Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет _____Автоматики і комп'ютеризованих технологій_____
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій, автоматизації та роботехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____перший (бакалаврський)_____

Розробка системи автоматизації – розумний будинок "Modular House"
(тема)

Виконав:

студент IV курсу, групи АКТАКІТ-20-1
Юсупов В. Т.
(прізвище, ініціали)

Спеціальність ____151 Автоматизація та комп'ютерно-інтегровані технології____
(код і повна назва спеціальності)

Тип програми __освітньо-професійна_____

Освітня програма _____Автоматизація та комп'ютерно-інтегровані технології___
(повна назва освітньої програми)

Керівник _____ст. викл. Теслюк С. І._____
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ ___Невлюдов І. Ш.__
(підпис) (прізвище, ініціали)

2024 р.

Я, як студент ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

«15» червня 2024 року  Юсупов В.Т.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 151 Автоматизація та комп`ютерно-інтегровані технології
Тип програми Освітньо-професійна
Освітня програма Автоматизація та комп`ютерно-інтегровані технології
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри КІТАР _____
(підпис)
«___» _____ 2024 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Юсупову Владиславу Тимуровичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розробка системи автоматизації – розумний будинок "Modular House"
Затверджена наказом по університету від 03.06.2024 р. № 544 Ст
2. Термін подання студентом роботи до екзаменаційної комісії 20.06.2024 р.
3. Вихідні дані до роботи
3.1 Фреймворк .NET;
3.2 Мова програмування C#;
3.3 Середовище розробки Rider;
3.4 Мікроконтролер Arduino UNO;
3.5 Середовище розробки Arduino IDE;
3.6 Мова програмування C++.
4. Перелік питань, що потрібно опрацювати в роботі
4.1 Вступ;
4.2 Аналіз недоліків існуючих систем;
4.3 Методи керування розумним будинком;
4.4 Складання макету системи;
4.5 Розробка коду мікроконтролера Arduino UNO;
4.6 Розробка коду серверної частини;
4.7 Проведення експерименту;
4.8 Розрахунки для моделювання системи автоматичного управління;
4.9 Охорона праці;
4.10 Висновки;
4.11 Додатки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій

Демонстраційний матеріал у вигляді презентації PowerPoint (*.pptx) – 19 с.

6. Консультанти розділів роботи

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Приміт-ка
1	Аналіз технічного завдання	29.04 – 011.05.24	виконано
2	Визначення методів розробки розумного будинку	13.05 – 18.05.24	виконано
3	Розробка макету системи	19.05 – 21.05.24	виконано
4	Розробка коду для мікроконтролера Arduino UNO	22.05 – 30.05.24	виконано
5	Розробка програми управління системою	31.05 – 10.06.24	виконано
6	Охорона праці	11.06 – 12.06.24	виконано
7	Подання роботи на перевірку на плагіат	13.06 – 15.06.24	виконано
8	Оформлення пояснювальної записки	15.06 – 15.06.24	виконано
9	Подання роботи на рецензію	15.06 – 16.06.24	виконано
10	Подання роботи на підпис зав. кафедри	17.06 – 19.06.24	виконано
11	Подання кваліфікаційної роботи в ЕК	20.06.2024	виконано

Дата видачі завдання 01.04.2024 р.

Студент  Юсупов В. Т.

(підпис)

Керівник роботи _____ ст. викл. Теслюк С. І.

(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 78 с., 2 табл., 63 рис., 4 дод., 21 джерел.

ВЕБ-СЕРВЕР, РОЗУМНИЙ БУДИНОК, СИСТЕМА АВТОМАТИЗАЦІЇ, .NET, ESP32, Wi-Fi.

Мета роботи – покращення роботи системи розумного будинку за рахунок розробки есперименталього макету з декількома режимами роботи в залежності від потреб користувача в режимі реального часу.

Об’єкт розробки – автоматизація процесу регулювання стану розумних модулів.

Предмет розробки – система розумного будинку на базі веб-сервісу.

Методи дослідження – аналіз існуючих систем, аналіз методів контролю параметрів, розробка лабораторного макету.

В кваліфікаційній роботі розглянуто актуальні питання за темою, запропоновано систему автоматизації розумного будинку.

Спроєктовано та створено лабораторний макет системи автоматизації розумного будинку. Також розроблено програмний продукт для управління системою, за допомогою якого здійснюється вмикання та вимикання системи, а також регулювання стану під’єднаних модулів.

Також була розроблена модель системи автоматичного управління згідно теорії автоматичного управління, завдяки якої є змога оптимізувати параметри модулів до стану бажаних значень показників.

ABSTRACT

Explanatory note: 78 p., 2 tabl., 63 fig., 4 adj., 21 sources.

.NET, AUTOMATION SYSTEM, ESP 32, MODULAR HOUSE, WEB SERVER, Wi-Fi.

The goal of the work is to develop a modular house automation system that has several operating modes depending on production needs and controls the state of connected smart modules in real-time.

The object of development is to automate the process of regulating the state of smart modules.

The subject of development is a smart home system based on a web service.

Research methods include analysis of existing systems, analysis of parameter control methods, and development of a laboratory mock-up.

The thesis addresses current issues on the topic and proposes a smart home automation system.

A laboratory mock-up of a smart home automation system was designed and created. Software was also developed to control the system, which allows turning the system on and off, as well as regulating the state of connected devices.

A model of an automatic control system was also developed according to automatic control theory, which makes it possible to optimize the parameters of modules to the desired state of indicators.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

API (Application Programming Interface) – відкритий інтерфейси програмування;

ASCII (American Standard Code for Information Interchange) – стандартний кодувальний набір символів;

GET – HTTP-метод, який використовується для отримання ресурсів із сервера;

HTTP (Hypertext Transfer Protocol) – протокол передачі даних, який використовується для передачі інформації між веб-браузерами та веб-серверами в Інтернеті;

Id (Identifier) – унікальний ідентифікатор;

IDE (Integrated Development Environment) – програмне середовище, призначене для полегшення процесу розробки програмного забезпечення;

IoT (Internet of Things) – інтернет речей;

LED (Light-Emitting Diode) – напівпровідниковий прилад, який випромінює світло при проходженні через нього електричного струму;

MQTT (Message Queuing Telemetry Transport) – протокол передачі повідомлень, який часто використовується в IoT;

PATCH – HTTP-метод, який використовується для часткового оновлення ресурсу на сервері;

POST – HTTP-метод, який використовується для надсилання даних на сервер з метою створення нового ресурсу;

UNO – назва плати мікроконтролера, розробленої компанією Arduino.

ВСТУП

На сьогоднішній день концепція розумного будинку стає все більш популярною. Автоматизація та інтелектуальне керування різними системами в оселі дозволяє підвищити рівень комфорту, безпеки та енергоефективності. Розвиток технологій Інтернету речей та доступність різноманітних пристроїв і рішень для автоматизації відкривають нові можливості для створення розумних будинків.

Одним з ключових аспектів розумного будинку є система керування освітленням. Автоматизація освітлення дозволяє не лише дистанційно керувати ввімкненням/вимкненням світла, але й регулювати рівень освітленості, створювати сценарії освітлення та інтегрувати систему з іншими розумними пристроями та системами в будинку. Це підвищує зручність використання, енергоефективність та надає додаткові можливості для персоналізації.

Дана кваліфікаційна робота присвячена розробці власної системи автоматизації розумного будинку, яка пропонує вдосконалення існуючих рішень та відкриває нові можливості для поліпшення якості життя.

Актуальність даної роботи полягає в тому, що вона пропонує рішення для автоматизації житлових приміщень, яке враховує сучасні тенденції та потреби користувачів. Запропонована система є гнучкою, масштабованою та легко інтегрується з іншими компонентами розумного будинку.

Мета роботи – покращення роботи системи розумного будинку за рахунок розробки есперимантального макету з декількома режимами роботи в залежності від потреб користувача в режимі реального часу.

Об’єкт розробки – автоматизація процесу регулювання стану розумних модулів.

Предмет розробки – система розумного будинку на базі веб-сервісу.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати існуючі системи розумного будинку;

– проаналізувати методи управління розумним будинком;

– розробити схему макета;

– обрати середу розробки;

– написати програму для автоматизації управління;

– оформити кваліфікаційну роботу згідно ДСТУ 3008:2015 [1], а також з методичними вказівками з підготовки й оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 151 Автоматизація та комп’ютерно-інтегровані технології [2-4].

Робота відповідає дев’ятій цілі сталого розвитку. Промисловість, інновації та інфарструктура.

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

Щоб провести аналіз технічного завдання для системи автоматизації розумного будинку, потрібно детально проаналізувати існуючі системи та їх недоліки. Також потрібно розглянути системи автоматизації компонентів, що саме можна автоматизувати.

1.1 Існуючі системи розумного будинку

Існує чимало систем по всьому світу аналогів розумного будинку. Найпопулярніша із них – вітчизняний виробник Ajax.

Ajax – охоронна сигналізація для квартири, будинку, офісу, яку легко встановити власноруч.

На рисунку 1.1 наведено комплект сигналізації Ajax Starter Kit.

Рисунок 1.1 – Комплект сигналізації Ajax Starter Kit [5]

Далі буде наведено основні компоненти системи Ajax.

Центральний блок. Серце системи, яке об’єднує всі пристрої. Підтримує зв’язок між сенсорами, сиренами та іншими пристроями. Має резервний акумулятор і кілька каналів зв’язку.

Датчики руху. Виявляють рух всередині будинку чи офісу. Моделі можуть мати додаткові функції, наприклад захист від тварин.

Датчики відкриття. Кріпляться на двері та вікна. Виявляють відкриття або злом.

Датчики розбиття скла. Реагують на звук розбиття скла. Зазвичай комбінуються з датчиками руху.

Датчики диму та вуглекислого газу. Виявляють дим або підвищену концентрацію вуглекислого газу. Можуть працювати автономно, сповіщаючи про небезпеку навіть при відсутності зв’язку із центральним блоком.

Датчики протікання води. Розміщуються у місцях можливих протікань. Своєчасно виявляють протікання та запобігають серйозним пошкодженням.

Сирени. Внутрішні та зовнішні. Подають гучний сигнал у випадку тривоги.

Клавіатури та брелоки. Використовуються для керування системою вмикання та вимикання. Брелоки можуть мати кнопки паніки.

Камери відеоспостереження. Можна інтегрувати з системою для отримання відео в режимі реального часу.

Далі буде наведено основні характеристики та переваги використання системи Ajax.

Надійність зв’язку. Використання власного протоколу Jeweller для бездротового зв’язку. Дальність зв’язку до 2000 метрів на відкритому просторі. Захист від глушіння та перехоплення сигналу.

Зручність керування. Мобільні додатки для iOS та Android. Веб-додаток для керування системою з комп’ютера. Можливість віддаленого керування та налаштування.

Інтелектуальні функції, автоматизація сценаріїв. Наприклад, увімкнення освітлення при виявленні руху. Інтеграція з розумним будинком через API та сторонні сервіси, наприклад Google Home, Amazon Alexa.

Безперебійна робота. Резервне живлення на випадок відключення електрики. Підтримка декількох каналів зв'язку GSM, Ethernet.

Масштабованість. Підтримка великої кількості пристроїв. До 100 у хабах початкового рівня і до 150 у більш просунутих моделях.

Установка та налаштування. Легка установка. Більшість пристроїв не потребують прокладки кабелів. Можливість самостійної установки користувачем.

Просте налаштування. Пристрої легко підключаються до хабу через QR-коди. Інтуїтивно зрозумілий інтерфейс додатків для налаштування та керування.

Охоронна система Ajax забезпечує високий рівень безпеки завдяки поєднанню надійності, інтелектуальних функцій та зручності використання. Її гнучкість та масштабованість дозволяють адаптувати систему під різні потреби, що робить її ідеальним рішенням для сучасних розумних будинків та офісів.

1.2 Аналіз недоліків існуючих систем

Існуючі системи розумного будинку, незважаючи на їхні численні переваги, мають певні недоліки та проблеми, які варто враховувати. Далі буде наведено певні проблеми існуючих розумних будинків.

Відсутність стандартизації. Через брак загальноприйнятих стандартів, системи розумного будинку від різних виробників часто несумісні одна з одною. Це ускладнює інтеграцію різних пристроїв та систем, змушуючи користувачів обмежуватися рішеннями одного постачальника.

Висока вартість. Впровадження повноцінної системи розумного будинку може бути досить дорогим, особливо для існуючих будинків, які потребують

значного переобладнання. Високі початкові інвестиції стримують масове поширення цих технологій.

Складність налаштування та використання. Деякі системи розумного будинку можуть бути складними у налаштуванні та використанні, що вимагає від користувачів певних технічних знань та навичок. Це може відлякувати потенційних користувачів, особливо серед людей похилого віку або тих, хто не є технічно підкованим.

Питання конфіденційності та безпеки. Оскільки системи розумного будинку зазвичай підключені до Інтернету, вони можуть бути вразливими до кібератак та порушень конфіденційності. Забезпечення належного рівня безпеки та захисту даних є важливим викликом.

Залежність від Інтернету. Багато функцій розумного будинку залежать від наявності стабільного підключення до Інтернету. Втрата зв'язку може призвести до тимчасової непрацездатності певних функцій або навіть всієї системи.

Проблеми сумісності з існуючою інфраструктурою. Інтеграція систем розумного будинку з існуючими будівельними системами та інфраструктурою, наприклад електрика, водопостачання, опалення тощо може бути складною та дорогою, особливо у старих будинках.

Обмежена підтримка та оновлення. Деякі системи розумного будинку можуть мати обмежену підтримку виробника або нерегулярні оновлення програмного забезпечення, що може зробити їх застарілими з часом.

Проблеми з надійністю та стабільністю. Оскільки системи розумного будинку складаються з багатьох взаємопов'язаних компонентів, вони можуть бути вразливими до збоїв або несправностей, що може призвести до тимчасової непрацездатності.

Енергоспоживання. Хоча системи розумного будинку можуть допомогти зменшити споживання енергії за рахунок ефективного керування освітленням, опаленням та кондиціонуванням, самі вони можуть споживати значну кількість електроенергії, особливо якщо вони постійно підключені та активні.

1.3 Визначення сфер автоматизації в концепції розумного будинку

У цьому розділі наведені різні аспекти життя та функціонування приміщення, які можна автоматизувати та оптимізувати за допомогою рішень розумного будинку. Далі будуть наведені найпопулярніші аспекти.

Керування освітленням є одним з найбільш очевидних і широко впроваджених застосувань технологій розумного будинку. Автоматизація освітлення дозволяє не тільки підвищити рівень зручності й комфорту, а й значно заощадити енергію та кошти на електроспоживання.

Основними функціями автоматизованої системи керування освітленням є автоматичне вмикання та вимикання світла, регулювання яскравості та інтенсивності світла, створення світлових сценаріїв, віддалене керування, інтеграція з іншими системами, енергоефективність.

