text = ["Used protocol: %s." % request.remote.scheme]
```

```
        text.append("Request came from %s." % request.remote.hostinfo)
        text.append("The server address used %s." % request.remote.hostinfo_local)

        claims = list(request.remote.authenticated_claims)
        if claims:
            text.append("Authenticated claims of the client: %s." % ", ".join(repr(c) for c in claims))
        else:
            text.append("No claims authenticated.")

        return aiocoap.Message(content_format=0,
                payload="\n".join(text).encode('utf8'))

# logging setup

logging.basicConfig(level=logging.INFO)
logging.getLogger("coap-server").setLevel(logging.DEBUG)

async def main():
    # Resource tree creation
    root = resource.Site()

    root.add_resource(['.well-known', 'core'],
            resource.WKCResource(root.get_resources_as_linkheader))
    root.add_resource(['time'], TimeResource())
    root.add_resource(['other', 'block'], BlockResource())
    root.add_resource(['other', 'separate'], SeparateLargeResource())
    root.add_resource(['whoami'], WhoAmI())

    await aiocoap.Context.create_server_context(root)

    # Run forever
    await asyncio.get_running_loop().create_future()

if __name__ == "__main__":
    asyncio.run(main())
```

7.2.2 Створити клієнтський скрипт, як зображено на рис. 7.4.

```
root@host2:/# cat coap_client_1.py
#!/usr/bin/python3

import logging
import asyncio

from aiocoap import *

logging.basicConfig(level=logging.INFO)

async def main():
    protocol = await Context.create_client_context()

    request = Message(code=GET, uri='coap://10.10.0.2/time')

    try:
        response = await protocol.request(request).response
    except Exception as e:
        print('Failed to fetch resource:')
        print(e)
    else:
        print('Result: %s\n%r'%(response.code, response.payload))

if __name__ == "__main__":
    asyncio.run(main())
```

Рисунок 7.4 — Клієнтський скрипт

7.2.3 Запускаємо серверний додаток

7.2.4 Запускаємо Wireshark (або аналог) для прослуховування бриджа

7.2.5 Запускаємо клієнтський скрипт

7.2.6 Зберігаємо отримані пакети в .pcap файл. У робочому вікні Wireshark повинно бути схожі пакети, як на рис. 7.5.

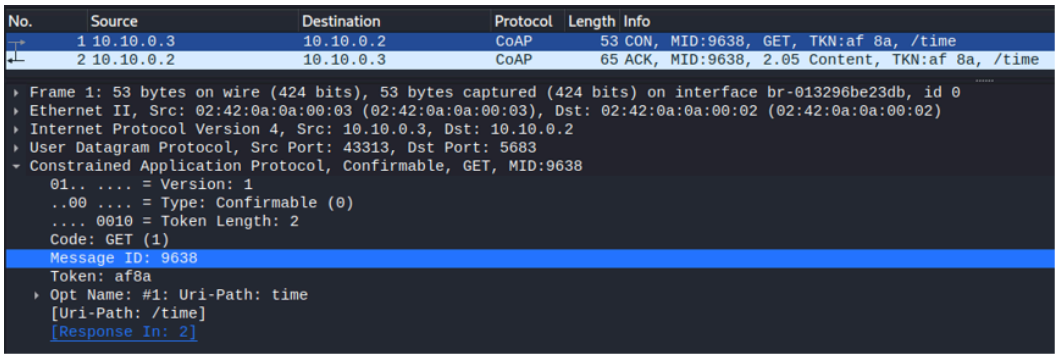

Рисунок 7.5 — Робоче вікно Wireshark із перехопленими CoAP пакетами

Завдання 7.3. Використання PUT методу

7.3.1 Клієнтський скрипт з використанням PUT методу

```
root@host2:/# cat coap_client_2.py
#!/usr/bin/python3

import logging
import asyncio

from aiocoap import *

logging.basicConfig(level=logging.INFO)

async def main():
    """Perform a single PUT request to localhost on the default port, URI
    "/other/block". The request is sent 2 seconds after initialization.

    The payload is bigger than 1kB, and thus sent as several blocks."""

    context = await Context.create_client_context()

    await asyncio.sleep(2)

    payload = b"The quick brown fox jumps over the lazy dog.\n" * 30
    request = Message(code=PUT, payload=payload, uri="coap://10.10.0.2/other/block")

    response = await context.request(request).response

    print('Result: %s\n%r'%(response.code, response.payload))

