

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

**ФАКУЛЬТЕТ ЕЛЕКТРОНІКИ
КАФЕДРА ЕЛЕКТРОННИХ ПРИСТРОЇВ ТА СИСТЕМ**

«На правах рукопису»
УДК _____

«До захисту допущено»
Завідувач кафедри

(підпис) Юлія ЯМНЕНКО
(ініціали, прізвище)
“ _____ ” _____ 2020р.

**Магістерська дисертація
на здобуття ступеня магістра**

зі спеціальністю _____ 171 Електроніка
(код і назва)

освітня програма (спеціалізація) _____ Електронні компоненти і системи

на тему: Система моніторингу та контролю для навчальної лабораторії «SmartLab»

Виконав (-ла): студент (-ка) II курсу, групи ДС-91мп
(шифр групи)

Пікож Андрій Володимирович _____
(прізвище, ім'я, по батькові) (підпис)

Науковий керівник ст. викл. Ганна САРИБОГА _____
(посада, науковий ступінь, вчене звання, ім'я ПРІЗВИЩЕ) (підпис)

Консультант _____
(назва розділу) (науковий ступінь, вчене звання, ім'я ПРІЗВИЩЕ) (підпис)

Рецензент д.т.н. проф. каф. АМЕС Сергій НАЙДА _____
(посада, науковий ступінь, вчене звання, науковий ступінь, ім'я ПРІЗВИЩЕ) (підпис)

Консультант
по нормоконтролю к.т.н., доц. Лариса БАТРАК _____
(посада, науковий ступінь, вчене звання, науковий ступінь, ім'я ПРІЗВИЩЕ) (підпис)

Засвідчую, що у цій магістерській дисертації
немає запозичень з праць інших авторів без
відповідних посилань.

Студент _____
(підпис)

Київ – 2020 року

**Національний технічний університет України
“Київський політехнічний інститут
імені Ігоря Сікорського”**

Факультет електроніки
(повна назва)

Кафедра Електронних пристроїв та систем
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-професійною програмою

Спеціальність 171 Електроніка
(шифр і назва)

Спеціалізація Електронні компоненти і системи

ЗАТВЕРДЖУЮ

Завідувач кафедри

(підпис) Ю.С. Ямненко
(прізвище ініціали)

«» 2020 року

**З А В Д А Н Н Я
НА МАГІСТЕРСЬКУ ДИСЕРТАЦІЮ СТУДЕНТУ**

Пікожу Андрію Володимировичу
(прізвище, ім'я, по батькові)

1. Тема проекту Система моніторингу та контролю для навчальної лабораторії «SmartLab»

науковий керівник дисертації Сарибога Г.В., ст.вик.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «05» листопада 2020 року № 3241-с

2. Строк подання студентом проекту 08 грудня 2020 року

3. Об'єкт дослідження Інтернет речей

4. Вихідні дані створення програмно-апаратної платформи для автоматизованої системи керування навчальною лабораторією SmartLab.

5. Перелік завдань, які потрібно розробити 1. Проаналізувати системи моніторингу та контролю параметрів. 2. Підібрати компоненти автоматизованої системи. 3. Обрати алгоритм прогнозування системи. 4. Розробити програмне забезпечення 5. Розробка макету виконавчого пристрою

6. Перелік графічного (ілюстративного) матеріалу Програмне забезпечення для системи моніторингу та контролю навчальної лабораторії «SmartLab», власне макет аудиторії, слайди презентації.

7. Перелік публікацій. 1) Ямненко Ю. С., Пікож А. В. Керування електроспоживанням MicroGrid за вартісним критерієм // Електронні системи та сигнали, 2019. – Том 2. - №1. – С. 10-15. 2) Сарибога Г.В., Пікож А.В, Савчин Ю.В. Система моніторингу для навчальної лабораторії «SmartLab» Міжнародна науково-практична конференція обдарованої учнівської та студентської молоді "Новітні технології сучасного суспільства»

8. Консультанти розділів дисертації

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв

9. Дата видачі завдання «28» жовтня 2020 року

КАЛЕНДАРНИЙ ПЛАН

/п	Назва етапів виконання магістерської дисертації	Строки виконання етапів магістерської дисертації	Примітка
	Огляд літератури	03.09.2020-20.09.2020	виконано
	Проаналізувати системи моніторингу та контролю параметрів.	21.09.2020-10.10.2020	виконано
	Підібрати компоненти автоматизованої системи.	11.10.2020-24.10.2020	виконано
	Розробити програмне забезпечення	28.10.2020-10.11.2020	виконано
	Розробка макету виконавчого пристрою	10.11.2020-28.11.2020	виконано
	Розробка стартап-проекту	29.11.2020– 04.12.2020	виконано