Датчики руху, датчики присутності та розклад можуть використовуватися для автоматичного ввімкнення освітлення у приміщеннях, коли там хтось знаходиться, і вимкнення, коли приміщення порожнє, уникаючи марного споживання енергії.

Замість бінарного вмикання та вимикання система може плавно регулювати рівень освітлення залежно від денного світла, присутності людей або заздалегідь визначених налаштувань.

Користувачі можуть налаштувати різні сценарії освітлення для різних випадків, таких як приглушене світло, м'яке декоративне світло, яскравіше фокусоване світло тощо. Перемикатися між ними одним натисканням кнопки.

Система розумного будинку дозволяє користувачам керувати освітленням за допомогою смартфону, планшета, голосових команд або централізованого інтерфейсу, навіть коли вони знаходяться поза межами будинку.

Освітлення може бути інтегроване з системами безпеки, наприклад увімкнення світла у разі виявлення руху або тривоги, системами клімат-

контролю, наприклад автоматичне регулювання штор та іншими функціями розумного будинку.

Автоматизоване керування освітленням може значно зменшити споживання електроенергії шляхом вимикання освітлення у непотрібних місцях та адаптації рівня освітлення до фактичних потреб.

Крім того, сучасні системи керування освітленням часто використовують енергоефективні світлодіодні лампи та можуть аналізувати дані про споживання електроенергії для виявлення можливостей подальшої оптимізації.

На рисунку 1.2 наведена розумна багатобарвна Wi-Fi лампа TP-Link Tapo L530E від виробника Samsung з можливістю регулювання яскравості та кольору.

Рисунок 1.2 – Wi-Fi лампа TP-Link Tapo L530E [6]

Автоматизація освітлення є привабливим і відносно простим способом підвищити зручність, безпеку та ефективність використання енергії в розумному будинку, що робить її однією з найпопулярніших функцій для впровадження. Управління кліматом. Автоматичне регулювання опалення, кондиціонування, вентиляції залежно від температури, вологості та присутності людей.

Забезпечення безпеки – одна з найважливіших функцій, яку можна автоматизувати в розумному домі. Інтегровані системи відеоспостереження, датчики руху, контролю доступу, сигналізації та сповіщення про небезпеку дозволяють власникам стежити за своєю оселею цілодобово й отримувати миттєві повідомлення про будь-які потенційні загрози.

Системи відеоспостереження з підключенням до інтернету дають можливість власникам спостерігати за своїм будинком з будь-якої точки світу через смартфон або веб-інтерфейс. Розумні відеокамери можуть виявляти рух, розпізнавати обличчя та навіть аналізувати зображення для виявлення підозрілої діяльності.

Розумні датчики руху можуть бути розміщені в стратегічних місцях будинку для відстеження пересування та виявлення вторгнень. Вони можуть активувати сигналізацію, освітлення чи навіть надсилати сповіщення на ваш смартфон у разі виявлення руху.

На рисунку 1.3 наведено розумну камеру Ajax TurretCam, яка відстежує рух та надсилає повідомленя користувачу.

Рисунок 1.3 – Розумна камера Ajax TurretCam [7]

Системи контролю доступу дозволяють власникам керувати тим, хто має доступ до їхнього будинку. Вони можуть надавати тимчасовий або постійний доступ визначеним особам за допомогою електронних ключів, кодів або біометричних даних, таких як відбитки пальців чи розпізнавання облич.

У разі виявлення потенційної загрози система безпеки може автоматично активувати сигналізацію та відправити сповіщення власникам, сусідам або відповідним службам безпеки.

Розумні сенсори можуть виявляти задимлення, витоки газу, протікання води та інші небезпечні ситуації та вживати заходів для їх усунення чи мінімізації шкоди.

Інтеграція різноманітних систем безпеки під єдиним інтерфейсом керування дозволяє власникам розумного будинку контролювати всі аспекти безпеки зручно та ефективно. Завдяки автоматизації та розумним сповіщенням вони можуть відчувати себе у безпеці незалежно від того, де вони знаходяться, маючи миттєвий доступ до інформації та можливість оперативно реагувати на будь-які інциденти.

Керування мультимедіа. Інтегрована система для управління аудіо, відео, домашнім кінотеатром за допомогою голосових команд або централізованого інтерфейсу.

Автоматизація побутових завдань. Автоматичне поливання рослин, прибирання за допомогою робота-пилососа, керування пральною машиною.

Моніторинг енергоспоживання. Відстеження та аналіз споживання енергії різними пристроями для оптимізації витрат.

Голосове керування. Інтеграція голосових асистентів для управління різними функціями будинку за допомогою голосових команд.

Інтеграція з розумними пристроями. Об'єднання різних розумних гаджетів, наприклад розумні лампочки, розетки, датчики тощо в єдину систему.

Дистанційне керування. Можливість керувати системами будинку з будь-якої точки світу за допомогою смартфона або веб-інтерфейсу.

1.4 Висновки до першого розділу

У даному розділі було проведено аналіз аналогів існуючих систем розумного будинку. Були розібрані позитивні сторони та недоліки аналогів.

Найпопулярніші недоліки аналогів – це висока вартість, незахищеність системи від кібератак, залежність від інтернету та велике енергоспоживання.

Було визначено сфери автоматизації в концепції розумного будинку.

Найпопулярніші сфери автоматизації – це керування освітленням, відстеження руху, керування мультимедіа, автоматизація побутових завдань та моніторинг енергоспоживання.

2 МЕТОДИ КЕРУВАННЯ РОЗУМНИМ БУДИНКОМ

2.1 Методи керування розумним будинком

Існує декілька методів керування розумним будинком:

– автоматичне керування;
– керування за допомогою дистанційного пульта та панелі керування;
– віддалене керування.

Автоматичне керування розумним будинком – це система, яка дозволяє контролювати та керувати різними пристроями та функціями в будинку за допомогою автоматизованих процесів та програмованої логіки. Ось основні аспекти автоматичного керування розумним будинком: центральний контролер або шлюз, датчики та виконавчі пристрої, алгоритми та програмована логіка, сценарії та розклади, віддалений доступ та керування, енергоефективність та безпека.

Центральний контролер або шлюз є головним компонентом системи. Це комп'ютерний пристрій, який збирає дані від різних датчиків та пристроїв у будинку, обробляє їх та відповідно керує виконавчими пристроями, такими як світло, опалення, кондиціонування тощо.

В розумному будинку використовуються різноманітні датчики для збору даних про навколишнє середовище, такі як датчики температури, руху, освітлення, диму. Також є виконавчі пристрої, які керують різними функціями, наприклад, вмикають або вимикають світло, регулюють температуру.

Центральний контролер використовує запрограмовані алгоритми та логіку для автоматичного керування пристроями на основі даних від датчиків та заданих користувачем правил або сценаріїв.

Користувачі можуть створювати власні сценарії та розклади для автоматичного керування різними функціями будинку. Наприклад, можна

встановити розклад для опалення або охолодження приміщень залежно від часу доби чи присутності людей.

Автоматичне керування розумним будинком може сприяти енергоефективності, вимикаючи непотрібні пристрої або регулюючи їх роботу відповідно до потреб.

Автоматичне керування розумним будинком забезпечує зручність, комфорт, енергоефективність та підвищену безпеку, дозволяючи будинку автоматично реагувати та адаптуватися до потреб мешканців та зовнішніх умов.

Найпоширенішими засобами керування будинком є дистанційне керування. Найчастіше використовується пульт дистанційного керування та панель управління.

Пульт керує модулями, приймаючи сигнали від приймопередавача та перетворюючи їх на команди, які передаються через електричну мережу. Приймопередавач може бути реалізований як окремий пристрій або вбудований в інші пристрої. Підключивши приймопередавач до звичайної розетки, можна за допомогою пульта вмикати та вимикати електричніпристрої, регулювати яскравість освітлення та інше. Існують різні методи передачі сигналів пульта: інфрачервоне випромінювання, радіохвилі та Bluetooth.

На рисунку 2.1 наведено типовий пульт для керування розумним будинком ELANHR2.

Рисунок 2.1 – Пульт для керування розумнимбудинком ELANHR2 [8]

Система віддаленого доступу дозволяє керувати будинком з комп'ютера, планшета чи телефону. Для цього потрібно мати сервер системи розумного будинку. Цей сервер підключається до локальної мережі з одного боку та до інформаційної служби керування з іншого. Після цього сервер може перенаправляти команди, які надходять через локальну мережу від мешканця, до керуючих пристроїв будинку.

Значною перевагою такого методу є можливість підключення до системи через інтернет, що дозволяє надсилати команди на сервер через інтернет з телефону чи ноутбука. Це означає, що для керування не обов'язково знаходитись в приміщенні – достатньо лише надіслати команду на сервер через інтернет. Завдяки такої можливості сервер розумного будинку може забезпечувати зворотний зв'язок з мешканцем, наприклад, транслювати в режимі реального часу відео з камер спостереження [9].

2.2 Існуючі аналоги компонентів розумного будинку

Існує чимало аналогів компонентів розумного будинку від різних виробників. Їх можна розділити на центральні керуючі пристрої, датчики руху, розумні освітлювальні прилади, розумні датчики температури та вологості, розумні розетки та вимикачі.

Центральні керуючі пристрої є ключовими компонентами в системах розумного будинку. Вони виконують роль мозку всієї системи, забезпечуючи зв'язок, інтеграцію та керування різними розумними пристроями та датчиками.

Далі наведено аналоги центральних керуючих пристроїв від найпопулярніших виробників:

– apple homekit – фірмовий набір інструментів від Apple для розумного дому;

– google assistant – голосовий помічник від Google, який керує пристроями дому.

На рисунку 2.2 наведено зовнішній вигляд програми Apple HomeKit.

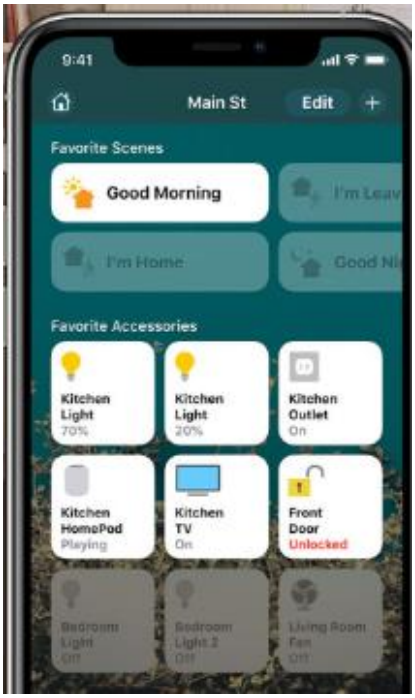

Рисунок 2.2 – Зовнішній вигляд програми Apple HomeKit [10]

Датчики руху є одними з ключових компонентів систем розумного будинку та безпеки. Вони використовуються для виявлення присутності людей або руху в приміщеннях чи на певних ділянках. Далі наведено аналоги датчиків руху від найпопулярніших виробників:

– aqara motion sensors – сумісні з багатьма системами розумного дому;
– smart things motion sensors – від розробника SmartThings;
– philips hue motion sensors – сумісні з системою Philips Hue.

На рисунку 2.3 наведено датчик руху від виробника Aqara Motion Sensors.

Рисунок 2.3 – Датчик руху від виробника Aqara Motion Sensors [11]

Розумні освітлювальні пристрої є одними з найпопулярніших компонентів систем розумного будинку. Вони дозволяють керувати освітленням за допомогою смартфонів, голосових команд або автоматичних сценаріїв Далі наведено аналоги освітлювальних пристроїв від найпопулярніших виробників:

– philips hue – розумні світлодіодні лампи та освітлювальні прилади;
– lifx – розумні світлодіодні лампи та освітлювальні прилади;
– tp-link kasa – різноманітні розумні освітлювальні пристрої.

На рисунку 2.4 наведено розумні світлодіодні лампи від виробника Philips Hue.

Рисунок 2.4 – Розумні лампи від виробника Philips Hue [12]

Розумні датчики температури та вологості відіграють важливу роль у системах розумного будинку, забезпечуючи моніторинг та контроль кліматичних умов всередині приміщень. Далі наведено аналоги датчиків температури від найпопулярніших виробників:

– aqara temperature/humidity sensors;
– smart things sensors;
– ecobee remote sensors.

На рисунку 2.5 наведено розумний датчик температури від виробника Aqara Temperature Sensors.

Рисунок 2.5 – Розумний датчик температури від виробника Aqara Temperature Sensors [13]

Розумні розетки та вимикачі є важливими компонентами систем розумного будинку, які дозволяють керувати підключеними до них пристроями та освітленням за допомогою смартфонів, голосових команд або автоматичних сценаріїв. Далі наведено розумні розетки від найпопулярніших виробників.

– wemo – лінійка розумних розеток та вимикачів від Belkin;
– ajax – лінійка розумних розеток;
– tp-link kasa – розумні розетки та вимикачі.

На рисунку 2.6 наведена розумна вбудована розетка Ajax Outlet Black.

Рисунок 2.6 – Розумна вбудована розетка Ajax Outlet Black [14]

2.3 Вибір компонентів макету та їх опис

У роботі буде використовуватись роутер та бінарний модуль лампочки. Роутер включатиме в себе:

- arduino UNO;
- модуль HW-069 4-розрядний 7-сегментний індикатор дисплей;
- мережний інтернет модуль W5500;
- радіо модуль з антеною NRF24L01 + PA + LNA;
- адаптер до радіо модуля.

Модуль лампочки включатиме в себе:

– arduino Nano;
– радіомодуль NRF24L01;
– резистор 220 Ом;
– світлодіод LED.

Також макет буде містити додаткові компоненти, такі як модуль живлення MB-102 та макетна плата 830 точок типу MB-102.

Arduino UNO – це популярна мікроконтролерна плата на базі мікроконтролера ATmega328P, яка використовується в різноманітних проектах, починаючи від простих навчальних експериментів до складних розробок. Вона має 14 цифрових входів/виходів, 6 аналогових входів, кварцовий резонатор на 16 МГц, USB-роз'єм, роз'єм для живлення та кнопку скидання.

На рисунку 2.7 наведено зображення Arduino UNO.

Рисунок 2.7 – Arduino UNO [15]

Ethernet Shield додає можливість підключення до мережі через Ethernet, що дозволяє Arduino обмінюватися даними з іншими пристроями в мережі або Інтернеті, використовуючи мікросхему W5100 для управління мережевим інтерфейсом.

На рисунку 2.8 наведено Ethernet модуль W5500 для Arduino UNO.

Рисунок 2.8 – Ethernet модуль W5500 для Arduino UNO [15]

Модуль NRF24L01 – це високочастотний радіомодуль для бездротової передачі даних на частоті 2,4 ГГц. Він використовується для зв'язку між різними пристроями на невеликих відстанях. Вбудована антена значно покращує стабільність та дальність зв'язку. Модуль може підтримувати декілька каналів передачі, забезпечуючи швидку і надійну комунікацію між пристроями, такими як сенсори, контролери та інші IoT-пристрої.