if __name__ == "__main__":
    asyncio.run(main())
```

Рисунок 7.6 — Приклад PUT запиту до серверу

7.3.2 Запускаємо Wireshark на прослуховування бріджа

7.3.3 Запускаємо клієнтський скрипт

7.3.4 Зберігаємо .pcap файл.

7.3.5 Провести аналіз пакету та визначити основні складові, такі як Ver, TKL і т.п. (рис. 7.1).

Додаткові завдання на лабораторну роботу

Завдання 7.4. Реалізувати симулятор пристрою, який може надсилати телеметрію (температуру, вологість, струм);

Завдання 7.5. Написати скрипт, який буде проводити моніторинг розробленого симулятору.

Контрольні запитання

1) Що таке CoAP?

2) В яких випадках доцільно його використовувати?

3) Протокол CoAP має чотири типи повідомлень: CON, NON, ACK та RST. Яке призначення кожного типу?

4) Які є опції (наприклад, URI-Path) та яке їх призначення?

5) Що таке DTLS (Datagram Transport Layer Security) та OSCORE (Object Security for Constrained RESTful Environments)?

Лабораторна робота № 8

Тема: Основи роботи з Node-RED

Мета роботи — ознайомитись із редактором Node-RED, отримати навички налаштування вузлів та їх комбінацій, конфігурування пошти, читання регістрів за допомогою Modbus, обробку системної інформації з використанням JS.

ТЕОРЕТИЧНІ ВІДОМОСТІ

Node-RED - це візуальний інструмент для програмування вузлів (nodes) та створення потоків (flows) в Internet of Things (IoT) та розробці програмного забезпечення. Цей інструмент дозволяє легко створювати IoT застосунки, об'єднуючи різні пристрої, сервіси та API шляхом з'єднання графічних вузлів у веб-інтерфейсі. Node-RED базується на Node.js та використовує JavaScript для створення потоків даних [1].

За допомогою Node-RED можна взаємодіяти з різними сенсорами, пристроями, базами даних та іншими ресурсами через візуальне перетягування та з'єднання вузлів, що репрезентують різні функції. Це дозволяє швидко створювати складні системи IoT або автоматизовані робочі процеси, мінімізуючи кодування із використанням готових вузлів.

Node-RED є відкритим програмним забезпеченням та має активну спільноту, яка постійно розширює набір доступних вузлів та функціональності для спрощення розробки IoT застосунків.

Робота із потоками

Створення потоків в Node-RED відбувається за допомогою веб-інтерфейсу, де ви можете перетягувати, з'єднувати та налаштовувати вузли для обробки даних. Ось загальний процес створення потоку:

1. **Запуск Node-RED:** Після встановлення Node-RED ви запускаєте його сервер, зазвичай через командний рядок або веб-інтерфейс.

2. **Створення нового потоку:** Після запуску ви створюєте новий потік (flow), який буде місцем для створення вашої програми. У веб-інтерфейсі це зазвичай відбувається за допомогою створення нового проекту або потоку.

3. **Додавання вузлів:** Виберіть вузли з панелі зліва (що представляють конкретні дії, обробники, вхідні або вихідні дані) та перетягніть їх на поле робочої області.

4. **Підключення вузлів:** З'єднайте вузли лініями, встановлюючи потік даних від одного вузла до іншого. Це визначає порядок, в якому дані будуть оброблятися в вашому потоці.

5. **Налаштування вузлів:** Кожен вузол має свої власні параметри налаштування, які можна налаштовувати через веб-інтерфейс. Наприклад, ви можете налаштовувати з'єднання з API, обробляти дані, виконувати умови та багато іншого.

6. **Виконання та тестування:** Після налаштування потоку ви можете запустити його для виконання та перевірки його працездатності. Ви можете спостерігати за потоком даних, використовуючи вбудований інтерфейс візуалізації Node-RED.

7. **Збереження та розгортання:** Після успішного створення потоку ви можете зберегти його та використовувати, а також розгорнути на відповідних пристроях або платформах IoT.

Програма побудована на Node.js (рис. 8.1).

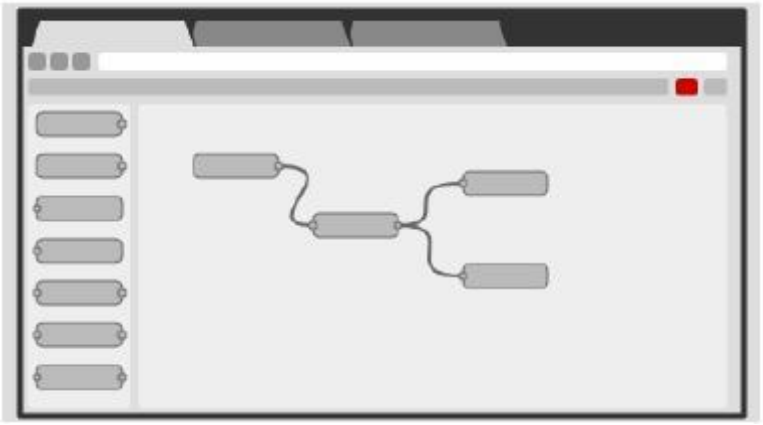

Рисунок 8.1 – Побудова схеми на Node.js

ПІДГОТОВКА ДО ВИКОНАННЯ ЛАБОРАТОРНОЇ РОБОТИ № 8

Перед виконанням лабораторної роботи рекомендується ознайомитись з відповідним розділом лекційного матеріалу курсу та засвоїти теоретичний матеріал.

ВИМОГИ ДО ОФОРМЛЕННЯ ЗВІТУ

Звіт з лабораторної роботи обов’язково повинен містити наступну інформацію:

- назва лабораторної роботи;

- мета роботи;

- постановка завдання і алгоритм його розв’язання;

- відповіді на завдання у текстовому форматі та графічними зображеннями.

Завдання на лабораторну роботу

Завдання 8.1. Встановити Node-RED

Завдання 8.2. Ознайомлення із Node-RED