Студент

(підпис)

Пікож А.В.
(ініціали, прізвище)

Науковий керівник дисертації

(підпис)

Сарибога Г.В.
(ініціали, прізвище)

АНОТАЦІЯ

В магістерській дисертації розглядається задача розробки програмно-апаратного забезпечення системи «SmartLab» для моніторингу та контролю умов перебування студентів і викладачів в навчальній аудиторії. В ході роботи розглянуто сучасні системи MicroGrid з використанням засобів та компонентів Інтернету Речей. Виконано огляд компонентів системи, зроблено порівняльні характеристики існуючих елементів та підібрано найбільш відповідні системі датчики та мікросхеми. Розглянуто алгоритми прогнозування MPC та обрано один для аналізу кліматичних умов в аудиторії. Спроектовано макет та підготовлено програмне забезпечення для демонстрації роботи.

Ключові слова: IoT, sensor, cloud computing, remote control, monitoring system, MicroGrid, microcontroller.

ANNOTATION

In the master's thesis the problem of developing software and hardware for the "SmartLab" system for monitoring and controlling the conditions of stay of students and teachers in the classroom. In the course of the work, modern MicroGrid systems using the means and components of the Internet of Things were considered. A review of the system components was made, comparative characteristics of existing elements were made and the most suitable sensors and microcircuits were selected. MPC models are considered and one is selected to analyze the climatic conditions in the classroom. The layout was designed and the software was prepared to demonstrate the work.

Keywords: IoT, sensor, cloud computing, remote control, monitoring system, MicroGrid, microcontroller.

ЗМІСТ

ВСТУП.....	7
1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ	10
1.1. Огляд автоматизованих систем моніторингу та контролю параметрів	10
1.2. Керування електроспоживанням MicroGrid.....	13
1.3. Принципи автоматизації системи моніторингу параметрів	16
1.4. Проблеми реалізації систем на базі інтернету речей	17
1.5. Аналіз систем моніторингу та контролю параметрів	19
Висновки до першого розділу	
2. ВИБІР ТА АНАЛІЗ КОМПОНЕНТІВ АВТОМАТИЗОВАНОЇ СИСТЕМИ	21
2.1. Модель автоматизації	21
2.2. Підсистеми моніторингу та контролю параметрів	22
2.3. Підбір та обґрунтування виконавчих пристроїв підсистем	23
Висновки до другого розділу	
3. . КЕРУВАННЯ СИСТЕМОЮ МОНІТОРИНГУ ЗА ДОПОМОГОЮ MPC	33
3.1. Dynamic Matrix Control (DMC)	35
3.2. Лінійні MPC.....	37
3.3. Нелінійні MPC	41
3.4. Explicit MPC.....	42
Висновки до третього розділу	
4. РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ	44
4.1. Опис Back-end частини.....	44
4.2. Інструменти та реалізація Back-end частини	48
4.3 Опис реалізації Front-end частини	50

Висновки до четвертого розділу

5. РОЗРОБКА СТАРТАП – ПРОЕКТУ	54
5.1. Опис ідеї проекту	54
5.2. Технологічний аудит ідеї проекту.....	56
5.3. Аналіз ринкових можливостей запуску стартап-проекту.....	57
5.4. Розроблення маркетингової програми стартап-проекту	60
ВИСНОВКИ	64
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	65
ABSTRACT	

ВСТУП

Актуальність теми. Із розвитком систем побутової автоматизації, а в більш глобальному розрізі – систем інтелектуального керування електроспоживанням, в тому числі з розподіленою генерацією за наявності альтернативних та відновлювальних джерел електроживлення [1], все більше уваги приділяється розробкам та дослідженням в напрямку моніторингу показників внутрішнього та оточуючого середовища, параметрів мікроклімату, біотелеметричних показників людини як користувача або оператора системи [2]. Ці, а також багато інших задач, пов'язаних із створенням та функціонуванням енергетично-інформаційної інфраструктури, вирішуються в рамках сучасних систем MicroGrid, що визначаються як інтегровані електроенергетичні комплекси з розподіленими енергоресурсами та різнотипними електричними навантаженнями, що разом діють як єдина система – автономно, паралельно з наявною електромережею, або у «острівному», чи відокремленому від існуючої мережі режимі [3].