На рисунку 2.9 наведено радіо модуль NRF24L01.

Рисунок 2.9 – Радіо модуль NRF24L01 [16]

Адаптер для модуля NRF24L01 забезпечує стабільне живлення та спрощує його підключення до мікроконтролерних плат, таких як Arduino. Він перетворює напругу від 5 В до 3,3 В, що необхідно для правильного функціонування модуля NRF24L01. Адаптер також містить додаткові конденсатори для стабілізації напруги, що допомагає уникнути перешкод і збоїв у роботі модуля.

На рисунку 2.10 наведено адптер до радіо модуля NRF24L01.

Рисунок 2.10 – Адаптер до радіо модуля NRF24L01 [16]

HW-069 – це чотирьох-значний семи-сегментний дисплей, який використовується для відображення числової інформації. Цей дисплей підходить для виведення коду безпеки або іншої важливої інформації під час підключення до пристрою. Він зазвичай підключається до мікроконтролера через інтерфейс I2C або SPI, що дозволяє легко інтегрувати його в систему. Відображення цифр забезпечується за допомогою світлодіодів, що робить його добре видимим навіть при яскравому освітленні.

На рисунку 2.11 наведено модуль HW-069.

Рисунок 2.11 – Модуль HW-069 чотирьох-розрядний семи-сегментний індикатор [17]

Основою макету буде плата для макетування, яка має 830 точок для контактів, її показано на рисунку 2.12. Вона має з’єднані точки по каналах A-E та F-J для кожного номера окремо, ще слід зауважити що плюс та мінус розділені посередині.

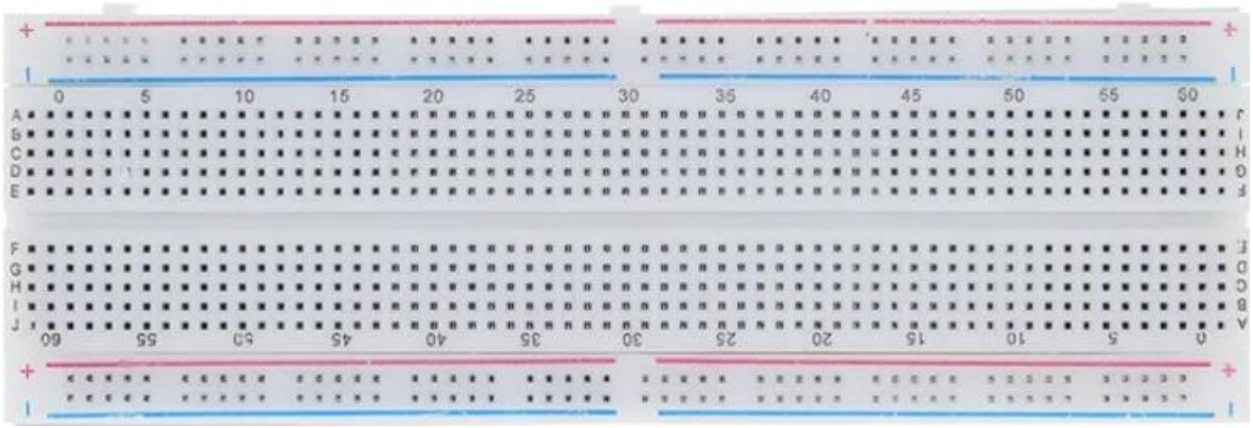

Рисунок 2.12 – Макетна плата на 830 точок типу MB-102 [18]

Модуль живлення забезпечує стабільне і регульоване живлення для мікроконтролера та підключених до нього компонентів. Він фільтрує та

регулює напругу від джерела живлення, щоб видавати стабільну напругу необхідного рівня. Такий модуль Та захищає компоненти від перенапруги. Тому було обрано в якості модуля живлення – модуль живлення MB-102.

На рисунку 2.13 наведено модуль живлення MB-102 для мікроконтролера Arduino UNO.

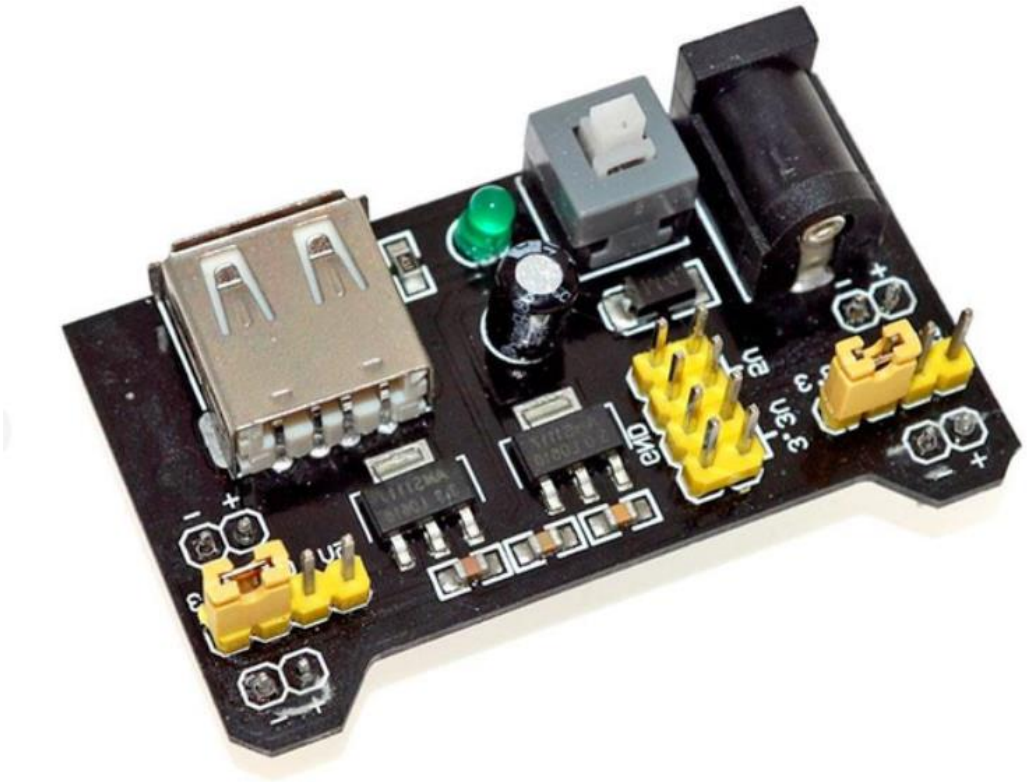

Рисунок 2.13 – Модуль живлення MB-102 для Arduino UNO [18]

Основа модулів, такі як лампочка або реле – плата Arduino Nano.

Arduino Nano – це мала плата на базі мікроконтролера ATmega328P. Вона має подібну функціональність до Arduino UNO, але в меншому форм-факторі. Плата має 14 цифрових входів/виходів, 8 аналогових виходів на працює від 5 В. Плату можна живити через Mini-B USB з’єднання, зовнішнє джерело живлення або батарею.

На відміну від UNO, Nano немає роз’єму для підключення зовнішнього джерела живлення. Програмування відбувається через USB. Дуже компактний

розмір та низьке енергоспоживання роблять її ідеальною для портативних модулів розумного будинку.

На рисунку 2.14 наведено зовнішній вигляд плати Arduino Nano.

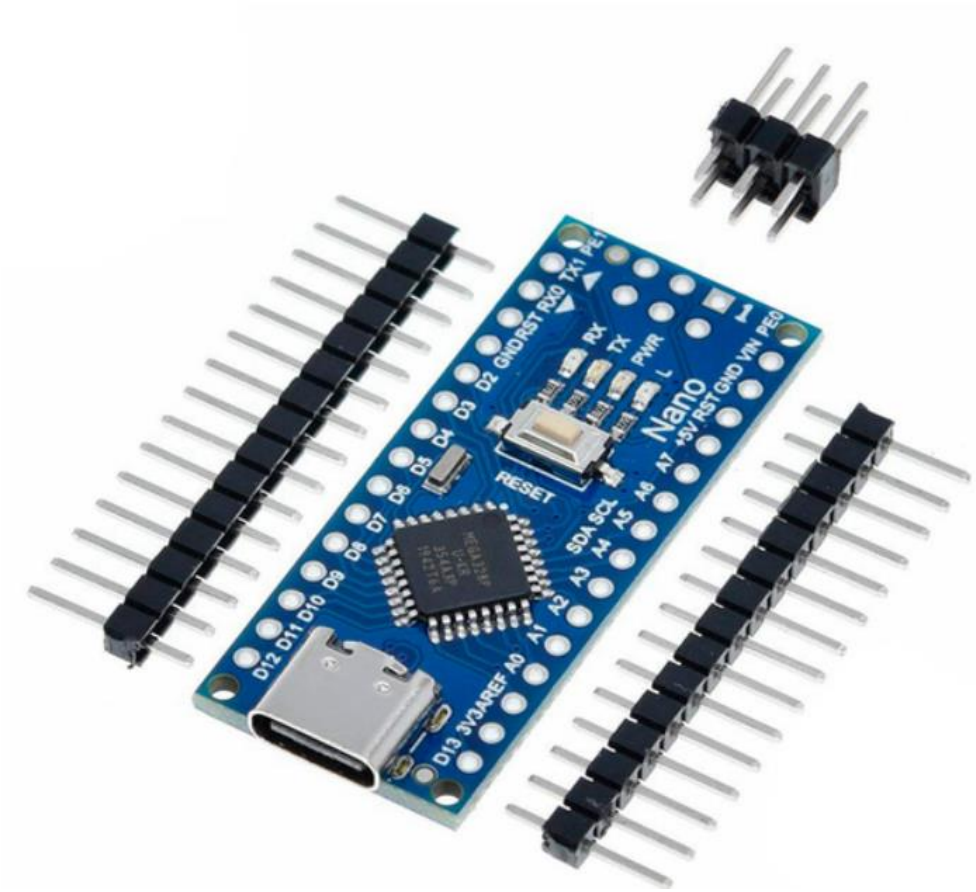

Рисунок 2.14 – Плата Arduino Nano [15]

Резистор – це компонент, який обмежує або контролює потік електричного струму в колі. Номінальний опір складає 220 Ом для протікання струму. Резистори використовують для обмеження струму для захисту компонентів від надмірного струму.

На рисунку 2.15 наведено зовнішній вигляд резистора 220 Ом.

Рисунок 2.15 – Резистор 220 Ом [19]

Діод – це компонент, який пропускає електрични струм лише в одному напрямку. В роботі буде використовуватися світлодіод для модуля лампочки.

На рисунку 2.16 наведено зовнішній вигляд світлодіода.

Рисунок 2.16 – Світлодіод LED [20]

2.4 Складання макету системи

Перед початком складання макету системи розташуємо по центру макетну плату 830 точок типу MB-102. Далі через неї будемо підключати компоненти.

Спочатку зберемо компонент роутера. Для цього необхідно розташувати мікроконтролер Arduino UNO та підключити до нього модуль HW-069 4-розрядний 7-сегментний індикатор дисплей. Далі до модуля інтернет мережі W5500 підключимо кабель мережі.

Для радіомодуля NRF24L01 доєднаємо антену через адаптер для модуля NRF24L01, який забезпечує стабільне живлення та спрощує підключення до мікроконтролерних плат, таких як Arduino.

Після того, як радіомодуль для роутера зібраний, підключимо його до роутера.

Далі зберемо модуль лампочки. Для цього необхідно розташувати мікроконтролер Arduino Nano. Підключимо до мікроконтролера радіомодуль NRF24L01 для прослуховування команд від роутера. Після цього підключимо резистор номіналом у 220 Ом для того, щоб діод не згорів від напруги. Останнім кроком встановимо світлодіод LED на мікроконтролері Arduino Nano.

Останнім кроком буде підключення мікроконтролерів до модуля живлення MB-102.

Зібраний макет системи наведено на рисунку 2.17.

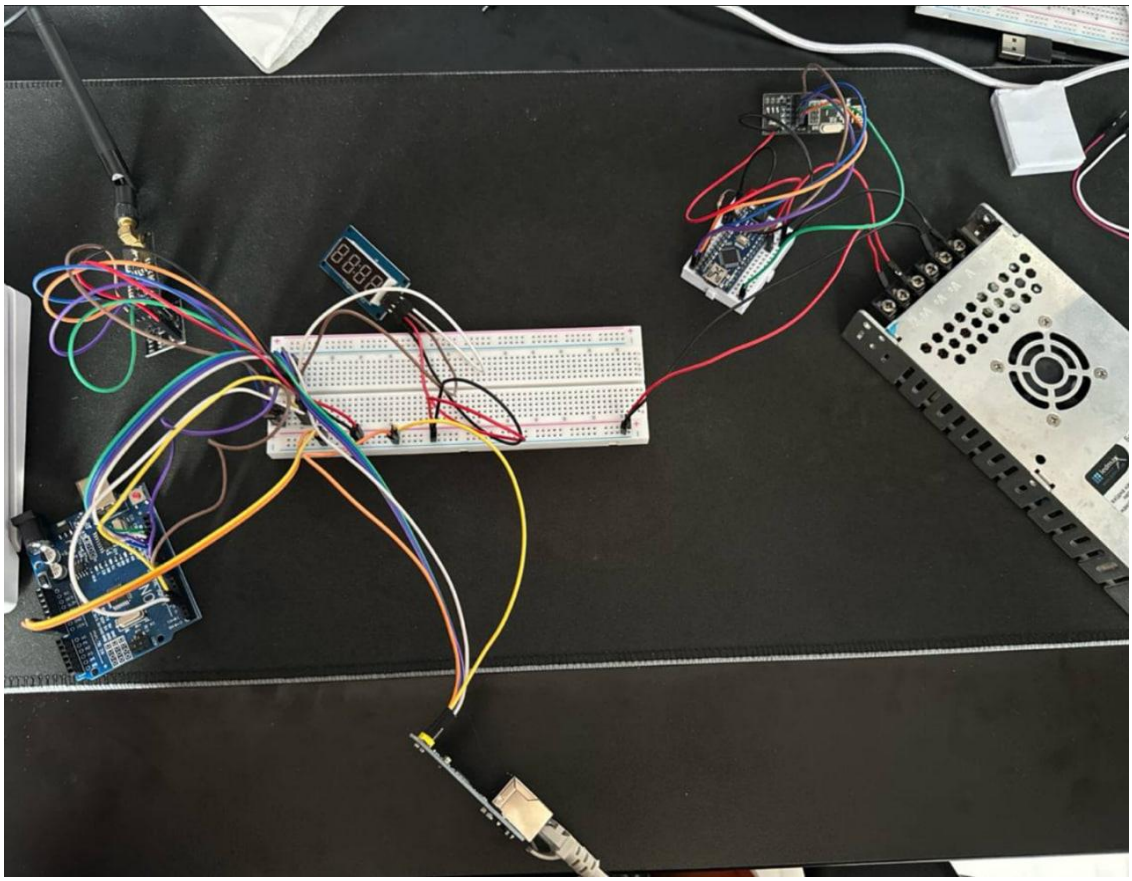

Рисунок 2.17 – Зібраний макет системи

2.5 Висновки до другого розділу

У даному розділі було проаналізовано існуючі аналоги компонентів розумного будинку. Було проведено пошук методів керування розумним будинком.