Програма, створена на Node-RED, складається з потоків, які виконуються як умовно незалежні програми. Кожен потік - це набір зв'язаних між собою вузлів, які обробляють інформацію та виконують конкретні функції. Ця ідеологія нагадує побудову програм на мові FBD із стандарту програмування ПЛК (IEC 61131-3). Однак, варто зазначити, що між цими мовами існують значні відмінності у підходах та функціоналі.

8.2.1 Запустіть Node-RED та ознайомтесь з інтерфейсом програми (рис. 8.2).

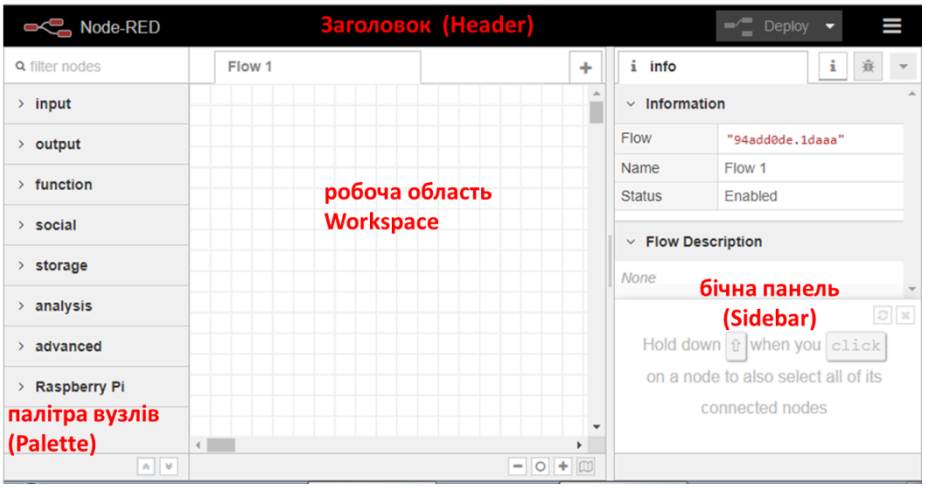

Рисунок 8.2 — Інтерфейс Node-RED

8.2.2 Розташуйте на робочій області вузли Inject та Debug. З’єднайте їх між собою, щоб вийшло як на рис. 8.3. Нявність блакитних кіл означає, що потрібно розгорнути проект, щоб задіяти зроблені зміни.

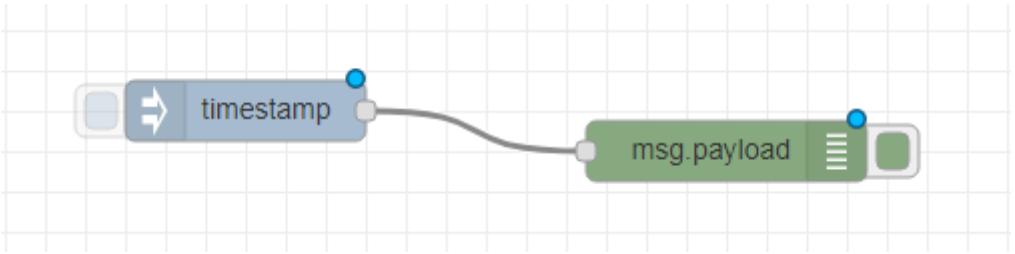

Рисунок 8.3 — Приклад з’єднання вузлів

8.2.3 Розгортаємо проект. Для цього потрібно натиснути кнопку Deploy (рис. 8.4).

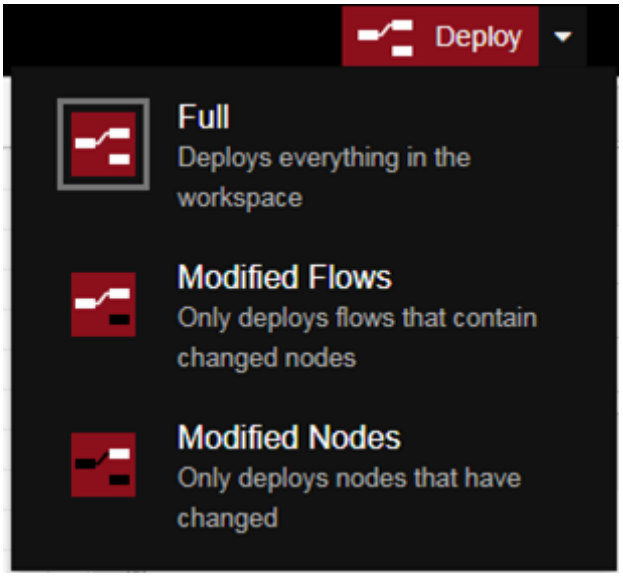

Рисунок 8.4 — Розгортання проекту

8.2.4 Якщо розгортання успішне, то повинне з’явитися повідомлення (рис. 8.5).

Рисунок 8.5 — Повідомлення про успішне розгортання

8.2.5 Натиснемо на кнопку з “жуком”, для відображення вікна із відладочними повідомленнями (рис. 8.6).

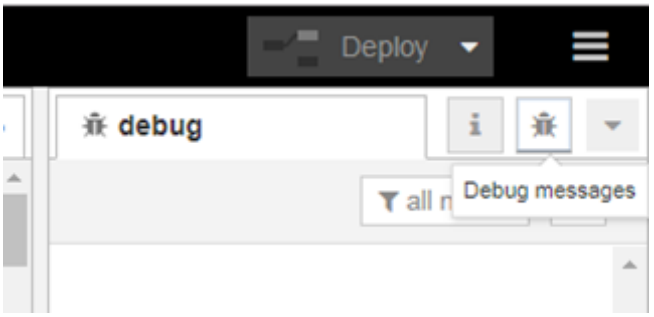

Рисунок 8.6 — Налагоджувальні повідомлення

8.2.6 В лівій частині вузла Inject є кнопка. Вона потрібна для ініціювання розрахунку ланцюжка вузлів. Натисніть на цю кнопку.

8.2.7 У вікні повідомлень повинно відображатись повідомлення про успішність вводу (рис. 8.7).

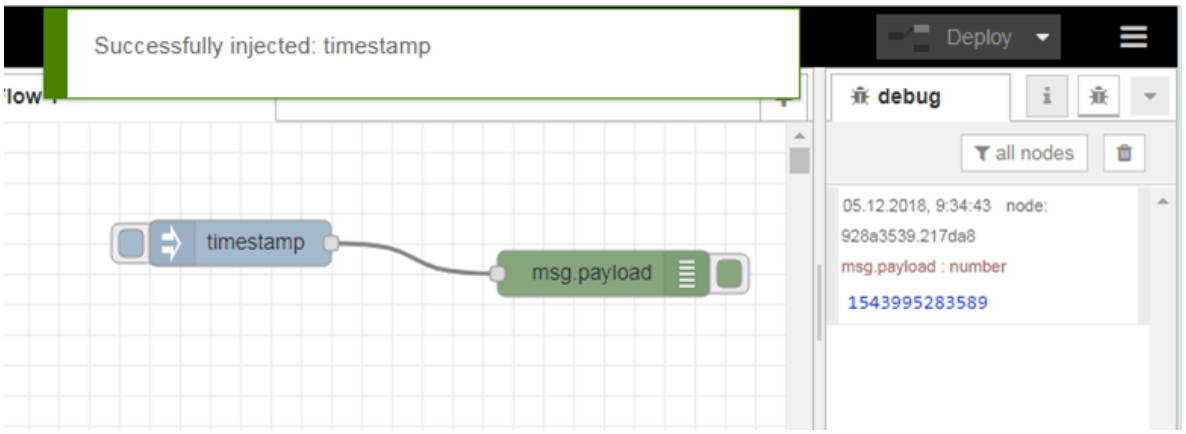

Рисунок 8.7 — Приклад вводу (Inject)

8.2.8 Зробіть подвійний клік по вузлу та налаштуйте вузли, яки показано на рис. 8.8: назви вузлів, тема і періодичність оновлення.

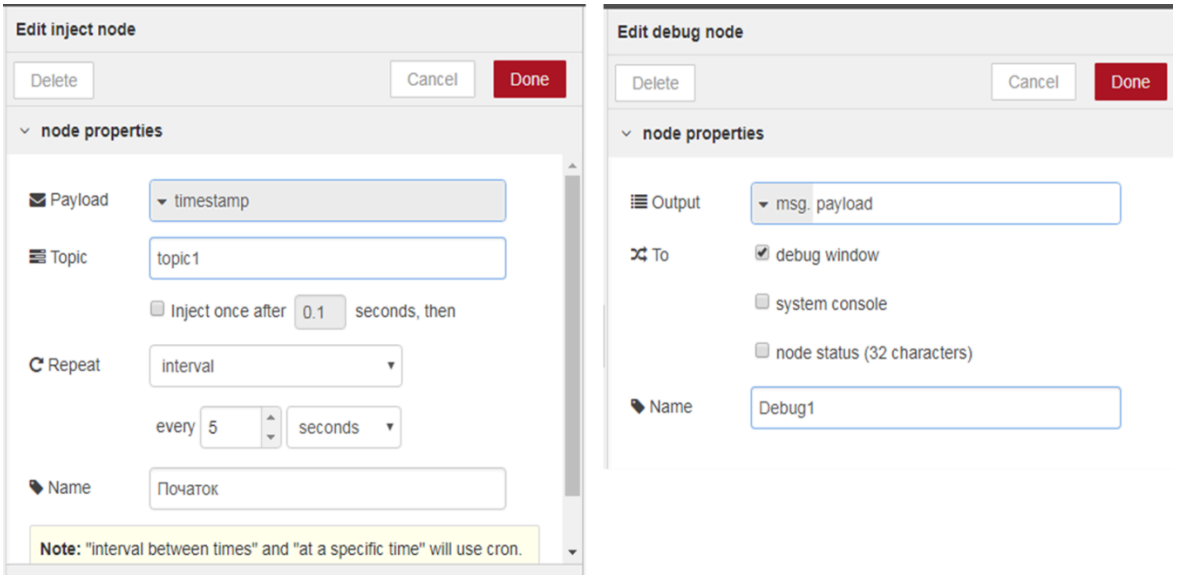

Рисунок 8.8 — Зміна властивостей вузлів

8.2.9 Розгортаємо проект та перевіряємо вміст вікна повідомлень.

8.2.10 Налаштуйте формування корисного навантаження (рис. 8.9). Розгорніть та спробуйте подивитись, як це працює.

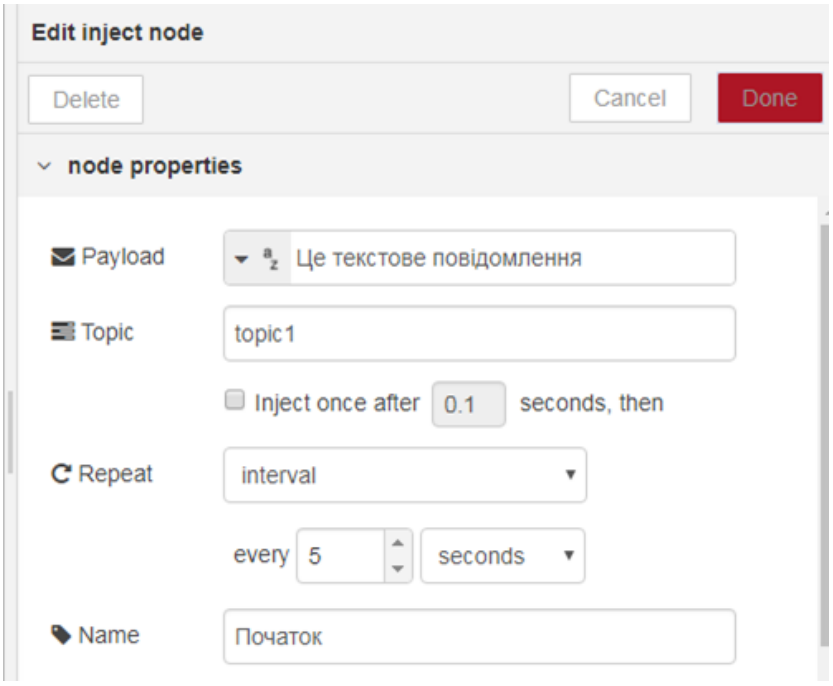

Рисунок 8.9 — Зміна вузла і налаштування корисного навантаження

8.2.11 Налаштуйте вузел “Початок” таким чином, щоб він формував корисне навантаження відміткою часу (Timestamp). Відмітка часу повинна відображатись кожні 5 секунд у вікні повідомлень.

8.2.12 Підготуйте програму, як зображено на рис. 8.10, використовуючи вузли delay та change. Для вузлів delay виставте затримки:

- delay 1s – 1 second

