КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ТАРАСА ШЕВЧЕНКА

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

До захисту допущено
Завідувач кафедри ІСТ
_____ Олександр КУЧАНСЬКИЙ
(підпис) (ім'я, ПРІЗВИЩЕ)
"___"_____ 2022р.

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

спеціальності 126 «Інформаційні системи та технології»

освітньої програми «Програмні технології інтернет речей»

на тему: «Створення бездротової мережі з використанням повторювача на базі ESP, для забезпечення гарантованого зв'язку стандарту Wi-Fi»

Виконав: студент 4 курсу, групи IP-41
(шифр групи)

Крук Назар _____ _____
(Ім'я, ПРІЗВИЩЕ) (підпис)

Керівник д.т.н. професор Михайло СТЕПАНОВ _____ _____
(посада, науковий ступінь, вчене звання, Ім'я, ПРІЗВИЩЕ) (підпис)

Консультант нормо контроль к.т.н., доцент Ростислав ЛІСНЕВСЬКИЙ _____
(назва розділу) (посада, вчене звання, науковий ступінь, Ім'я, ПРІЗВИЩЕ) (підпис)

Рецензент професор кафедри Прикладної радіоелектроніки КПІ, д.т.н., професор Володимир Дружинін _____ _____
(посада, науковий ступінь, вчене звання, науковий ступінь, Ім'я, ПРІЗВИЩЕ) (підпис)

Засвідчую, що у пояснювальна записка не має запозичень з праць інших авторів без відповідних посилань.

Здобувач освіти _____
(підпис)

Київ – 2022 року

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра Інформаційні системи та технології
Освітній рівень Бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітня програма Програмні технології інтернет речей

ЗАТВЕРДЖУЮ
Завідувач кафедри ІСТ
Олександр КУЧАНСЬКИЙ
_____
«__» _____ 2022 року

ЗАВДАННЯ
НА ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ БАКАЛАВРА

Здобувач освіти: Назар КРУК

Група: ІР-41

1. **Тема кваліфікаційна робота бакалавра**: «Створення бездротової мережі з використанням повторювача на базі ESP, для забезпечення гарантованого зв'язку стандарту Wi-Fi».

Затверджена протоколом засідання кафедри ІСТ №05/21_22 від 03.12.2021 року

2. **Строк подання студентом готової роботи** – «21» червня 2022 р.

3. **Вихідні дані до роботи**: схема існуючої мережі, проект розширення мережі (схема, алгоритми налаштування та роботи повторювача, роботи програми та її процедур), програмний код повторювача, реалізоване розширення мережі на базі ESP8266 ESP-01S, заміри швидкості Інтернет-зв’язку та затримок, їх аналіз.

4. **Зміст роботи**: 1 АНАЛІЗ ТА ОПИС СФЕРИ ВИКОРИСТАННЯ МІКРОКОНТРОЛЕРІВ. ОГЛЯД ГОТОВИХ РІШЕНЬ (опис об’єкту дослідження, існуючі рішення, Wi-Fi подовжувач з додаткового маршрутизатора (використання другого маршрутизатора у якості точки доступу), комерційний Wi-Fi повторювач, створення Wi-Fi повторювача на базі Raspberry Pi, створення Wi-Fi повторювача на базі LuaNode32, обрання рішення); 2 РОЗРОБКА ПРОЕКТУ МЕРЕЖІ (огляд існуючої мережі, опис компонентів мережі,

компоненти існуючої мережі, ESP8266 версії ESP-01S, створення проекту розширення мережі); 3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ЕКСПЕРИМЕНТ ТА АНАЛІЗ ЕФЕКТИВНОСТІ РІШЕННЯ (опис алгоритмів функціонування, розробка програмного забезпечення мікроконтролера, налаштування мікроконтролера, введення мікроконтролера в експлуатацію, збір даних та аналіз ефективності рішення).

5. **Перелік графічного матеріалу**: схеми існуючої мережі, розширення мережі; алгоритми налаштування та роботи повторювача, роботи програми та її процедур; інтерфейс веб-сторінки налаштування повторювача; лінійні діаграми результатів замірів швидкості Інтернет-зв’язку та затримок.

6. **Календарний план виконання роботи:**

Етапи виконання кваліфікаційної роботи бакалавра	**Термін виконання**	**Результат виконання**
1. Вибір тематики кваліфікаційної роботи бакалавра	01.09.2021-01.10.2021	виконано
2. Наказ про затвердження тем кваліфікаційної роботи бакалавра та призначення керівників	03.12.2021	виконано
3. Розробка плану кваліфікаційної роботи бакалавра і його погодження з керівником	25.12.2021	виконано
4. Написання I розділу кваліфікаційної роботи	19.03.2022	виконано
5. Написання II розділу кваліфікаційної роботи	25.04.2022	виконано
6. Написання III розділу кваліфікаційної роботи	29.04.2022	виконано
7. Підготовка висновків і пропозицій	04.05.2021	виконано
9. Попередній захист кваліфікаційної роботи	12.05.2022	виконано
9. Перевірка на плагіат	13.05.2022-15.06.2022	виконано
10. Нормоконтроль	02.06.2022-06.06.2022	виконано

11. Рецензування кваліфікаційної роботи бакалавра і представлення роботи на кафедру в друкованому вигляді	15.06.2022	виконано
12. Захист кваліфікаційної роботи бакалавра	23.06.2022-24.06.2022	

Дата видачі завдання «__» _____ 2022 р.

Керівник роботи: д.т.н. професор Михайло СТЕПАНОВ _____ (підпис)

Завдання прийняв до виконання:
Здобувач освіти на освітньому рівні «бакалавр» 4-го курсу групи ІР-41

Назар КРУК _____
(Власне Ім'я, ПРІЗВИЩЕ) (підпис)

АНОТАЦІЯ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

Факультет інформаційних технологій

Кафедра інформаційних систем та технологій

Освітня програма «Програмні технології інтернет речей»

Кваліфікаційна робота бакалавра Назара КРУКА

Тема роботи: «Створення бездротової мережі з використанням повторювача на базі ESP, для забезпечення гарантованого зв'язку стандарту Wi-Fi».

Мета кваліфікаційної роботи бакалавра – розробка програмного забезпечення для мікроконтролера для повторення сигналу з основного маршрутизатора, проектування та реалізація розширення домашньої Wi-Fi мережі шляхом повторення сигналу з основного маршрутизатора.

Об’єкт дослідження – використання технології IoT для розумного будинку.

Предмет дослідження – мікроконтролер ESP8266 версії ESP-01S в мережі Wi-Fi.

Апробація результатів. Основні положення і результати досліджень, викладені у проекті, пройшли апробацію на XVIII Міжнародній науково-технічній конференції «Проблеми інформатизації» у грудні 2021-го року.

Кваліфікаційна робота бакалавра складається зі змісту, вступу, основної частини, яка включає чотири розділи, висновків та списку використаних джерел. Всього 57 сторінок.

КЛЮЧОВІ СЛОВА: IoT, Wi-Fi, повторювач, мікроконтролер, ESP, ESP8266, ESP-01S.

ABSTRACT

TARAS SHEVCHENKO NATIONAL UNIVERSITY OF KYIV

Faculty of Information Technology

Department of Information Systems and Technology

Educational Program "Software Technologies of the Internet of Things"

Qualification work of master Nazar KRUK.

Work topic: "Creating a wireless network using an ESP-based repeater to ensure guaranteed Wi-Fi connectivity."

The purpose of the bachelor’s qualification work is to develop software for a microcontroller to repeat the signal from the main router, design and implement the extension of the home Wi-Fi network by repeating the signal from the main router.

The object of research is use of IoT technology for a smart home.

The subject of research is the ESP8266 version ESP-01S microcontroller in Wi-Fi network.

Approbation of results. The main provisions and results of the research presented in the project were tested at the XVIII International Scientific and Technical Conference "Problems of Informatization" in December 2021.

The bachelor's qualification work consists of the content, introduction, main part, which includes four sections, conclusions and a list of sources used. Total 57 pages.

KEY WORDS: IoT, Wi-Fi, repeater, microcontroller, ESP, ESP8266, ESP-01S.

ЗМІСТ

ВСТУП

Ми живемо у чудові часи, коли розвиток технологій набрав надзвичайний темп, і усіма пристроями власного розумного будинку, як от обігрівач, пральна машина, бойлер тощо, можна керувати з будь-якої точки земної кулі, де є підключення до мережі Інтернет. Проте кожному з пристроїв з підтримкою технології IoT на основі Wi-Fi необхідне підключення, і використання безпровідних технологій мають свої обмеження - схильність до впливу шумів, інших мереж, особливо у розповсюдженому діапазоні 2.4Ггц, або ж банальна віддаленість від джерела сигналу.

Актуальні події, а саме пандемія коронавірусу та військовий стан по всій Україні, внесли свої корективи у звичне життя – заради безпеки життя та здоров’я люди більшу частину часу залишаються удома та потребують постійного доступу до актуальної інформації, як то стан повітряної тривоги у селі, де сирени відсутні як такі, проте є доступним Інтернет-зв’язок.

Саме з цієї причини виникає необхідність у повторювачах та розширювачах Wi-Fi сигналу. Ринок наповнений різноманітними комерційними рішеннями, хоча й існують значно дешевші, на основі мікроконтролерів та мікрокомп’ютерів, і часто таких простих пристроїв цілком достатньо.

Задачею даної бакалаврської роботи є дослідження готових рішень, існуючої мережі, опис її компонентів та компоненту для розширення області Wi-Fi сигналу, створення проекту її розширення, написання програмного коду та введення повторювача в експлуатацію.

У якості існуючої мережі обрано власну домашню Wi-Fi мережу у приватному двоповерховому будинку з підвалом. Маршрутизатор розташований на другому поверсі для забезпечення максимальної швидкості сигналу на кінцевих пристроях, таких як персональний комп’ютер та смартфони, отже, його переміщення недоцільне. У підвалі, який є технічним приміщенням та використовується у якості укриття під час повітряної тривоги, сигнал відсутній як такий через широкі бетонні міжповерхові перекриття та підігрів підлоги з металевих труб, що заважають проходженню. Необхідно розробити проект

розширення мережі, за якого будуть можливими доступ в інтернет з будь-якої кімнати будинку, отримання інформації щодо повітряної тривоги та текстове онлайн-листування у месенджерах на кшталт Telegram та Viber за адекватну ціну. Необхідним є саме розширення вже існуючої мережі – власник будинку не потребує жодних покращень чи замін і не має для цього необхідного бюджету.

1. АНАЛІЗ ТА ОПИС СФЕРИ ВИКОРИСТАННЯ МІКРОКОНТРОЛЕРІВ. ОГЛЯД ГОТОВИХ РІШЕНЬ

1.1. Опис об’єкту дослідження

Технологія Інтернету речей (IoT) зростає настільки швидко, що вже затьмарила звичайні технологічні системи з точки зору особливостей та функціональних можливостей. Застосування IoT у сферах охорони здоров’я, науки про життя та роздрібної торгівлі революціонізує бізнес, особливо лабораторії та охорону здоров’я, надаючи їм незмірну користь. Одним із таких застосувань IoT є створення так званих розумних будинків.

Розумний дім – це оселя, в якій існує своя домашня мережа та підключені до Інтернету смарт-пристрої для віддаленого моніторингу та керування приладами та системами, наприклад, опалення та освітлення, а також автоматизації. Така система націлена на забезпечення безпеки та комфорту, а також енергоефективності житла, її компоненти обмінюються даними між собою, а керування часто реалізовано у вигляді зручного додатку на смартфон, планшет чи інший кінцевий пристрій (рис. 1.1).

Альтернативи та покращення звичних побутових речей від сфери IoT вже існують майже у кожному аспекті, і з кожним днем їх стає все більше:

- смарт-телевізори можуть завантажувати медіаконтент (відео, музика тощо) з мережі Інтернет, а деякі з них вчаться розпізнавати людські жести та мову;
- завдяки розумним термостатам, їх власники можуть створювати графіки опалення та провітрення кімнат, коли це потрібно, керувати пристроями дистанційно, що забезпечує максимальний комфорт, а також отримувати сповіщення щодо власного стану та споживання електроенергії;
- інтелектуальні системи освітлення, що отримають інформацію з датчиків руху та сонячного світла, працюють тільки тоді, коли це

необхідно, що економить електроенергію та робочий ресурс пристроїв;

- догляд за тваринами та рослинами значно спрощують автоматизовані годівниці та системи поливу;
- розумні системи відеоспостереження та контролю дверей і вікон надають можливість контролю приміщень за відсутності, надавати та забороняти доступ конкретним відвідувачам, мають функціонал сповіщення поліції;
- моніторинг систем електро-, газо- та водопостачання забезпечують контроль над станом обладнання та має можливість відключення у разі виникнення небезпечної ситуації, таких як різке підвищення напруги тощо;
- кухонна смарт-техніка значно спрощує життя завдяки автоматизації звичних щоденних процесів, від заварки свіжої чашки кави у заданий час до перевірки терміну придатності продуктів у холодильнику.

Рисунок 1.1 – Зображення можливостей технології розумного дому [1]

1.2. Існуючі рішення

Ще років 10 тому IoT був великою перспективою в майбутньому, оскільки вона дозволяє належним чином контролювати сутності за допомогою постійної та підтримуючої мережевої інфраструктури. Існує велика ймовірність того, що цифровий і фізичний світ співпрацюють як єдине ціле за допомогою комп'ютерних структур. Концепція Інтернету речей відкриває свої двері для багатьох кінцевих споживачів, таких як звичайні люди, галузі та підприємства, а також уряди. І з кожним роком компаній та різноманітних рішень у цій сфері стає дедалі більше. Варто зазначити, що українські компанії, які спеціалізуються на рішеннях у сфері IoT, входять у топ компаній за даними сайту clutch.co [2], а Indeema Software, що знаходиться у Львові, взагалі посідає перше місце рейтингового списку.

При розробці власної системи необхідно опиратись на досвід інших розробників, проаналізувати вже готові рішення, тож далі описані декілька з багатьох прикладів.

1.2.1. Wi-Fi подовжувач з додаткового маршрутизатора (використання другого маршрутизатора у якості точки доступу)

При виникненні проблеми відсутності Wi-Fi сигналу у певній частині приміщення та недоцільності зміни розташування основного маршрутизатора, першою на думку спадає можливість придбання ще одного, можливо, навіть аналогічного, та підключення до вже існуючого. За словами автора статті [3], для використання у якості точки доступу, достатньо відключити деякі функції і підключити по кабелю до іншого маршрутизатора, або до модему (рис. 1.2).

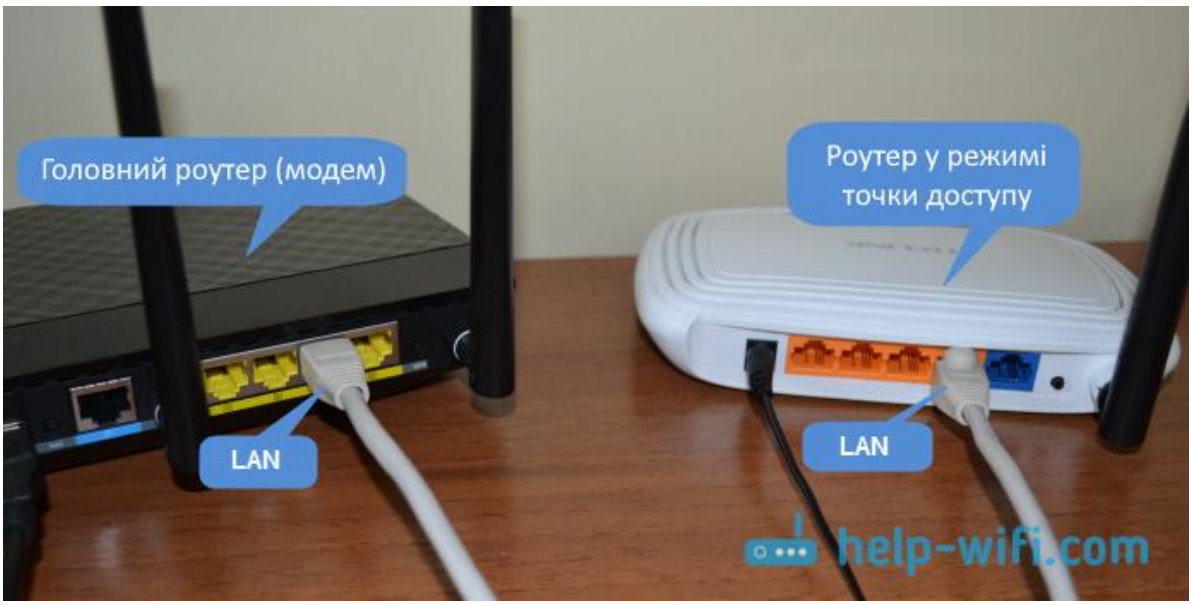


Рисунок 1.2 – Наочне використання другого маршрутизатора у якості точки доступу

Налаштування пристроїв кожного виробника відрізняється, хоч і не суттєво – найбільша різниця полягає у різних інтерфейсах програмного забезпечення та розташувань певних вкладок меню тощо. Принцип налаштування виглядає наступним чином:

1. Відкриття налаштувань маршрутизатора (зазвичай це перехід за IP адресою 192.168.1.1 у будь-якому браузері на пристрої, що підключений до мережі).
2. Налаштування статичної IP адреси.
3. Відключення DHCP.
4. Налаштування мережі.
5. Збереження налаштувань та перезавантаження пристрою.

Дане рішення є досить надійним, особливо при використанні однакових або ж аналогічних маршрутизаторів одного виробника, просте у налаштуванні, проте достатньо дорогим та громіздким – вартість найпростішого маршрутизатора починається від 500 гривень, і його потім необхідно підключити за допомогою дроту, що не завжди є можливим та несе за собою додаткові витрати.

1.2.2. Комерційний Wi-Fi повторювач

Не варто забувати, що на ринку побутової електроніки існують незаслужено обділені увагою Wi-Fi повторювачі (рис. 1.3), які достатньо увімкнути у розетку, підключити до основної мережі та користуватись замість того, щоб переставляти маршрутизатор та прокладати додаткові дроти по приміщенню. Автор статті [4] наводить декілька важливих моментів у використанні ретрансляторів:

- комерційні повторювачі часто налаштовані так, щоб мережа залишалась мережа з тією самою назвою та паролем, що й основна – тобто, відбувається копіювання налаштувань маршрутизатора;
- усі пристрої, які вже були підключені до домашньої мережі, будуть автоматично підключатись до кращого джерела сигналу, коли знаходитимуться у зоні дії маршрутизатора чи повторювача, що робить їх використання зручнішим;
- через те, що підключені смартфони, планшети, телевізори, персональні комп’ютери тощо підключені до однієї мережі, налаштування локальної мережі та взаємодії між пристроями значно спрощується – наприклад, можна налаштувати DLNA сервер і дивитися фільми з комп'ютера на телевізорі, хоча й вони підключені до різних джерел сигналу.

Рисунок 1.3 – Схематичне зображення принципу роботи Wi-Fi повторювача

Дійсно, використання такого пристрою є значно простішим в налаштуванні та експлуатації рішенням, проте й воно має свої недоліки. По суті, ретранслятор приймає Wi-Fi сигнал від головного маршрутизатора та передає його далі, тобто виступає у ролі підсилювача. Його завдання – прийняти певну Wi-Fi мережу, і передати її далі. Отже, він не зможе бути настільки ж ефективним, як маршрутизатор, та роздавати сигнал швидше, ніж отримує – в результаті ми маємо лише «підтримку» сигналу там, куди він не може дістати з основної точки доступу. Також варто зазначити, що найдешевші рішення в середньому коштують від 620 гривень, що трохи дорожче за другий маршрутизатор.

1.2.3. Створення Wi-Fi повторювача на базі Raspberry Pi

Коли йде мова про IoT та розумні будинки, не варто оминати область програмованих мікроконтролерів, які надають «ціле море» можливостей аматорам та ентузіастам, що прагнуть покращення власного добробуту та автоматизації оселі. Одними з найпопулярніших є рішення компанії The Raspberry Pi Foundation, а саме мікрокомп’ютери Raspberry Pi (рис. 1.4). У даній статті [5] наведена покрокова інструкція щодо налаштування найпопулярніших моделей на роботу у якості Wi-Fi повторювача.

Рисунок 1.4 – Raspberry Pi у саморобному корпусі з Wi-Fi адаптером

В результаті ми маємо дешевий і енергоефективний спосіб збільшити загальний діапазон Wi-Fi, хоча й присутня особливість, а саме передача вже прийнятого сигналу. Також окрім Raspberry Pi необхідні два адаптери Wi-Fi, один з яких має підтримувати функціонал точки доступу. Є можливість створення подовжувача Wi-Fi з одним адаптером, проте це потребує прокладення кабелю до нього.

Отже, дане рішення потребує мікрокомп’ютер та два адаптери, живлення від розетки або акумулятора та програмування, є дешевшим гнучким рішенням, що має відкритий код, тож власник розуміє, що саме виконує система.

1.2.4. Створення Wi-Fi повторювача на базі LuaNode32

Серед саме мікроконтролерів одними з найпопулярніших рішень є продукція компанії Espressif Systems, а саме сімейство ESP. У даному репозиторії на Github [6] надано інструкції та скомпільовані файли прошивки для налаштування LuaNode32 (рис. 1.5) у якості повторювача.

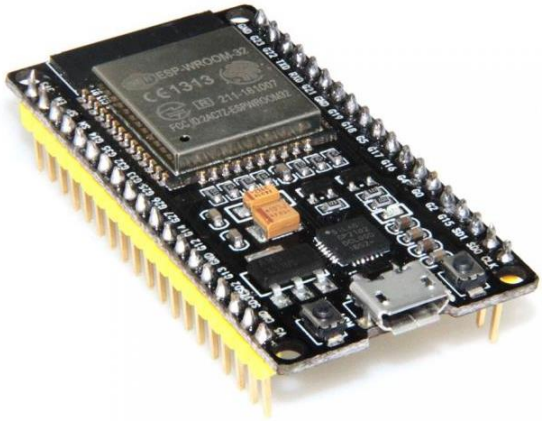


Рисунок 1.5 – Wi-Fi модуль LuaNode32 з ESP-WROOM-32 [7]

LuaNode32 – це модуль розробника, заснований на базі мікромодуля ESP-WROOM-32 компанії Espressif, який виділяється своїми компактними розмірами та підтримує технології Wi-Fi, Bluetooth та Bluetooth Low Energy, а також має усю необхідну периферію для зручної роботи «з коробки» (Flash пам'ять, роз'єм для SD карти, інфрачервоний порт, інтерфейс для підключення ємнісної сенсорної панелі тощо).

Процес запису готового ПЗ є досить простим, проте слід чітко дотримуватись покрокового виконання інструкції: LuaNode32 підключається по USB до персонального комп’ютера з утилітою та скомпільованим кодом, у Flash Download Tool (рис. 1.6) обираються файли, COM-порт та натискається кнопка START, при чому протягом запису на модулі має бути затиснута фізична кнопка BOOT. Після цього мікроконтролер підключається до живлення, а користувач має провести налаштування мережі: кінцевий пристрій з браузером підключається до ESP32 за статичною IP-адресою, і на сторінці запущеного веб-серверу вводяться дані для автентифікації (рис. 1.7).