Найпопулярніші методи керування розумним будинком – автоматичне керування, керування за допомогою дистанційного пульта, віддалене керування.

У даному розділі було обрано необхідні компоненти для системи. На основі обраних компонентів було складено макет системи.

3 РОЗРОБКА СИСТЕМИ

3.1 Вибір середи розробки

В якості середи розробки для серверної частини було обрано Rider.

Rider – це кросплатформне інтегроване середовище розробки (Integrated Development Environment) від компанії JetBrains, призначене для роботи з різними технологіями .NET.

Rider – це комфортна для розробка середи, тому що по-перше абезпечує інтелектуальне автодоповнення коду, рефакторинг, навігацію по коду, підсвічування синтаксису та інші зручні функції для підвищення продуктивності розробників. По-друге, Rider має вбудовану систему відстеження помилок та налагодження для полегшення процесу знаходження та виправлення помилок у коді.

На рисунку 3.1 наведено зовнішній вигляд програми Rider.

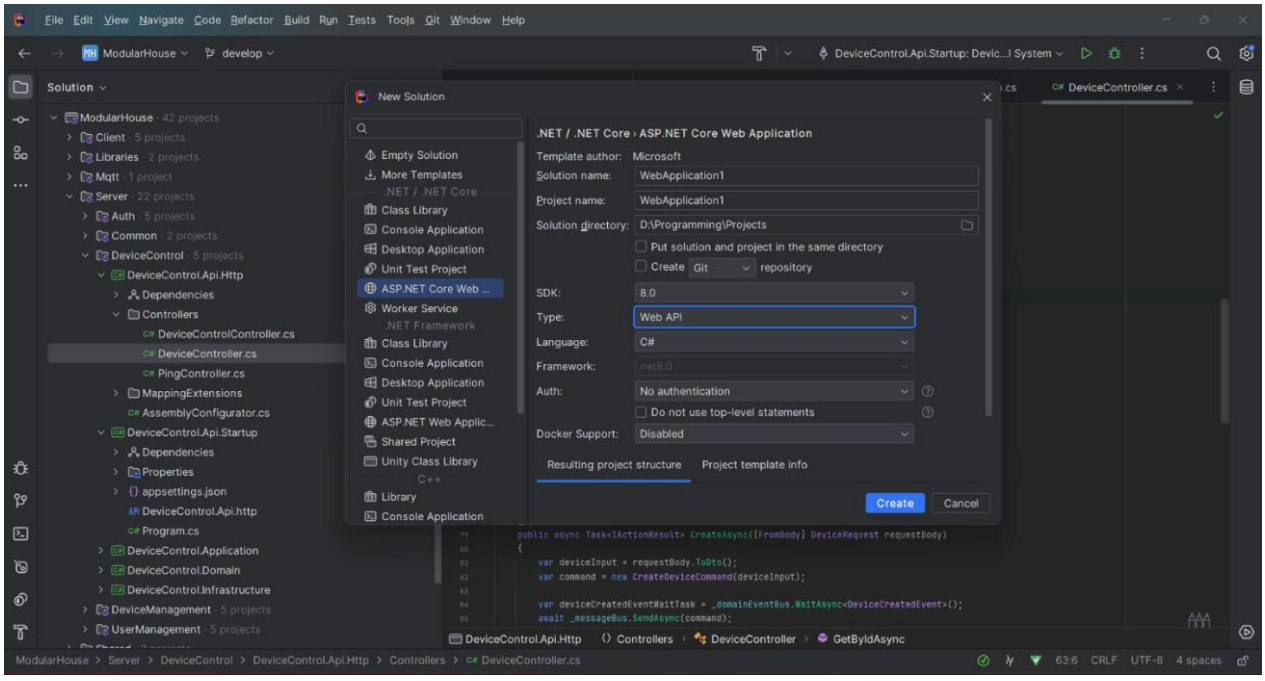

Рисунок 3.1 – Зовнішній вигляд Rider IDE

У якості середи розробки для апаратної частини було обрано Arduino IDE.

Arduino IDE – це крос-платформне середовище розробки, призначене для написання програмного коду, компіляції та завантаження його на плати Arduino. Редактор коду забезпечує середовище для написання, редагування та форматування програмного коду. Підтримує підсвічування синтаксису та автодоповнення для полегшення процесу програмування. Компілятор коду компілює код, написаний на мові програмування Arduino (базується на C/C++), у машинний код, який може бути завантажений на плату Arduino.

На рисунку 3.2 зображено зовнішній вигляд логотипу Arduino IDE.

Рисунок 3.2 – Логотип Arduino IDE

3.2 Розробка коду для мікроконтролера Arduino UNO

Спочатку потрібно загрузити бібліотеки для компонентів макету та для веб серверу через який буде спілкуватися з програмою управління системою,

ініціалізувати піни, створити об'єкти для керування дисплеєм, модулями та веб-сервером. Далі розроблюється блок-схема роботи мікроконтролера.

На рисунку 3.3 наведено блок-схему роботи мікроконтролера.

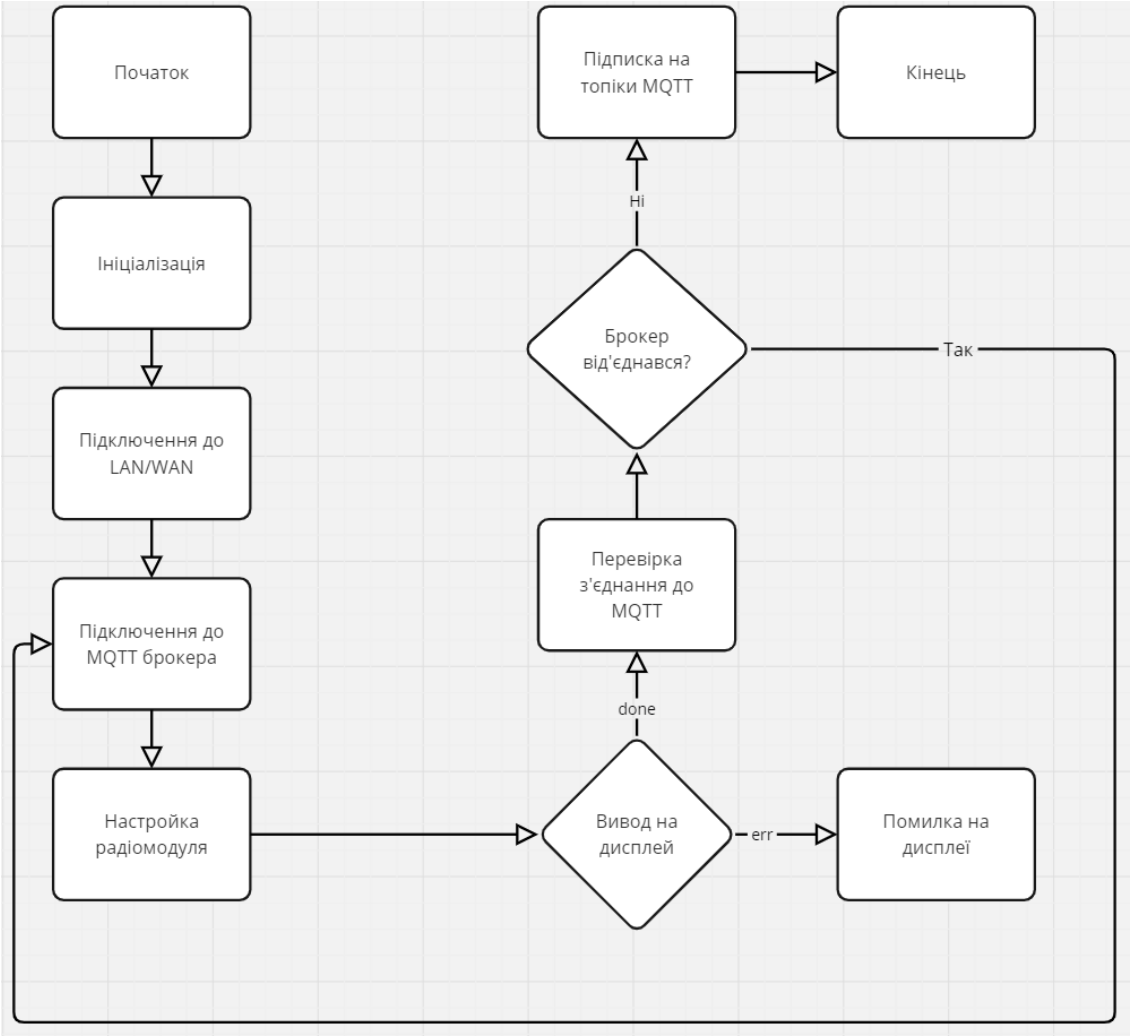

Рисунок 3.3 – Блок-схема роботи мікроконтролера

Після розробки блок-схеми потрібно написати код. Написаний код роутера та модуля лампочки описано далі. Повна версія коду наведена у додатку А.

Ініціалізація роутера розпочинається із підключення необхідних залежностей, бібліотек та визначення констант.

На рисунку 3.4 наведено визначення констант та підключення бібліотек роутера.

```
#define SERIAL_PORT 115200

#define PRIVATE_ID "rtr01"

#define MQTT_SERVER_ADDRESS "192.168.50.11"
#define MQTT_SERVER_PORT 1883
#define MQTT_CLIENT_ID "arduino-router-1"
#define MQTT_AUTH_USERNAME "UserName"
#define MQTT_AUTH_PASSWORD "Password"

#define MQTT_ENDPOINT_VERITY "verify"
#define MQTT_ENDPOINT_CONTROL "control"

#define RF24_CE_PIN 7
#define RF24_CSN_PIN 8

#define RADIO_PAYLOAD_SIZE 32
#define RADIO_SEND_MAX_RETRY_COUNT 5

#define DISPLAY_CLK_PIN 2
#define DISPLAY_DIO_PIN 3


#include <SPI.h>
#include <Ethernet.h>
#include <PubSubClient.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <printf.h>
#include <TM1637Display.h>
```

Рисунок 3.4 – Ініціалізація роутера.

Далі визначаються змінні.

На рисунку 3.5 наведено визначення змінних файлу router.ino.

```
33  // Ethernet settings
34  byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };
35  char mqttServer[] = MQTT_SERVER_ADDRESS;
36  int mqttPort = MQTT_SERVER_PORT;
37  EthernetClient ethClient;
38  PubSubClient mqttClient(ethClient);
39
40  // NRF24L01 settings
41  RF24 radio(RF24_CE_PIN, RF24_CSN_PIN);
42
43  // Display settings
44  /*
45     --A--
46    |     |
47    F     B
48    |     |
49     --G--
50    |     |
51    E     C
52    |     |
53     --D--
54  */
55  TM1637Display display(DISPLAY_CLK_PIN, DISPLAY_DIO_PIN);
56  uint8_t d_char = SEG_B | SEG_C | SEG_D | SEG_E | SEG_G;
57  uint8_t e_char = SEG_A | SEG_D | SEG_E | SEG_F | SEG_G;
58  uint8_t i_char = SEG_C;
59  uint8_t n_char = SEG_C | SEG_E | SEG_G;
60  uint8_t o_char = SEG_C | SEG_D | SEG_E | SEG_G;
61  uint8_t r_char = SEG_E | SEG_G;
62  uint8_t t_char = SEG_D | SEG_E | SEG_F | SEG_G;
63
64  // Global variables
65  char verifyMqttTopic[sizeof(MQTT_ENDPOINT_VERITY) + sizeof(PRIVATE_ID) + 2] {};
66  char controlMqttTopic[sizeof(MQTT_ENDPOINT_CONTROL) + sizeof(PRIVATE_ID) + 2] {};
67  bool isError = false;
68  unsigned long displayClearTime = 0;
```

Рисунок 3.5 – Визначення змінних файлу router.ino

Після підключення усіх необхідних залежностей та визначення змінних у файлі визначаються методи.

На рисунку 3.6 наведено порядок виклику методів у файлі router.ino.

```
71    void setup();
72    void loop();
73    bool setupEthernet();
74    void setupMqtt();
75    void ensureConnectMQTT();
76    void mqttCallback(char* topic, uint8_t* payload, unsigned int length);
77    void handleControlCallback(uint8_t* payload, unsigned int length);
78    void handleVerifyCallback(uint8_t* payload, unsigned int length);
79    bool setupRadio();
80    void sendMessageToModule(char* moduleId, uint8_t* data, uint8_t dataLength);
81    void sendRadioMessage(uint8_t* pipeAddress, uint8_t* payload, uint8_t payloadLength);
82    void setupDisplay();
83    void showInitText();
84    void showDoneText();
85    void showErrorText();
86    void showCodeOnDisplay(unsigned int code);
```

Рисунок 3.6 – Визначення методів у файлі router.ino

Функція setup() викликається один раз під час запуску чи перезавантаження Arduino. У цій функції відбувається налаштування всіх модулів. Також ініціалізація змінних та підключення залежностей.

На рисунку 3.7 наведено код методу setup().

```
89  void setup() {
90    Serial.begin(SERIAL_PORT);
91    printf_begin();
92
93    setupDisplay();
94    showInitText();
95
96    bool setupEthernetResult = setupEthernet();
97    bool setupRadioResult = setupRadio();
98
99    if (!setupEthernetResult || !setupRadioResult) {
100     isError = true;
101     showErrorText();
102     return;
103   }
104
105   setupMqtt();
106   showDoneText();
107 }
```

Рисунок 3.7 – Код методу setup()

Після виклику методу Serial.begin(SERIAL_PORT) викликається метод налаштування дисплею setupDisplay(). Дисплей не потребує додаткових налаштувань, він готовий до використання відразу після ініціалізації.

На рисунку 3.8 наведено код методу setupDisplay().

```
264     void setupDisplay() {
265       display.setBrightness(0x0f);
266       display.clear();
267     }
```

Рисунок 3.8 – Код методу setupDisplay()

Після налаштування дисплею викликається метод showInitText(), який виводить на дисплей поточний стан програми – ініціалізування (INIT).

На рисунку 3.9 наведено код методу showInitText().

```
270   void showInitText() {
271     uint8_t data[] { i_char, n_char, i_char, t_char };
272     display.clear();
273     display.setSegments(data);
274   }
```

Рисунок 3.9 – Код методу showInitText()

Далі відбувається настройка модуля Ethernet – Arduino намагається підключитися до локальної мережі LAN, а потім і до глобальної мережі WAN через Ethernet–кабель, реєструючи MAC-адресу маршрутизатора.