- 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

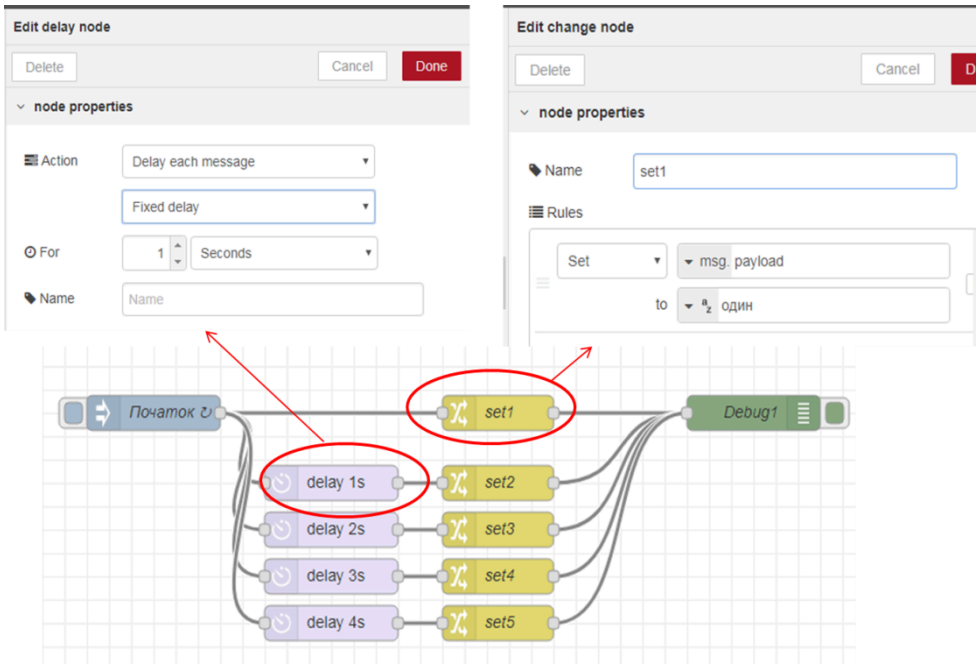

Рисунок 8.10 — Додавання вузлів затримки та зміни

8.2.13 Зробіть розгортання програми та очікуйте в окні повідомлень очікуваний вивід.

8.2.14 Використаємо вузол function, який застосує метод toLocaleString() для виведення в форматі дати та часу (рис. 8.11).

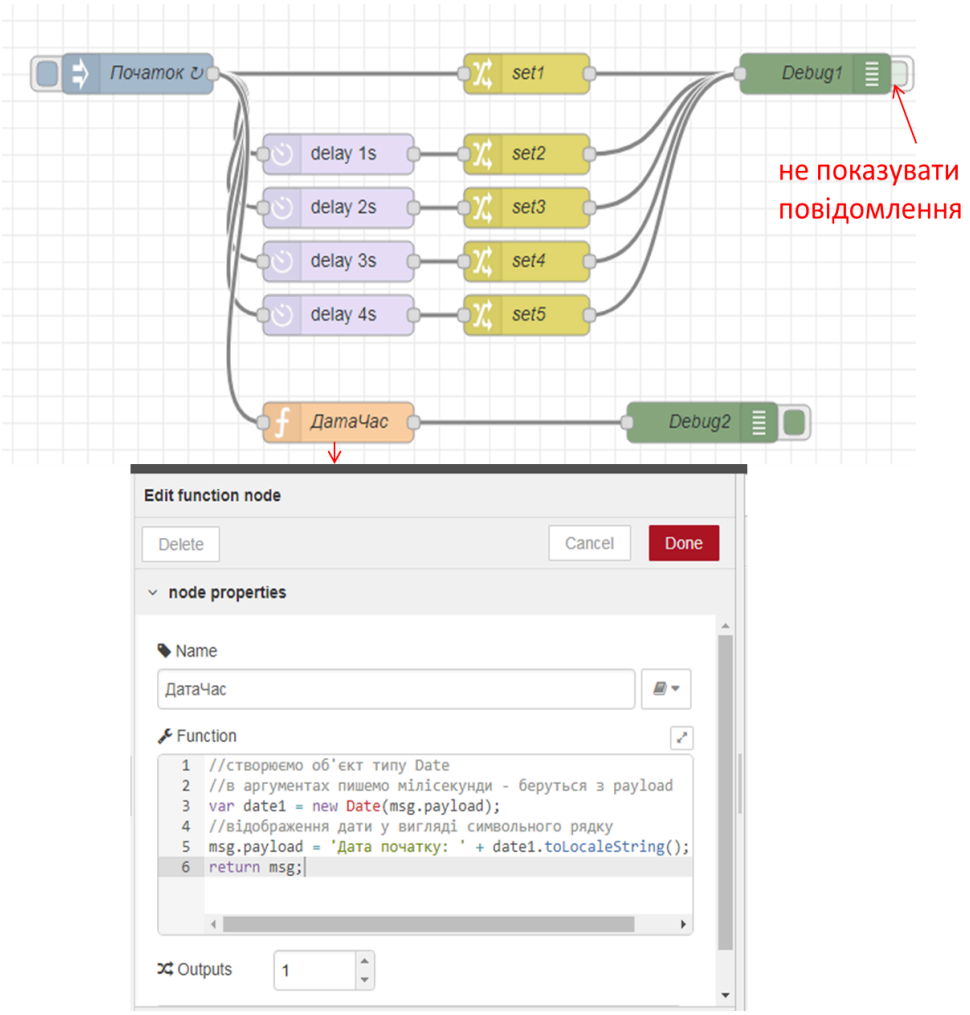

Рисунок 8.11 — Приклад використання function

Завдання 8.3. Робота з поштою

8.3.1 Дізнайтеся налаштування поштового сервера, який Ви будете використовувати для відправки повідомлень. Рекомендується створити новий акаунт для тестування.

8.3.2 До потрібних параметрів входять: SMTP Server, port. Для деяких поштових сервісів такі дані наведені на рис. 8.12.

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	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	465(SSL) або 587(TLS)	port	993	

Рисунок 8.12 — Приклади налаштувань для вхідних повідомлень

8.3.3 До потрібних параметрів для отримання повідолень відносяться POP3 Server або IMAP Server, port.

8.3.4 Увімкніть IMAP (рис. 8.13).

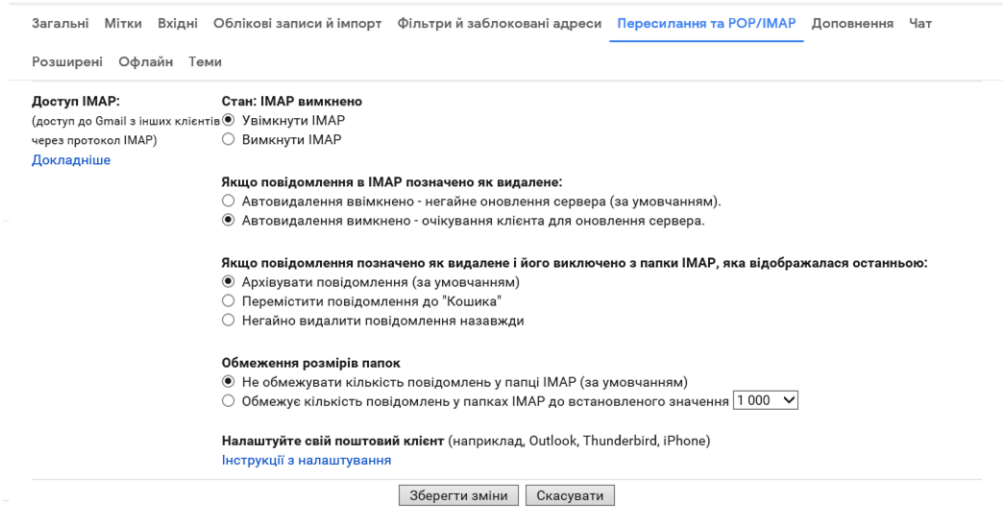

Рисунок 8.13 — Приклад включення IMAP на поштовому сервісі

8.3.5 Деякі поштові сервіси вимагають дозвіл для неперевірених додатків (рис. 8.14).

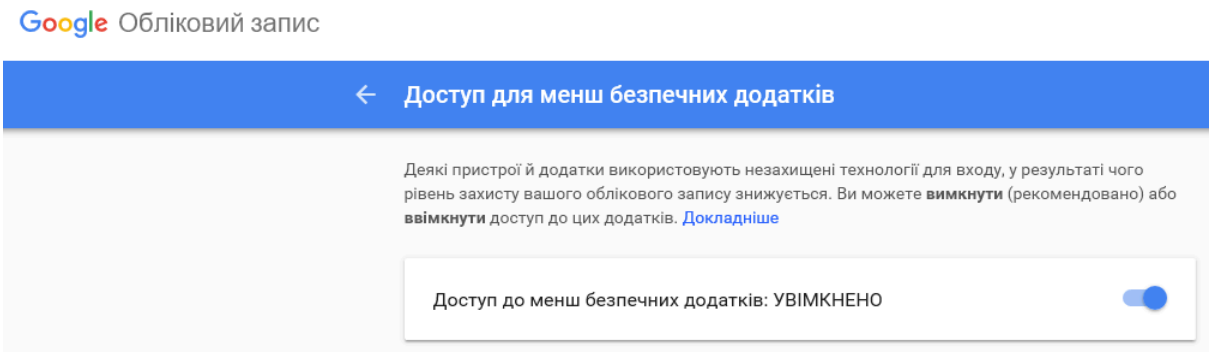

Рисунок 8.14 — Приклад увімкнення дозволу для небезпечних додатків

8.3.6 Налаштуйте вузли, як показано на рис. 8.15.

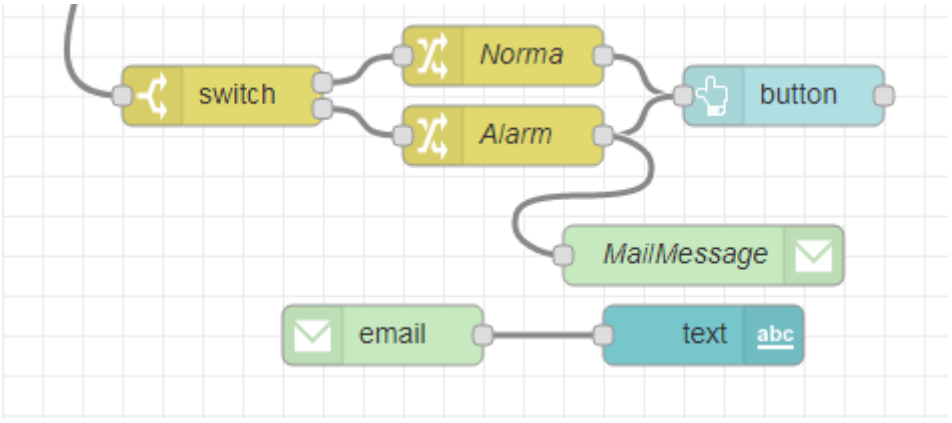

Рисунок 8.15 — Схема із доданим поштовим вузлом

8.3.7 Налаштування вузлів показано на рис. 8.16 та 8.17. Заповніть поле отримувача свою поштову скриньку.

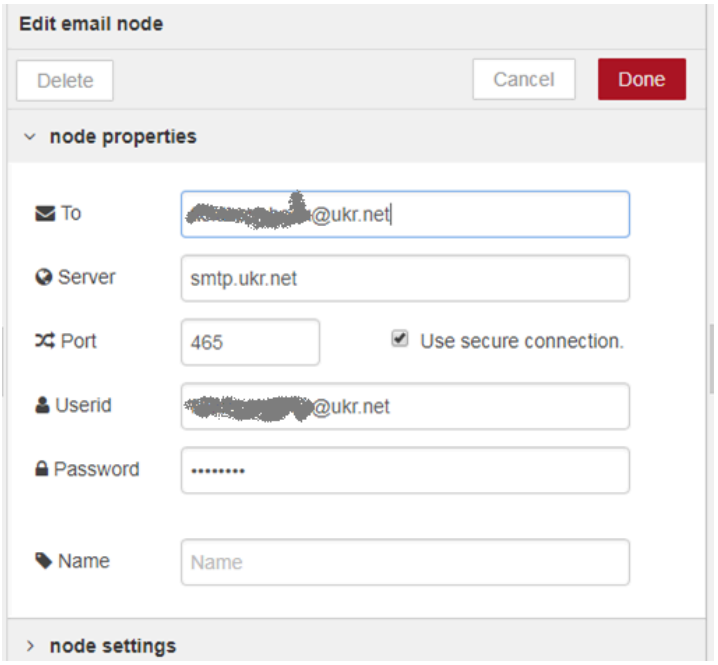

Рисунок 8.16 — Налаштування SMTP на вузлі

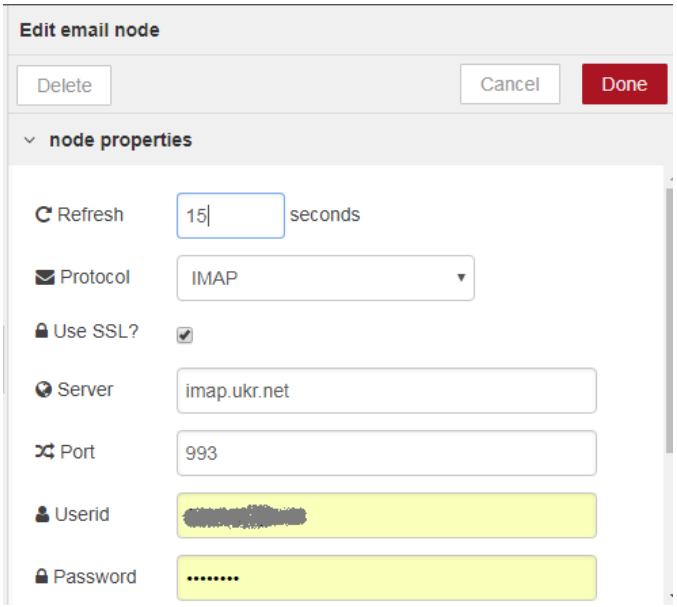

Рисунок 8.17 — Налаштування IMAP на вузлі

8.3.8 Ініціюйте ситуацію, яка спровокує надсилання повідомлення. Для цього потрібно виставити значення заданої температури. Повідомлення з пошти повинно відображатися на веб-сторінці у текстовому полі (рис. 8.18).

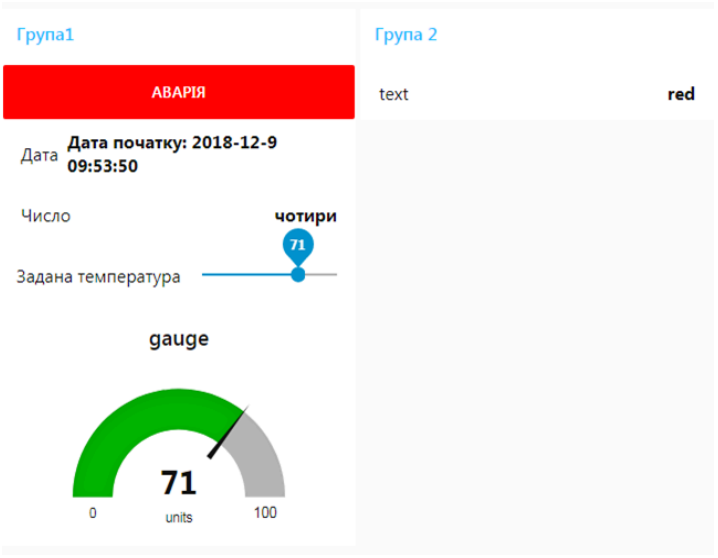

Рисунок 8.18 — Приклад очікуваного повідомлення при перевищенні температури

Завдання 8.4. Робота з Modbus

8.4.1 Встановіть пакет Modbus (node-red-contrib-modbustcp) з використанням Manage Palette (рис. 8.21).