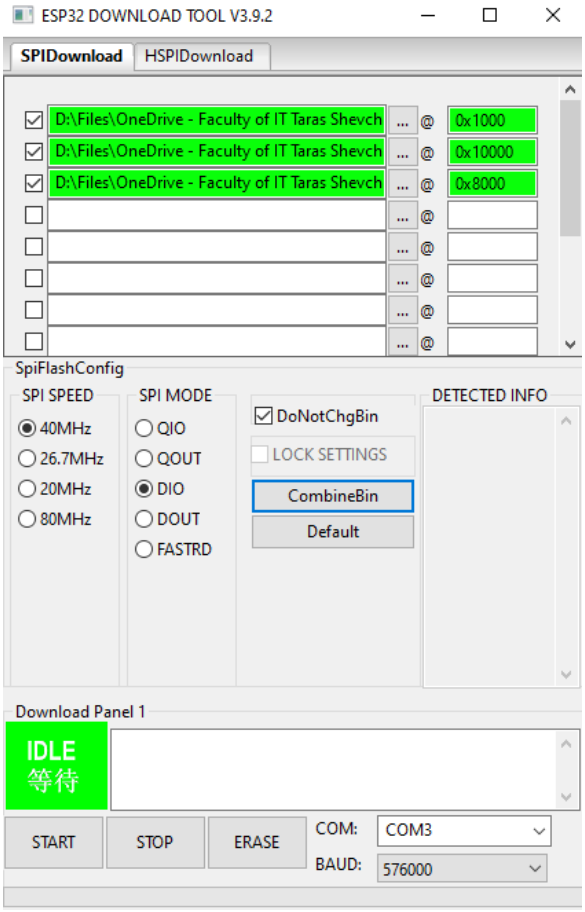


Рисунок 1.6 – Запис прошивки програмою компанії Espressif

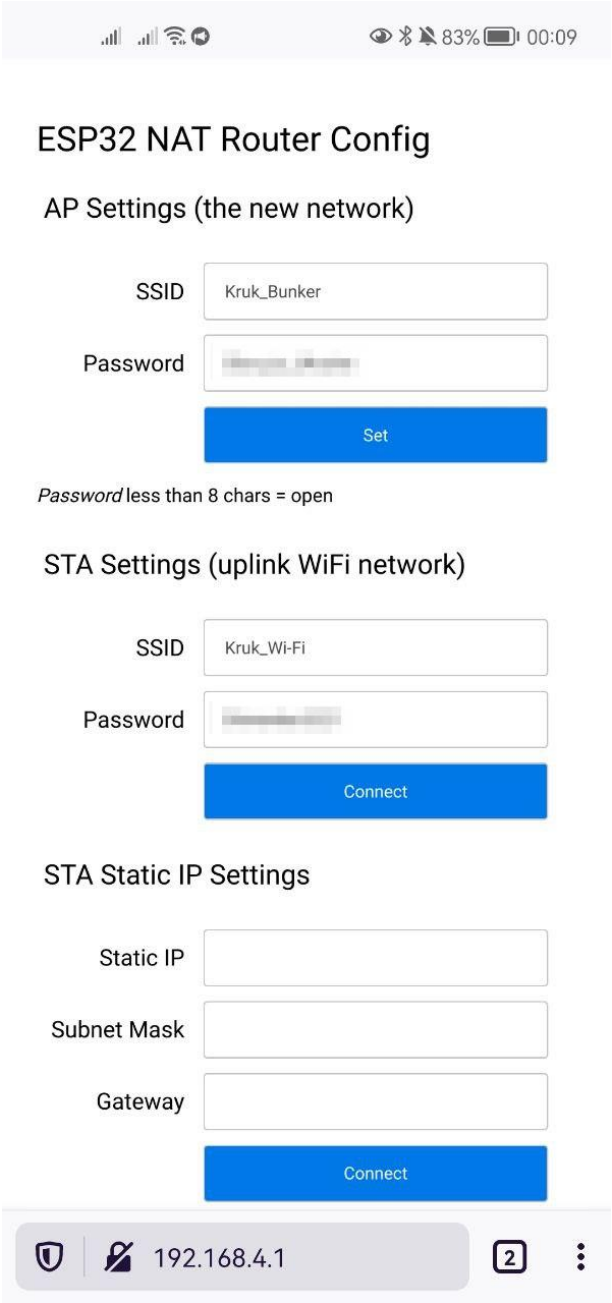


Рисунок 1.7 – Налаштування мережі зі смартфону з браузеру Firefox

В результаті ми маємо бюджетне та енергоефективне рішення щодо збільшення загальний діапазон Wi-Fi з урахуванням того, що сигнал повторюється.

1.3. Обрання рішення

Для простішого порівняння та обрання рішення створено таблицю з основними характеристиками розглянутих варіантів.

Таблиця 1.1 – Порівняння існуючих рішень

Рішення	**Підключення системи**	**Спосіб розширення**	**Відкритість системи**	**Мінімальна вартість системи**
1	2	3	4	5
Додатковий маршрутизатор	Живлення від розетки, вита пара	Подовжувач	Закрита	~550+₴
Готовий повторювач	Живлення від розетки, Wi-Fi	Повторювач	Закрита	~620+₴
Raspberry Pi	Живлення від розетки, вита пара або Wi-Fi	Подовжувач або повторювач	Відкрита	~400+₴
LuaNode32	Живлення від розетки або акумулятора, Wi-Fi	Повторювач	Відкрита	~300+₴

Згідно з низькими вимогами до розширення мережі та досить обмеженим бюджетом, найкращим з розглянутих рішень є використання LuaNode32 на чіпі ESP-WROOM-32, проте через наявність хоч і відкритого, але чужого коду та необхідності розбиратись у його алгоритмі роботи, що є нераціональною витратою великої кількості часу, а також доступність пристроїв для придбання, для подальшої роботи обрано набір зі старшої та дешевшої моделі ESP8266 версії ESP-01S, USB перехідника для програмування і налагодження модулів та резистора на 10 кОм для прошивання – вартість не перевищує 140₴ без урахування блоку живлення.

1.4. Висновок до розділу 1

У даному розділі описано об’єкт дослідження та перспективи розвитку технологій IoT та розумного дому розглянуто існуючі рішення: використання

другого маршрутизатора, готового повторювача або ж програмування мікрокомп’ютера. Кожне з них має свої переваги, недоліки та особливості, проте найбільш підходящим з точки зору співвідношення ціни до задоволення потреб є некомерційне створення повторювача власноруч.

У якості пристрою для забезпечення сигналу у підвальному приміщенні обрано повторювач на базі мікроконтролера ESP8266 версії ESP-01S з наступних причин:

- обмежений бюджет;
- невисокі вимоги до отримуваного сигналу;
- необхідність повного контролю роботи пристрою;
- доступність пристрою для придбання.

2. РОЗРОБКА ПРОЕКТУ МЕРЕЖІ

2.1. Огляд існуючої мережі

До обраного приватного будинку проведено оптоволоконний кабель та підключено тарифний план зі швидкістю Інтернет з’єднання до 100Мб/с. Оптоволоконний кабель підключений до абонентського терміналу на першому поверсі, від якого прокладена вита пара до безпровідного маршрутизатору на другому поверсі. Даний пристрій є єдиним можливим підключенням до мережі Інтернет у всьому будинку. Емпіричним методом досліджено якість сигналу Wi-Fi у кожній окремій кімнаті (рис. 2.1).

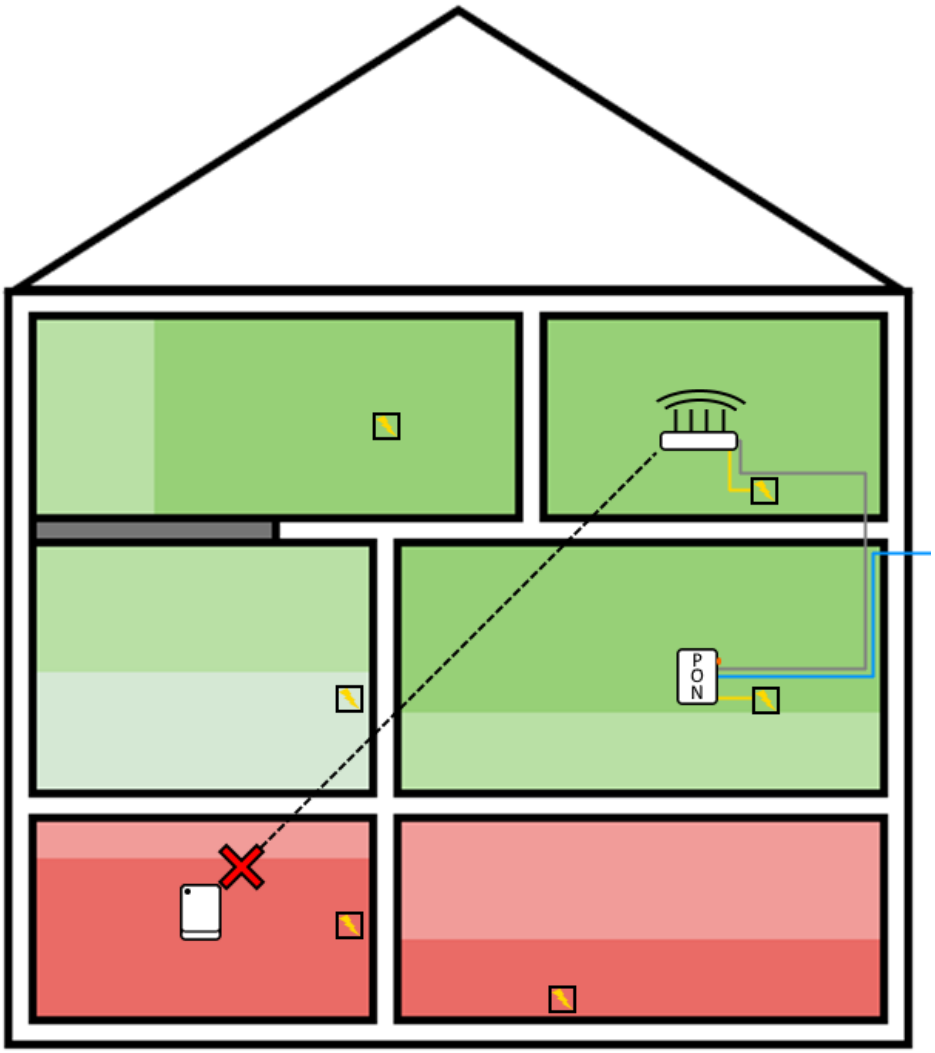


Рисунок 2.1 – Схема якості Wi-Fi сигналу існуючої мережі

На схемі будинку окреслені відносні розміри кімнат, місцерозташування розеток, маршрутизатора, абонентського терміналу та кінцевого пристрою,

з’єднання між ними: синім зображено оптоволоконний кабель, сірим – вита пара, жовтим – дроти живлення, пунктиром – зв’язок по Wi-Fi. Також є сходовий отвір, фоновим кольором показано якість зв’язку, градація якої наведена у таблиці 2.1.

Таблиця 2.1 – Градація якості Wi-Fi сигналу

Колір	**Якість (%)**
1	2
	81-100
	61-80
	41-60
	21-40
	1-20
	0

Варто зазначити, що на схемі відсутня якість 21-40%, а також є різкий перепад від 61+% до 20-% – це зумовлено тим, що сигнал окрім одного бетонного міжповерхового перекриття має пройти ще одне, в якому присутній підігрів підлоги, що постає додатковою перешкодою.

2.2. Опис компонентів мережі

Для кращого розуміння даної мережі необхідно розглянути кожен з компонентів та записати його характеристики.

2.2.1. Компоненти існуючої мережі

Існуюча мережа складається з оптоволоконного кабелю, який проведено від провайдера, абонентського терміналу ONU ZTE ZXHN F601, витої пари та безпровідного маршрутизатора Huawei WS5200. Абонентський термінал та оптоволоконний кабель обрано саме такі через те, що єдиний провайдер, який надає послуги у населеному пункті, в якому розташований обраний приватний будинок, пропонує клієнтам лише їх. Маршрутизатор і виту пару обрано з урахуванням їх характеристик, таких як діапазони частот, інтерфейси та швидкість передачі даних, а також репутації компанії-виробника.

Таблиця 2.2 – Компоненти існуючої мережі

Зображення	Назва	Характеристики
1	2	3
	ONU ZTE ZXHN F601	Інтерфейси: 1 x PON порт SC/UPC, 1 x RJ45 10/100/1000Мб/с. Швидкість передачі даних: 2.5Гб/с (downlink) / 1.25Гб/с (uplink) Підтримувані стандарти: IEEE 802.3ah, 802.3u, 802.3ab. Віртуальні приватні мережі (VLAN): IEEE 802.1Q, 802.1P, 802.1ad. Функції безпеки: обмеження швидкості портів, контроль пропускної спроможності, контроль штормів пакетів (storm control), аутентифікація IEEE 802.1x, підтримка шифрування даних, механізм аутентифікації ONU.
	Huawei WS5200	Діапазони частот: 2.4ГГц + 5ГГц. Інтерфейси: 1 x WAN порт 10/100/1000Мб/с, 3 x 3 LAN порт 10/100/1000Мб/с. Швидкість передачі даних: LAN 1Гб/с, Wi-Fi 1167Мб/с. Стандарти зв’язку: IEEE 802.11b/g/a, 802.11n, 802.11v, Wi-Fi 5 (802.11ac). Функції безпеки: батьківський контроль, гостьовий доступ, захист від DoS-атак, міжмережевий екран SPI, фільтрація MAC-адрес.

2.2.2. ESP8266 версії ESP-01S

Wi-Fi модуль ESP-01S на базі чіпу ESP8266 (рис. 2.2) призначений для вирішення різноманітних задач – наприклад, для IP-камер, мобільної електроніки або ж використання у бездротових сенсорах тощо. «ESP8266 народився, щоб стати основою майбутнього IoT».

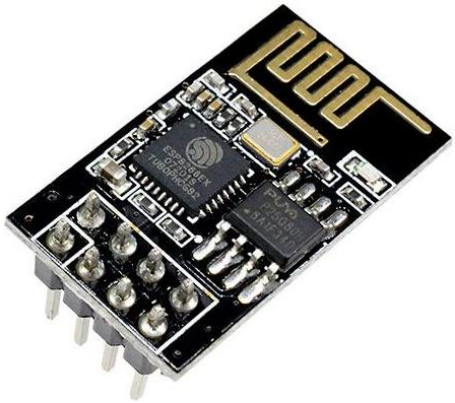


Рисунок 2.2 – Wi-Fi модуль ESP8266 версії ESP-01S [8]

Для коректної роботи даного модуля з програматором необхідно замкнути перемичкою чи резистором 10кОм контакти 8 (VCC) та 4 (CH_PD).

Характеристики та особливості:

- протоколи: 802.11 b/g/n;
- технології: Wi-Fi Direct (P2P), soft-AP;
- частотний діапазон: 2.4 ~ 2.5 ГГц;
- захист: WPA/WPA2;
- вбудований стек TCP/IP;
- максимальний час пробудження та відправки пакетів: 22 мс;
- енергоспоживання: до 10 мкА;
- розміри: 24.5x14 мм.

«USB-перехідник для програмування і налагодження Wi-Fi модулів ESP-01 і ESP-01S» (рис. 2.3) або ж програматор зроблений з мікросхемою CH340G. Пристрій також має усю необхідну периферію для зручної роботи та зручний перемикач між режимами налаштування та програмування.

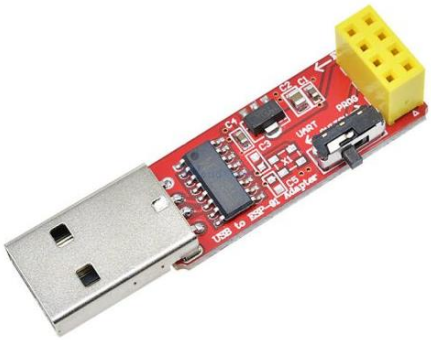


Рисунок 2.3 – USB-перехідник для програмування і налагодження модулів ESP-01 і ESP-01S [9]

Характеристики та особливості:

- робоча напруга: 4.5-5.5В;
- максимальний струм: 300 мА;
- перемикач: режими налагодження та програмування;
- розміри: 1.5 х 5 см.

2.3. Створення проекту розширення мережі

Після формулювання вимог до системи, а саме забезпечення доступу до мережі Інтернет з підвалу та будь-якої кімнати за адекватну ціну, обрання пристрою, наведення інформації про існуючу домашню мережу та створення її схеми, необхідно спроектувати її розширення. Вибір конкретного рішення відбувся з урахуванням причин, наведених раніше у першому розділі.

У розглянутих існуючих прикладах розширення мережі описано LuaNode32 та готове ПЗ, яке виконує функцію роздачі сигналу Wi-Fi за наявності живлення. Отже, з використанням прошитого мікроконтролера з чіпом ESP-WROOM-32 емпіричним методом виявлено, що для забезпечення у підвалі зв’язку прийнятної якості з усіх можливих та перевірених варіантів підключення повторювача до електроживлення єдиним працюючим рішенням є розміщення повторювача над кімнатою.

На (рис. 2.4) зображено якість сигналу Wi-Fi від ESP-01S у кожній окремій кімнаті. До початкової схеми додано повторювач, фоновим кольором показано якість зв’язку, градація якої наведена у таблиці 2.1. Як видно зі схеми, новий

пристрій розміщений на достатній відстані відносно маршрутизатора, щоб мати підключення до його мережі, а також передавати сигнал на кінцевий пристрій у підвалі.

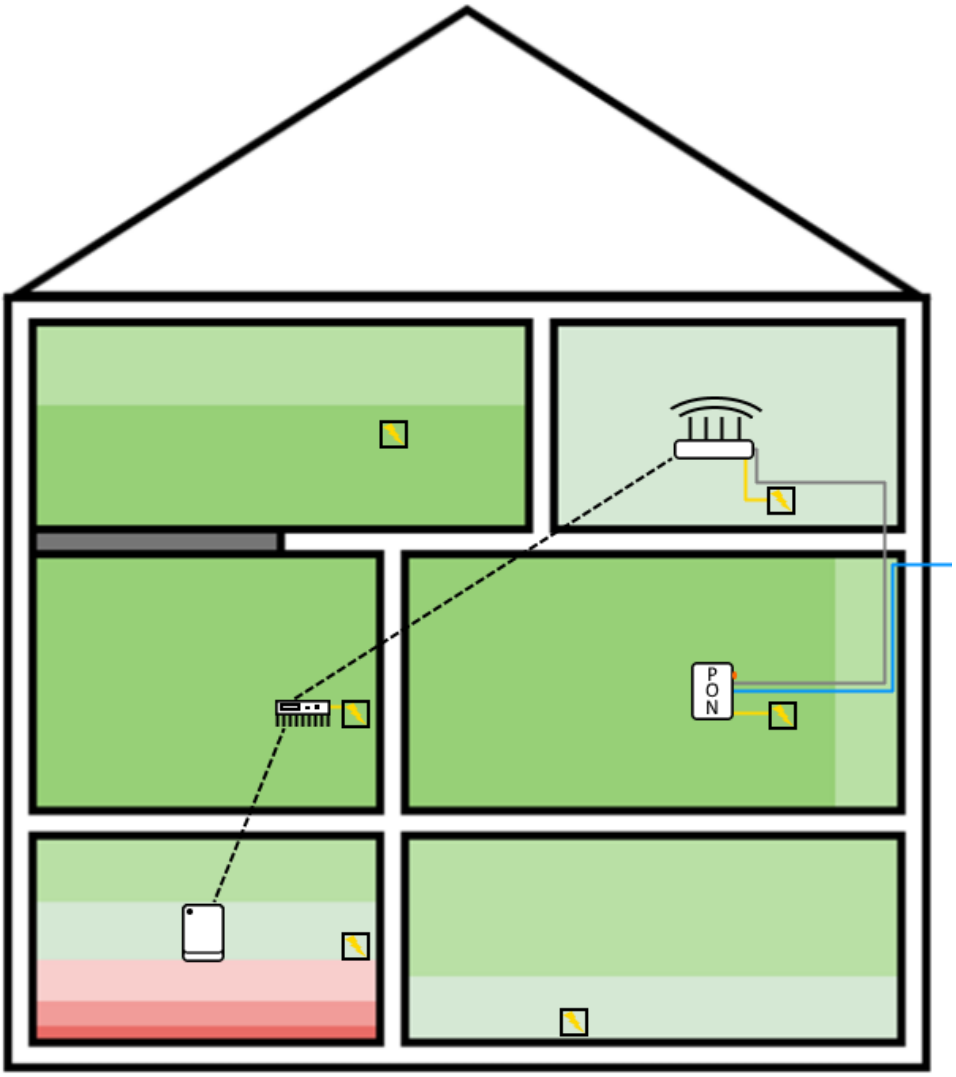


Рисунок 2.4 – Схема якості Wi-Fi сигналу розширення мережі

Варто зазначити, що технічне приміщення має товсті стіни та міжповерхове перекриття з металевими трубами, тому якість сигналу є нестабільним і коливається в середньому від 10 до 70%, проте високий відсоток не гарантує швидкого з’єднання – це залежить від роботи маршрутизатора, наявності сигналу сусідських Wi-Fi мереж та інших шумів.

Для реалізації Wi-Fi повторювача на базі ESP8266 необхідно написати програмний код мовою Arduino (діалект C/C++). Для кращого розуміння правил

та алгоритмів роботи мікроконтролера необхідно побудувати блок-схеми (рис. 2.5-2.7). Для цього використано веб-додаток створення діаграм Draw.io [10].

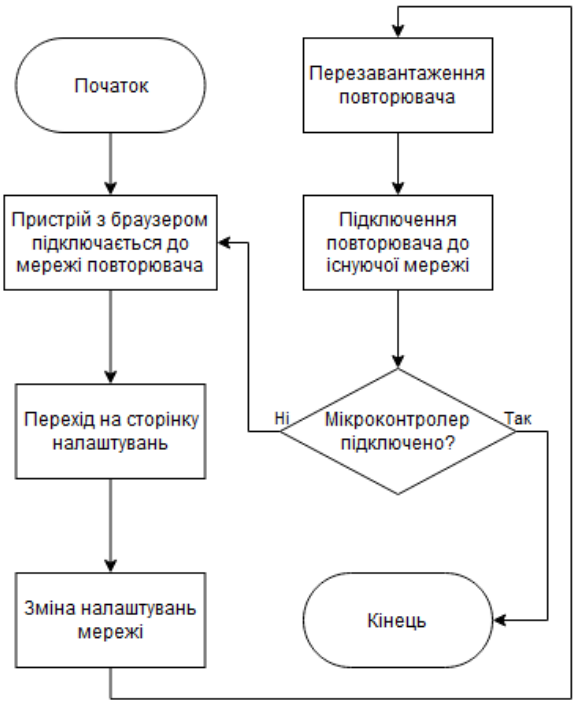


Рисунок 2.5 – Алгоритм налаштування повторювача

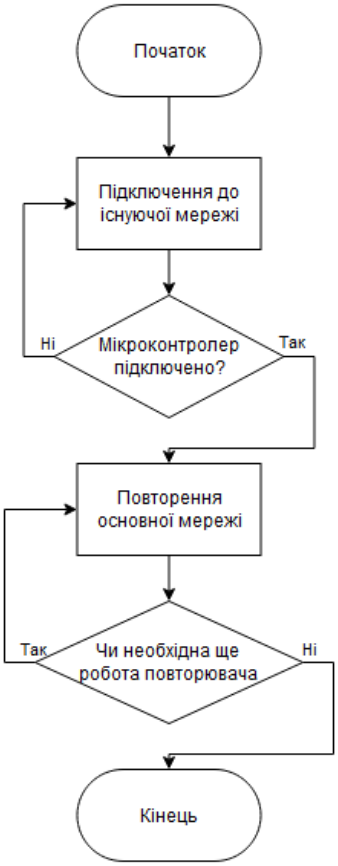


Рисунок 2.6 – Алгоритм роботи налаштованого повторювача

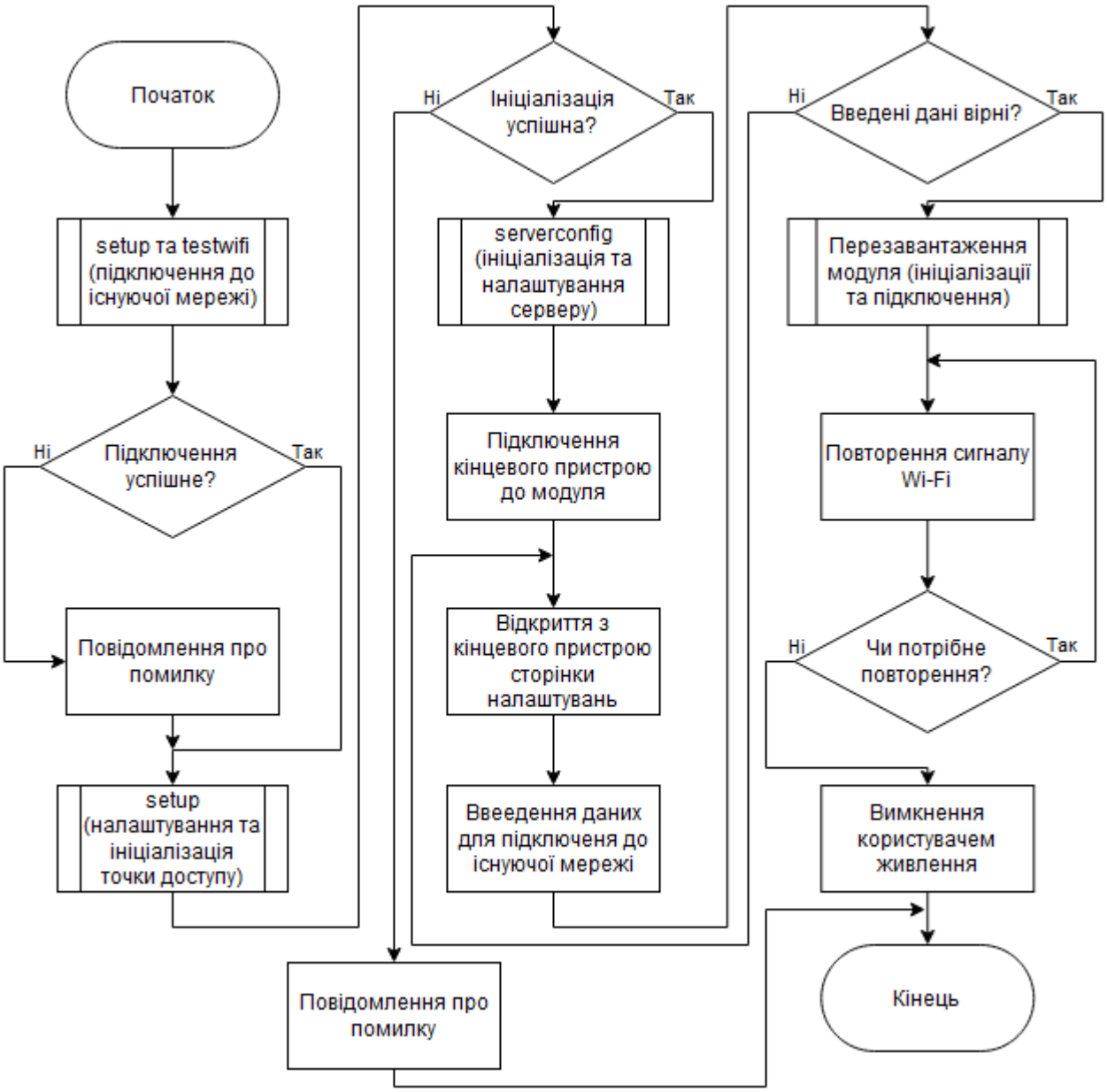


Рисунок 2.7 – Алгоритм роботи програми

2.4. Висновок до розділу 2

У даному розділі описано існуючу домашню мережу, пристрої, з яких вона складається, та модуль ESP8266 версії ESP-01S з програматором, змодельовано схематичний розріз будинку та розташування пристроїв і їх з’єднання між собою та мережею електроживлення, а також спроектовано розширення мережі та побудовано алгоритми налаштування, роботи повторювача та програми.

3. РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ, ЕКСПЕРИМЕНТ ТА АНАЛІЗ ЕФЕКТИВНОСТІ РІШЕННЯ

3.1. Опис алгоритмів функціонування

Після побудови блок-схем з алгоритмами налаштування та роботи мікроконтролера, а також роботи програми, для кращого розуміння структури та визначення необхідних функцій майбутнього коду, варто також детальніше описати процедури алгоритму (рис. 3.1-3.3). Слід зазначити, що у наступних блок-схемах для зручності відображені деякі блоки з основної схеми.

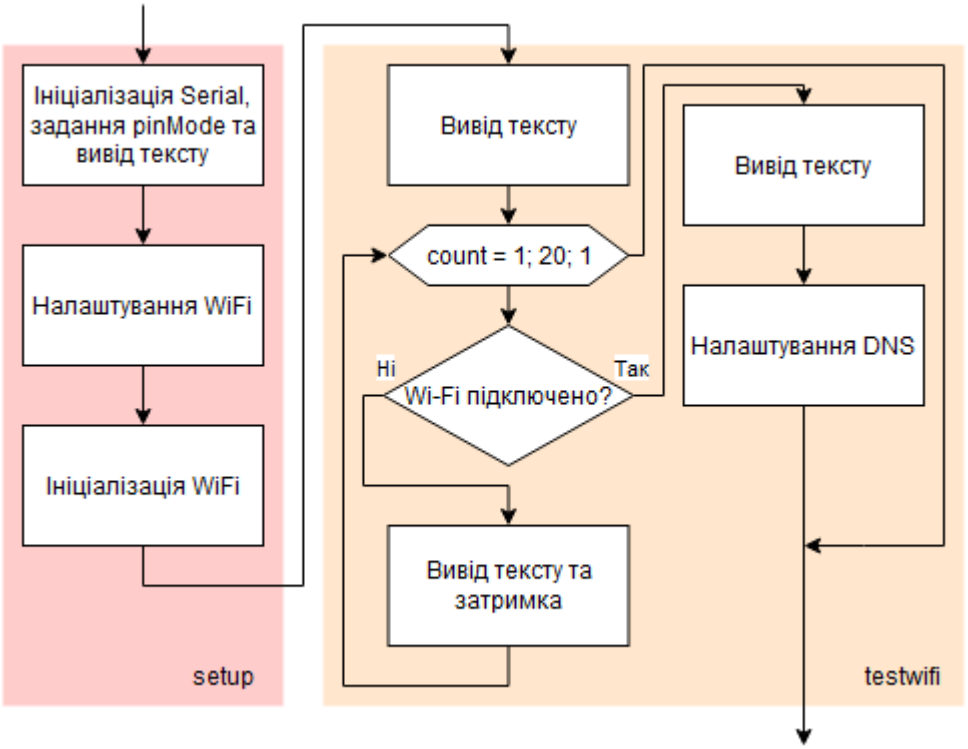


Рисунок 3.1 – Перша процедура – підключення до існуючої мережі

Більшість програм на мові Arduino починаються з підключення бібліотек та оголошення необхідних у подальшому змінних. Після цього виконується функція setup: ініціалізація Serial.begin, задання пінів pinMode та вивід тексту про початок роботи. Далі необхідно налаштувати WiFi та ініціалізувати за допомогою команди WiFi.begin. За виведення інформації про підключення відповідає функція testwifi:

1. відбувається вивід тексту та цикл щодо перевірки статусу підключення до 20 разів – якщо статус підключення негативний, кожного разу виводиться «.» та спрацьовує затримка перед наступною перевіркою;
2. після успішного підключення виводиться текст, відбувається налаштовується DNS для точки доступу та перехід до наступного кроку;

2.1. у разі ж відсутнього підключення протягом усіх 20 перевірок виводиться текст та відбувається перехід до наступного кроку.

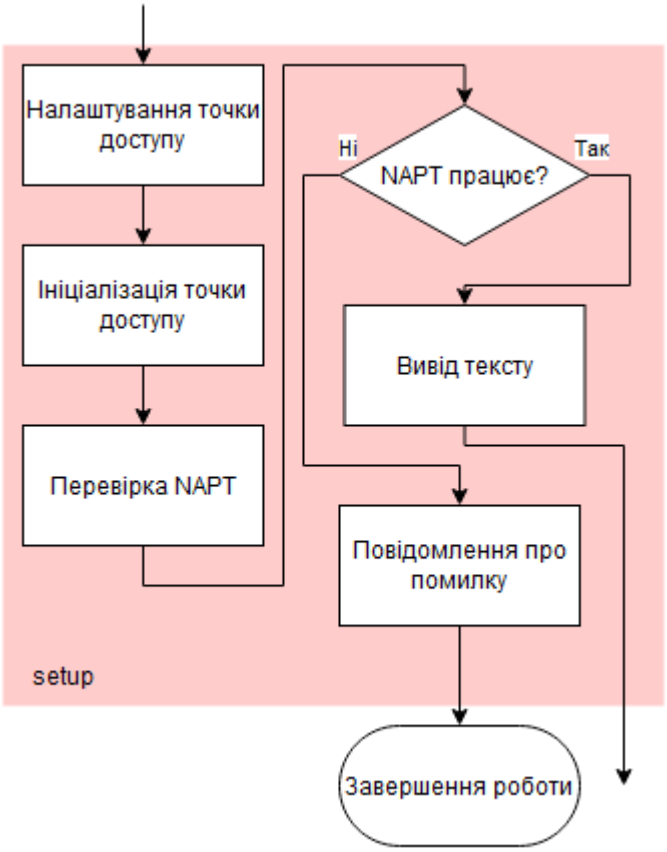


Рисунок 3.2 – Друга процедура – налаштування та ініціалізація точки доступу

Наступним кроком є налаштування точки доступу: задається IP-адреса, мережевий шлюз та маска підмережі. Далі відбувається запуск точки доступу та перевірка працездатності NAPT – перевіряється кожен з портів та, у разі позитивного результату, вивід тексту та перехід до наступного кроку.

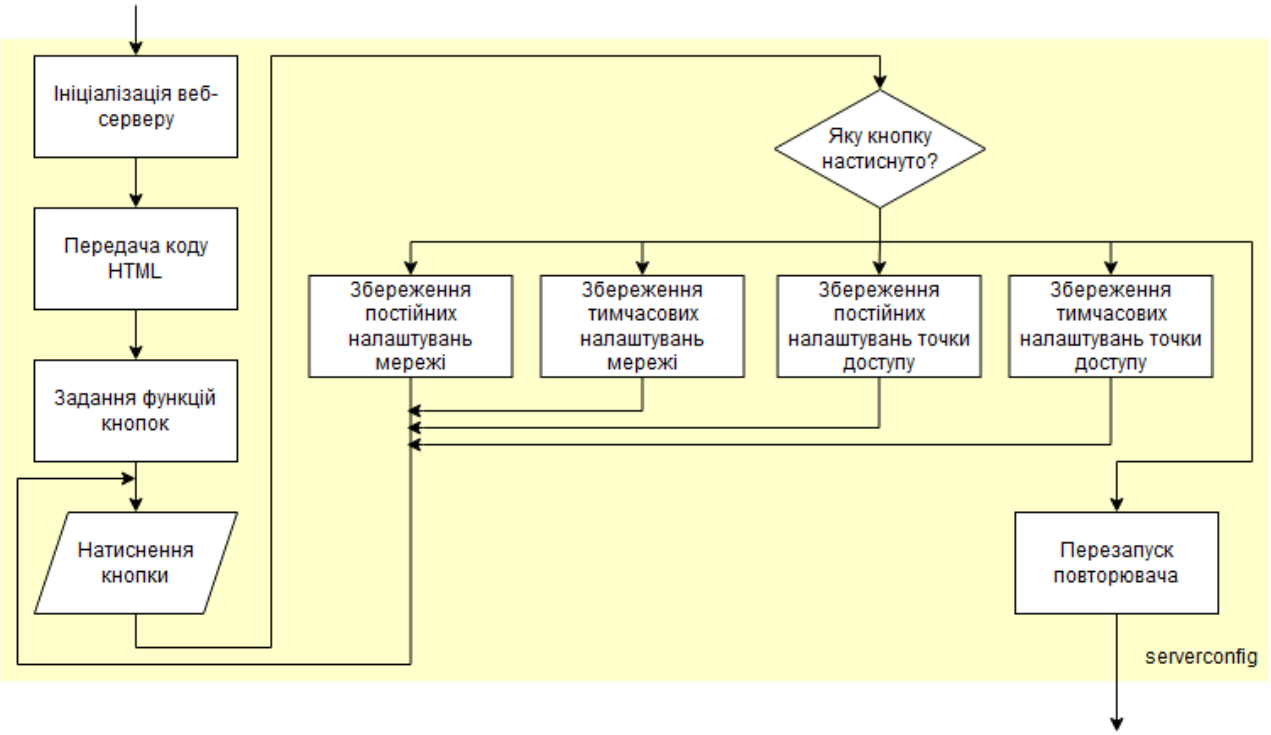


Рисунок 3.3 – Третя процедура – ініціалізація та налаштування серверу

Третя процедура починається з ініціалізації веб-серверу server.begin – він необхідний для створення сторінки налаштування повторювача з інтуїтивним інтерфейсом. Далі на запущений сервер передається увесь код мовою HTML завдяки заданню події server.on та відправки server.send. Після цього необхідно задати функції на натискання кожної кнопки, кожна з яких має унікальну назву, завдяки чому server.on усе фіксує. Для активації нових налаштувань повторювач може потребувати перезапуску, тож відбувається перехід до наступного кроку.

Четверта процедура є звичайним запуском коду з початку та його виконанням до цього ж моменту, хіба що з новими налаштуваннями, тож створення її блок-схеми є нераціональним.

Після успішної ініціалізації та підключення до існуючої мережі повторювач Wi-Fi сигналу виконуватиме свою роботу до відсутності живлення через відключення користувачем чи інші причини та виведення з ладу компонентів повторювача або програматора, що відбудеться не скоро навіть за умов безперервного використання.

3.2. Розробка програмного забезпечення мікроконтролера

Arduino – це не тільки компанія та мова програмування, Arduino є цілою платформою електроніки з відкритим кодом, яка зосереджена на запровадження у життя людей простої та доступної електроніки для покращення його якості. Плати Arduino достатньо просто налаштовуються, здатні зчитувати вхідні дані та видавати вихідні, контактувати між собою тощо – можливості результуючих пристроїв залежать лише від компонентів та фантазії розробника програмного коду. Arduino серед конкурентів виділяється бюджетністю продукції, простотою та кросплатформенністю середовища розробки, відкритістю та розширюваністю програмного та апаратного забезпечення.

Для написання коду використано найновішу версію Arduino IDE 1.8.19 (рис. 3.4).

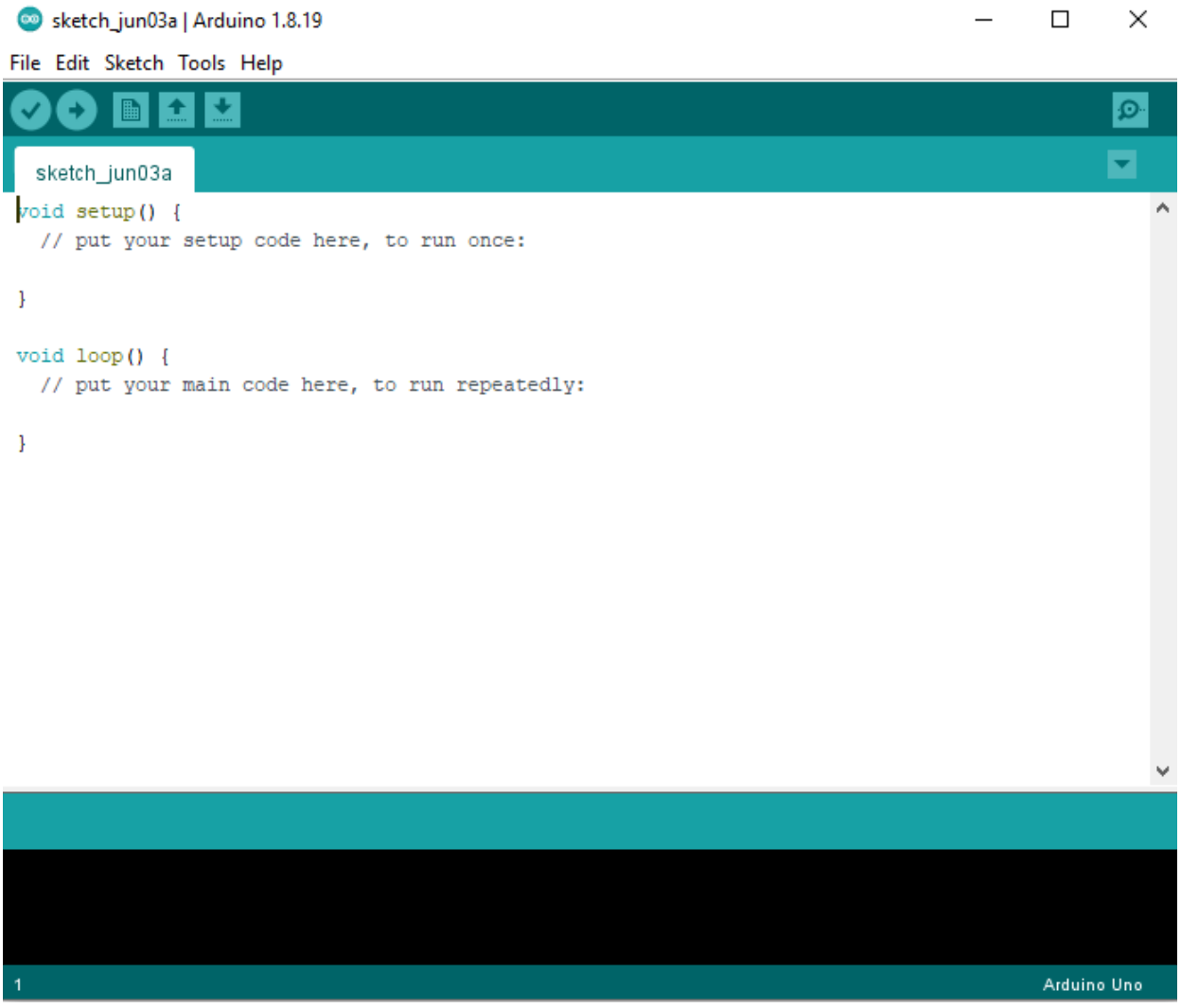


Рисунок 3.4 – Інтерфейс Arduino IDE

Базова комплектація Arduino IDE включає в себе лише стандартні бібліотеки та інструменти з платами AVR (Uno, Nano, Mini тощо), тож перед початком роботи необхідно додати усе необхідне для ESP8266 (рис. 3.5-3.6).

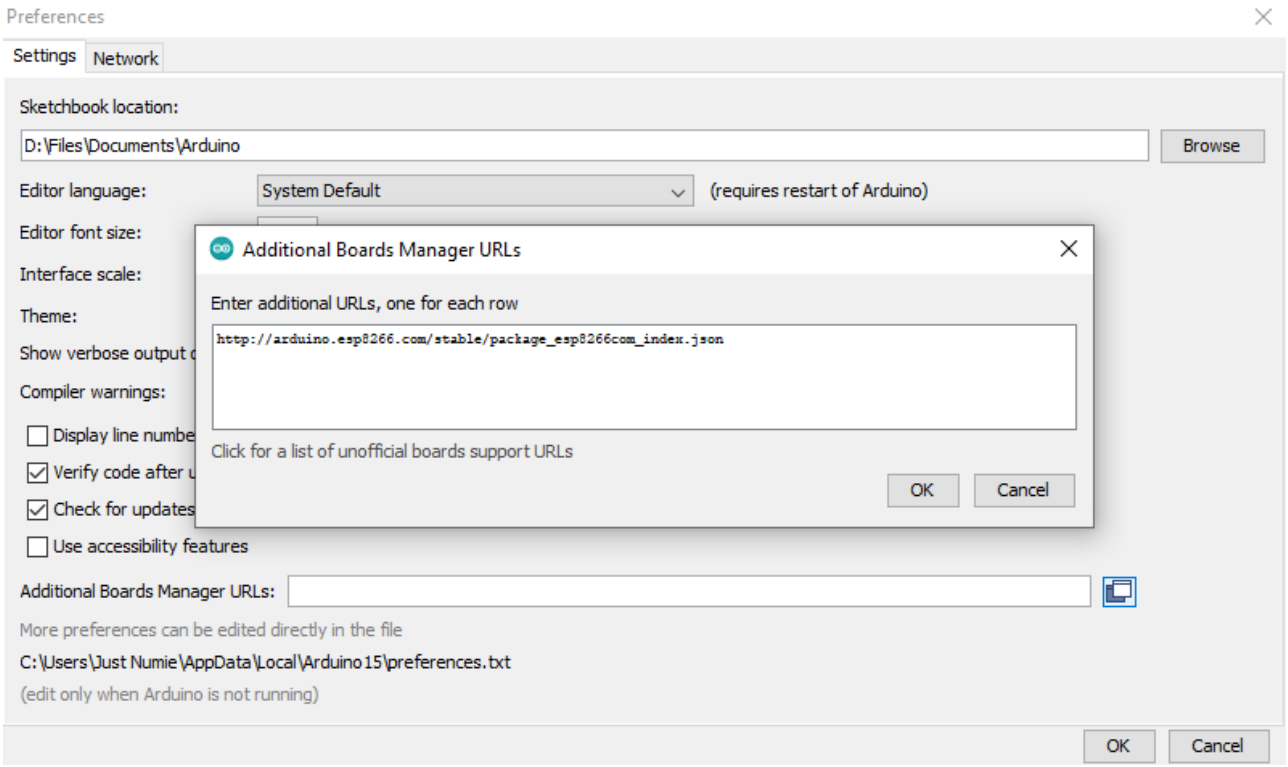


Рисунок 3.5 – Додання у Arduino IDE списку бібліотек для ESP8266

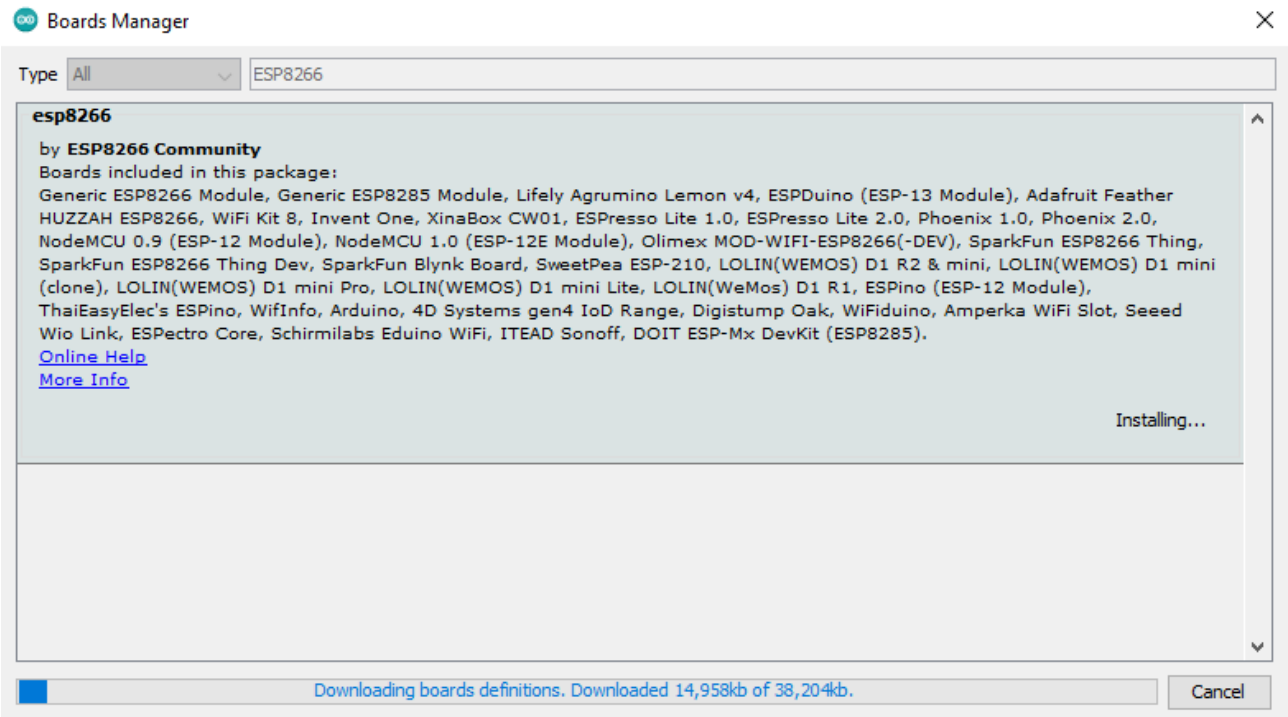


Рисунок 3.6 – Встановлення драйверів та інструментів для ESP8266

Після успішного встановлення середовище для написання коду готове. Спочатку підключено усі необхідні бібліотеки, а саме ESP8266WiFi для команд модулю Wi-Fi, ESP8266WebServer для запуску та роботи веб сервера, завдяки якому реалізовано налаштування повторювача, LwIP – незалежна реалізація набору протоколів TCP/IP для систем з малими обсягами оперативної пам'яті, яка чудово підходить для даного мікроконтролера, та FS для завантаження файлів у енергонезалежну пам'ять. Також оголошено змінні HAVE_NETDUMP для реалізації ведення журналів, NAPT та NAPT_PORT – для трансляції мережевих адрес та портів, ssid та password – для збереження даних мережі основного маршрутизатора. Функція dump при виклику виводить логи (рис. 3.7).

```
#if LWIP_FEATURES && !LWIP_IPV6
#define HAVE_NETDUMP 0

#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <lwip/napt.h>
#include <lwip/dns.h>
#include <LwipDhcpServer.h>
#include <FS.h>

#define NAPT 1000
#define NAPT_PORT 10
const char* ssid = "";
const char* password = "";

#if HAVE_NETDUMP
#include <NetDump.h>

void dump(int netif_idx, const char* data, size_t len, int out, int success) {
  (void)success;
  Serial.print(out ? F("out ") : F(" in "));
  Serial.printf("%d ", netif_idx);
}
#endif
```