На рисунку 3.10 наведено код методу setupEthernet().

```
125  bool setupEthernet() {
126    for (byte attempt = 1; attempt <= 5; attempt++) {
127      if (Ethernet.begin(mac) == 1) {
128        printf("Ethernet init success\n");
129        return true;
130      }
131      printf("Ethernet init error. Retry %i\n", attempt);
132      delay(1000);
133    }
134    printf("Ethernet init error. Abort\n");
135    return false;
136  }
```

Рисунок 3.10 – Код методу setupEthernet()

Далі відбувається налаштування радіо модуля викликом методу setupRadio().

На рисунку 3.11 наведено код методу setupRadio().

```
225  bool setupRadio() {
226    if (!radio.begin()) {
227      printf("RF24 init fail. Abort");
228      return false;
229    }
230
231    radio.setAutoAck(1);
232    radio.setPayloadSize(RADIO_PAYLOAD_SIZE);
233    radio.setChannel(0x60);
234    radio.setPALevel(RF24_PA_LOW);
235    radio.setDataRate(RF24_1MBPS);
236    radio.powerUp();
237    radio.stopListening();
238
239    printf("RF24 init success\n\n");
240    radio.printDetails();
241    printf("\n");
242    radio.printPrettyDetails();
243    printf("\n");
244    return true;
245  }
```

Рисунок 3.11 – Код методу setupRadio()

Якщо метод setupEthernet() або setupRadio() поверне false, то далі буде виведено на дисплей роутера повідомлення з помилкою (err).

На рисунку 3.12 наведено код метода showErrorText().

```
285  void showErrorText() {
286    uint8_t data[] { 0, e_char, r_char, r_char };
287    display.clear();
288    display.setSegments(data);
289  }
```

Рисунок 3.12 – Код методу showErrorText()

Далі Arduino намагається підключитися до MQTT-брокера за вказаними адресою, портом, логіном та паролем.

На рисунку 3.13 наведено код методу setupMqtt().

```
139  void setupMqtt() {
140    strcat(verifyMqttTopic, MQTT_ENDPOINT_VERITY);
141    strcat(verifyMqttTopic, "/");
142    strcat(verifyMqttTopic, PRIVATE_ID);
143
144    strcat(controlMqttTopic, MQTT_ENDPOINT_CONTROL);
145    strcat(controlMqttTopic, "/");
146    strcat(controlMqttTopic, PRIVATE_ID);
147
148    mqttClient.setServer(mqttServer, mqttPort);
149    mqttClient.setCallback(mqttCallback);
150    ensureConnectMQTT();
151  }
```

Рисунок 3.13 – Код методу setupMqtt()

Після цього Arduino UNO переконується чи було з’єднання з MQTT успішним через виклик методу ensureConnectMQTT().

На рисунку 3.14 наведено код методу ensureConnectMQTT().

```
154  void ensureConnectMQTT() {
155    while (!mqttClient.connected()) {
156      if (mqttClient.connect(MQTT_CLIENT_ID, MQTT_AUTH_USERNAME, MQTT_AUTH_PASSWORD)) {
157        printf("MQTT Broker (re)connect success\n");
158
159        if (!mqttClient.subscribe(verifyMqttTopic))
160          printf("Fail subscribe. MQTT topic \"%s\"\n", verifyMqttTopic);
161        else
162          printf("Success subscribe. MQTT topic \"%s\"\n", verifyMqttTopic);
163
164        if (!mqttClient.subscribe(controlMqttTopic))
165          printf("Fail subscribe. MQTT topic \"%s\"\n", controlMqttTopic);
166        else
167          printf("Success subscribe. MQTT topic \"%s\"\n", controlMqttTopic);
168      } else {
169        printf("MQTT Broker connection error, rc=%d\n", mqttClient.state());
170        delay(2000);
171      }
172    }
173  }
```

Рисунок 3.14 – Код методу ensureConnectMQTT()

Якщо усі методи були успішно виконані, то на дисплей виводиться повідомлення з успішним завершенням методу setup() – done.

На рисунку 3.15 наведено код методу showDoneText().

```
277  void showDoneText() {
278    uint8_t data[] { d_char, o_char, n_char, e_char };
279    display.clear();
280    display.setSegments(data);
281    displayClearTime = millis() + 5000;
282  }
```

Рисунок 3.15 – Код методу showDoneText()

Після виклику метода setup() викликається циклічний метод loop(), де спочатку перевіряється підключення до MQTT-брокера викликом метода ensureConnectMQTT(). Якщо з'єднання втрачено, воно відновлюється, і Arduino підписується на необхідні топіки "control/{privateId}" та

"verify/{privateId}", де privateId – це унікальна константа, вбудована в кожний пристрій.

На рисунку 3.16 наведено код метода loop().

```
110  void loop() {
111    if (isError) {
112      return;
113    }
114
115    if (displayClearTime != 0 && millis() >= displayClearTime) {
116      display.clear();
117      displayClearTime = 0;
118    }
119
120    ensureConnectMQTT();
121    mqttClient.loop();
122  }
```

Рисунок 3.16 – Код метода loop()

Використовуючи бібліотеку радіомодуля, коли отримується повідомлення від MQTT, то викликається функція mqttCallback(). У цій функції перевіряється, який конкретний топік було використано.

На рисунку 3.17 наведено код метода mqttCallback().

```
176  void mqttCallback(char* topic, byte* payload, unsigned int length) {
177    printf("\nMsg on MQTT Topic \"%s\"\n", topic);
178
179    if (strcmp(topic, verifyMqttTopic) == 0)
180      handleVerifyCallback(payload, length);
181    else if (strcmp(topic, controlMqttTopic) == 0)
182      handleControlCallback(payload, length);
183  }
```

Рисунок 3.17 – Код метода mqttCallback()

Якщо топік verify – то на дисплеї виводяться необхідні цифри для верифікації підключення модуля.

На рисунку 3.18 наведено код метода handleVerifyCallback().

```
186  void handleVerifyCallback(uint8_t* payload, unsigned int length) {
187    if (length != 4) {
188      printf("Verify MQTT msg invalid. Payload size != 4. Abort\n");
189      return;
190    }
191
192    unsigned int code = 0;
193    for (byte i = 0; i < 4; i++) {
194      int symbolAsInt = payload[i] - '0';
195      if (symbolAsInt < 0 || symbolAsInt > 9) {
196        printf("Varify MQTT msg invalid. Value is not digit: %i. Abort\n", payload[i]);
197        return;
198      }
199      code = code * 10 + symbolAsInt;
200    }
201
202    showCodeOnDisplay(code);
203  }
```

Рисунок 3.18 – Код метода handleVerifyCallback()

Для топіку verify, payload складається з 4 байтів. Кожен байт відповідає символу ASCII, що дозволяє використовувати не лише цифри, а й літери.

Кожен байт розшифровується, і, якщо він виявляється цифрою, конвертується у відповідний код з 4 цифр і відображається на дисплеї для верифікації підключення модуля викликом метода showCodeOnDisplay().

На рисунку 3.19 наведено код метода showCodeOnDisplay().

```
292  void showCodeOnDisplay(unsigned int code) {
293    if (code / 10000 > 0) {
294      printf("Code %i contains > 4 digits. Abort\n", code);
295      return;
296    }
297
298    display.clear();
299    display.showNumberDec(code, true);
300    displayClearTime = millis() + 60000;
301  }
```

Рисунок 3.19 – Код метода showCodeOnDisplay()

Якщо топік control – то відправляється повідомлення через радіомодуль на відповідний модуль у форматі mqtt, де вказується topic – тема повідомлення, та тіло повідомлення payload, яке створено у форматі масиву байт.

На рисунку 3.20 наведено код метода handleVerifyCallback().

```
206  void handleControlCallback(uint8_t* payload, unsigned int length) {
207    if (length < 6) {
208      printf("Control MQTT msg invalid. Payload size < 6. Abort\n");
209      return;
210    }
211
212    // pipeAddress is the moduleId
213    uint8_t pipeAddress[6] {};
214    unsigned int dataLength = length - 5;
215    uint8_t data[dataLength]{};
216
217    memcpy(pipeAddress, payload, 5);
218    pipeAddress[5] = '\0';
219    memcpy(data, payload + 5, dataLength);
220
221    sendRadioMessage(pipeAddress, data, dataLength);
222  }
```

Рисунок 3.20 – Код метода handleVerifyCallback()

Для топіку control, payload складається мінімум з 6 байтів. Перші 5 байтів – це privateId модуля, де кожен символ представлений у форматі ASCII. Тобто на сервері зберігається рядок, ідентичний тому, що "вбудований" у модуль. Цей рядок за допомогою кодування ASCII конвертується у масив байтів, де кожен байт відповідає символу.

Решта байтів, починаючи з 6-го включно, містять дані, які передаються на модуль по радіоканалу.

В димпломній роботі використовується лише один тип модулів – binary. Отже, і лампочка, і реле можуть мати значення true або false. Цей стан може бути записаний в один байт.

Отже, для керування модуля лампочки, треба відіслати повідомлення на топік control з payload у розмірі 6 байт, де перші 5 байтів – це ідентифікатор модуля, а шостий байт – байт "стану". Цей 1 байт передається по радіоканалу на потрібний модуль. Модуль його приймає і обробляє:

– якщо значення дорівнює 0 – вимикає пристрій;
– якщо значення дорівнює 1 – вмикає пристрій;
– якщо надійшло інше значення – ігнорується.

Радіоповідомлення відправляється завдяки виклику метода sendRadioMessage().

На рисунку 3.21 наведено код метода sendRadioMessage().

Далі буде приведено та описано код модуля лампочки module-single-led.ino.

Із початку визначаються змінні, підлючаються залежності та бібліотеки у файлі module-single-led.ino.

На рисунку 3.22 наведено код іціціалізації фалу module-single-led.ino.

```
248  void sendRadioMessage(uint8_t* pipeAddress, uint8_t* payload, uint8_t payloadLength) {
249    radio.flush_tx();
250    radio.openWritingPipe(pipeAddress);
251
252    for (byte i = 0; i < RADIO_SEND_MAX_RETRY_COUNT; i++) {
253      bool success = radio.write(payload, payloadLength);
254      if (success) {
255        printf("RF24 msg sent\n");
256        return;
257      }
258      printf("Error on RF24 msg. Retry %i\n", i + 1);
259    }
260    printf("Error on RF24 msg. Abort\n");
261  }
262
263
264  void setupDisplay() {
265    display.setBrightness(0x0f);
266    display.clear();
267  }
```

Рисунок 3.21 – Код метода sendRadioMessage()

```
1   #define SERIAL_PORT 115200
2
3   #define PRIVATE_ID "mdl01"
4
5   #define RF24_CE_PIN 9
6   #define RF24_CSN_PIN 10
7
8   #define RADIO_PAYLOAD_SIZE 32
9
10  #define LED_PIN 6
11
12
13  #include <SPI.h>
14  #include <nRF24L01.h>
15  #include <RF24.h>
16  #include <printf.h>
17
18
19  // NRF24L01 settings
20  RF24 radio(RF24_CE_PIN, RF24_CSN_PIN);
```

Рисунок 3.22 – Іцініалізація файлу module-single-led.ino

Далі визначаються методи файлу module-single-led.ino.

На рисунку 3.23 наведено визначення методів у файлі module-single-led.ino.

```
23      void setup();
24      void loop();
25      void startRadio();
26      void readRadio();
27      void setupLed();
28      void toggleLed(bool isOn);
```

Рисунок 3.23 – Визначення методів у файлі module-signle-led.ino

Із початку викликається метод setup(), в якому ініціалізується світлодіод LED та радіо модуль.

На рисунку 3.24 наведено код методу setup().

```
31  void setup() {
32    Serial.begin(SERIAL_PORT);
33    printf_begin();
34
35    setupLed();
36    startRadio();
37  }
```

Рисунок 3.24 – Код методу setup()

У методі setupLed() налаштовується цифровий вивід діоду.

На рисунку 3.25 наведено код методу setupLed()

```
93      void setupLed() {
94        pinMode(LED_PIN, OUTPUT);
95      }
```

Рисунок 3.25 – Код методу setupLed()

Далі викликається метод startRadio(), в якому відбувається налаштування для запуску радіо модуля, який в свою чергу далі буде приймати по радіо каналу повідомлення від роутера.

На рисунку 3.26 наведено код метода startRadio().

```
45  void startRadio() {
46    if (!radio.begin()) {
47      printf("RF24 init fail. Abort");
48      while (true) {
49        delay(1);
50      }
51    }
52
53    radio.setAutoAck(1);
54    radio.setPayloadSize(RADIO_PAYLOAD_SIZE);
55
56    uint8_t pipeAddress[6] = PRIVATE_ID;
57    radio.openReadingPipe(0, pipeAddress);
58
59    radio.setChannel(0x60);
60    radio.setPALevel(RF24_PA_LOW);
61    radio.setDataRate(RF24_1MBPS);
62
63    radio.powerUp();
64    radio.startListening();
65
66    printf("RF24 init success\n\n");
67    radio.printDetails();
68    printf("\n");
69    radio.printPrettyDetails();
70  }
```

Рисунок 3.26 – Код метода startRadio()

Після цього кроку завершується метод setup(), і далі викликається циклічний метод loop(). Цей метод викликає метод readRadio(), який в совю чергу чекає на повідомлення та у разі отримання повідомлення, обробляє його.

На рисунку 3.27 наведено код метода loop().

```
40      void loop() {
41        readRadio();
42      }
```

Рисунок 3.27 – Код метода loop()

Метод readRadio() отримує по радіоканалу повідомлення та обробляє його.

На рисунку 3.28 наведено код метода readRadio().

```
73  void readRadio() {
74    if (!radio.available()) {
75      return;
76    }
77
78    uint8_t receivedData[RADIO_PAYLOAD_SIZE + 1];
79    radio.read(&receivedData, RADIO_PAYLOAD_SIZE);
80
81    Serial.print("Received data: ");
82    for (byte i = 0; i < RADIO_PAYLOAD_SIZE; i++) {
83      Serial.print(receivedData[i]);
84      Serial.print(" ");
85    }
86    Serial.println();
87
88    bool isOn = receivedData[0] == 1;
89    toggleLed(isOn);
90  }
```

Рисунок 3.28 – Код метода readRadio()

Із повідомлення отримується команда увімкнення або вимикання діода, і далі викликається метод toggleLed(isOn), в який передається команда.

На рисунку 3.29 наведено код метода toggleLed().

```
98  void toggleLed(bool isOn) {
99    if (isOn)
100     digitalWrite(LED_PIN, HIGH);
101   else
102     digitalWrite(LED_PIN, LOW);
103 }
```

Рисунок 3.29 – Код метода toggleLed()

Якщо передано true, то діод вмикається, у іншому разі діод вимикається.