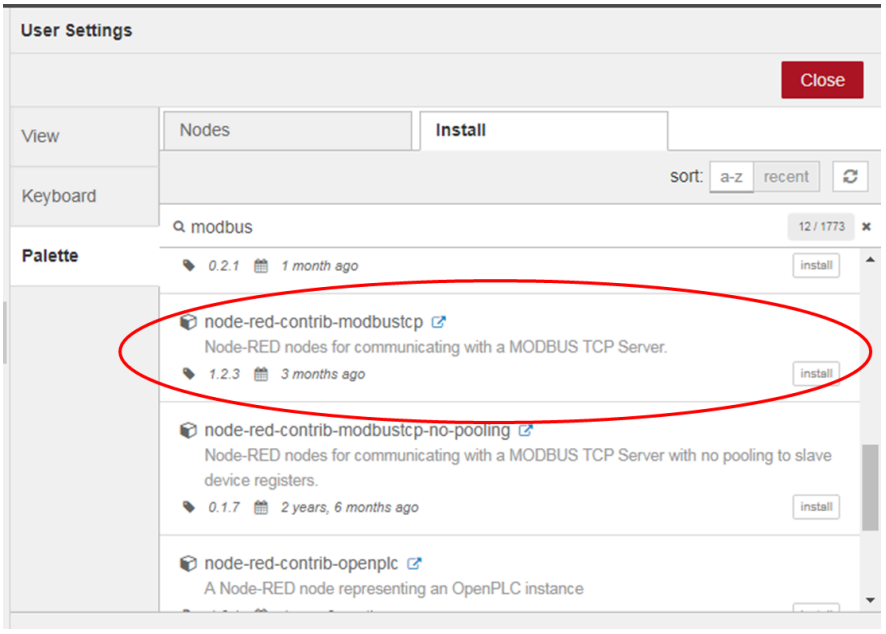

Рисунок 8.21 — Пакет modbus

8.4.2 Завантажте Modbus PLC Simulator (Mod_Rssim) з https://sourceforge.net/projects/modrssim/. Як альтернативу, можна використовувати https://sourceforge.net/projects/modrssim2/

8.4.3 Завантажте та встановіть дистрибутив https://sourceforge.net/projects/modrssim/files/SimSetup.msi/download

8.4.4 Запустіть на виконання Modbus PLC Simulator C:\Program Files (x86)\EmbeddedIntelligence\Mod_Rssim. Виставте значення в Prot: Modbus TCP (рис. 8.21).

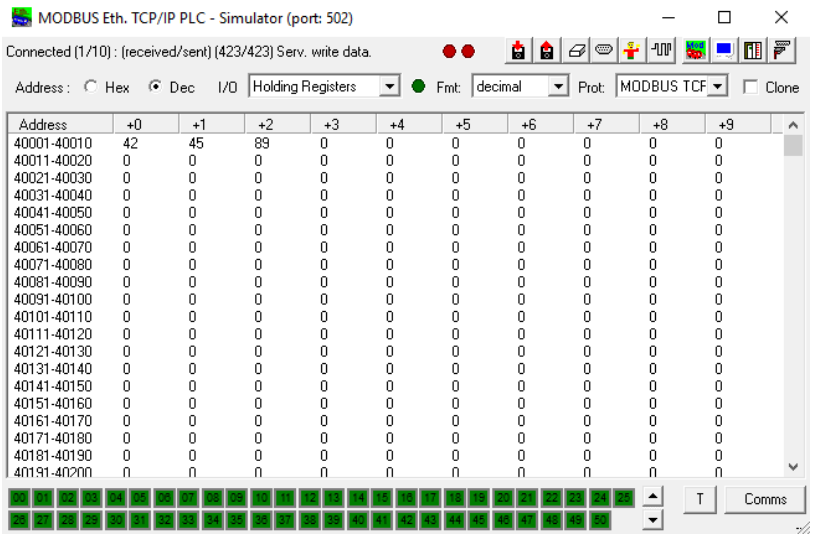

Рисунок 8.21 — Значення в Prot: Modbus TCP

8.4.5 Документацію можете подивитись за посиланням https://flows.nodered.org/node/node-red-contrib-modbustcp

8.4.6 З палітри Inputs додайте елемент modbustcp-read, зайдіть в налаштування. Клікніть на кнопку з олівцем для створення нового серверу (рис. 8.22).

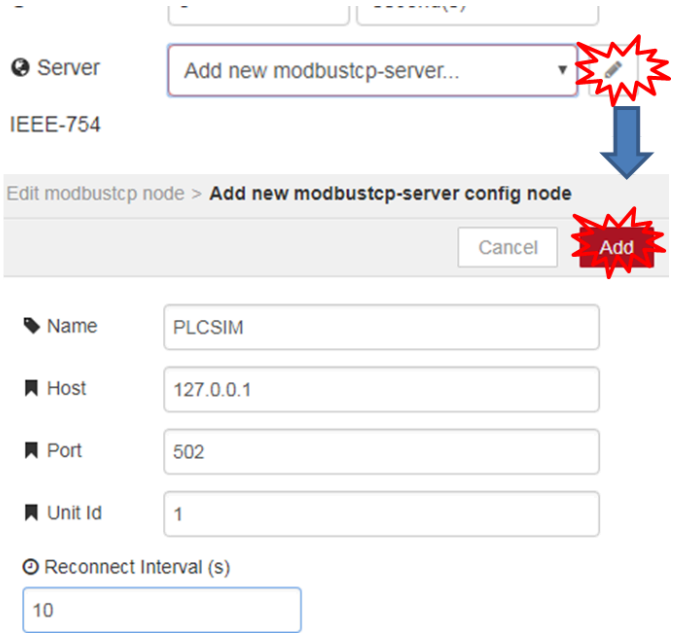

Рисунок 8.22 — Створення нового серверу

8.4.7 Зробіть зчитування десяти Holding регістрів починаючи з 0-го. Приклад зображено на рис. 8.23.

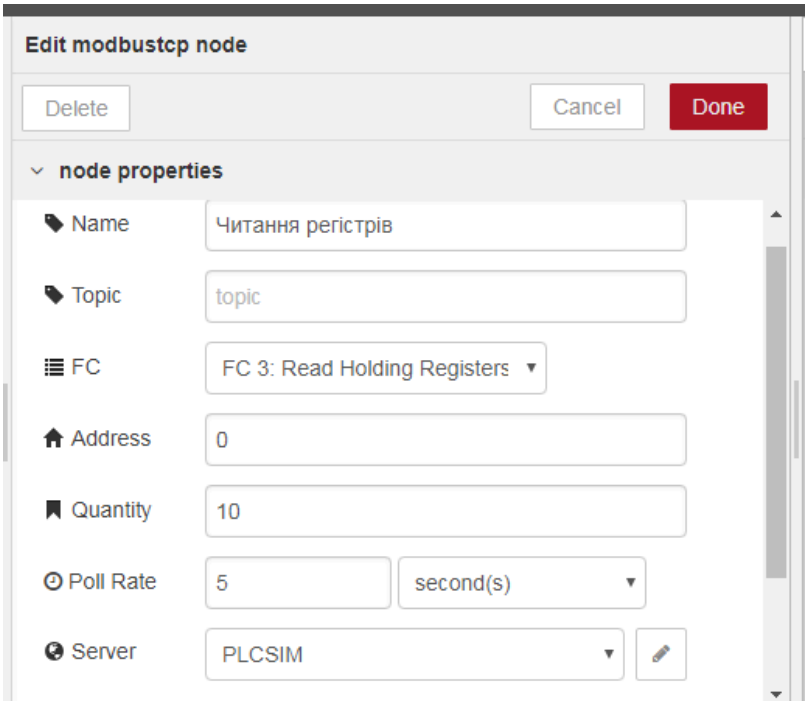

Рисунок 8.23 — Зчитування десяти Holding регістрів

8.4.8 Підготуйте вузли таким чином, як це на рис. 8.24. Розгорніть проект та деактивуйте усі виводи debug окрім останнього.

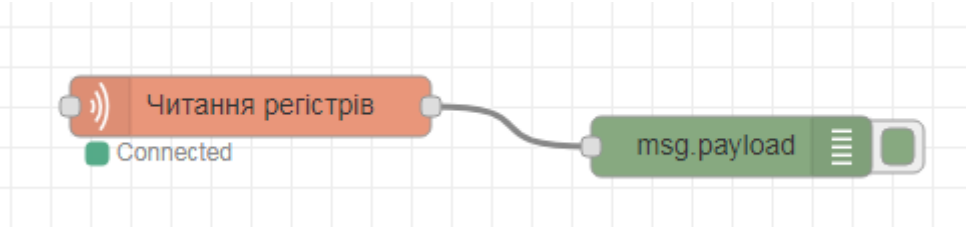

Рисунок 8.24 – Фрагмент програми

8.4.9 Використовуючи програму Mod_RSsim, оновіть значення перших десяти регістрів. Включіть вікно для відладочних повідомленьg. Перевірте, що у вікні є вивід масиву зі значеннями регістрів (рис. 8.25).

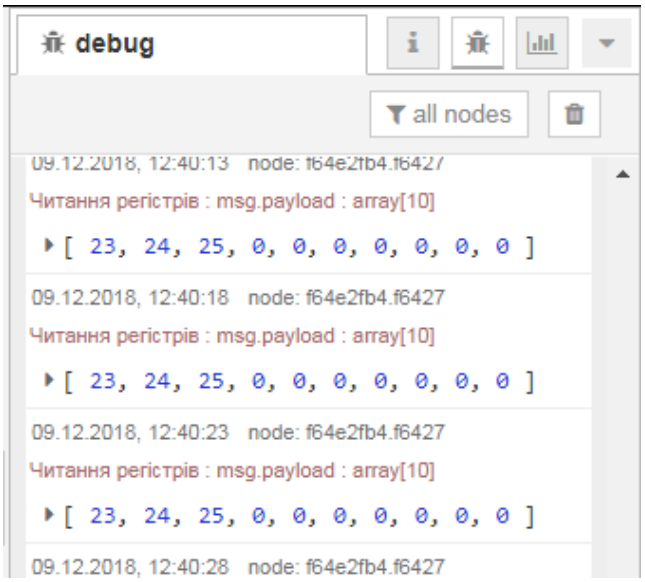

Рисунок 8.25 – Фрагмент програми

8.4.10 Оскільки msg.payload представляє собою масив з десяти елементів, то перед виводом на WEB-сторінку його потрібно підготувати.

8.4.11 Реалізуйте вузли таким чином, як зображено на рис. 8.26 та 8.27. Перевірте правильність роботи програми.

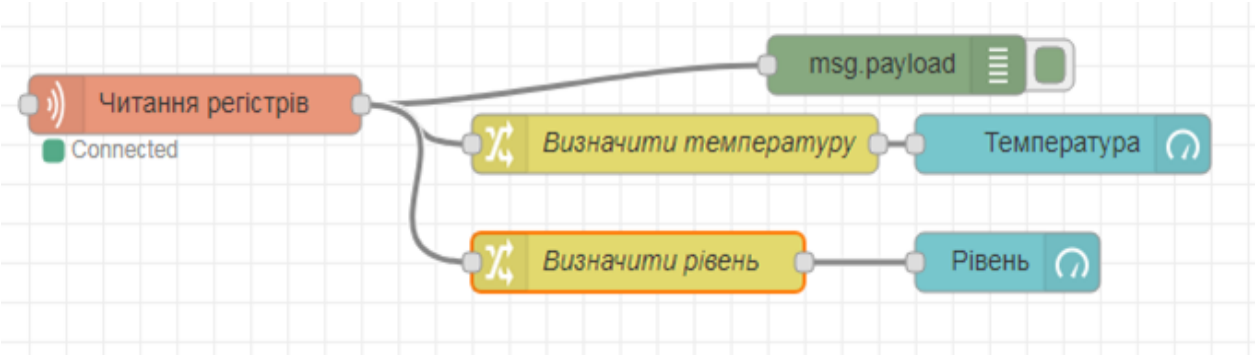

Рисунок 8.26 — Модифікація програми

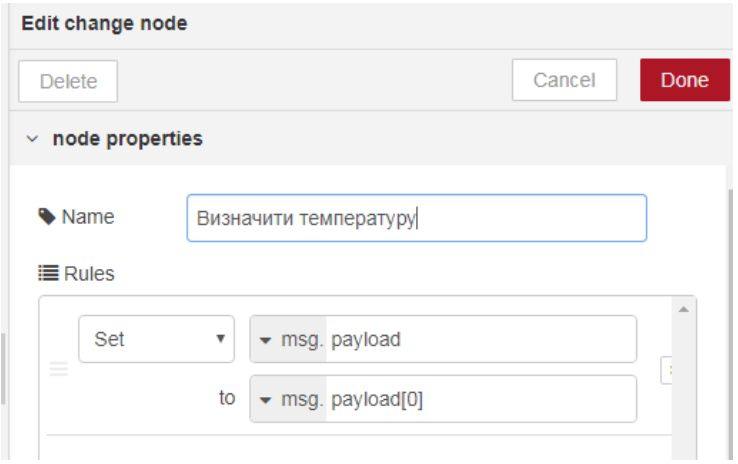

Рисунок 8.27 — Зміна налаштувань на вузлі change

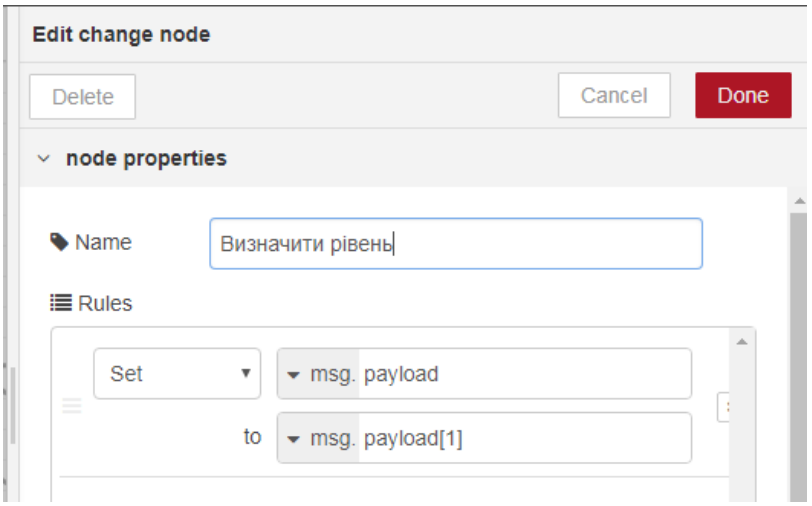

Рисунок 8.28 — Зміна налаштувань на вузлі change

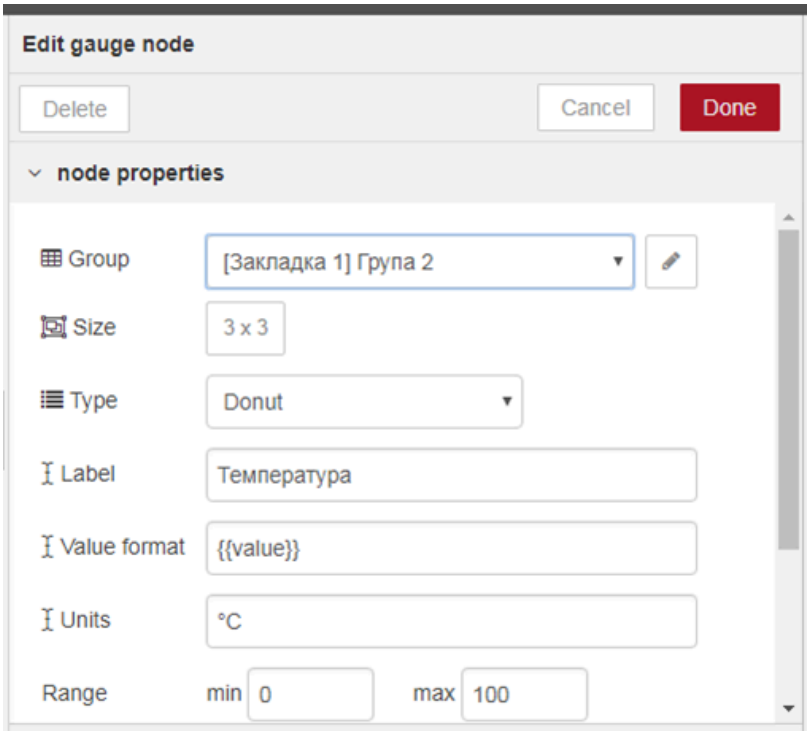

Рисунок 8.29 — Зміна налаштувань на вузлі change

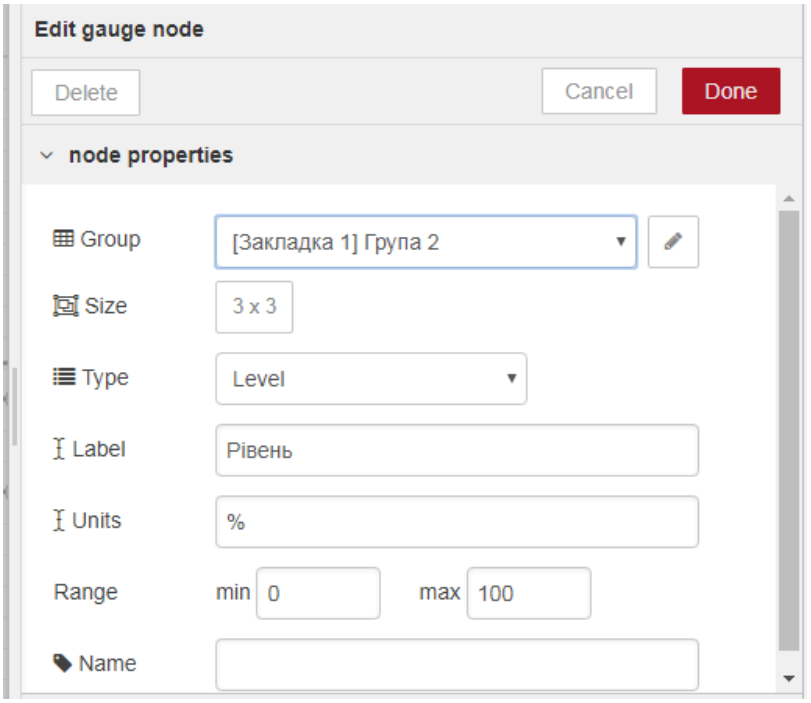