Рисунок 3.7 – Код підключення бібліотек, підключення змінних та функція dump

Функція setup у програмах мовою Arduino зазвичай прописується у кінці. У даному випадку ініціалізовано порт Serial зі швидкістю передачі даних 115200 біт/с, режим роботи 2 піна, а після затримки у 2000мс виводиться текст,

вкионується перевірка на наявність файлу bootstrap.css та викликається функція dump; відбувається налаштування роботи модуля Wi-Fi, а саме режим роботи та ssid з password для підключення до мережі, викликається функція testwifi, яка описана далі, а також задаються IP-адреса, шлюз за замовчуванням і маска підмережі для точки доступу самого повторювача; перевіряються порти та працездатність трансляції мережевих адрес та портів NAPT і викликається функція serverconfig для налаштування веб-серверу (рис. 3.8-3.9).

```
void setup() {
  Serial.begin(115200);
  pinMode(2, OUTPUT);
  delay(2000);
  Serial.printf("\n\nWi-Fi повторювач ESP8266 версії ESP-01S\n");
  Serial.printf("Автор: Крук Назар. Слава Україні!\n");
  Serial.printf("Доступна пам'ять при запуску: %d\n", ESP.getFreeHeap());

  //css check
  if(!SPIFFS.begin()) {
    Serial.println("Помилка монтування файлової системи!");
    return;
  }
  File file = SPIFFS.open("/bootstrap.css", "r");
  if(!file) {
    Serial.println("Помилка відкриття bootstrap.css!");
    return;
  }
  else {
    file.close();
    Serial.println("Файл bootstrap.css наявний!");
  }

#if HAVE_NETDUMP
  phy_capture = dump;
#endif

  //set radio type to N
  WiFi.setPhyMode(WIFI_PHY_MODE_11N);
  WiFi.mode(WIFI_AP_STA);
  WiFi.persistent(false);
  //use stored credentials to connect to network
  WiFi.begin(ssid, password);
  testwifi();
  //set IP Address, Gateway and Subnet
  WiFi.softAPConfig(
    IPAddress(192, 168, 4, 1),
    IPAddress(192, 168, 4, 1),
    IPAddress(255, 255, 255, 0));
  //use stored credentials to create AP
  WiFi.softAP(WiFi.softAPSSID(), WiFi.softAPPSK());
```

Рисунок 3.8 – Функція setup

```
  //NAPT-test
  Serial.printf("Доступна пам'ять до NAPT-тесту: %d\n", ESP.getFreeHeap());
  err_t ret = ip_napt_init(NAPT, NAPT_PORT);
  Serial.printf("ip_napt_init(%d,%d): ret=%d (OK=%d)\n", NAPT, NAPT_PORT, (int)ret, (int)ERR_OK);
  if (ret == ERR_OK) {
    ret = ip_napt_enable_no(SOFTAP_IF, 1);
    Serial.printf("ip_napt_enable_no(SOFTAP_IF): ret=%d (OK=%d)\n", (int)ret, (int)ERR_OK);
    if (ret == ERR_OK) {
      Serial.printf("\nWi-Fi мережа '%s' з паролем '%s' та IP-адресою '%s' налаштована та запущена!\n",
    }
  }
  Serial.printf("Доступна пам'ять після NAPT-тесту: %d\n", ESP.getFreeHeap());
  if (ret != ERR_OK) {
    Serial.printf("NAPT-тест невдалий\n");
  }

  serverconfig();
}
```

Рисунок 3.9 – Продовження функції setup

Слід зазначити, що у разі неуспішного NAPT-тесту повторювач сигналу фізично не зможе працювати, тож для виведення відповідної інформації написано альтернативну функцію setup (рис. 3.10).

```
#else

void setup() {
  Serial.begin(115200);
  Serial.printf("\n\nNAPT не підтримується на даній конфігурації\n");
}

#endif
```

Рисунок 3.10 – Альтернативна функція setup

Функція testwifi, що викликається при ініціалізації, виконує наступні задачі: після затримки у 1000мс виводить текст про початок тестування підключення до основної мережі за заданими параметрами (якщо нові дані ще не було надано користувачем, очевидно, підключення не відбудеться), керує станом світлодіоду завдяки команді digitalWrite та до 20 разів запускає цикл, у якому й відбувається спроба підключення, після чого повертається значення true – у разі невдачі друкує символ та після затримки у 1000мс збільшує лічильник на одиницю та повертає значення false (рис. 3.11).

```
bool testwifi() {
  delay(1000);
  Serial.printf("\nТестування підключення до '%s'\n", WiFi.SSID().c_str());
  int count = 0;
  digitalWrite(2, LOW);
  while (count < 20) {
    //connection testing
    if (WiFi.status() == WL_CONNECTED) {
      Serial.printf("\nWi-Fi підключено! \nSTA: %s (dns: %s / %s)\n\n",
                    WiFi.localIP().toString().c_str(),
                    WiFi.dnsIP(0).toString().c_str(),
                    WiFi.dnsIP(1).toString().c_str());

      //give DNS servers to AP side
      dhcpSoftAP.dhcps_set_dns(0, WiFi.dnsIP(0));
      dhcpSoftAP.dhcps_set_dns(1, WiFi.dnsIP(1));
      digitalWrite(2, HIGH);
      return true;
    }
    Serial.print(".");
    delay(1000);
    count++;
  }
  Serial.printf("\nНеможливо з'єднатись з мережею Wi-Fi! Підєднайтесь до точки
  return false;
}
```

Рисунок 3.11 – Функція testwifi

Для реалізації веб-серверу з налаштуваннями повторювача спочатку необхідно створити веб-сторінку, проте середа розробки Arduino IDE націлена лише на роботу з мовою Arduino, тож для написання коду мовою розмітки HTML обрано PyCharm, а саме Community Edition 2021.3.1 (рис. 3.12), яка є безкоштовною та підтримує «з коробки» роботу з файлами конфігурації, Python, HTML та CSS.

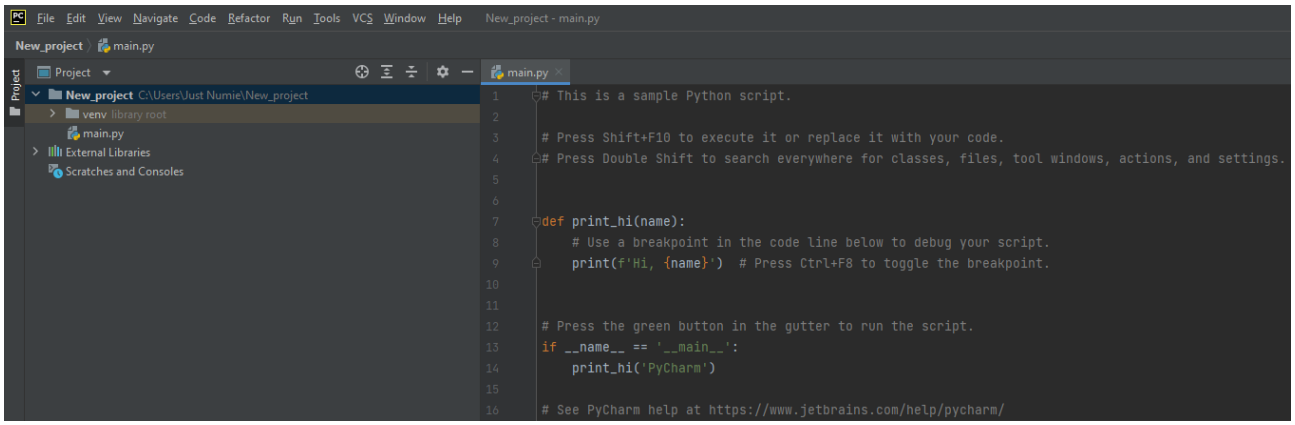


Рисунок 3.12 – Інтерфейс PyCharm

Початок коду містить основну інформацію про сторінку, таку як тип документа, мову, ширину вмісту, та назву сторінки (рис. 3.13).

```
<!DOCTYPE html>
<html lang='uk'>
<meta name='viewport' content='width=device-width, initial-scale=1.0' charset=utf-8>
<head>
    <title>Wi-Fi повторювач ESP8266 версії ESP-01S</title>
    <link href='bootstrap.css' rel='stylesheet' type='text/css'>
    <style...>
</head>
<body>
    <div class='container-fluid'>
        <h1>Wi-Fi повторювач ESP8266 версії ESP-01S</h1>
        <h5>Автор: Крук Назар. Слава Україні!</h5><br>
        <h2>Налаштування Wi-Fi мережі</h2>
        <form method='post'>
            <label class='form-label'>SSID:</label><br>
            <input name='stassid' placeholder=' WiFi.SSID()' length=32 class='form-control'><br>
            <label class='form-label'>Пароль:</label><br>
            <input type='password' placeholder='********' name='stapsk' minlength=8 maxlength=63 class='form-control'>
            <div class='form-text'>Мінімальна довжина - 8 символів! Інакше залиште порожнім.</div><br class='br-plus'>
            <button type='submit' formaction='stasettings' class='btn btn-primary'>Зберегти</button>
            <button type='submit' formaction='tempstasettings' class='btn btn-outline-primary'>Зберегти до перезапуску</button>
        </form>
        <h2>Налаштування точки доступу</h2>
        <form method='post'>
            <label class='form-label'>SSID:</label><br>
            <input name='apssid' placeholder=' WiFi.softAPSSID() ' length=32 class='form-control'><br>
            <label class='form-label'>Пароль:</label><br>
            <input type='password' placeholder=' WiFi.softAPPSK() ' name='appsk' minlength=8 maxlength=63 class='form-control'>
            <div class='form-text'>Мінімальна довжина - 8 символів! Інакше залиште порожнім.</div><br class='br-plus'>
            <button type='submit' formaction='apsettings' class='btn btn-primary'>Зберегти</button>
            <button type='submit' formaction='tempapsettings' class='btn btn-outline-primary'>Зберегти до перезапуску</button>
        </form>
        <h2>Додатково</h2><br class='br-plus'>
        <form method='post'>
            <button type='submit' formaction='reboot' class='btn btn-warning'>Перезавантаження повторювача</button><br>
            <button type='submit' formaction='stastandard' class='btn btn-outline-warning'>Стандартні налаштування Wi-Fi мережі</button><br>
            <button type='submit' formaction='apstandard' class='btn btn-outline-warning'>Стандартні налаштування точки доступу</button>
        </form>
    </div>
</body>
</html>
```

Рисунок 3.13 – Код веб-сторінки для веб-сервера ESP8266

У основній частині коду (тег body) прописані наступні елементи:

- заголовки форм;
- теги форм form з методом передачі даних;
- заголовки полів для введення даних;
- поля введення ssid для основної мережі та точки доступу з відповідними заповнювачами, які беруться з флешпам’яті ESP8266;
- поля введення password для основної мережі та точки доступу з заповнювачами у вигляді символів «*», що приховують дійсні символи;

- кнопки постійного та тимчасового збереження налаштувань для основної мережі та точки доступу з командами, які перехоплює веб-сервер та виконує відповідні функції;
- кнопки перезавантаження повторювача та встановлення стандартних налаштувань з відповідними командами.

Даний код в цілому є зручним для читання, проте для передачі потребує внесення у змінну content, тож інтерпретований у наступному вигляді (рис. 3.14).

```
//server with settings
ESP8266WebServer server(80);
String content;
void serverconfig() {
  server.begin();

  //page
  server.on("/", []() {
    //base
    content = "<!DOCTYPE html><html lang='uk'><meta name='viewport' content='width=device-width, initial-scale=1.0' ch
    content += "<head><title>Wi-Fi повторювач ESP8266 версії ESP-01S</title><link href='data/bootstrap.css' rel='style
    content += "<style> .form-control{max-width: 400px;} br{display: block; content: '';margin-top: 5px;} .br-plus{dis
    content += "<body><div class='container-fluid'><h1>Wi-Fi повторювач ESP8266 версії ESP-01S</h1><h5>Автор: Крук Наз
    //network settings
    content += "<form method='post'><label class='form-label'>SSID:</label><br><input name='stassid' placeholder='";
    content += WiFi.SSID();
    content += "' length=32 class='form-control'><br><label class='form-label'>Пароль:</label><br><input type='passwor
    content += "<div class='form-text'>Мінімальна довжина - 8 символів! Інакше залиште порожнім.</div><br class='br-pl
    content += "<button type='submit' formaction='tempstasettings' class='btn btn-outline-primary'>Зберегти до перезап
    //AP settings
    content += "<form method='post'><label class='form-label'>SSID:</label><br><input name='apssid' placeholder='";
    content += WiFi.softAPSSID();
    content += "' length=32 class='form-control'><br><label class='form-label'>Пароль:</label><br><input type='passwor
    content += WiFi.softAPPSK();
    content += "' name='appsk' minlength=8 maxlength=63 class='form-control'><div class='form-text'>Мінімальна довжина
    content += "<button type='submit' formaction='apsettings' class='btn btn-primary'>Зберегти</button>";
    content += "<button type='submit' formaction='tempapsettings' class='btn btn-outline-primary'>Зберегти до перезапу
    //features
    content += "<form method='post'><button type='submit' formaction='reboot' class='btn btn-warning'>Перезавантаження
    content += "<button type='submit' formaction='stastandard' class='btn btn-outline-warning'>Стандартні налаштування
    content += "<button type='submit' formaction='apstandard' class='btn btn-outline-warning'>Стандартні налаштування
    server.send(200, "text/html", content);
  });
```

Рисунок 3.14 – Код запуску веб-сервера та інтерпретована веб-сторінка

Для коректної роботи веб-серверу, а саме відображення чинних ssid, перезапуску пристрою та передачі нових налаштувань з відповідних полів для вводу шляхом натиснення кнопок, необхідно прописати окремі функції (рис. 3.15). Реалізований функціонал включає у себе наступне:

- функція onNotFound, яка виконується у випадку, якщо веб-сервер не зміг знайти шукану сторінку – у даній реалізації це фактично неможливо, проте така «заглушка» є необхідною;
- функція on з параметром /statesetting, яка виконується після натискання кнопки постійного збереження налаштувань для

підключення до існуючої мережі – зчитує введений текст у відповідних полях, присвоює їх у змінні та у разі наявності більше одного символу у змінній ssid виводить текст та запускає Wi-Fi з внесеними параметрами і функцію testwifi;

- функція on з параметром /tempstatesetting, яка виконується після натискання кнопки тимчасового збереження налаштувань для підключення до існуючої мережі – зчитує введений текст у відповідних полях, присвоює їх у змінні та у разі наявності більше одного символу у змінній ssid виводить текст та запускає Wi-Fi з внесеними параметрами і функцію testwifi;
- функція on з параметром /apsetting, яка виконується після натискання кнопки постійного збереження налаштувань для точки доступу – зчитує введений текст у відповідних полях, присвоює їх у змінні та у разі наявності більше одного символу у змінній ssid виводить текст та запускає Wi-Fi з внесеними параметрами і функцію testwifi;
- функція on з параметром /tempapsetting, яка виконується після натискання кнопки тимчасового збереження налаштувань для точки доступу – зчитує введений текст у відповідних полях, присвоює їх у змінні та у разі наявності більше одного символу у змінній ssid виводить текст та запускає Wi-Fi з внесеними параметрами і функцію testwifi; виводить текст та запускає Wi-Fi з внесеними параметрами і функцію testwifi;
- функція on з параметром /reboot, яка виконується після натискання кнопки перезавантаження – виконує повне перезавантаження пристрою;
- функції on з параметрами /stastandard та /apstandard, які виконується після натискання кнопки стандартних налаштувань – встановлює стандартні налаштування мережі та точки доступу відповідно.

```
server.onNotFound([]() {
  server.send(404, "text/plain", "Тож... Яким чином ми тут опинились?");
});

//network settings
server.on("/stasettings", []() {
  String stassid = server.arg("stassid");
  String stapsk = server.arg("stapsk");
  if (stassid.length() > 0) {
    server.send(200, "text/plain", "Налаштування отримано!");
    Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n", stassid.c_str(), stapsk.c_str());
    WiFi.persistent(true);
    WiFi.begin(stassid, stapsk);
    testwifi();
  }
});
server.on("/tempstasettings", []() {
  String stassid = server.arg("stassid");
  String stapsk = server.arg("stapsk");
  if (stassid.length() > 0) {
    server.send(200, "text/plain", "Налаштування отримано!");
    Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n", stassid.c_str(), stapsk.c_str());
    WiFi.persistent(false);
    WiFi.begin(stassid, stapsk);
    testwifi();
  }
});

//AP settings
server.on("/apsettings", []() {
  String apssid = server.arg("apssid");
  String appsk = server.arg("appsk");
  if (apssid.length() > 0) {
    server.send(200, "text/plain", "Налаштування отримано!");
    Serial.printf("\n\nНалаштування точки доступу \nSSID: %s \nПароль: %s \n", apssid.c_str(), appsk.c_str());
    WiFi.persistent(true);
    WiFi.softAP(apssid, appsk);
  }
});

server.on("/tempapsettings", []() {
  String apssid = server.arg("apssid");
  String appsk = server.arg("appsk");
  if (apssid.length() > 0) {
    server.send(200, "text/plain", "Налаштування отримано!");
    Serial.printf("\n\nНалаштування точки доступу (до перезавантаження) \nSSID: %s \nПароль: %s \n", apssid.c_str(), appsk.c_str());
    WiFi.persistent(false);
    WiFi.softAP(apssid, appsk);
  }
});

//features
server.on("/reboot", []() {
  server.send(200, "text/plain", "Перезавантаження повторювача...");
  Serial.printf("\n\nПерезавантаження повторювача...");
  delay(5000);
  ESP.reset();
});
server.on("/stastandard", []() {
  String stassid = "";
  String stapsk = "";
  server.send(200, "text/plain", "Встановлення стандартних налаштувань Wi-Fi мережі");
  Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n", stassid.c_str(), stapsk.c_str());
  WiFi.persistent(true);
  WiFi.begin(stassid, stapsk);
  testwifi();
});
server.on("/apstandard", []() {
  String apssid = "ESP_62F21C";
  String appsk = "";
  server.send(200, "text/plain", "Встановлення стандартних налаштувань точки доступу");
  Serial.printf("\n\nНалаштування точки доступу \nSSID: %s \nПароль: %s \n", apssid.c_str(), appsk.c_str());
  WiFi.persistent(true);
  WiFi.softAP(apssid, appsk);
});
```

Рисунок 3.15 – Функції веб-серверу

Після функції setup у програмах мовою Arduino зазвичай прописується функція loop – основний код, який виконується до логічного завершення або ж

вимкнення мікроконтролера. У даному випадку має постійно працювати сервер – це реалізовано за допомогою команди server.handleClient; для того, щоб після підключення повторювача до живлення не від комп'ютера, який виводить текст у відповідну консоль, користувач міг чітко розуміти, чи є підключення до основної мережі, реалізовано блимання світлодіоду кожні 1000мс та постійне світіння у разі відсутності сигналу (рис. 3.16).

```
void loop() {
  server.handleClient();
  if (WiFi.status() != WL_CONNECTED)
    digitalWrite(2, LOW);
    delay(1000);
    digitalWrite(2, HIGH);
    delay(1000);
  }
  else {
    digitalWrite(2, HIGH);
  }
}
```

Рисунок 3.16 – Функція loop

Далі готову програму необхідно верифікувати та записати на ESP-01S, а також налаштувати пристрій та ввести в експлуатацію.

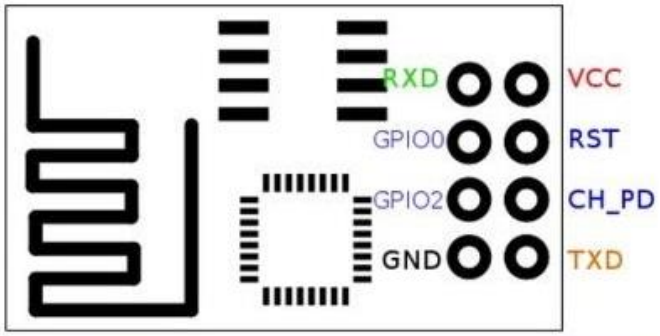
3.3. Налаштування мікроконтролера

Після створення проекту мережі та написання програмного коду необхідно налаштувати мікроконтролер перед введенням в експлуатацію. Першим кроком є верифікація програми. Arduino IDE має вбудований функціонал компіляції коду, яка запускається натисненням кнопки «Verify». Компілятор виводить у консоль інформацію щодо розмірів виконуваних сегментів, таких як інструкція для флеш-пам'яті, код у ній, в оперативній пам'яті, ініціалізовані змінні, константи та нульові змінні; у разі успішної верифікації також буде виведено максимальний об'єм пам'яті пристрою, розмір скетчу у байтах та відсоткове відношення до максимуму, об'єм динамічної пам'яті пристрою, використання її глобальними змінними та відсоткове відношення до максимуму, а також залишок для локальних змінних (рис. 3.17).

```
Executable segment sizes:
ICACHE : 32768           - flash instruction cache
IROM   : 305952          - code in flash         (default or ICACHE_FLASH_ATTR)
IRAM   : 27385   / 32768 - code in IRAM          (IRAM_ATTR, ISRs...)
DATA   : 1504  )         - initialized variables (global, static) in RAM/HEAP
RODATA : 5380  ) / 81920 - constants             (global, static) in RAM/HEAP
BSS    : 26048 )         - zeroed variables      (global, static) in RAM/HEAP
Sketch uses 340221 bytes (68%) of program storage space. Maximum is 499696 bytes.
Global variables use 32932 bytes (40%) of dynamic memory, leaving 48988 bytes for local variables. Maximum is 81920 bytes.
```

Рисунок 3.17 – Процес верифікації коду

На сторінці товару ESP8266 ESP-01S [8] вказана інформація щодо необхідності замикання контактів для коректної роботи з програматором. Для цього необхідні резистор на 10кОм та обладнання для пайки. Слід зазначити, що припаювання не є обов'язковим – достатньо лише замикання, тож за відсутності відповідної апаратури ніжки резистора можна обкрутити навколо контактів (рис. 3.19). Для обрання необхідних контактів використано схему плати та таблицю контактів ESP8266 ESP-01S (рис. 3.18).



Label	Signal
VCC	3.3V (3.6V max) supply voltage
GND	Ground
TXD	Transmit Data (3.3V level)
RXD	Receive Data (3.3V level!)
CH_PD	Chip Power down: (LOW = power down active)
GPIO0	General Purpose I / O 0
GPIO2	General Purpose I / O 2
RST	Reset (reset = LOW active)

Рисунок 3.18 – Схема плати та таблиця контактів ESP8266 ESP-01S [8]

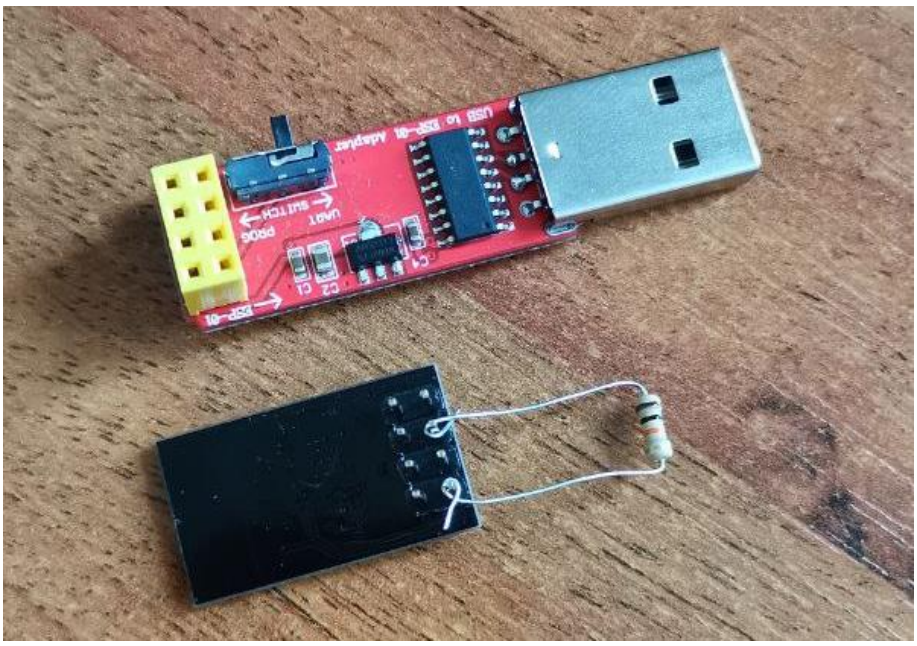


Рисунок 3.19 – Замикання резистором 10кОм контактів 8 (VCC) та 4 (CH_PD)

Після застосування резистора та об’єднання мікроконтролера з програматором, необхідно підключити останній у USB-порт комп’ютера з запущеною Arduino IDE – відповідний драйвер або буде встановлено автоматично (наприклад, ОС Windows 10 визначає підключені пристрої та шукає підходящі драйвери), або треба встановити вручну, завантаживши з офіційного або перевіреного сайту ретейлера [9]. У разі успішного підключення у списку доступних COM-портів з’явиться новий, який необхідно вибрати.