3.3 Розробка серверної частини та управління системою

Для системи потрібна керуюча програма, за допомогою якої можна буде легко контролювати станом модуля лампочки. Програма повинна мати простий та зрозумілий інтерфейс. У якості інтерфейсу програми використовується API, який надає усі необхідні контролери та ендпоінти для взаємодії із серверною частиною.

Серверна частина розроблена на мові C# за допомогою фреймворку .NET версії 8.0.

У якості інтерфейсу серверної частини, яка взаємодіє із апаратною було створено мікросервис DeviceControlSystem, який надає у якості інтерфейсу API для взаємодії між клієнтом та сервером. API містить у собі контролери, саме DeviceController.cs та DeviceControlController.cs.

Контролер DeviceControler.cs створений для керування модулями. Завдяки ньому можна взаємодіяти із серверною частиною та отримувати, створювати, оновляти та видаляти модулі.

Контролер DeviceControlController.cs створений для керування станом модулів. Завдяки ньому можна вмикати або вимикати бінарні модулі, які доєднані до роутера.

Також у серверній частині використовується ще один мікросервис на мові C# за допомогою фреймворку .NET версії 8.0 Mqtt.Broker.

Мікросервіс Mqtt.Broker – це брокер повідомлень для взаємодії серверною частиною із роутером, на базі механізму публікації/підписки.

Брокер містить у собі 2 теми для взаємодії з апаратною частиною: control та verify. Control відповідає за публікацію повідомлень для контролю станом модуля, а verify – для публікації повідомлень, пов’язаних із верифікацією під час додавання модуля до роутера.

Код контролерів приведено у додатку Б.

3.4 Проведення експерименту системи

Взаємодія із компонентами системи проходить завдяки спілкуванню серверної частини та роутера. Спілкування із роутером на себе бере брокер повідомлень Mqtt на стороні серверної частини, тож для взаємодії із компонентами системи необхідно взаємодіяти із серверною частиною.

Взаємодія із серверною частиною реалізована завдяки інтерфейсу серверної частини API. HTTP запити будемо відправляти за допомогою утіліти Swagger.

Для початку експерименту необхідно запустити апаратну частину. Після вмикання системи, на дисплей виводиться повідомленя стану системи. Наразі система у стані ініціалізації тому буде виведено init.

На рисунку 3.30 наведено поточний стан системи.

Рисунок 3.30 – Стан ініціалізації системи

Якщо під час ініціалізції виникає помилка, то на дисплей роутера виводиться повідомлення помилки Err.

На рисунку 3.31 зображено вигляд дисплею, де під час ініціалізації роутера виникла помилка.

Рисунок 3.31 – Стан помилки під час ініціалізації Err

Якщо помилки не виникає, то на дисплей роутера виводиться done.

На рисунку 3.32 зображено вдалий стан ініціалізування роутера.

Рисунок 3.32 – Вдале ініціалізування роутера

В якості експерименту проведемо вмикання та вимикання модуля лампочки. Для реалізації експерименту необхідно створити роутер та модуль лампочки. Взаємодія буде із мікросервісом DeviceControlSystem.

Для створення роутера необхідно надіслати HTTP POST запит, URI якого "api/devices".

На рисунку 3.33 наведено запит на створення роутера.

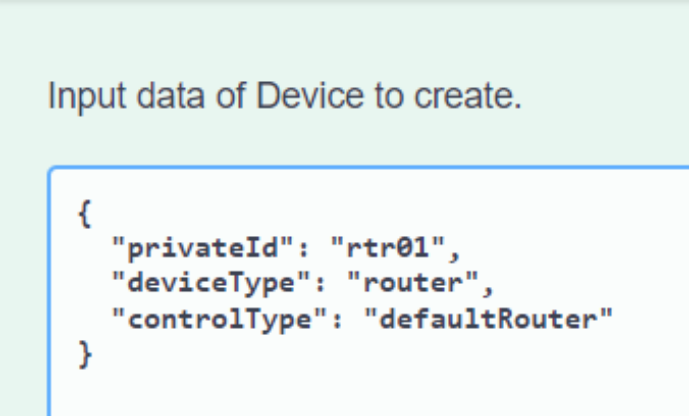

Рисунок 3.33 – Запит на створення роутера

Після відправки запиту отримаємо результат. Результат запиту повернув код 200, що позначає успіх операції. Також повернувся унікальний ідентифікатор id роутера.

Результат заптиту створення роутера наведено на рисунку 3.34.

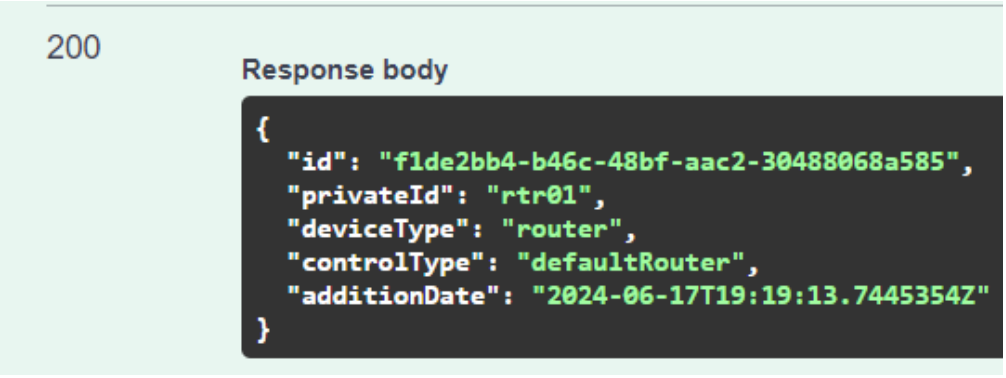

Рисунок 3.34 – Результат запиту на створення роутера

Далі аналогічно створюємо модуль лампочки. Відсилаємо аналогічний запит, але змінимо тіло запиту.

Запит на створення модуля лампочки наведено на рисунку 3.35.

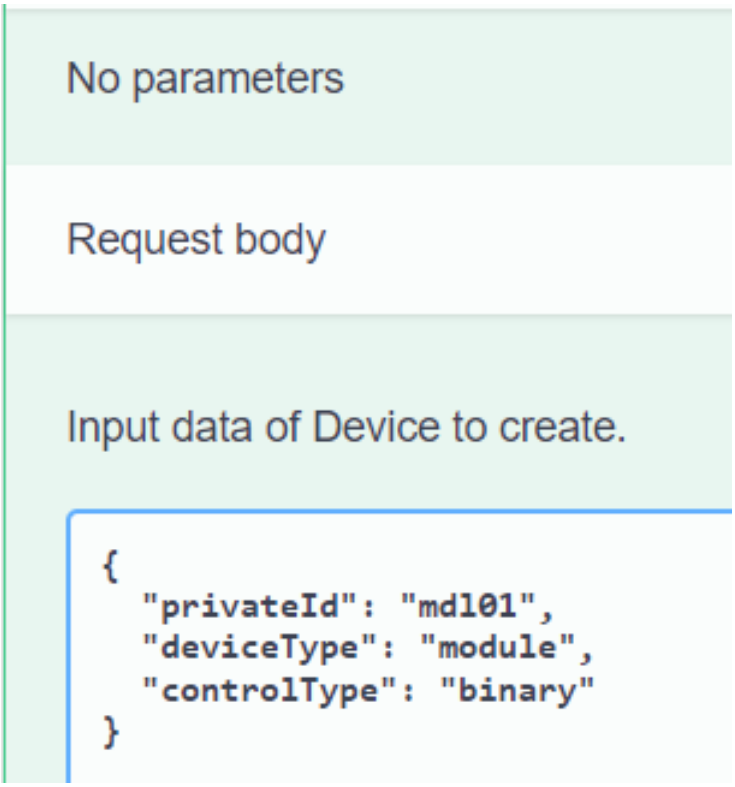

Рисунок 3.35 – Запит на створення модуля лампочки

В результаті отримаємо id модуля лампочки, який можна використоувати для подальшого маніпулювання цим модулем через роутер.

Результат запиту на створення модуля лампочки наведено на рисунку 3.36.

Code | Details

200

Response body


```
{
  "id": "b3fb661e-34b4-4d03-a852-8d51bc3e719b",
  "privateId": "md101",
  "deviceType": "module",
  "controlType": "binary",
  "additionDate": "2024-06-17T19:21:44.8801049Z"
}
```


Рисунок 3.36 – Результат запиту на створення модуля лампочки

Для того, щоб переконатися, що модулі дійсно створилися, надішлемо запит на сервер для отримання усіх існуючих модулів HTTP GET, URI якого "api/devices". Сервер надає відповідь у вигляді масиву із двох модулів.

Результат запиту наведено на рисунку 3.37.

Code | Details

200

Response body


```
{
  "list": [
    {
      "id": "f1de2bb4-b46c-48bf-aac2-30488068a585",
      "privateId": "rtr01",
      "deviceType": "router",
      "controlType": "defaultRouter",
      "additionDate": "2024-06-17T19:27:42.9320429Z"
    },
    {
      "id": "b3fb661e-34b4-4d03-a852-8d51bc3e719b",
      "privateId": "md101",
      "deviceType": "module",
      "controlType": "binary",
      "additionDate": "2024-06-17T19:27:42.932089Z"
    }
  ],
  "totalItemsCount": 2
}
```


Рисунок 3.37 – Результат запиту на отримання усіх створених модулів

Для перевірки стану модуля лампочки, чи увімкнений він, надішлемо запит на сервер HTTP GET, URI якого "api/devices/{deviceId}/control/module-state", де deviceId – це унікальний ідентифікатор створеного модуля лампочки, який ми отримали раніше. В результаті відповідь із сервера повертає поле value, яке дорівнює false. Це означає, що лампочка вимкнена.

На рисунку 3.38 наведено запит та відповідь із сервера.

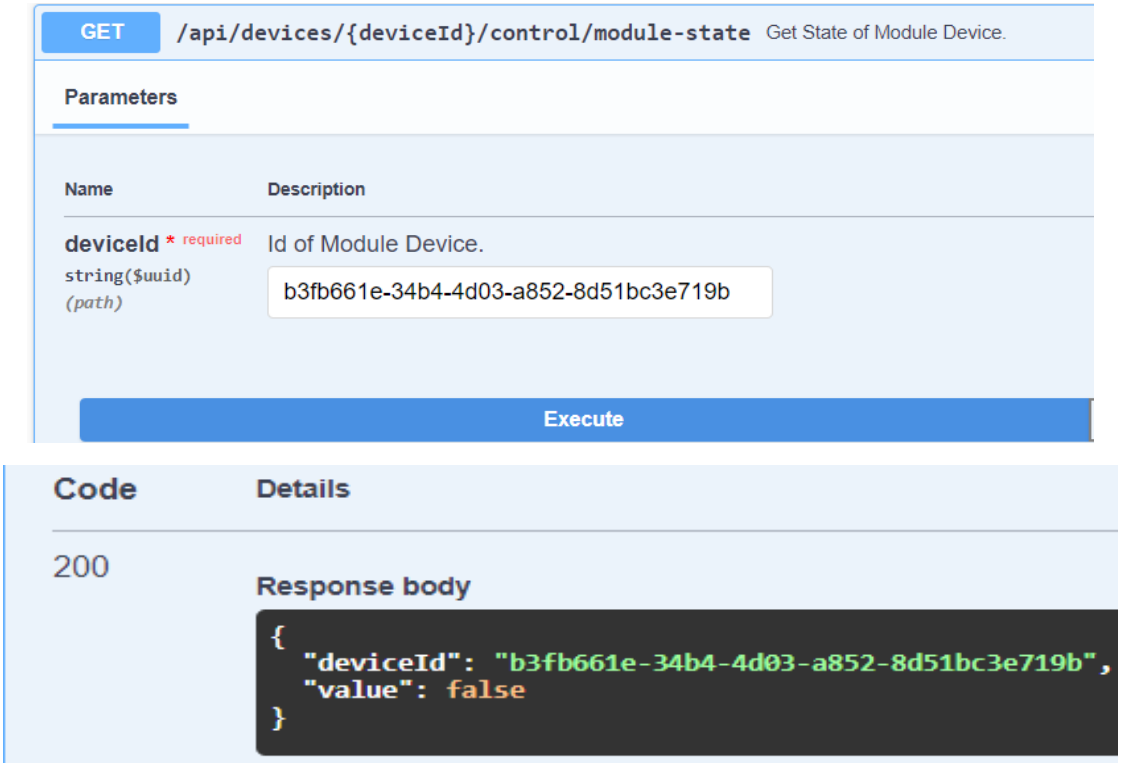

Рисунок 3.38 – Результат запиту поточного стану подуля лампочки

Для того, щоб увімкнути модуль лампочки, необхідно надіслати HTTP PATCH запит, URI якого "/api/devices/{deviceId}/control/module-state", де deviceId – унікальний ідентифікатор модуля лампочки. Також треба передати тіло запиту у форматі json, де вказується поле value із значенням true, тобто увімкнути модуль лампочки. В результаті запиту отримано код 200, що позначає успіх операції та value true, що відображає поточний стан модуля лампочки. Якщо value дорівнює true, то лампочка увімкнена.

Запит та результат наведено на рисунку 3.39.