Рисунок 8.30 — Зміна налаштувань на вузлі change

8.4.12 Щоб записати нові значення регістрів, можна використати modbustcp-write. Змініть програму таким чином, щоб вона була як показано на рис. 8.31. Зробіть перевірку, що відбувається зміна значення Holding регістру

в Mod_RSsim. Зміна повинна відбуватись через через елемент «Задана температура».

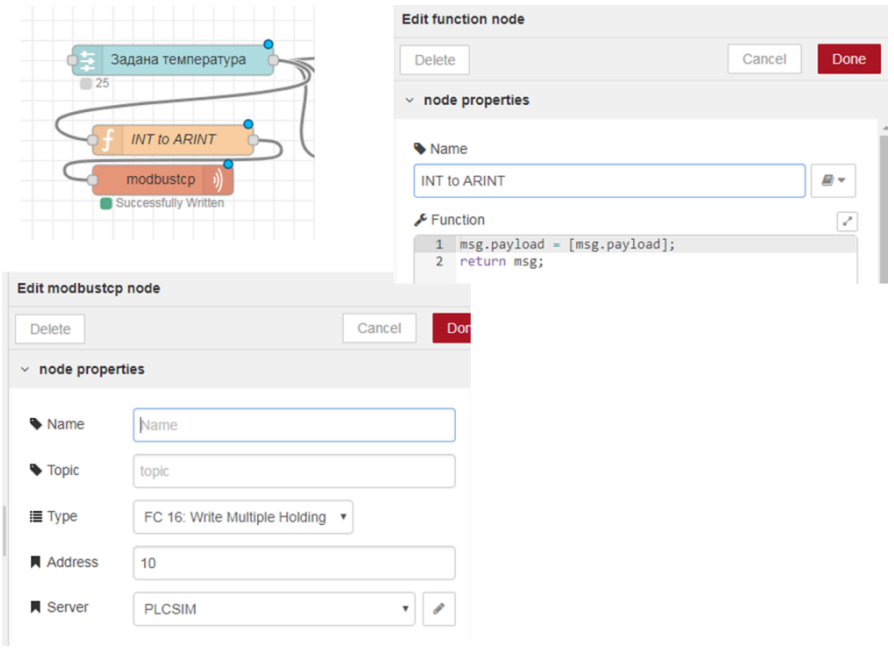

Рисунок 8.31 — Зміна налаштувань на вузлі change

Завдання 8.5. Обробка інформації системи з використанням JS

8.5.1 Встановіть модуль node-red-contrib-os (рис. 8.32).

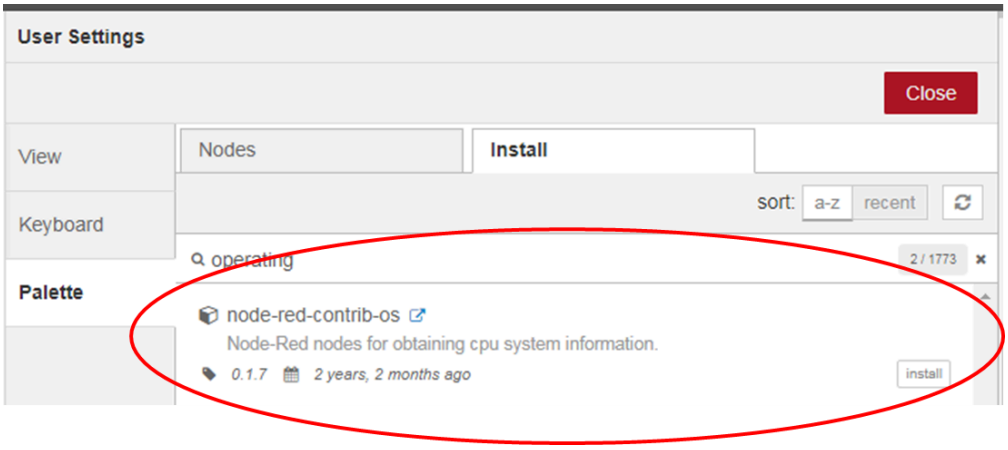

Рисунок 8.32 — Встановлення модуля node-red-contrib-os

8.5.2 Після успішної установки модуля, має бути вузол типу Networkintf. Приклад зображено на рис. 8.33.

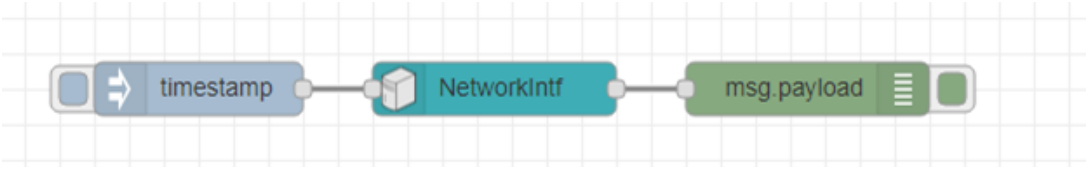

Рисунок 8.33 — Використання вузла типу Networkintf

8.5.3 Розгорніть програму та перевпевнетісь у тому, що інформація надається у вигляді JS об’єкту. Він має включати в себе об’єкт NetworkInterfaces (рис. 8.34).

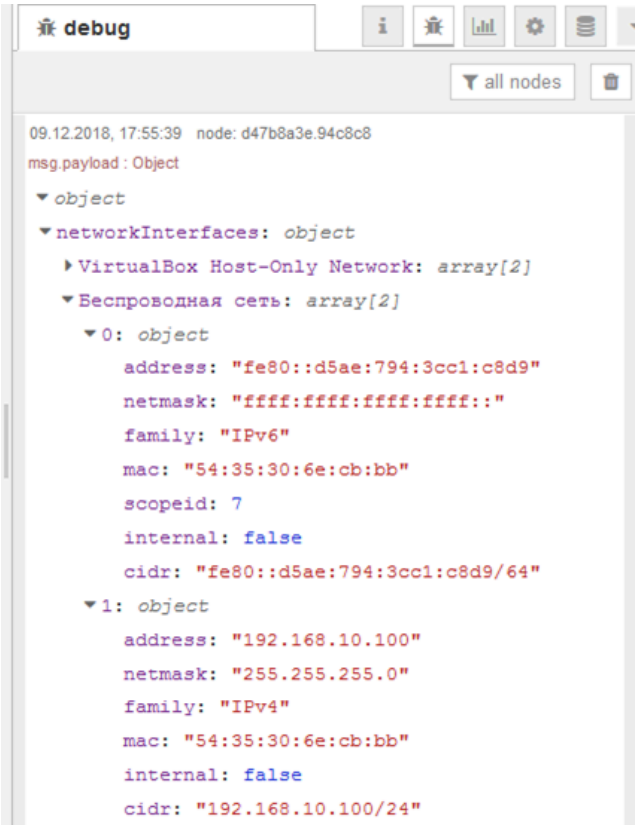

Рисунок 8.34 — Приклад виводу мережевої інформації

8.5.4 Розробимо програму для виводу MAC-адрес (рис. 8.35-8.36).

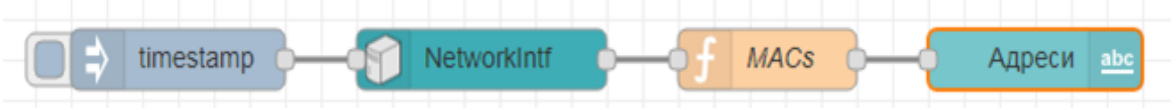

Рисунок 8.35 — Схема вузлів для виводу MAC-адрес комп’ютера

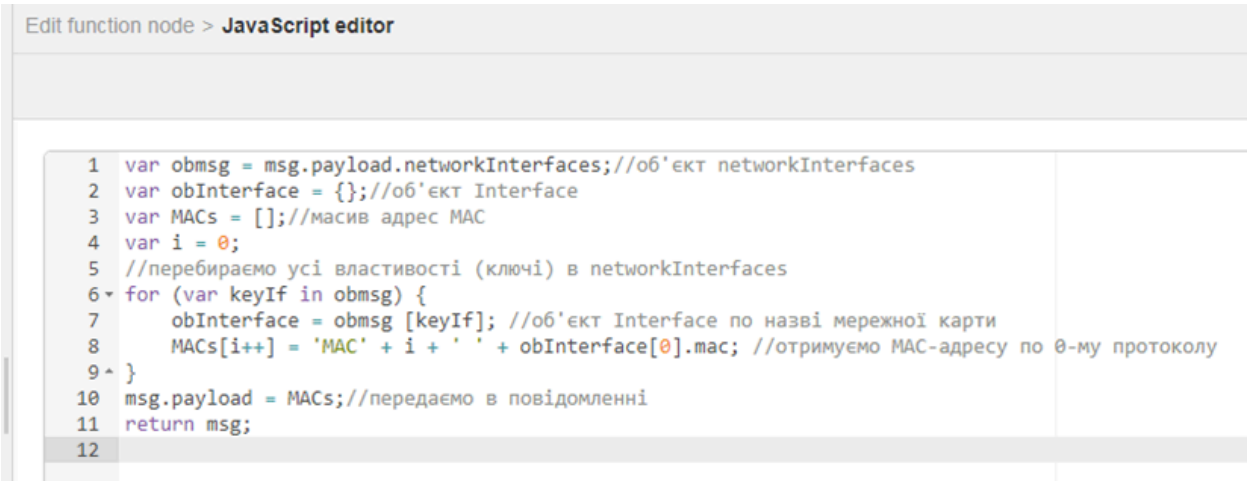

Рисунок 8.36 — Програма виводу MAC-адрес комп’ютера

Контрольні запитання

1) Що таке Node-Red та для чого він потрібен?

2) Які вузли існують та які функції мають?

3) Як змінити властивості вузла?

4) Як налаштувати пошту?

5) Що таке бібліотека Modbus та навіщо потрібна?

Навчальні матеріали та ресурси

1. Технології створення інтернету речей. Комп’ютерний практикум [Електронний ресурс] : навчальний посібник для здобувачів ступеня магістра за освітньою програмою «Інформаційне забезпечення робототехнічних систем» за спеціальністю 126 «Інформаційні системи та технології» / Жураковський Б. Ю., Федорова Н. В., Гаврилко Є. В., Зенів І. О. ; КПІ ім. Ігоря Сікорського. – Електронні текстові дані (1 файл: 6,28 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2021. – 127 с. Режим доступу: https://ela.kpi.ua/handle/123456789/46169;

2. Жураковський Б.Ю Технології інтернету речей. Навчальний посібник [Електронний ресурс] / Б. Ю. Жураковский, І. О. Зенів // КПІ ім. Ігоря Сікорського. – 2021. – 503 с. Режим доступу до ресурсу: https://ela.kpi.ua/handle/123456789/42078;