Обране середовище розробки також має функціонал запису скетчів та файлів на мікроконтролери тощо, а саме скрипт esptool v3.0, написаний мовою Python - він запускається натисненням кнопки «Upload». У консоль виводиться інформація про обраний порт та спробу підключення – у разі успіху відображається наявний чіп, функціонал, частота кристалу та MAC-адреса, після чого відбувається сам процес запису, що супроводжується коментарями про виконувану роботу, таку як стан заглушки, налаштування та розмір флеш-пам’яті, адреси байтів запису та відсоток прогресу і результат (кількість записаних байтів, стиснутих, адреса початку, витрачений час та ефективність запису у кб/с) (рис. 3.20).

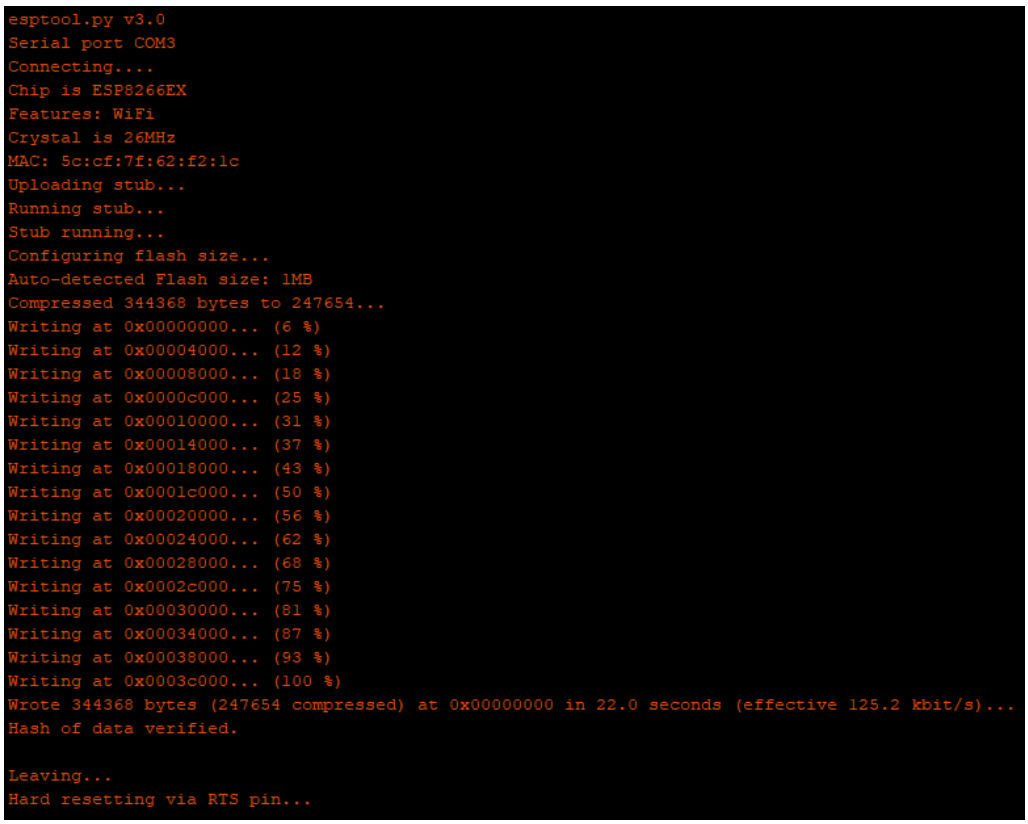


Рисунок 3.20 – Процес запису коду на модуль

Процес запису файлу bootstrap.css у флеш-пам’ять модуля (рис. 3.21) для подальшого його підключення веб-сервером відбувається схожим чином – як видно зі скріншоту, виводиться та ж сама інформація, що і при записі скетчу, окрім інформації про файл та флеш-пам’ять від бібліотеки файлової системи FS (SPIFFS).

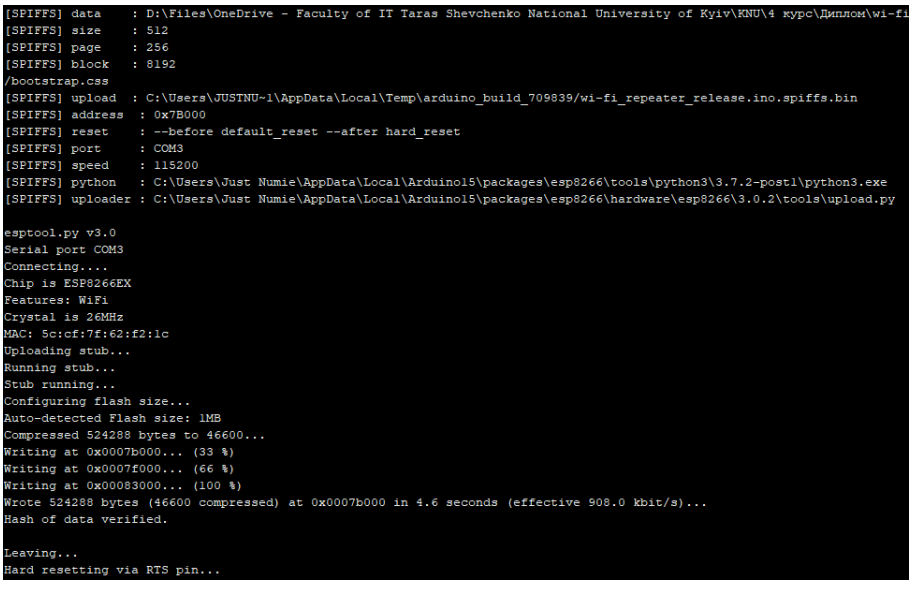


Рисунок 3.21 – Процес запису файлу bootstrap.css на модуль

Після запису коду та файлу скрипт перезавантажує пристрій, проте для функціонування ESP-01S за підключення до програматора необхідно перевести другий з початкового режиму PROG, що використовується для запису коду на модуль тощо, у режим UART (універсального асинхронного приймача/передавача) шляхом зміни положення фізичного перемикача (рис. 3.22).

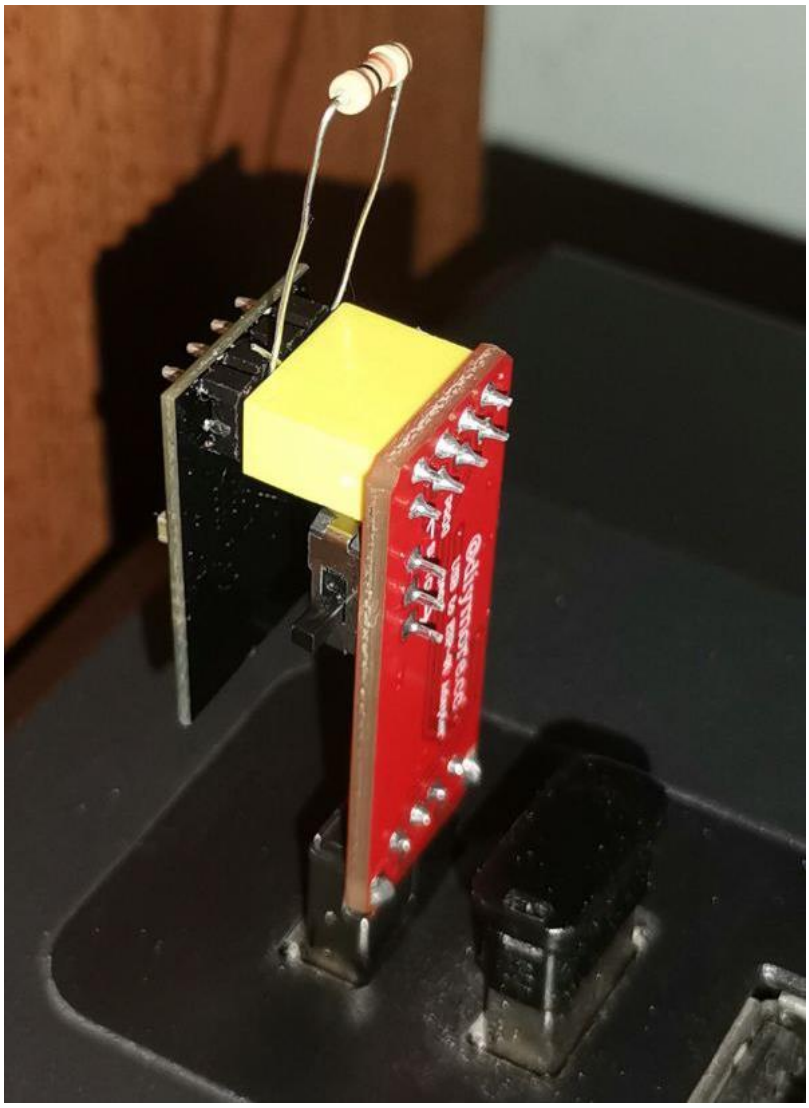


Рисунок 3.22 – Підключений програматор з мікроконтролером у режимі UART

У даному режимі програматор є, так би мовити, мостом між комп’ютером та мікроконтролером, а також забезпечую для другого необхідне живлення.

Написана програма запускається одразу ж, як підключена до мережі електропостачання, та виконує задані функції. Для моніторингу необхідно відкрити вікно консолі (рис. 3.23) порту COM3, до якого підключений

програматор. Спочатку повторювач намагається під’єднатись до неіснуючої мережі, так як при першому запуску не має у флеш-пам’яті жодних налаштувань. Після невдалої спроби точці доступу задається назва з порядковим номером чіпу, відбувається NAPT-тестування та запускається точка доступу зі заздалегідь заданою IP-адресою.

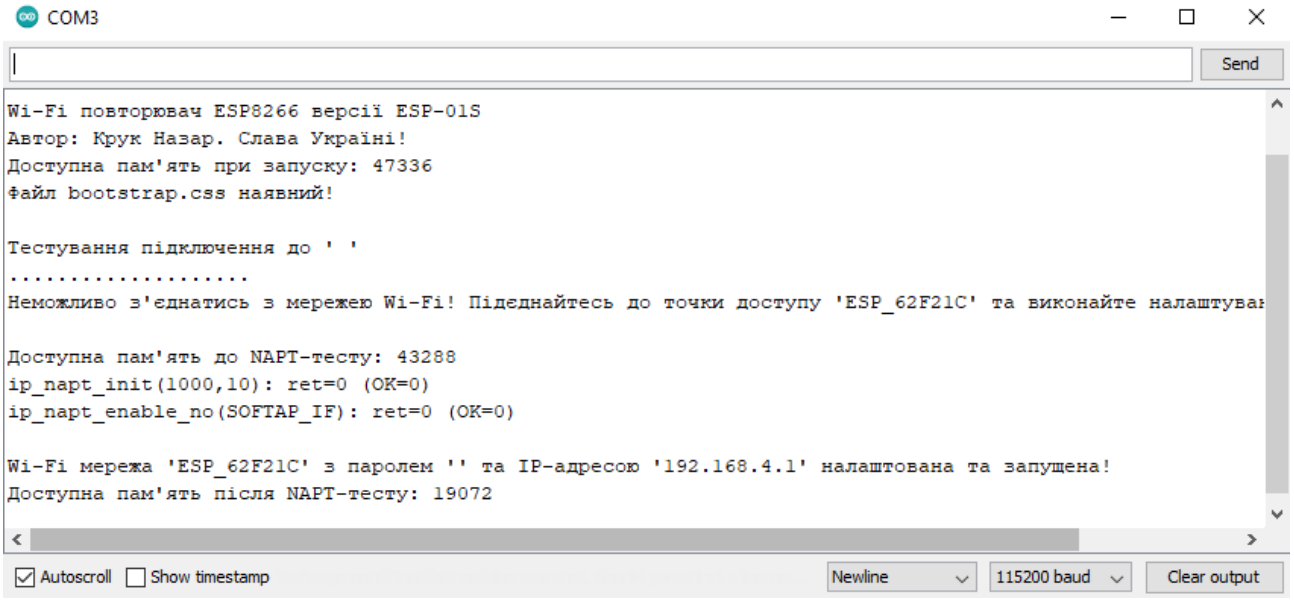


Рисунок 3.23 – Вивід інформації у консоль

Для внесення нових налаштувань необхідно з будь-якого кінцевого пристрою з браузером під’єднатись до точки доступу (у даному випадку стандартними налаштуваннями є ssid «ESP_62F21C» та відсутній пароль), в адресний рядок ввести раніше отриману IP-адресу, після чого користувачу буде доступна веб сторінка з формами (рис. 3.24).

Веб-сторінка налаштувань складається з форми налаштувань існуючої мережі, налаштувань точки доступу та окремо винесеної кнопки перезавантаження пристрою. Для подальшого використання повторювача введено дані Wi-Fi мережі обраного приватного будинку - відбулось вдале підключення, після чого введено дані і для точки доступу. Дане підключення є тестовим, тож обраний спосіб збереження є «Зберегти до перезапуску» - відповідна інформація виведена у консоль порту COM3 (рис 3.25).

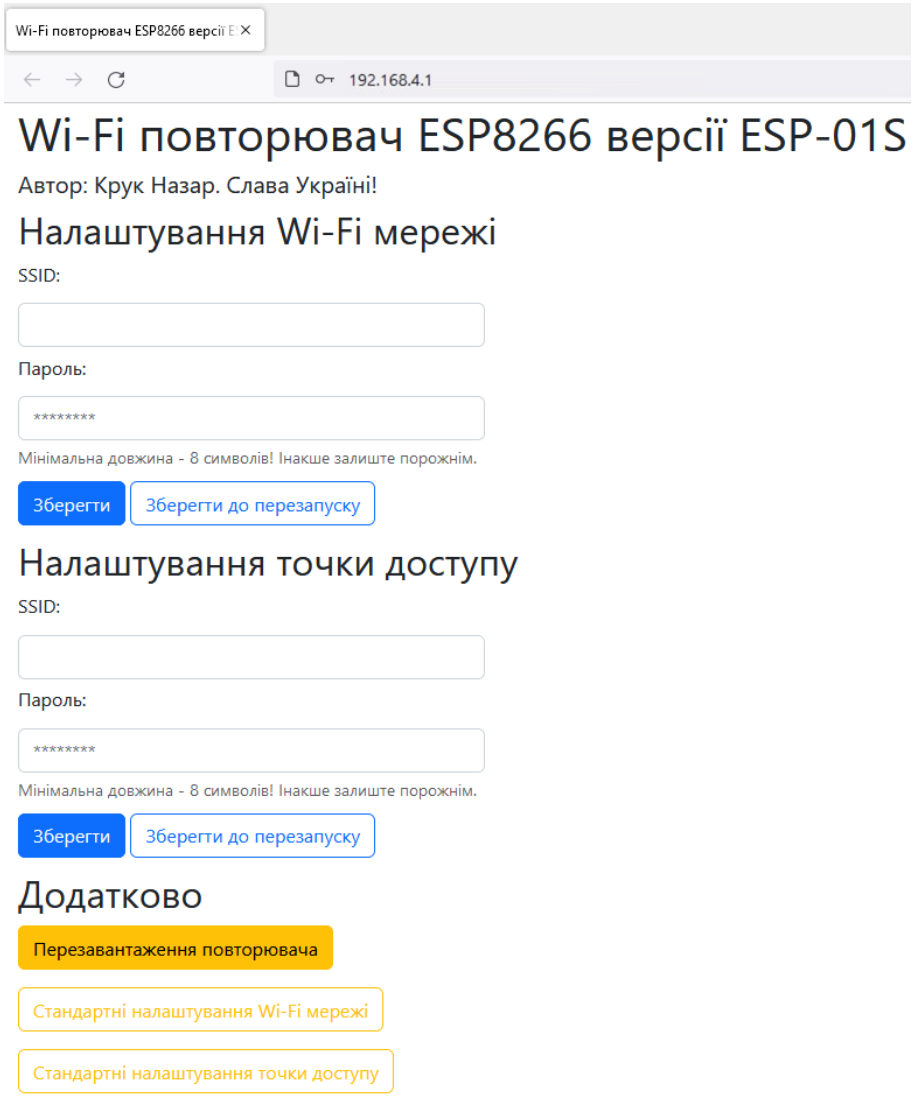


Рисунок 3.24 – Налаштування повторювача з персонального комп’ютера

```
Спроба підключення до 'Kruk_Wi-Fi' з паролем '[redacted]'

Тестування підключення до 'Kruk_Wi-Fi'
....
Wi-Fi підключено!
STA: [redacted] (dns: [redacted] / (IP unset))



Налаштування точки доступу
SSID: Kruk_Bunker
Пароль: Glory_to_Ukraine
```

Рисунок 3.25 – Вивід інформації щодо налаштувань у консолі

Тестове налаштування мікроконтролера виконано успішно – його можна відключити від персонального комп’ютера.

3.4. Введення мікроконтролера в експлуатацію

За проектом розширення мережі, який розроблено у другому розділі, повторювач необхідно розмістити та підключити для живлення у кімнаті над підвалом – так як програматор вміє контролювати вхідну напругу, у якості підключення використано стандартний зарядний блок для телефонів компанії Samsung з силою струму 1А та напругою 5В, чого більш ніж достатньо для забезпечення працездатності мікроконтролера (рис. 3.26).

Рисунок 3.26 – Підключення повторювача до мережі живлення

З отриманням живлення повторювач одразу починає виконувати записану програму – про результат підключення до основної мережі та активації точки доступу можна дізнатись завдяки сигналам світлодіоду. Для налаштування мережі було обрано цільовий кінцевий пристрій – смартфон. Серед наявних мереж окрім основної та сусідських зафіксовано безпарольну «ESP_62F21C». При переході за тією ж IP-адресою, що і у тестовому налаштуванні, відкривається веб-сторінка з налаштуваннями (рис. 3.27). Для подальшої

експлуатації введено ті ж самі налаштування, але обрано спосіб збереження «Зберегти».

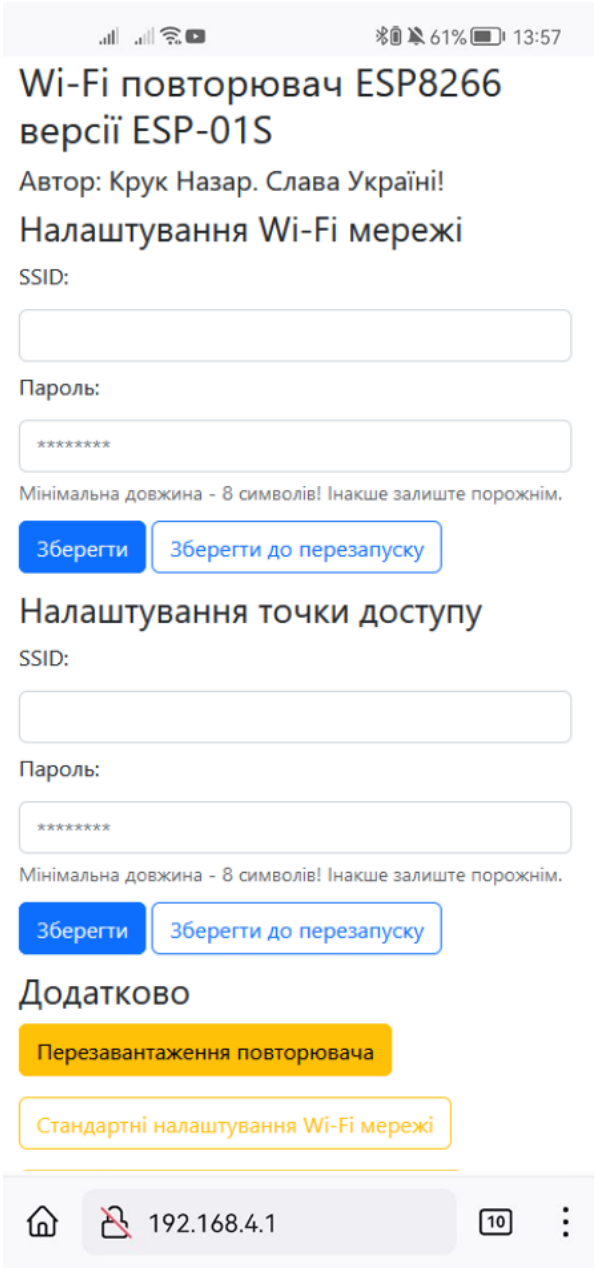
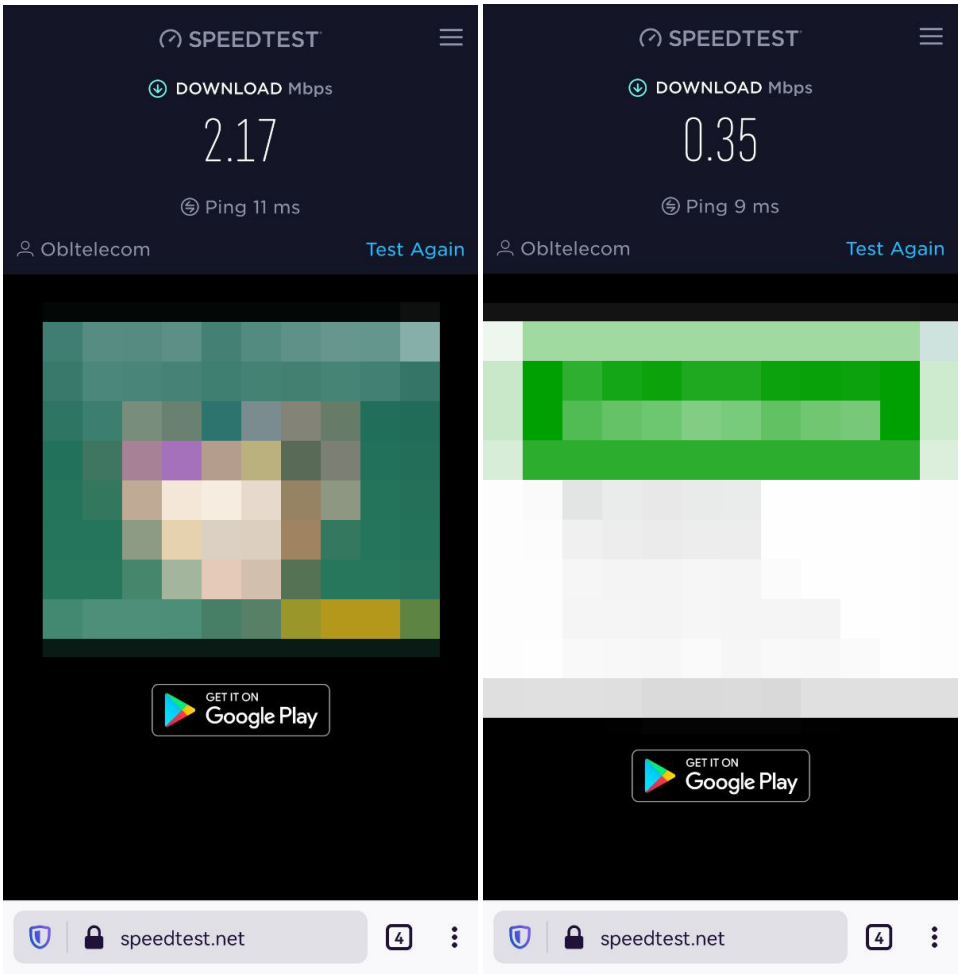


Рисунок 3.27 – Налаштування повторювача з кінцевого пристрою

Після успішного налаштування необхідно підключитись до повторювача, як до нової мережі (у даному випадку налаштуваннями є ssid «Kruk_Bunker» та пароль «Glory_to_Ukraine»), а також бажано провести заміри швидкості Інтернет-зв’язку. Для виконання цієї задачі розроблена велика кількість інструментів, з яких було обрано один з найвідоміших – веб-додаток SPEEDTEST (рис. 3.28).



а) б)

Рисунок 3.28 – Заміри швидкості Інтернету у веб-додатку SPEEDTEST [11] а) біля повторювача б) у підвалі

Як видно з результатів, повторювач працює належним чином та задовольняє вимоги до розширення мережі, проте для кращого розуміння можливостей даної системи варто провести декілька замірів та їх аналізів.