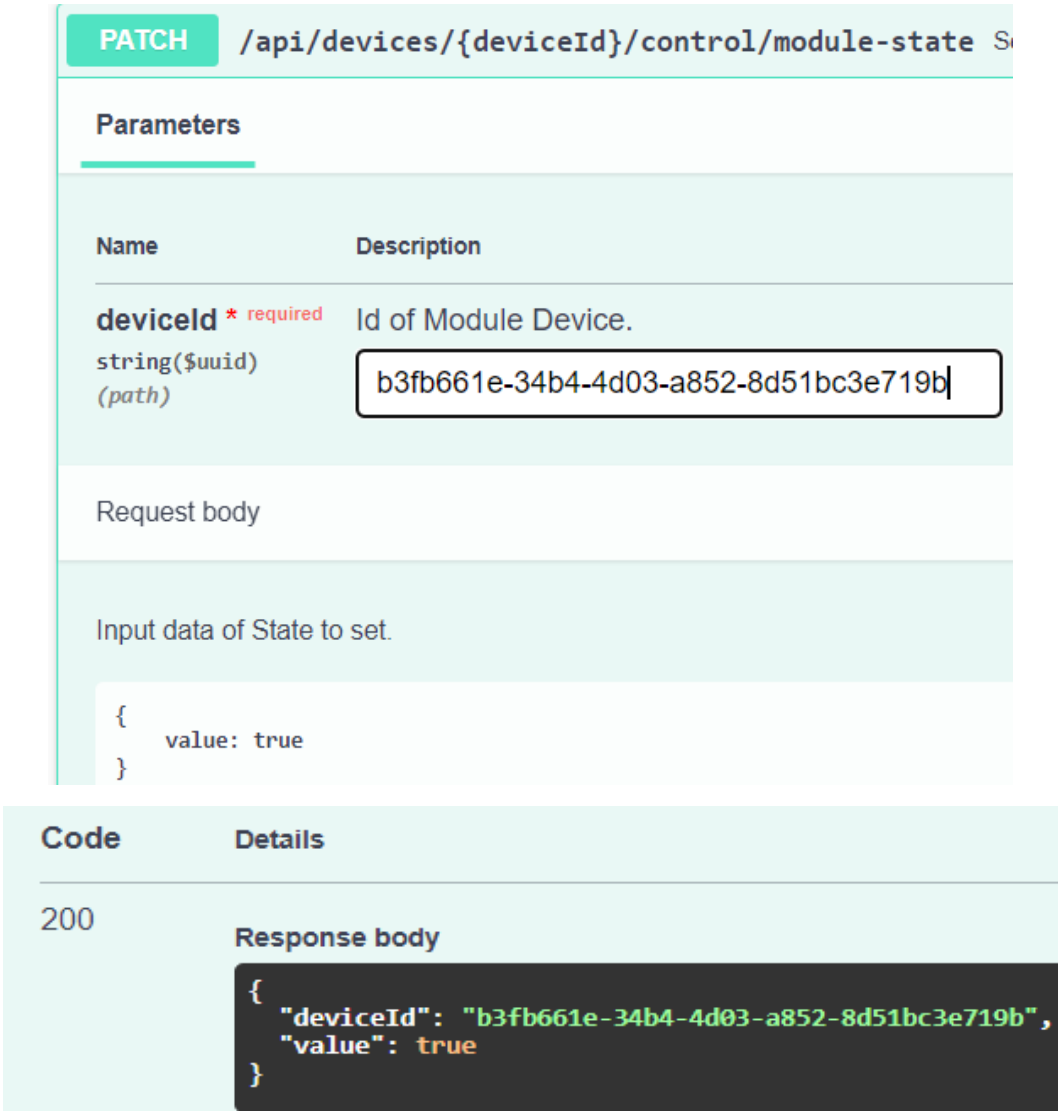

Рисунок 3.39 – Запит та результат увімкненя модуля лампочки

Надішлемо наступний запит на сервер, щоб переконатися що модуль лампочки дійсно увімкнений: HTTP GET, URI якого "api/devices/{deviceId}/control/module-state", де deviceId – це унікальний ідентифікатор створеного модуля лампочки, який ми отримали раніше. В результаті відповідь із сервера повертає поле value, яке дорівнює true. Це означає, що модуль лампочки дійсно увімкнений.

На рисунку 3.40 наведено запит та відповідь із сервера.

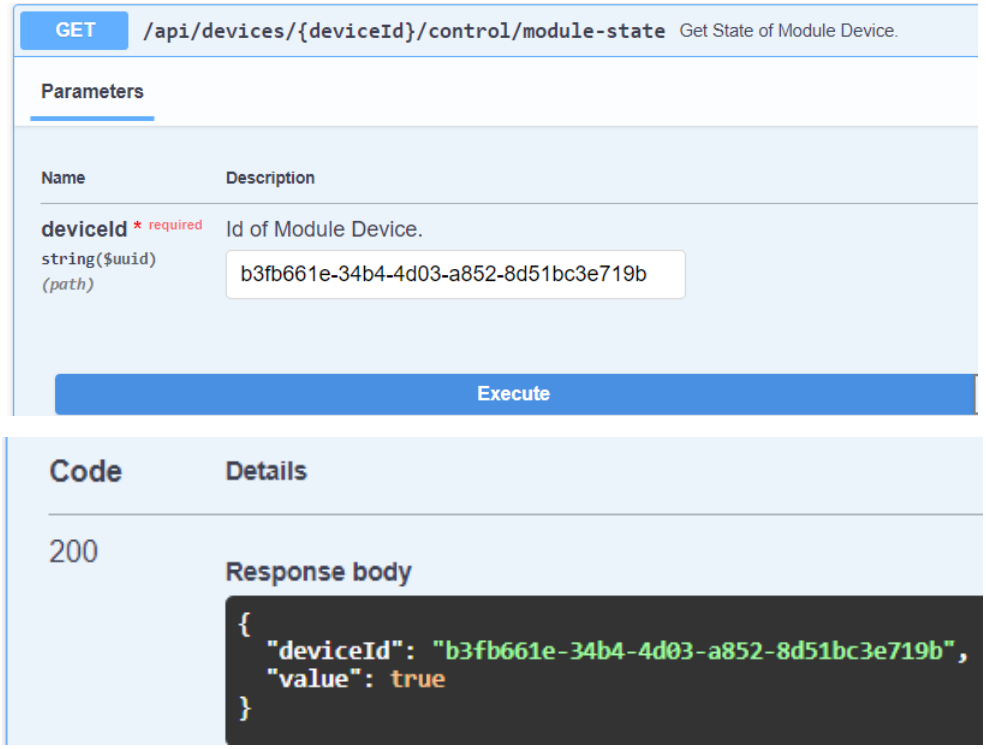

Рисунок 3.40 – Запит та відподвідь із сервера на отримання стану модуля лампочки

Після виконання запиту на увімкнення лампочки, діод модуля лампочки увімкнувся.

На рисунку 3.41 наведено вигляд стану модуля лампочки у активному стані.

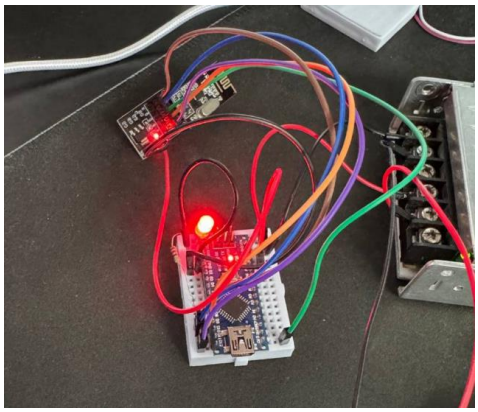

Рисунок 3.41 – Модуль лампочки у активному стані

Тепер вимкнемо модуль лампочки. Для цього надішлемо один із попередніх запитів на сервер, в якому вкажемо в якості value false, що значить вимкнути. Надішлемо HTTP PATCH метод, URI якого: "/api/devices/{deviceId}/control/module-state", де deviceId – унікальний ідентифікатор модуля лампочки. Також надішлемо тіло запиту у форматі json, де вкажемо value false, тобто вимкнути.

Запит та його результат наведено на рисунку 3.42.

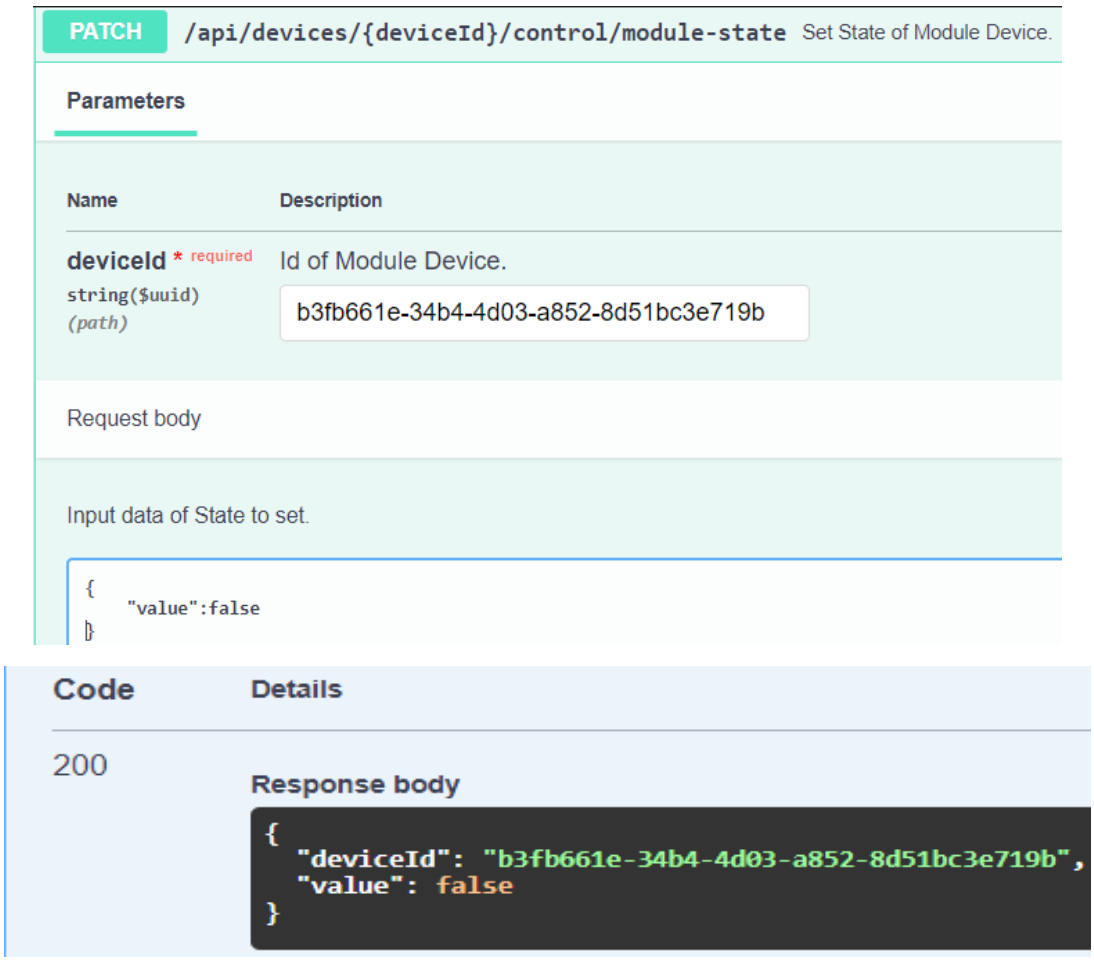

Рисунок 3.42 – Запит та його результат на вимкнення модуля лампочки

Після запиту вимкнення модуля лампочки, модуль лампочки вимкнувся.

На рисунку 3.43 наведено зовнішній вигляд модуля лампочки у вимкненому стані.

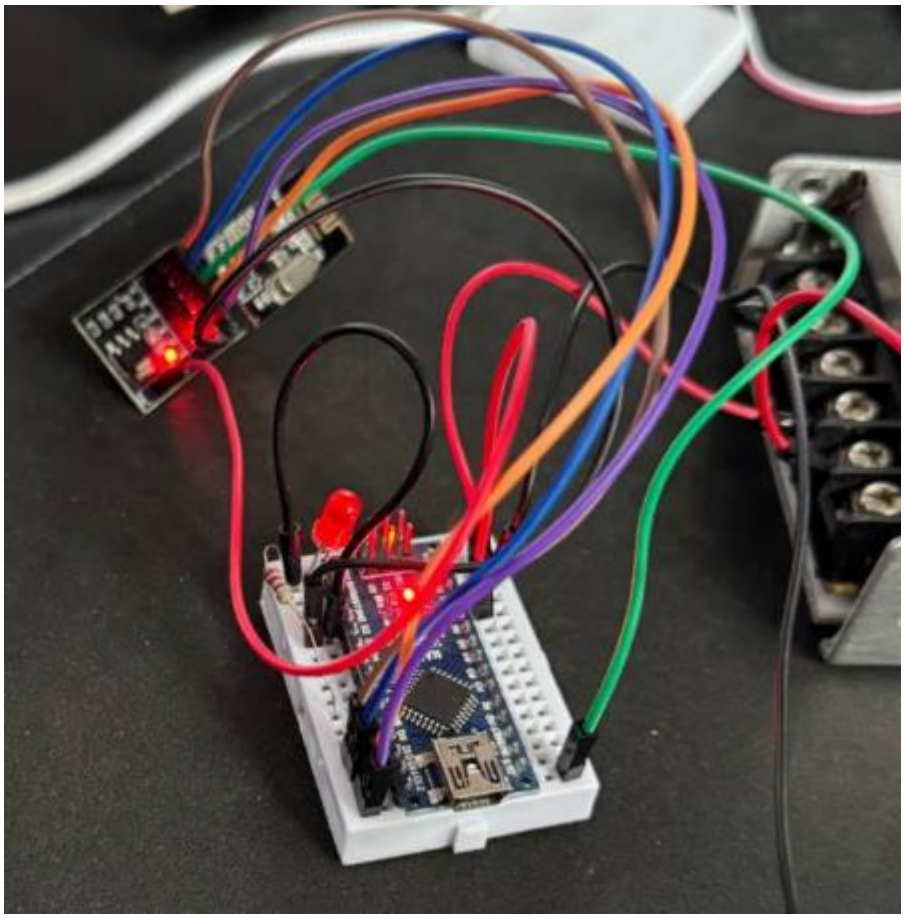

Рисунок 3.43 – Вигляд модуля лампочки у вимкненому стані

Отже, для того щоб взаємодіяти із модулями системи, треба надсилати запит на серверну частину вказуючи унікальні ідентифіктори модулів та обрати необхідний запит на маніпулювання модулем.

3.5 Розрахунки для моделювання системи автоматичного управління

В ТАУ (теорії автоматичного управління) при моделюванні системи проводять її синтез та аналіз, а вони напряму зв’язані із математичною моделлю, ця модель виходить в результаті математичного опису системи [15].

Для побудови математичної моделі, яка використовується в теорії автоматичного управління, нам необхідно поєднати отримані рівняння в єдину систему. Ця модель повинна описувати взаємодію між вхідними параметрами,

контролером та виконавчими механізмами для досягнення бажаних вихідних параметрів.

Модель системи. Виконавчий механізм можна розглядати як систему автоматичного керування споживанням електроенергії. Основні компоненти системи:

– виконавчий механізм;
– датчики споживання електроенергії;
– контролер (мікропроцесор або мікроконтролер);
– користувацький інтерфейс для налаштування параметрів.

Почнемо з моделі споживання електроенергії. Нехай потужність, що споживається навантаженням, позначається як P(t). Виконавчий механізм має можливість керувати навантаженням за допомогою зміни напруги або частоти.

Для спрощення припустимо, що виконавчий механізм може вмикати або вимикати навантаження. Тоді стан виконавчого механізму можна описати як дискретну змінну u(t), де u(t) = 1 означає "ввімкнено", а u(t) = 0 означає "вимкнено".

Побудова регулятора. Розглянемо пропорційно-інтегральний (ПІ) регулятор для підтримання споживання енергії на заданому рівні. Передатна функція виконавчого механізму може бути виражена у формулі (3.1):

$$H(s) = \frac{K}{T_s + 1}, \tag{3.1}$$

де K – коефіцієнт підсилення,

T – постійна часу системи.

ПІ регулятор описується у рівнянні (3.2):

$$G(s) = K_p + \frac{K_i}{s}, \tag{3.2}$$

де K_p – пропорційний коефіцієнт,

K_i – інтегральний коефіцієнт.