3. Жураковський, Б. Ю. Безпровідні технології для управління смарт-середовищами. Навчальний посібник для виконання лабораторних робіт [Електронний ресурс] : навч. посіб. для студентів спеціальності 126 «Інформаційні системи та технології» денної та заочної форми навчання / Жураковський Б. Ю., Нікітін В. А. ; КПІ ім. Ігоря Сікорського. – Електронні

текстові дані (1 файл: 10.05 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2023. – 122 с. Режим доступу: https://ela.kpi.ua/handle/123456789/57341;

4. What are character special and block special files in a unix system? [Електронний ресурс] - https://unix.stackexchange.com/questions/60034/what-are-character-special-and-block-special-files-in-a-unix-system;

5. Socat [Електронний ресурс] - https://linux.die.net/man/1/socat;

6. Setting Up Minicom [Електронний ресурс] - https://developer.ridgerun.com/wiki/index.php/Setting_Up_Minicom;

7. UART [Електронний ресурс] — https://www.analog.com/en/analog-dialogue/articles/uart-a-hardware-communication-protocol.html;

8. Fundamentals of RS-232 Serial Communications. [Електронний ресурс]: https://www.analog.com/en/technical-articles/fundamentals-of-rs232-serial-communications.html;

9. Exploring Internet of Things with Cisco Packet Tracer. [Електронний ресурс]: https://skillsforall.com/course/exploring-iot-cisco-packet-tracer?utm_source=netacad.com&utm_medium=referral&utm_campaign=packet-tracer&courseLang=en-US&userlogin=0;

10. Програмування пристроїв Інтернету речей: лабораторний практикум [Електронний ресурс] : навчальний посібник для студентів спеціальності 121 «Інженерія програмного забезпечення» (освітня програма «Програмне забезпечення комп’ютерних та інформаційно-пошукових систем») / КПІ ім. Ігоря Сікорського ; уклад.: Л. М. Олещенко, Я. В. Хіцко. – Електронні текстові дані (1 файл: 2,64 Мбайт). – Київ : КПІ ім. Ігоря

Сікорського, 2019. – 47 с. Режим доступу: https://ela.kpi.ua/handle/123456789/28080;

11. Яременко, К. М. Автоматизація планування розумного будинку : магістерська дис. : 8.05010103 Системне проектування / Яременко Костянтин Миколайович. - Київ, 2018. - 81 с. Режим доступу: https://ela.kpi.ua/handle/123456789/26929.