3.5. Збір даних та аналіз ефективності рішення

Протягом часу експлуатації усього проведено 30 замірів швидкості Інтернет-зв’язку та затримок пакетів – скріншоти кожного з них наведені у Додатку Б, для зручності побудовано діаграми (рис 3.29 та рис. 3.31), обидві відсортовані по швидкості біля повторювача від більшої до найменшої.

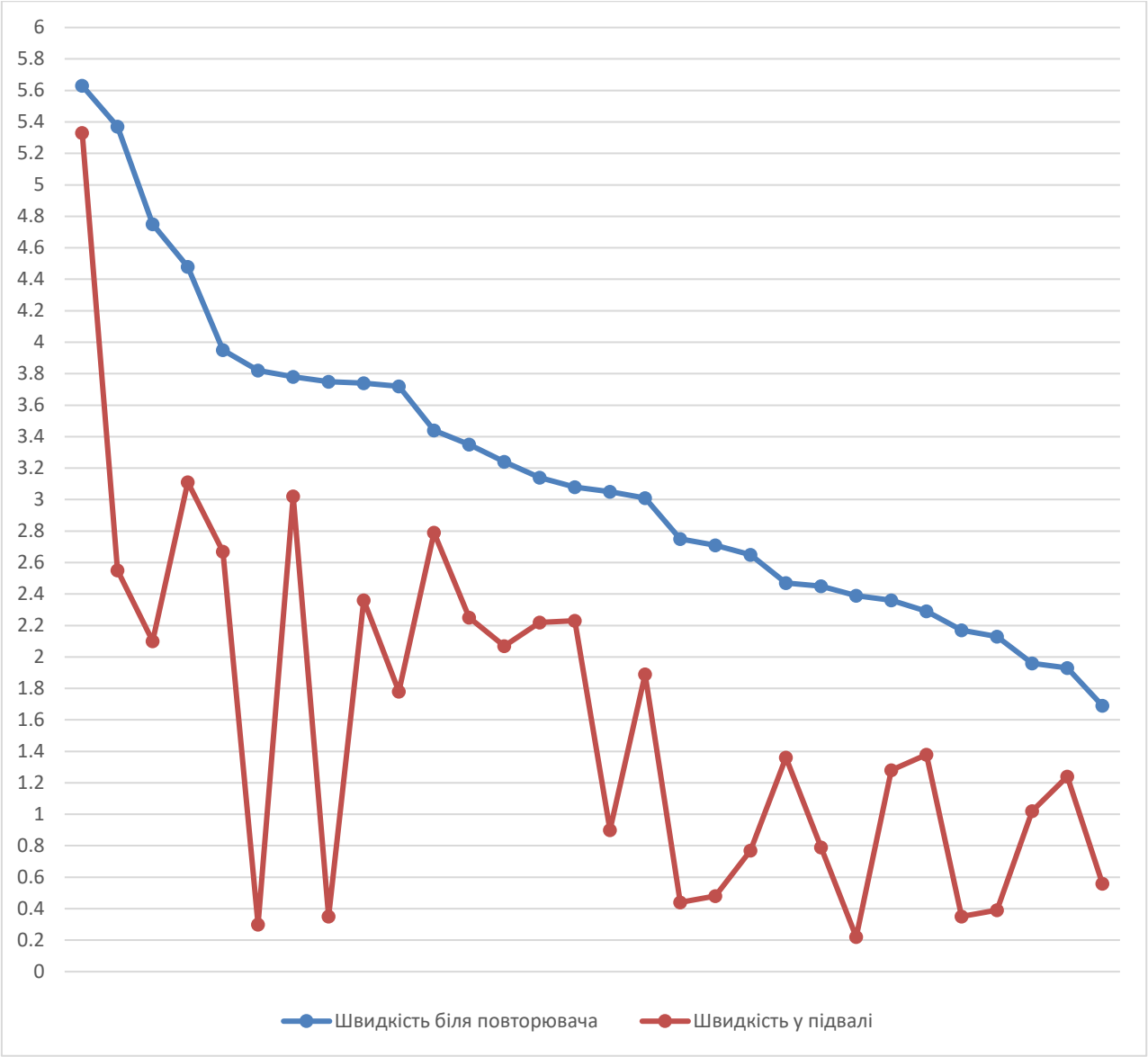


Рисунок 3.29 – Лінійна діаграма замірів швидкості Інтернет-зв’язку (Мб/с)

При огляді першої діаграми помітно, що швидкість у підвалі хоч і залежить від швидкості біля самого повторювача, проте знаходиться під впливом інших факторів, таких як широке міжповерхове перекриття та його матеріал, швидкість сигналу від маршрутизатора, шум (сусідні Wi-Fi мережі та інше), розташування кінцевого пристрою тощо. Найбільша зафіксована швидкість у підвалі 5.33Мб/с, а найменша – 0.22Мб/с. Незважаючи на часто низьку якість зв’язку (11 з 30 замірів швидкості у підвалі показали результат менше 1Мб/с), він є прийнятним для онлайн-листування та отримання інформації щодо стану повітряної тривоги. Більш того, найбільшої швидкості, за даними з довідки веб-сервісу YouTube [12] (рис. 3.30), достатньо для комфортного перегляду потокового відео у роздільній

HD 1080p- у більшості ж випадків (18 з 30 замірів швидкості у підвалі показали результат більше 1.1Мб/с) можна переглядати відео з роздільною здатністю SD 480p.

Роздільна здатність відео	Рекомендована стабільна швидкість
4K	20 Мбіт/с
HD 1080p	5 Мбіт/с
HD 720p	2,5 Мбіт/с
SD 480p	1,1 Мбіт/с
SD 360p	0,7 Мбіт/с

Рисунок 3.30 – Вимоги веб-сервісу YouTube до швидкості зв’язку для комфортного перегляду відео

При огляді другої діаграми помітна відсутність залежності так званої затримки (ping) у підвалі від затримки біля повторювача. Хорошою затримкою, яка спостерігається у переважній більшості випадків, для Wi-Fi сигналу є менше 40мс (29 з 30 замірів), і лише один замір з затримкою у 104мс є прийнятним – у діапазоні 40-110мс.

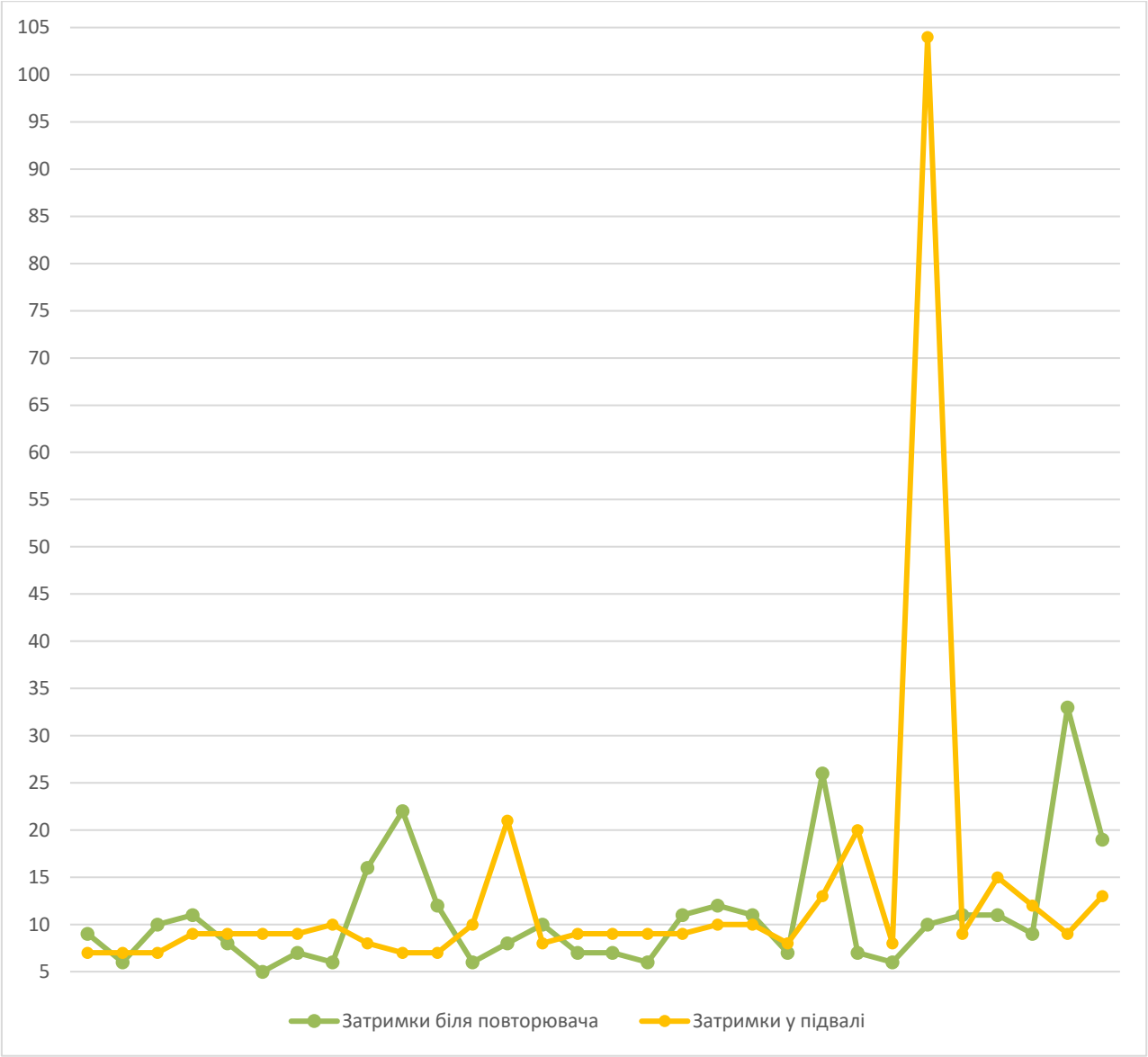


Рисунок 3.31 – Лінійна діаграма замірів затримок (мс)

3.6. Висновок до розділу 3

У даному розділі описано алгоритми функціонування, а саме процедури алгоритму роботи програми, та обрані середовища розробки, процес налаштування Arduino IDE для роботи з ESP8266 і розробка програмного коду повторювача покроково; налаштування та експлуатація мікроконтролера, а саме замикання контактів резистором, верифікація та запис програмного коду, вивід інформації у консоль, налаштування повторювача з персонального комп'ютера та смартфону, підключення до мережі живлення, проведено 30 замірів швидкості зв'язку, дані яких зібрано у лінійні діаграми та проаналізовано на відповідність вимогам до розширення мережі.

ВИСНОВКИ

У результаті виконання даної кваліфікаційної роботи бакалавра створено програмне забезпечення для мікроконтролера, а також реалізовано фактичне розширення домашньої Wi-Fi мережі шляхом повторення сигналу з основного маршрутизатора.

Протягом розробки та створення системи досягнуто наступних результатів:

1. Сформульовано завдання та розглянуто його актуальність. Кожному з пристроїв з підтримкою технології IoT на основі Wi-Fi необхідне підключення до мережі, а використання безпровідних технологій мають свої обмеження - схильність до впливу шумів, інших мереж, особливо у розповсюдженому діапазоні 2.4Ггц, або ж банальна віддаленість від джерела сигналу. Саме з цієї причини виникає необхідність у повторювачах та розширювачах Wi-Fi сигналу.
2. Описано об’єкт дослідження. Розумний дім – це оселя, в якій існує своя домашня мережа та підключені до неї смарт-пристрої для віддаленого моніторингу та керування приладами та системами з підключенням до Інтернету, наприклад, опалення та освітлення, а також автоматизації, і націлена на забезпечення безпеки та комфорту, а також енергоефективності житла, її компоненти обмінюються даними між собою, а керування часто реалізовано у вигляді зручного додатку на смартфон, планшет чи інший кінцевий пристрій.
3. Розглянуто існуючі рішення, одне з яких перевірено на практиці, створено порівняльну таблицю та обрано оптимальне. Найкращим є використання LuaNode32 на чіпі ESP-WROOM-32, проте через наявність хоч і відкритого, але чужого коду та необхідності розбиратись у його алгоритмі роботи, для подальшої роботи обрано набір зі старшої та дешевшої моделі ESP8266 версії ESP-01S.

4. Обрано та описано існуючу мережу, створено її схему, емпіричним шляхом визначено якість Інтернет-зв’язку. До обраного приватного будинку проведено оптоволоконний кабель та підключено тарифний план зі швидкістю Інтернет з’єднання до 100Мб/с. Оптоволоконний кабель підключений до абонентського терміналу на першому поверсі, від якого прокладена вита пара до безпровідного маршрутизатору на другому поверсі. Даний пристрій є єдиним підключенням до мережі Інтернет у всьому будинку.
5. Описано компоненти мережі. Вона складається з оптоволоконного кабелю, який проведено від провайдера, абонентського терміналу ONU ZTE ZXHN F601, витої пари та безпровідного маршрутизатора Huawei WS5200. Абонентський термінал та оптоволоконний кабель обрано саме такі через те, що єдиний провайдер, який надає послуги у населеному пункті, в якому розташований обраний приватний будинок, пропонує клієнтам лише їх. Маршрутизатор і виту пару обрано з урахуванням їх характеристик, таких як діапазони частот, інтерфейси та швидкість передачі даних, а також репутації компанії-виробника.
6. Створено проект розширення існуючої мережі, а саме схема, алгоритми налаштування та роботи повторювача, роботи програми та її процедур. Для реалізації повторювача мережі технології Wi-Fi на базі ESP8266 необхідно написати програмний код мовою Arduino (діалект C/C++). Для кращого розуміння правил та алгоритмів роботи мікроконтролера необхідно побудувати блок-схеми.
7. Створено програмний код та покроково описано процеси його написання і налаштування повторювача. Скетч верифіковано, записано на мікроконтролер, тестове налаштування повторювача виконано успішно – його можна відключити вводити в експлуатацію.
8. Повторювач введено в експлуатацію, проведено та проаналізовано вимірювання швидкості зв’язку та затримок. швидкість у підвалі хоч

і залежить від швидкості біля самого повторювача, проте знаходиться під впливом інших факторів, таких як широке міжповерхове перекриття та його матеріал, швидкість сигналу від маршрутизатора, шум (сусідні Wi-Fi мережі та інше), розташування кінцевого пристрою тощо. Найбільша зафіксована швидкість у підвалі 5.33Мб/с, а найменша – 0.22Мб/с. Незважаючи на часто низьку якість зв’язку (11 з 30 замірів швидкості у підвалі показали результат менше 1Мб/с), він є прийнятним для онлайн-листування та отримання інформації щодо стану повітряної тривоги.

Поставлена задача виконана, мета кваліфікаційної роботи бакалавра досягнута.

ПЕРЕЛІК ВИКОРИСТАНИХ ІНФОРМАЦІЙНИХ ДЖЕРЕЛ

1. Стаття «Наскільки твій “розумний будинок” може стати розумним?» [Електронний ресурс] – Режим доступу до ресурсу: https://vn.20minut.ua/analitika-i-publicistika/naskilki-tviy-rozumniy-budinok-mozhe-stati-rozumnim-10525652.html (дата звертання 11.05.2022)
2. Рейтинговий список clutch.io [Електронний ресурс] – Режим доступу до ресурсу: https://clutch.co/developers/internet-of-things?sort_by=ReviewRating (дата звертання 11.05.2022)
3. Стаття про використання маршрутизатора у якості точки доступу [Електронний ресурс] – Режим доступу до ресурсу: https://help-wifi.com/sovety-po-nastrojke/kak-router-sdelat-tochkoj-dostupa-wi-fi/ (дата звертання 11.05.2022)
4. Стаття про використання Wi-Fi повторювача [Електронний ресурс] – Режим доступу до ресурсу: https://help-wifi.com/poleznoe-i-interesnoe/chto-takoe-wi-fi-repetir-povtoritel-kak-on-rabotaet-i-chto-znachit-router-v-rezhime-repitera/ (дата звертання 11.05.2022)
5. Стаття про створення Wi-Fi повторювача на базі Raspberry Pi [Електронний ресурс] – Режим доступу до ресурсу: https://pimylifeup.com/raspberry-pi-wifi-extender/ (дата звертання 11.05.2022)
6. Репозиторій Wi-Fi повторювача на базі LuaNode32 [Електронний ресурс] – Режим доступу до ресурсу: https://github.com/martin-ger/esp32_nat_router
7. Сторінка товару LuaNode32 [Електронний ресурс] – Режим доступу до ресурсу: https://arduino.ua/prod2041-wi-fi-modyl-luanode32-s-esp-32 (дата звертання 12.05.2022)
8. Сторінка товару ESP8266 ESP-01S [Електронний ресурс] – Режим доступу до ресурсу: https://arduino.ua/prod2892-esp-01s-wi-fi-modyl-esp8266 (дата звертання 12.05.2022)

9. Сторінка товару USB перехідник для програмування і налагодження модулів ESP-01S [Електронний ресурс] – Режим доступу до ресурсу: https://arduino.ua/prod3262-usb-perehodnik-dlya-programmirovaniya-i-otladki-modylei-esp-01-i-esp-01s (дата звертання 12.05.2022)
10. Веб-додаток створення діаграм Draw.io [Електронний ресурс] – Режим доступу до ресурсу: https://app.diagrams.net/ (дата звертання 17.05.2022)
11. Веб-додаток заміру швидкості Інтернету SPEEDTEST [Електронний ресурс] – Режим доступу до ресурсу: https://www.speedtest.net (дата звертання 17.05.2022)
12. Стаття про вимоги до кінцевого пристрою та якості Інтернет-зв'язку у довіднику «YouTube Довідка» [Електронний ресурс] – Режим доступу до ресурсу: https://support.google.com/youtube/answer/78358?hl=uk (дата звертання 17.05.2022)

ДОДАТОК А

Код програми

Листів 9

Розробник Крук Н.Б.

Керівник Степанов М.М.

Київ - 2022

```
#if LWIP_FEATURES && !LWIP_IPV6
#define HAVE_NETDUMP 0

#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <lwip/napt.h>
#include <lwip/dns.h>
#include <LwipDhcpServer.h>
#include <FS.h>

#define NAPT 1000
#define NAPT_PORT 10
const char* ssid = "";
const char* password = "";

#if HAVE_NETDUMP
#include <NetDump.h>

void dump(int netif_idx, const char* data, size_t len, int out, int
success) {
  (void)success;
  Serial.print(out ? F("out ") : F(" in "));
  Serial.printf("%d ", netif_idx);
}
#endif

bool testwifi() {
  delay(1000);
  Serial.printf("\nТестування підключення до '%s'\n",
WiFi.SSID().c_str());
  int count = 0;
  digitalWrite(2, LOW);
  while (count < 20) {
```

```
    //connection testing
    if (WiFi.status() == WL_CONNECTED) {
      Serial.printf("\nWi-Fi підключено! \nSTA: %s (dns: %s / %s)\n\n",
                    WiFi.localIP().toString().c_str(),
                    WiFi.dnsIP(0).toString().c_str(),
                    WiFi.dnsIP(1).toString().c_str());

      //give DNS servers to AP side
      dhcpSoftAP.dhcps_set_dns(0, WiFi.dnsIP(0));
      dhcpSoftAP.dhcps_set_dns(1, WiFi.dnsIP(1));
      digitalWrite(2, HIGH);
      return true;
    }
    Serial.print(".");
    delay(1000);
    count++;
  }
  Serial.printf("\nНеможливо з'єднатись з мережею Wi-Fi! Підєднайтесь до
точки доступу '%s' та виконайте налаштування повторювача!\n\n",
WiFi.softAPSSID());
  return false;
}

//server with settings
ESP8266WebServer server(80);
String content;
void serverconfig() {
  server.begin();

  //page
  server.on("/", []() {
    //base
    content = "<!DOCTYPE html><html lang='uk'><meta name='viewport'
content='width=device-width, initial-scale=1.0' charset=utf-8>";
```

```
    content += "<head><title>Wi-Fi повторювач ESP8266 версії ESP-
01S</title><link href='data/bootstrap.css' rel='stylesheet'
type='text/css'></head>";
    content += "<style> .form-control{max-width: 400px;} br{display:
block; content: '';margin-top: 5px;} .br-plus{display: block; content:
''; margin-top: 10px;} button{margin-bottom: 10px;}</style>";
    content += "<body><div class='container-fluid'><h1>Wi-Fi повторювач
ESP8266 версії ESP-01S</h1><h5>Автор: Крук Назар. Слава
Україні!</h5><br><h2>Налаштування Wi-Fi мережі</h2>";
    //network settings
    content += "<form method='post'><label class='form-
label'>SSID:</label><br><input name='stassid' placeholder='";
    content += WiFi.SSID();
    content += "' length=32 class='form-control'><br><label class='form-
label'>Пароль:</label><br><input type='password' placeholder='********'
name='stapsk' minlength=8 maxlength=63 class='form-control'>";
    content += "<div class='form-text'>Мінімальна довжина - 8 символів!
Інакше залиште порожнім.</div><br class='br-plus'><button type='submit'
formaction='stasettings' class='btn btn-primary'>Зберегти</button>";
    content += "<button type='submit' formaction='tempstasettings'
class='btn btn-outline-primary'>Зберегти до
перезапуску</button></form><h2>Налаштування точки доступу</h2>";
    //AP settings
    content += "<form method='post'><label class='form-
label'>SSID:</label><br><input name='apssid' placeholder='";
    content += WiFi.softAPSSID();
    content += "' length=32 class='form-control'><br><label class='form-
label'>Пароль:</label><br><input type='password' placeholder='";
    content += WiFi.softAPPSK();
    content += "' name='appsk' minlength=8 maxlength=63 class='form-
control'><div class='form-text'>Мінімальна довжина - 8 символів! Інакше
залиште порожнім.</div><br class='br-plus'>";
    content += "<button type='submit' formaction='apsettings' class='btn
btn-primary'>Зберегти</button>";
```

```
    content += "<button type='submit' formaction='tempapsettings'
class='btn btn-outline-primary'>Зберегти до
перезапуску</button></form><h2>Додатково</h2><br class='br-plus'>";
    //features
    content += "<form method='post'><button type='submit'
formaction='reboot' class='btn btn-warning'>Перезавантаження
повторювача</button><br>";
    content += "<button type='submit' formaction='stastandard' class='btn
btn-outline-warning'>Стандартні налаштування Wi-Fi мережі</button><br>";
    content += "<button type='submit' formaction='apstandard' class='btn
btn-outline-warning'>Стандартні налаштування точки
доступу</button></form></div>";
    server.send(200, "text/html", content);
  });
  server.onNotFound([]() {
    server.send(404, "text/plain", "Тож... Яким чином ми тут
опинились?");
  });

  //network settings
  server.on("/stasettings", []() {
    String stassid = server.arg("stassid");
    String stapsk = server.arg("stapsk");
    if (stassid.length() > 0) {
      server.send(200, "text/plain", "Налаштування отримано!");
      Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n",
stassid.c_str(), stapsk.c_str());
      WiFi.persistent(true);
      WiFi.begin(stassid, stapsk);
      testwifi();
    }
  });
  server.on("/tempstasettings", []() {
    String stassid = server.arg("stassid");
```

```
    String stapsk = server.arg("stapsk");
    if (stassid.length() > 0) {
      server.send(200, "text/plain", "Налаштування отримано!");
      Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n",
stassid.c_str(), stapsk.c_str());
      WiFi.persistent(false);
      WiFi.begin(stassid, stapsk);
      testwifi();
    }
  });

  //AP settings
  server.on("/apsettings", []() {
    String apssid = server.arg("apssid");
    String appsk = server.arg("appsk");
    if (apssid.length() > 0) {
      server.send(200, "text/plain", "Налаштування отримано!");
      Serial.printf("\n\nНалаштування точки доступу \nSSID: %s \nПароль:
%s \n", apssid.c_str(), appsk.c_str());
      WiFi.persistent(true);
      WiFi.softAP(apssid, appsk);
    }
  });
  server.on("/tempapsettings", []() {
    String apssid = server.arg("apssid");
    String appsk = server.arg("appsk");
    if (apssid.length() > 0) {
      server.send(200, "text/plain", "Налаштування отримано!");
      Serial.printf("\n\nНалаштування точки доступу (до перезавантаження)
\nSSID: %s \nПароль: %s \n", apssid.c_str(), appsk.c_str());
      WiFi.persistent(false);
      WiFi.softAP(apssid, appsk);
    }
  });
```

```
  //features
  server.on("/reboot", []() {
    server.send(200, "text/plain", "Перезавантаження повторювача...");
    Serial.printf("\n\nПерезавантаження повторювача...");
    delay(5000);
    ESP.reset();
  });
  server.on("/stastandard", []() {
    String stassid = "";
    String stapsk = "";
    server.send(200, "text/plain", "Встановлення стандартних налаштувань
Wi-Fi мережі");
    Serial.printf("\n\nСпроба підключення до '%s' з паролем '%s' \n",
stassid.c_str(), stapsk.c_str());
    WiFi.persistent(true);
    WiFi.begin(stassid, stapsk);
    testwifi();
  });
  server.on("/apstandard", []() {
    String apssid = "ESP_62F21C";
    String appsk = "";
    server.send(200, "text/plain", "Встановлення стандартних налаштувань
точки доступу");
    Serial.printf("\n\nНалаштування точки доступу \nSSID: %s \nПароль: %s
\n", apssid.c_str(), appsk.c_str());
    WiFi.persistent(true);
    WiFi.softAP(apssid, appsk);
  });
}

void setup() {
  Serial.begin(115200);
  pinMode(2, OUTPUT);
```

```
  delay(2000);
  Serial.printf("\n\nWi-Fi повторювач ESP8266 версії ESP-01S\n");
  Serial.printf("Автор: Крук Назар. Слава Україні!\n");
  Serial.printf("Доступна пам'ять при запуску: %d\n", ESP.getFreeHeap());

  //css check
  if(!SPIFFS.begin()) {
    Serial.println("Помилка монтування файлової системи!");
    return;
  }
  File file = SPIFFS.open("/bootstrap.css", "r");
  if(!file) {
    Serial.println("Помилка відкриття bootstrap.css!");
    return;
  }
  else {
    file.close();
    Serial.println("Файл bootstrap.css наявний!");
  }

#if HAVE_NETDUMP
  phy_capture = dump;
#endif

  //set radio type to N
  WiFi.setPhyMode(WIFI_PHY_MODE_11N);
  WiFi.mode(WIFI_AP_STA);
  WiFi.persistent(false);
  //use stored credentials to connect to network
  WiFi.begin(ssid, password);
  testwifi();
  //set IP Address, Gateway and Subnet
  WiFi.softAPConfig(
    IPAddress(192, 168, 4, 1),
```

```
    IPAddress(192, 168, 4, 1),
    IPAddress(255, 255, 255, 0));
  //use stored credentials to create AP
  WiFi.softAP(WiFi.softAPSSID(), WiFi.softAPPSK());

  //NAPT-test
  Serial.printf("Доступна пам'ять до NAPT-тесту: %d\n",
ESP.getFreeHeap());
  err_t ret = ip_napt_init(NAPT, NAPT_PORT);
  Serial.printf("ip_napt_init(%d,%d): ret=%d (OK=%d)\n", NAPT, NAPT_PORT,
(int)ret, (int)ERR_OK);
  if (ret == ERR_OK) {
    ret = ip_napt_enable_no(SOFTAP_IF, 1);
    Serial.printf("ip_napt_enable_no(SOFTAP_IF): ret=%d (OK=%d)\n",
(int)ret, (int)ERR_OK);
    if (ret == ERR_OK) {
      Serial.printf("\nWi-Fi мережа '%s' з паролем '%s' та IP-адресою
'%s' налаштована та запущена!\n", WiFi.softAPSSID(),
WiFi.softAPPSK().c_str(), WiFi.softAPIP().toString().c_str());
    }
  }
  Serial.printf("Доступна пам'ять після NAPT-тесту: %d\n",
ESP.getFreeHeap());
  if (ret != ERR_OK) {
    Serial.printf("NAPT-тест невдалий\n");
  }

  serverconfig();
}

#else

void setup() {
  Serial.begin(115200);
```

```
  Serial.printf("\n\nNAPT не підтримується на даній конфігурації\n");
}

#endif

void loop() {
  server.handleClient();
  if (WiFi.status() != WL_CONNECTED) {
    digitalWrite(2, LOW);
    delay(1000);
    digitalWrite(2, HIGH);
    delay(1000);
  }
  else {
    digitalWrite(2, HIGH);
  }
}
```