Аналіз стійкості системи. Розглянемо характеристичне рівняння замкненої системи (3.3):

$$1 + G_{(s)}H_{(s)} = 0. \tag{3.3}$$

Підставимо вирази для $G_{(s)}$ та $H_{(s)}$ у рівняння (3.4):

$$1 + \left(K_p + \frac{K_i}{S}\right)\frac{K}{T_s+1} = 0. \tag{3.4}$$

Розв'яжемо характеристичне рівняння для визначення стійкості системи (3.5):

$$(T_s + 1) + K_pK + \frac{K_iK}{s} = 0. \tag{3.5}$$

Налаштування параметрів регулятора. Застосовуючи методи ТАУ (наприклад, метод кореневих годографів або частотних характеристик), можна визначити оптимальні значення K_p і K_i для досягнення бажаних динамічних характеристик системи.

Для конкретного прикладу наведемо параметри:

– K = 1;
– T = 0,1;
– Kp = 2;
– Ki = 1.

Складемо характеристичне рівняння (3.6):

$$1 + \left(2 + \frac{1}{s}\right)\frac{1}{0{,}1s+1} = 0. \tag{3.6}$$

Спростимо рівняння (3.7 – 3.9):

$$(0{,}1s + 1) + 2 + \frac{1}{s} = 0, \tag{3.7}$$

$$0{,}1s^2 + s + 2s + 1 + \frac{1}{s} = 0, \tag{3.8}$$

$$0{,}1s^3 + 3s^2 + s + 1 = 0. \tag{3.9}$$

Для аналізу стійкості системи автоматичного керування за допомогою характеристичного рівняння можна застосувати критерій Раута-Гурвіца.

Основна мета полягає в тому, щоб визначити, чи знаходяться всі корені цього рівняння в лівій півплощині комплексної площини Re(s) < 0, що є необхідною умовою для стійкості системи.

Застосуємо критерій Раута-Гурвіца для аналізу стійкості. Критерій полягає у побудові таблиці Раута і перевірці, чи всі елементи першого стовпчика таблиці мають однаковий знак.

Розглянемо характеристичне рівняння (3.10):

$$0{,}1s^3 + 3s^2 + s + 1 = 0. \tag{3.10}$$

Побудова таблиці Раута. Заповнюємо перші два рядки таблиці. Перший рядок: 0,1 (коефіцієнт при s^3) і 1 (коефіцієнт при s). Другий рядок: 3 (коефіцієнт при s^2) і 1 (вільний член).

Результат побудови таблиці Раута наведено у таблиці 3.1.

Таблиця 3.1 – Таблиця Раута.

s^3	0,1	1
s^2	3	1

Визначимо елементи наступних рядків. Перший елемент третього рядка (3.11):

$$\frac{3 \cdot 1 - 0{,}1 \cdot 1}{3} = \frac{3 - 0{,}1}{3} = \frac{2{,}9}{3} \approx 0{,}9667. \quad (3.11)$$

Другий елемент третього рядка: 0 (немає наступного коефіцієнта).

Оновлену таблицю Раута наведено у таблиці 3.2.

Таблиця 3.2 – Продовження таблиці Раута.

s^3	0,1	1
s^2	3	1
s^1	0,9667	0
s^0	1	

Перевіримо елементи першого стовпця:

– 0,1;

– 3;

– 0,9667;

– 1.

Всі елементи першого стовпця мають однаковий знак (позитивний). Отже, оскільки всі елементи першого стовпця у таблиці 3.2 Раута позитивні,

то система є стійкою. Це означає, що всі корені характеристичного рівняння мають від'ємну дійсну частину, тобто система повертається до рівноважного стану після збурень.

3.6 Охорона праці

Система автоматизації розумного будинку має забезпечувати безпечні умови праці, знижувати ризики виробничого травматизму, професійних захворювань та інші небезпечні фактори. Основні кроки для забезпечення охорони праці включають ідентифікацію небезпек, оцінку ризиків і впровадження відповідних заходів безпеки.

У цьому розділі буде виявлення можливих чинників небезпек при експлуатації системи автоматизації розумного будинку, що спричиняють можливість виробничого травматизму, професійних захворювань, отруєнь, пожеж, вибухів, забруднення навколишнього середовища [24].

Тож до виробничого травматизму можна віднести:

– неправильне поводження з електронними компонентами;
– несправність модулів або компонентів системи.

До професійних захворювання та отруєнь:

– недостатня вентиляція.

До пожежі та вибухів:

– короткі замикання в електронних компонентах;
– наявність легкозаймистих матеріалів у приміщенні.

Забруднення навколишнього середовища можна віднести тільки викиди шкідливих речовин в атмосферу.

Для того щоб уникнути потрібно регулярно проводити заходи з охорони праці. Навіть така, на перший погляд проста, система може призвести до серйозних наслідків.

Основне що потрібно знати про охорону праці системи автоматизації для управління якістю повітря у виробничому приміщенні це:

– ні в якому разі не сувати нічого до рухомих частин системи;

– вчасно обслуговувати елементи системи, так як це може призвести до захворювання, отруєнь та інших проблем;

– не обслуговувати систему поки вона повністю не знеструмиться;

– якщо при запуску системи з її компонента підіймається дим то систему слід вимкнути та знеструмити, після чого перевірити компонент на дефекти;

– не слід ставити легкозаймистих матеріалів біля спліт-систем, при включенні обігрівачу вони можуть спалахнути;

ні в якому разі не запускати систему при несправності компонентів системи.

3.7 Висновки до третього розділу

У третьому розділі роботи було обрано середовище для розробки програмного забезпечення, а саме Arduino IDE та Rider IDE.

Було розроблено та описано код для таких компонентів системи, як роутер та модуль лампочки.

Була розроблена серверна частина для управління системи. Був описаний код серверної частини.

У цьому розділі був проведений експеримент системи для вмикання та вимикання модуля лампочки.

Після експерименту було проведено розрахунки для моделювання системи автоматичного управління.

ВИСНОВКИ

Метою роботи було покращення роботи системи розумного будинку за рахунок розробки есперимантального макету з декількома режимами роботи в залежності від потреб користувача в режимі реального часу. Для досягнення поставленої мети було виконано наступні завдання:

– аналіз недоліків існуючих систем;
– аналіз методів керування розумним будинком;
– складання макету системи;
– розробка коду мікроконтролера Arduino UNO;
– розробка коду серверної частини;
– проведення експерименту.

У першому розділі було проведено аналіз аналогів існуючих систем розумного будинку. Були розібрані позитивні сторони та недоліки аналогів.

Найпопулярніші недоліки аналогів – це висока вартість, незахищеність системи від кібератак, залежність від інтернету та велике енергоспоживання.

Було визначено сфери автоматизації в концепції розумного будинку.

Найпопулярніші сфери автоматизації – це керування освітленням, відстеження руху, керування мультимедіа, автоматизація побутових завдань та моніторинг енергоспоживання.

У другому розділі було проаналізовано існуючі аналоги компонентів розумного будинку. Було проведено пошук методів керування розумним будинком.

Найпопулярніші методи керування розумним будинком – автоматичне керування, керування за допомогою дистанційного пульта, віддалене керування.

Також було обрано необхідні компоненти для системи. На основі обраних компонентів було складено макет системи.

У третьому розділі роботи було обрано середовище для розробки програмного забезпечення, а саме Arduino IDE та Rider IDE.

Було розроблено та описано код для таких компонентів системи, як роутер та модуль лампочки.

Була розроблена серверна частина для управління системи. Був описаний код серверної частини.

Також було проведено експеримент системи для вмикання та вимикання модуля лампочки. Після експерименту було проведено розрахунки для моделювання системи автоматичного управління.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008-15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015-06-22. К. Держстандарт України, 2017. 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи бакалавра для студентів усіх форм навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І.Ш. Невлюдов, А.О. Андрусевич, О.В. Токарєва, С.П. Новоселов, О.В Сичова. Харків: ХНУРЕ, 2024. 55 с.

3. Положення про кваліфікаційну роботу здобувача вищої освіти на другому (магістерському) рівні [Електронний ресурс] : Наказ ХНУРЕ від 06 травня 2024 р. No 143. – Режим доступу : https://nure.ua/wp-content/uploads/ Main_Docs_NURE/143-vid-06.05.2024-pro-vvedennja-v-diju-rishennja-vchenoiradi -universitetu.pdf.

4. Положення про академічну доброчесність [Електронний ресурс]: Наказ ХНУРЕ від 02 лютого 2024 р. No 50. – Режим доступу: https://nure.ua/wpcontent/uploads/Main_Docs_NURE/polozhennja-pro-akademi-chnudobrochesnist.pdf.

5. Комплект сигналізації Ajax Starter Kit. [Електронний ресурс] /– Режим доступу: www / URL: https://elmir.ua/ua/alarm_kits/alarm_kit_ajax_starterkit_black.html.

6. Розумна багатобарвна Wi-Fi лампа TP-Link Tapo L530E(2-pack). [Електронний ресурс] /– Режим доступу: www / URL: https://samsungshop.com.ua/smart-things/rozumni-lampi/rozumna-bagatobarvna-wi-fi-lampa-tp-link-tapo-l530e2-pack.html.

7. IP-камера Ajax TurretCam 5 Мп, 4 мм White (000039308). [Електронний ресурс] /– Режим доступу: www / URL: https://rozetka.com.ua/ua/ajax-turretcam-5mp-4mm-white/p404500740/?gad_source=1&gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2thk_wEKAEXWZthrA0ySBx7XWM8MCYj7HhJG0lymLZ2qTHKFocjTP0aArfwEALw_wcB.

8. Портативні контролери для керування розумним будинком. [Електронний ресурс] /– Режим доступу: www / URL: https://z-wave.com.ua/ua/g855993-portativnye-kontrollery-upravleniya.

9. Особливості технології HomeKit. [Електронний ресурс] /– Режим доступу: www / URL: https://ela.kpi.ua/server/api/core/bitstreams/e7a6dce1-8984-442b-a5bf- 74a468e46708/content.

10. Apple HomeKit smart home guide. [Електронний ресурс] /– Режим доступу: www / URL: https://thegadgetflow.com/blog/apple-homekit-smart-home/.

11. Датчик руху Aqara Human Body Sensor (RTCGQ11LM). [Електронний ресурс] /– Режим доступу: www / URL: https://elmir.ua/datchiki_i_sensory_dlya_umnogo_doma/motion_sensor_aqara_human_body_sensor_rtcgq11lm.html?gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2tSPIZGp6E3TkVWs6MbmWsj-mxgf_UqLlKF87RXPa8Pl7Gx5jXmflkaAlCBEALw_wcB.

12. Розумні лампи Philips Hue White. [Електронний ресурс] /– Режим доступу: www / URL: https://geekhouse.com.ua/ua/p1368236067-umnye-led-lampochki.html.

13. Датчик температури і вологості Maxus ZigBee TH Sensor (AirVision-Z-TH). [Електронний ресурс] /– Режим доступу: www / URL: https://elmir.ua/sensors_for_alarm_systems/temperature_and_humidity_sensor_maxus_zigbee_th_sensor_airvision-z-th.html?gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2uLOdwathyEt4tz0gqaRmHVLVb4cAE7XhOtHXOr9m8A8_bTrW2eUGMaAsFOEALw_wcB.

14. Розумна вбудована розетка Ajax Outlet Black. [Електронний ресурс] /– Режим доступу: www / URL: https://rozetka.com.ua/ua/ajax-saob/p421730133/?gad_source=1&gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2uSLUQTpyw2fzY3OZAcDXXEwVBsRCzfkXxjrBaKod3hvjzELgd669EaAiONEALw_wcB.

15. Плата мікроконтролера UNO. [Електронний ресурс] /– Режим доступу: www / URL: https://rozetka.com.ua/ua/249475206/p249475206/?gad_source=1&gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2uYlxFZ0fj9qXiLwOO5yxuoe6of_C_pLfXZt7qDtsiWuNqDO9uWOCwaAi2qEALw_wcB.

16. Радіомодуль NRF24L01. [Електронний ресурс] /– Режим доступу: www / URL: https://evse.com.ua/ua/radiomodul-NRF24L01-pa-lna-24ggts-1000m-transiver-arduino-pic-stm32?gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2tZ0tniSMLu9AXaKi6Imihx2Tgitreuoi6hGJLT17oY2qz6LPBxab4aAsWJEALw_wcB.

17. Модуль HW-069 4-розрядний 7-сегментний інликатор. [Електронний ресурс] /– Режим доступу: www / URL: https://myproject.com.ua/modul-4-rozryadniy-7-segmentniy-ndikator-tm1637-z-dvokrapkoyu-ua.html?gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2v9L_UcdxZ41k6TORrefLftPFm5QapVqiaTna2szfyXbfqM3gPaQNgaAn56EALw_wcB&utm_source=google&utm_medium=cpc&utm_campaign=New_Company.

18. Макетна плата 830 MB-102. [Електронний ресурс] /– Режим доступу: www / URL: https://uamper.com/index.php?route=product/product&path=94&product_id=68&gad_source=1&gclid=Cj0KCQjwvb-zBhCmARIsAAfUI2tEC4ZpEJJhJuRopE30zvDJJSGfuc5ETmUtOyE3NjVqT9qbbkDk8vUaArWeEALw_wcB.

19. Резистор 1 кОм 0.25 Вт. [Електронний ресурс] /– Режим доступу: www / URL: https://ohrana.ua/uk/rezistor-1-kom-0-125w.html.

20. Світлодіоди 5мм 10шт червоні. [Електронний ресурс] /– Режим доступу: www / URL: https://fullspectrum.com.ua/product/svitlodiody-5mm-10sht-chervoni/?gad_source=4&gclid=Cj0KCQjw4MSzBhC8ARIsAPFOuyUefXsfn74GUkmgsSq0pv2N8rdURslaW3rO4s2kTH6tfQ37Xftu0FYaAimCEALw_wcB.

21. Невлюдов І. Ш. Комп'ютерно-інтегровані технології виробництва технічних засобів автоматизації. Частина 1: підручник для студентів заклавдів вищої осіти ; Харків. Нац. Ун-т радіоелектроніки. – Харків : ФОП Панов А.М., 2021. – 604 с. ISBN 978-617-7947-67-6

22. Невлюдов І.Ш. Виробничі процеси та обладнання об'єктів автоматизації. Збірник задач: Навчальний посібник / І.Ш. Невлюдов, А.О. Андрусевич, Г.В. Пономарьова, А.О. Функендорф. Кривий Ріг: КК НАУ. 2018. – 332 с.

23. Невлюдов І.Ш. Технічні засоби автоматизації: Підручник / І.Ш. Невлюдов, А.О. Андрусевич, О.І. Филипенко, Н.П. Демська, С.П. Новоселов. – Кривий Ріг: Криворізький коледж НАУ, 2019. – 366 с.

24. Перелік заходів та засобів з охорони праці. [Електронний ресурс] /– Режим доступу: www / URL: https://www.kmu.gov.ua/npas/1316161.