ДОДАТОК Б

Скріншоти замірів швидкості сигналу

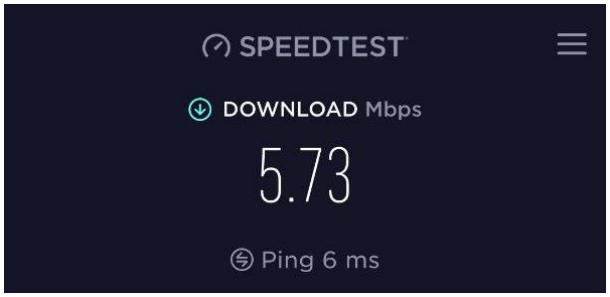
Листів 6

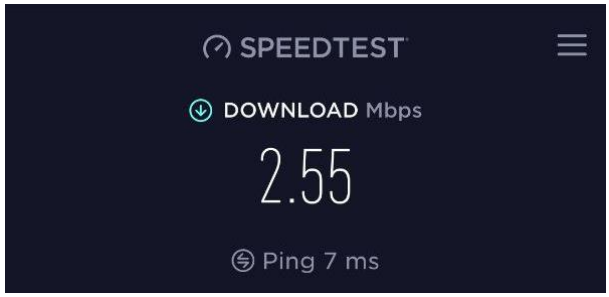
Розробник Крук Н.Б.

Керівник Степанов М.М.

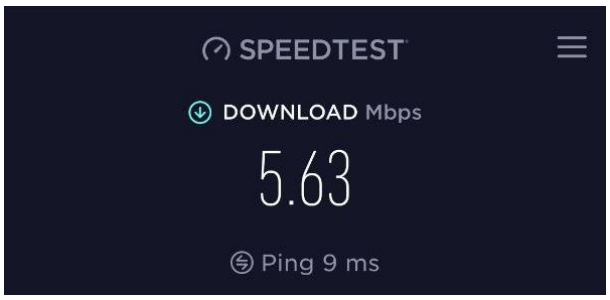
Київ - 2022

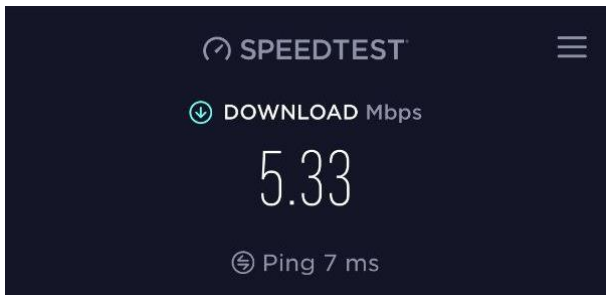
Замір 1:



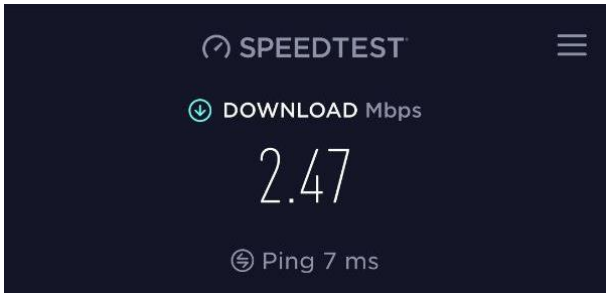


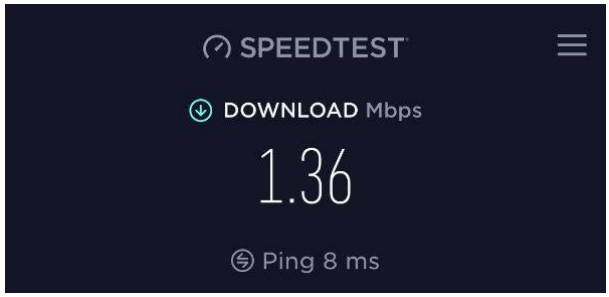
Замір 2:



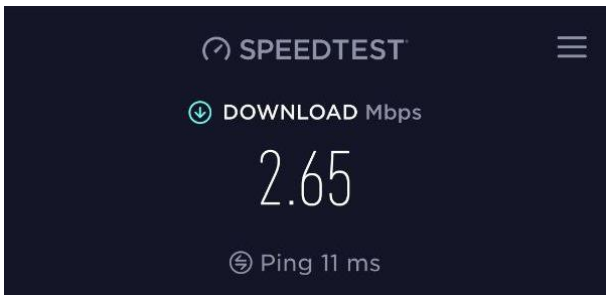


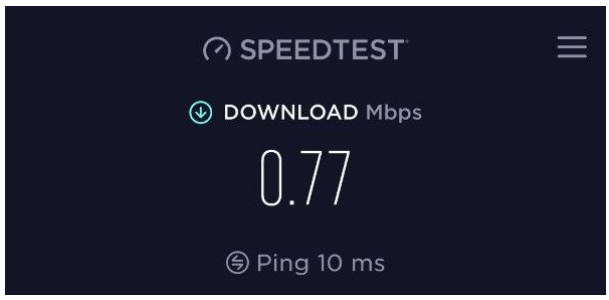
Замір 3:



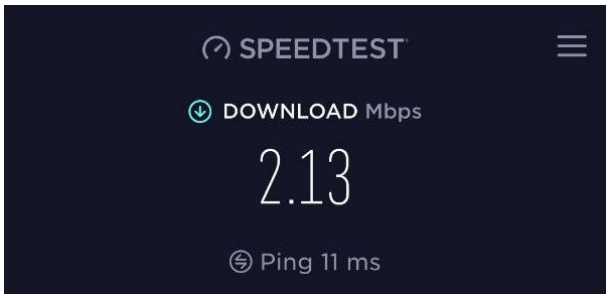


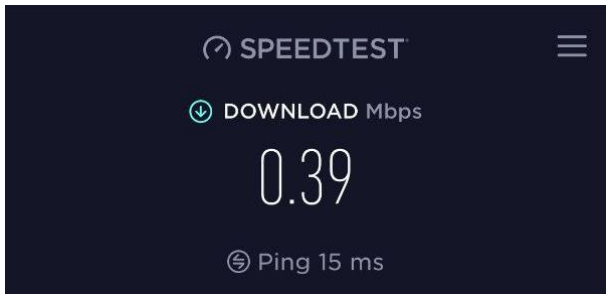
Замір 4:



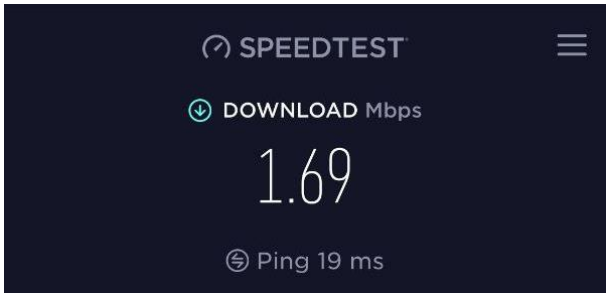


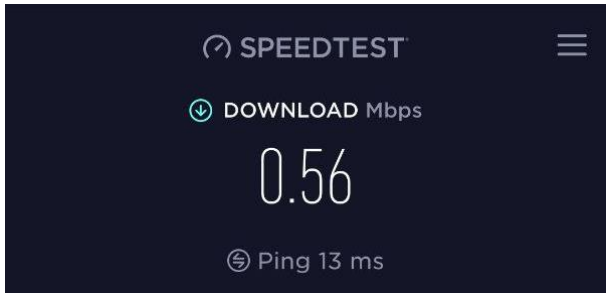
Замір 5:



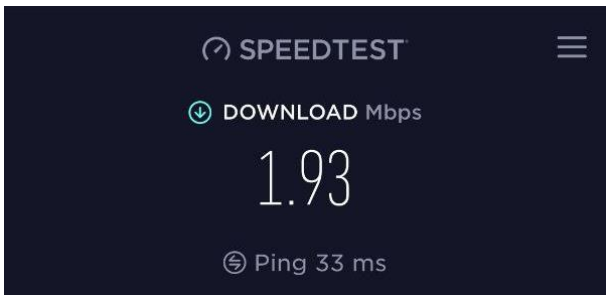


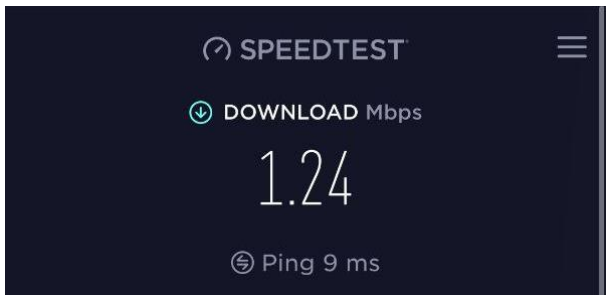
Замір 6:



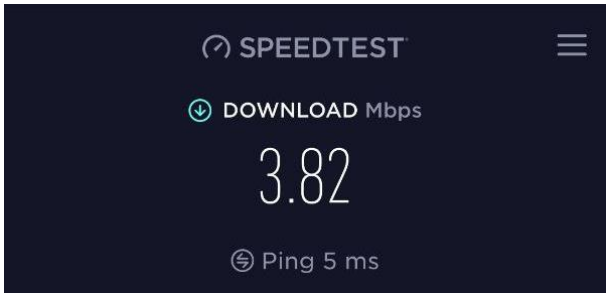


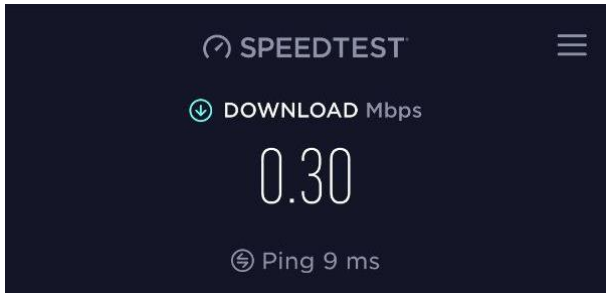
Замір 7:



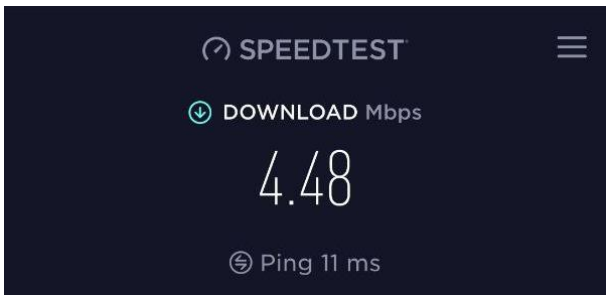


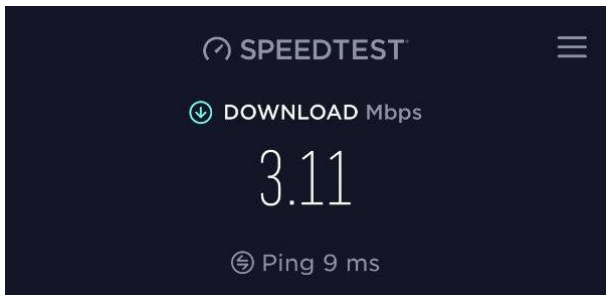
Замір 8:



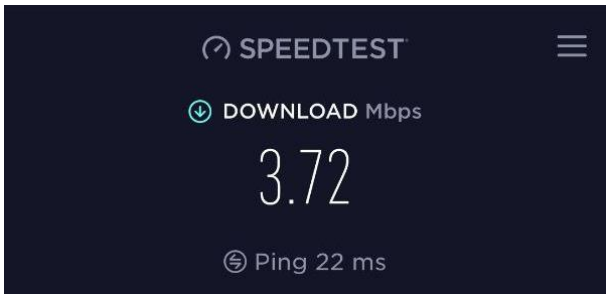


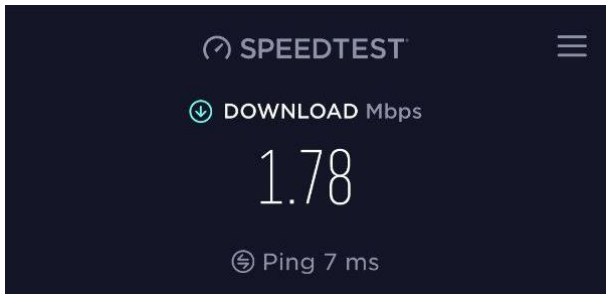
Замір 9:



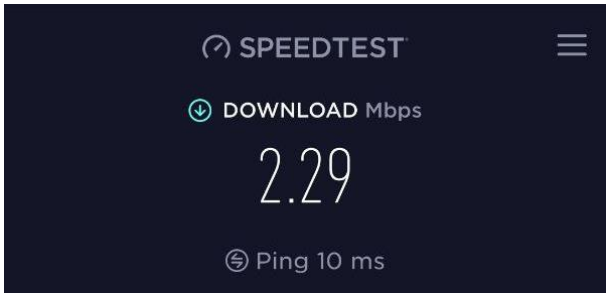


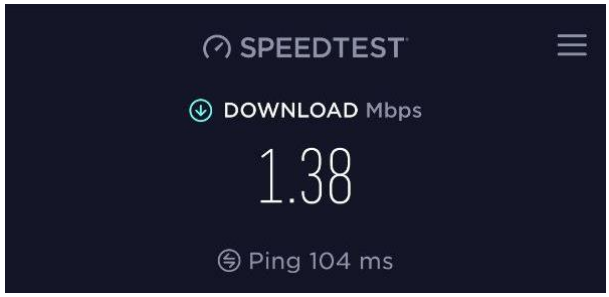
Замір 10:



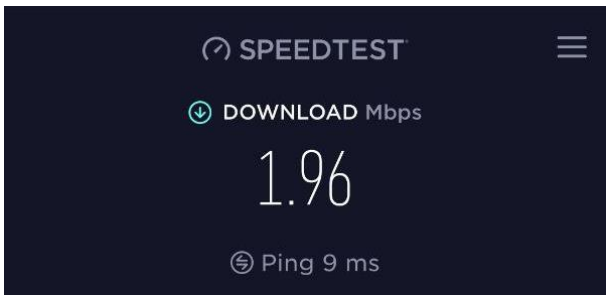


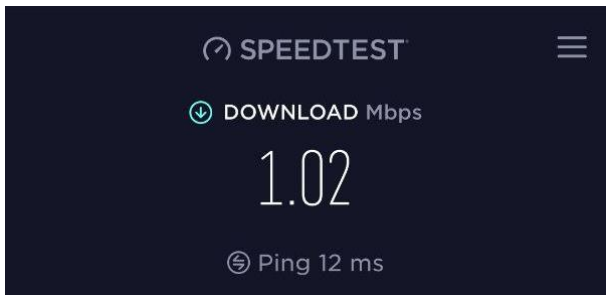
Замір 11:



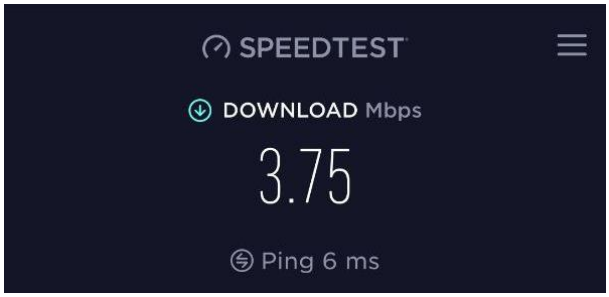


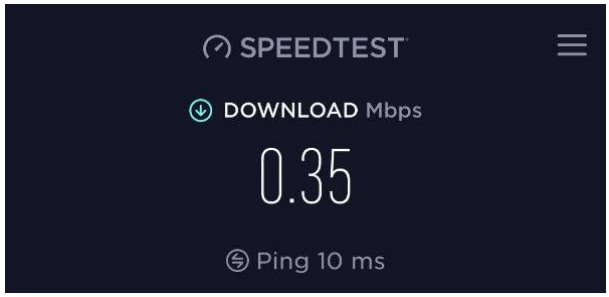
Замір 12:



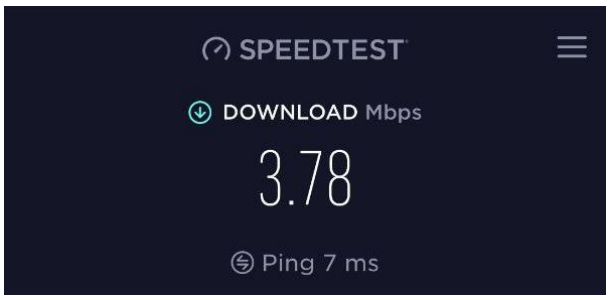


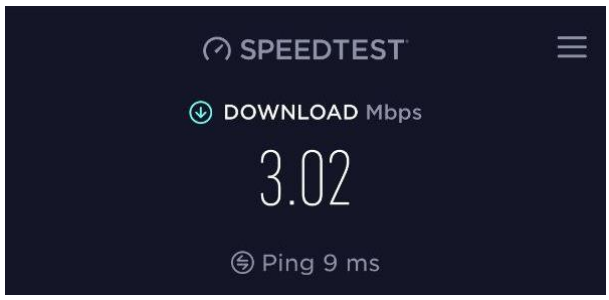
Замір 13:



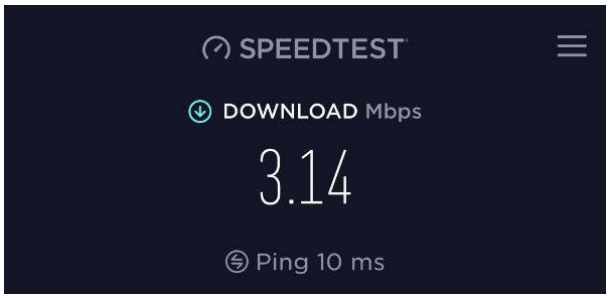


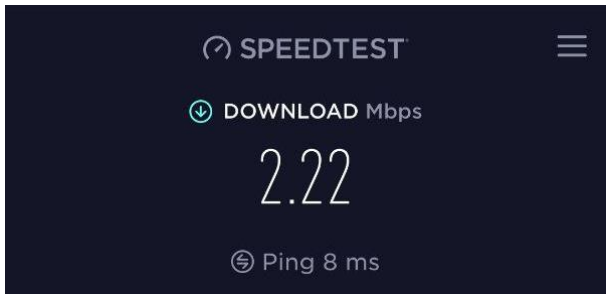
Замір 14:



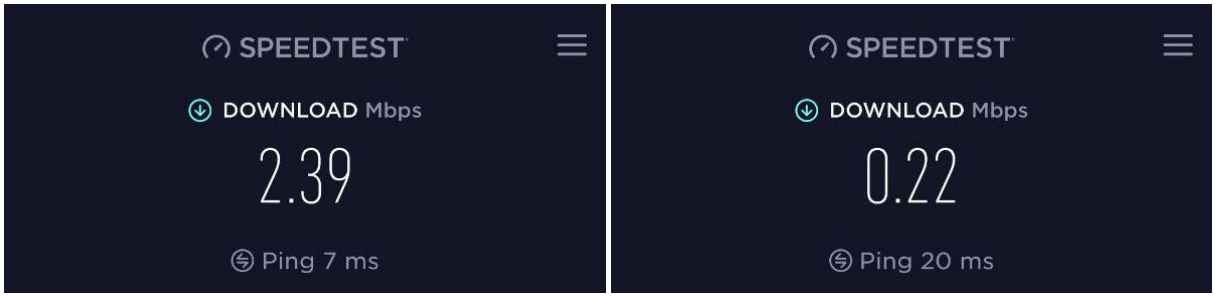


Замір 15:

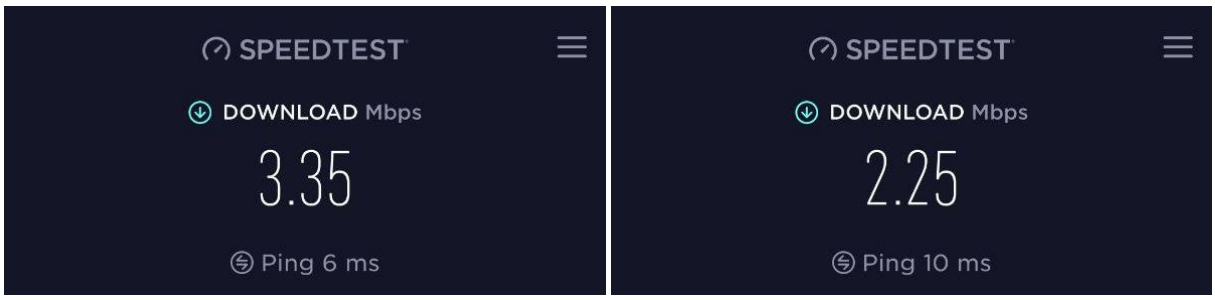




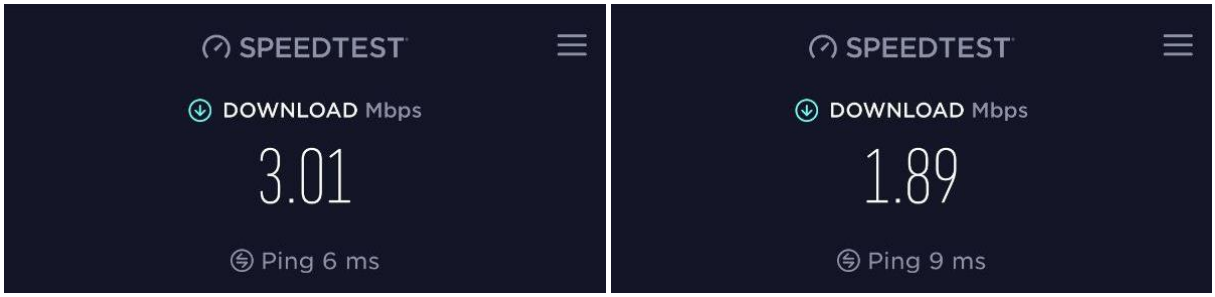
Замір 16:



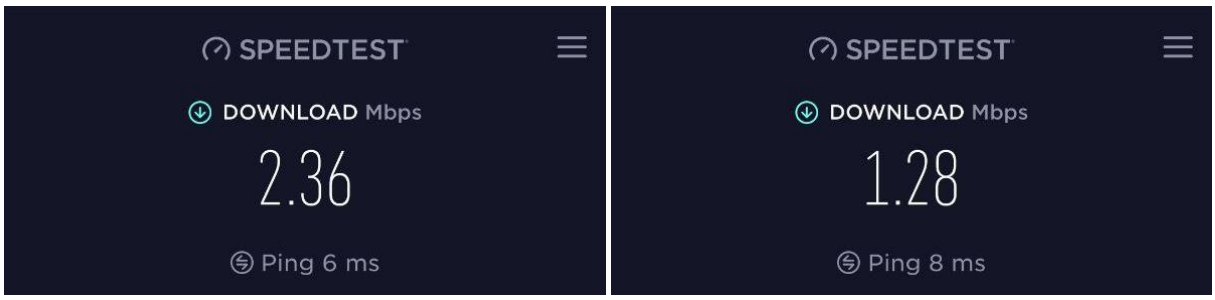
Замір 17:



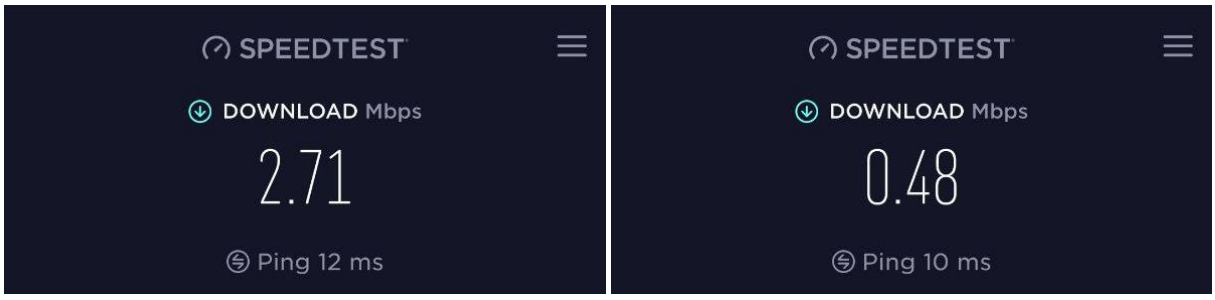
Замір 18:



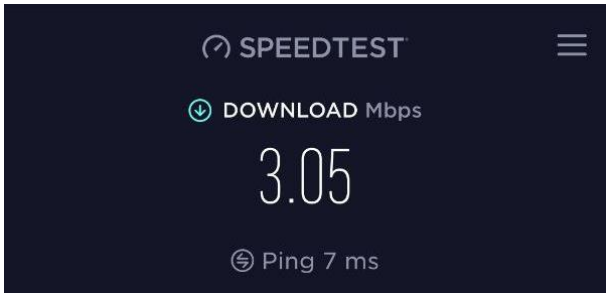
Замір 19:

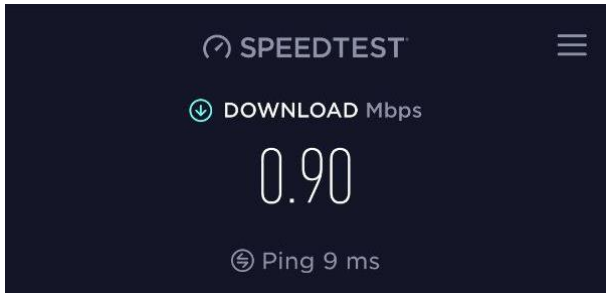


Замір 20:

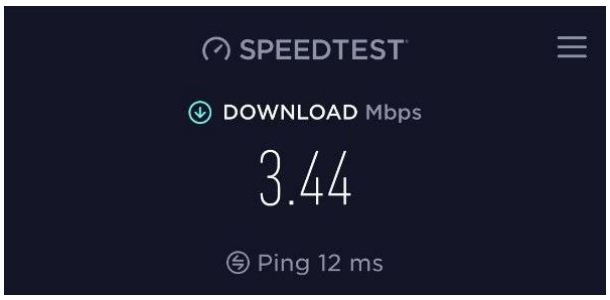


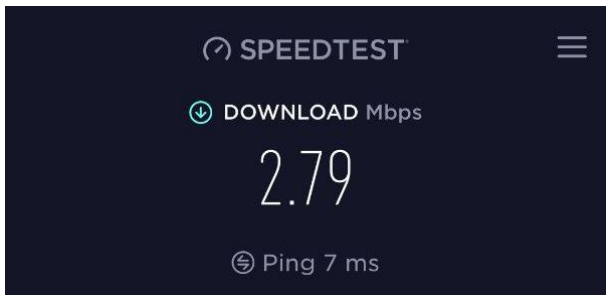
Замір 21:



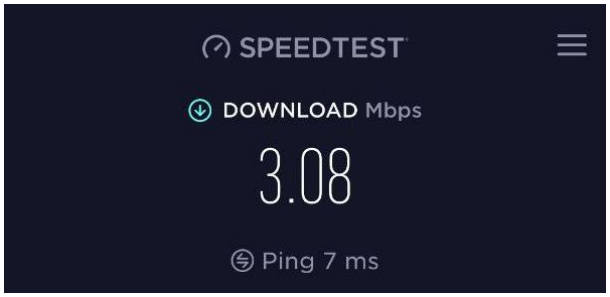


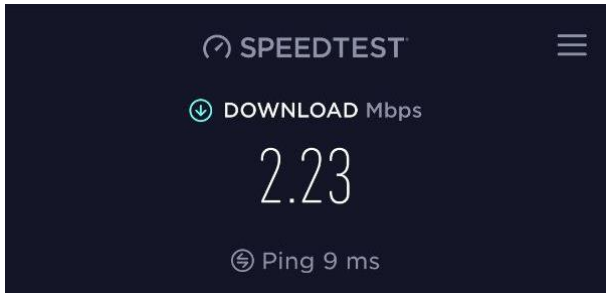
Замір 22:



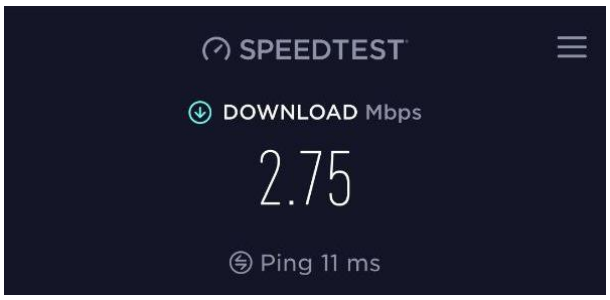


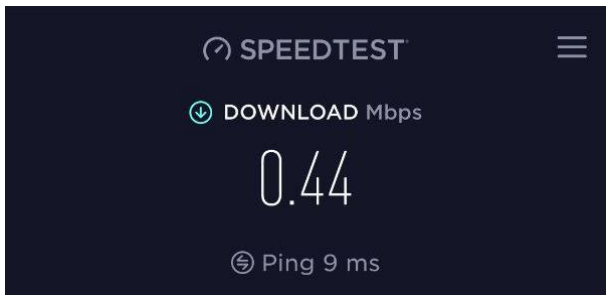
Замір 23:



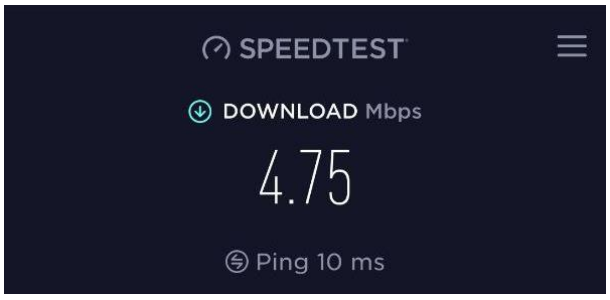


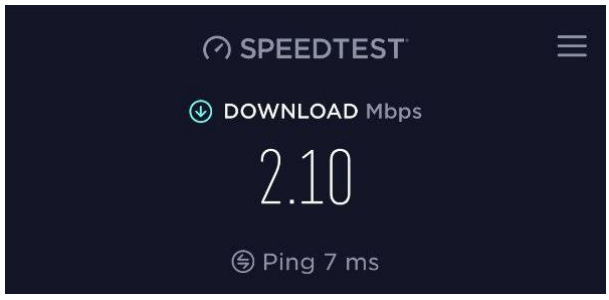
Замір 24:



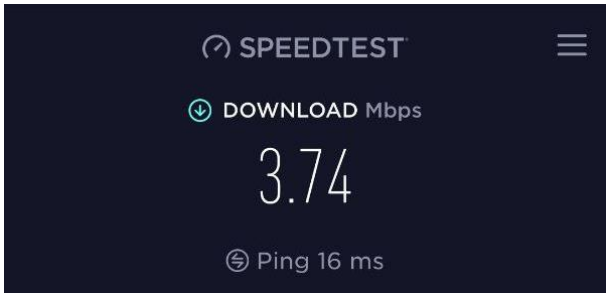


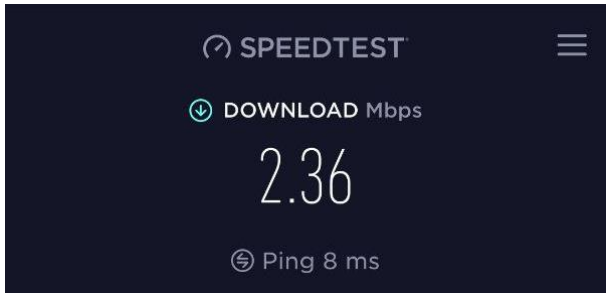
Замір 25:



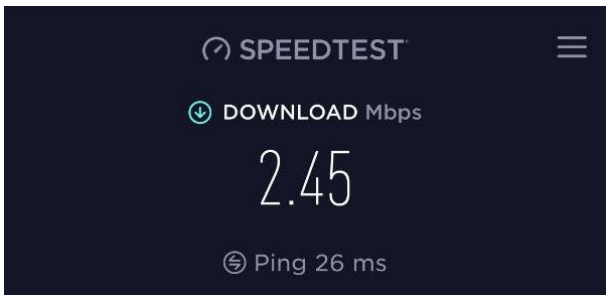


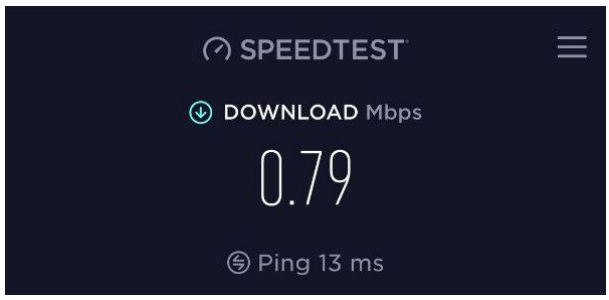
Замір 26:



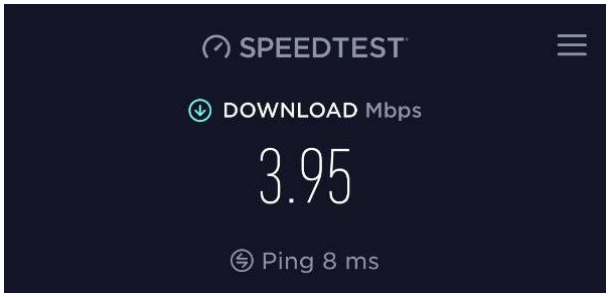


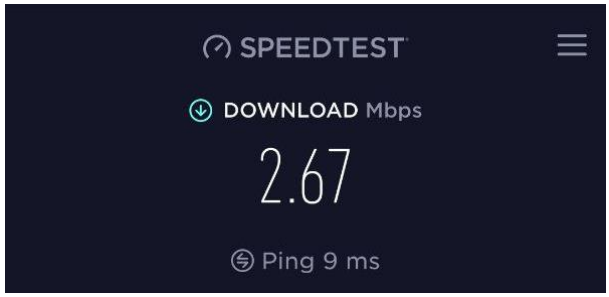
Замір 27:



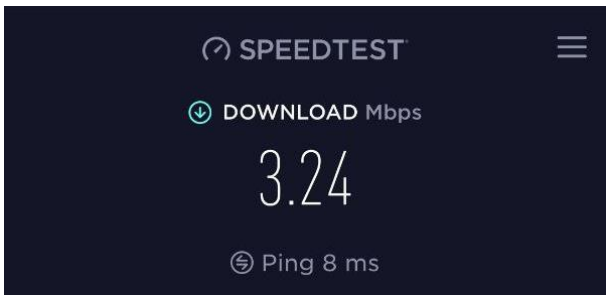


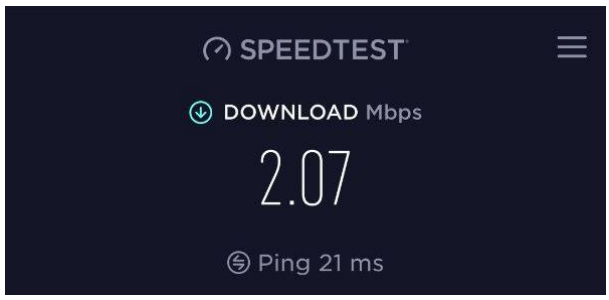
Замір 28:



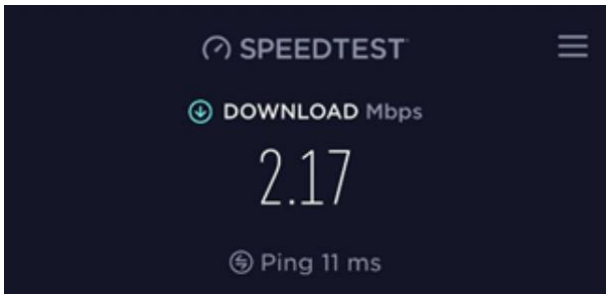


Замір 29:





Замір 30:



ДОДАТОК В

Скріншоти презентації дипломної роботи

Листів 2

Розробник Крук Н.Б.

Керівник Степанов М.М.

Київ - 2022

Рисунок 1 – Титульний слайд

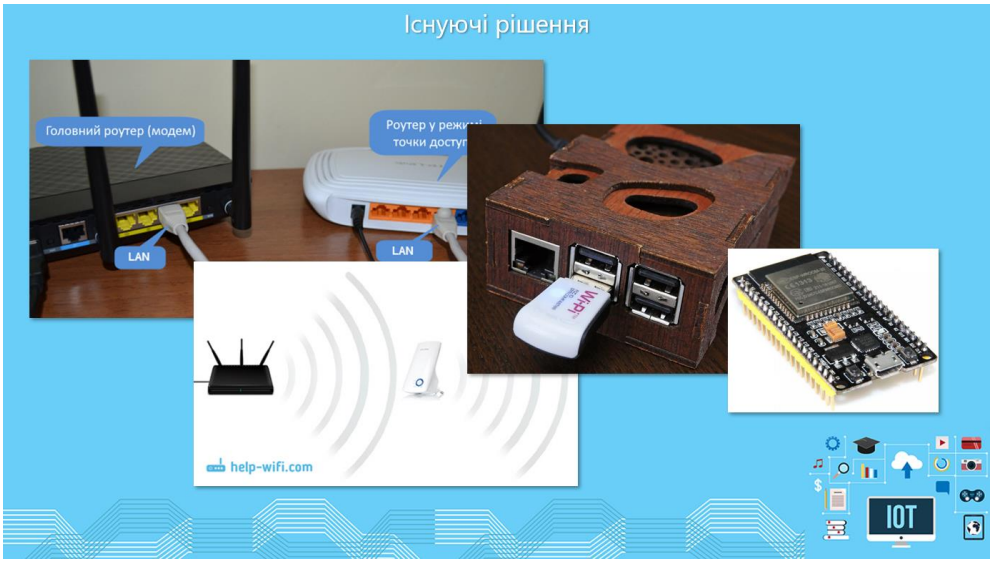


Рисунок 2 – Існуючі рішення

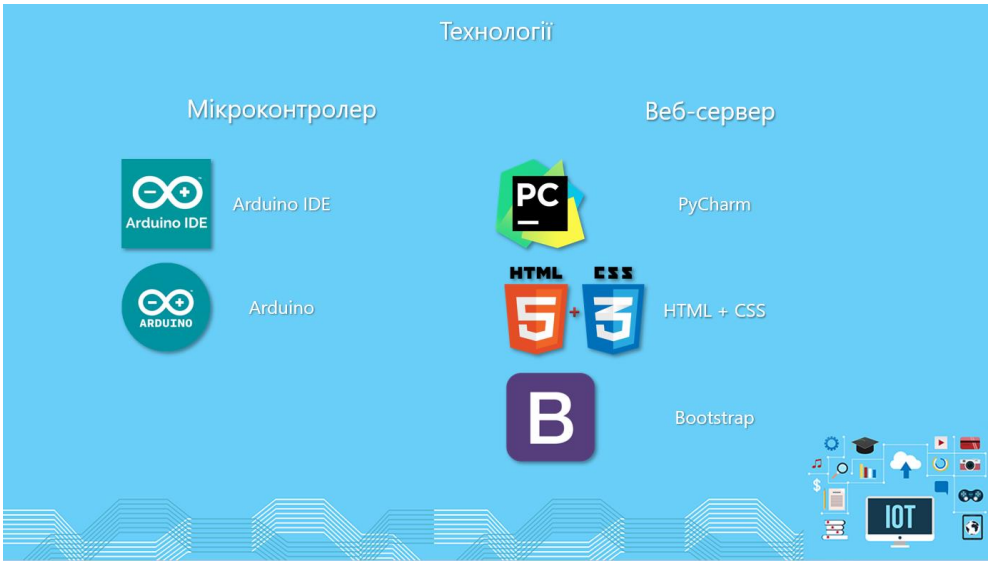


Рисунок 3 – Використані технології

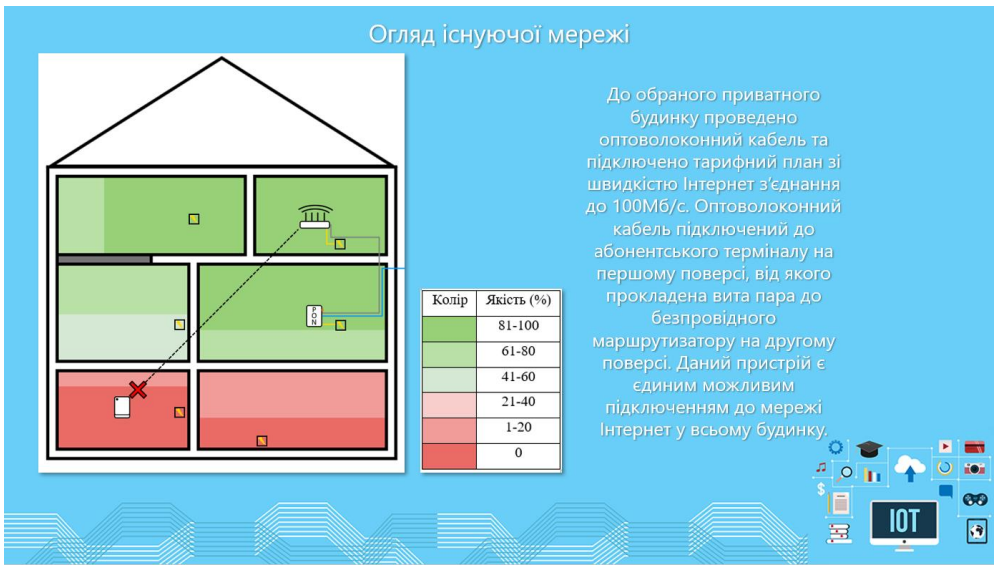


Рисунок 4 – Огляд існуючої мережі

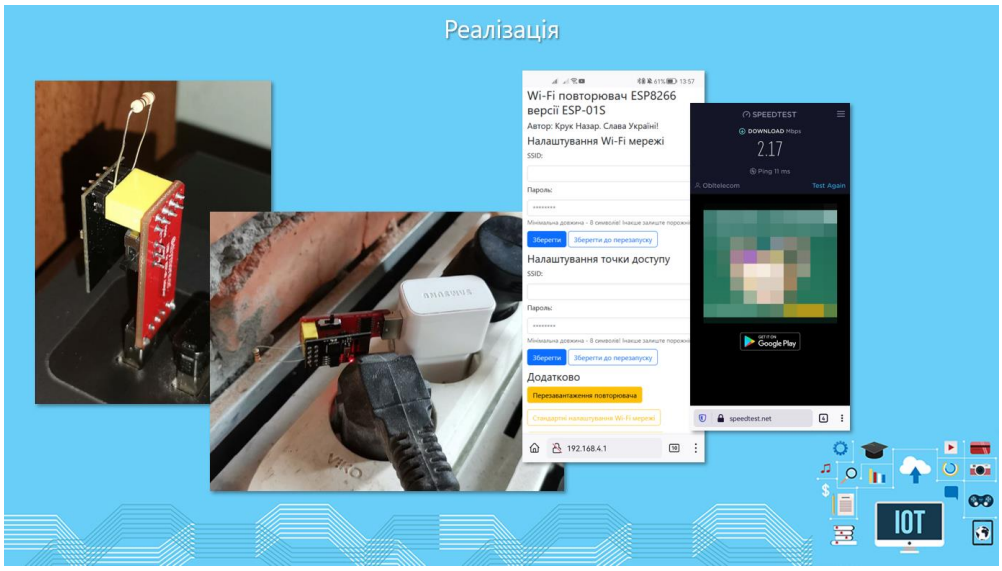


Рисунок 5 – Реалізація

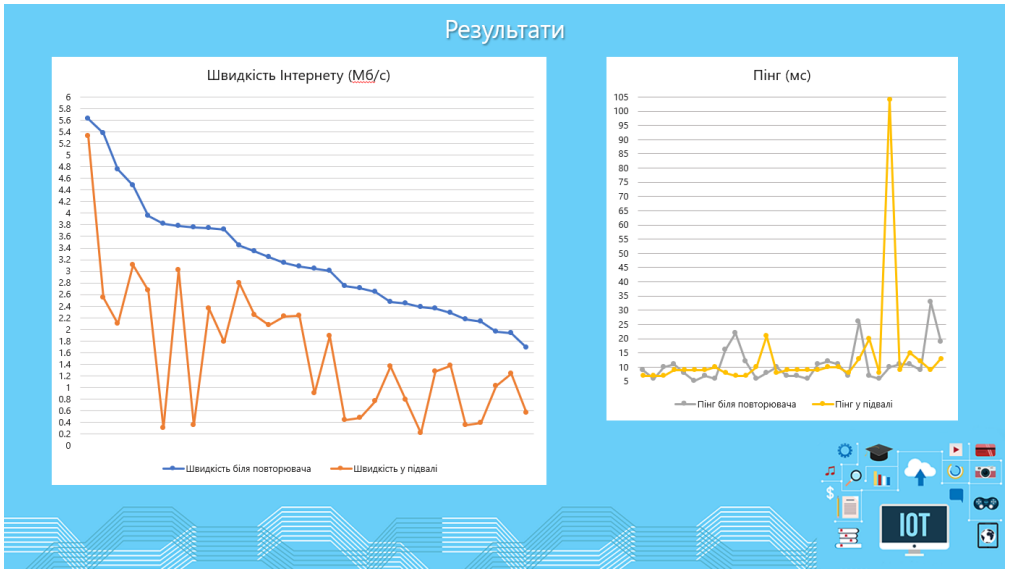


Рисунок 6 - Результати