Для узгодження між собою режимів роботи та інформаційних зв'язків між пристроями, що входять до складу таких систем, успішно застосовується концепція Інтернету речей (IoT – Internet of Things) [4], що являє собою не лише обчислювальну мережу із сукупністю датчиків, але й набір взаємопов'язаних технологій:

- 1) вимірювання, передавання, обробки та зберігання даних;
- 2) робототехніки;
- 3) штучного інтелекту, включаючи методи машинного навчання, нечіткої логіки, штучних нейронних мереж;
- 4) обробки і зберігання великих даних (Big Data) [4].

Широке застосування вбудованих (embedded) систем і технологій збору і передачі інформації в сукупності з пристроями та алгоритмами обробки даних сприяють все більшій популярності та розширенню сфер застосування Інтернету речей навіть на рівні побутового користувача [1].

Зв'язок роботи з науковими програмами, планами, темами. Дослідження за темою магістерської дисертації проводилося у відповідності до наукових напрямків кафедри промислової електроніки КПІ ім. І. Сікорського.

Мета і завдання досліджень. Метою даної роботи є дослідження існуючих засобів використання IoT, а також виконано огляд комерційних і некомерційних хмарних платформ Інтернету речей. Результатом проведених досліджень є практична частина роботи, що являє собою програмно-апаратну платформу для автоматизованої системи керування навчальною лабораторією SmartLab.

Об'єкт дослідження: Інтернет речей

Предмет дослідження: Способи та засоби автоматизації проектування навчальної лабораторії на базі хмарних платформ Інтернету речей.

Методи дослідження. Для демонстрації виконання поставлених в роботі завдань було сконструйовано макет реальної аудиторії. При розробці інтерфейсу програми та функціоналу була використана така мова програмування як JavaScript, з використанням фреймворку Vue.js.

Практичне значення одержаних результатів. Отримані результати можуть використовуватись у майбутніх дослідженнях за даним напрямком, враховуючи переваги та недоліки результатів проведеної роботи. Також розроблена програма може бути використана, як основа для створення лабораторного практикуму на кафедрі.

Об'єкт дослідження: Інтернет речей

Предмет дослідження: Способи та засоби автоматизації проектування навчальної лабораторії на базі хмарних платформ Інтернету речей

Апробація результатів дисертації. Матеріали досліджень були розглянуті на конференції «Електроніка-2019» та на міжнародній науково-технічній конференції «Smart-технології в енергетиці та електроніці-2020».

Публікації. Основний зміст дисертаційної роботи відображено у 2 наукових працях: 1. Ямненко Ю. С., Пікож А. В. Керування електроспоживанням MicroGrid за вартісним критерієм // Електронні системи та сигнали, 2019. – Том 2. - №1. – С. 10-15. 2. Сарибоба Г.В., Пікож А.В, Савчин Ю.В. Система моніторингу для навчальної лабораторії «SmartLab» Міжнародна науково-практична конференція обдарованої учнівської та студентської молоді "Новітні технології сучасного суспільства»

Структура та обсяг роботи. Магістерська робота складається зі вступу, п'яти розділів, висновків, списку використаних джерел зі 41 найменувань. Загальний обсяг магістерської роботи становить 81 сторінок, у тому числі 67 сторінок основного тексту, 19 рисунків та 17 таблиць.

РОЗДІЛ 1. АНАЛІЗ ІСНУЮЧИХ РІШЕНЬ

1.1. Огляд автоматизованих систем моніторингу та контролю параметрів

MicroGrid (у електромережі) - група взаємопов'язаних навантажень та розподілених енергетичних ресурсів із визначеними електричними межами, що утворюють локальну електроенергетичну систему на рівнях напруги розподілу, яка діє як єдиний керований об'єкт і здатна працювати як у мережевому, так і в автономному режимі [5].

Інтенсивність теоретичних і експериментальних робіт в області побудови MicroGrid обумовлена постійним зростанням енергоспоживання, розширенням типів, що генерують і споживають енергію пристроїв. Системи передачі електроенергії також удосконалюються, розширюються і резервуються, що надає нові можливості в маневруванні передачею електроенергії. На рис.1.1 наведено схематичне зображення системи MicroGrid.

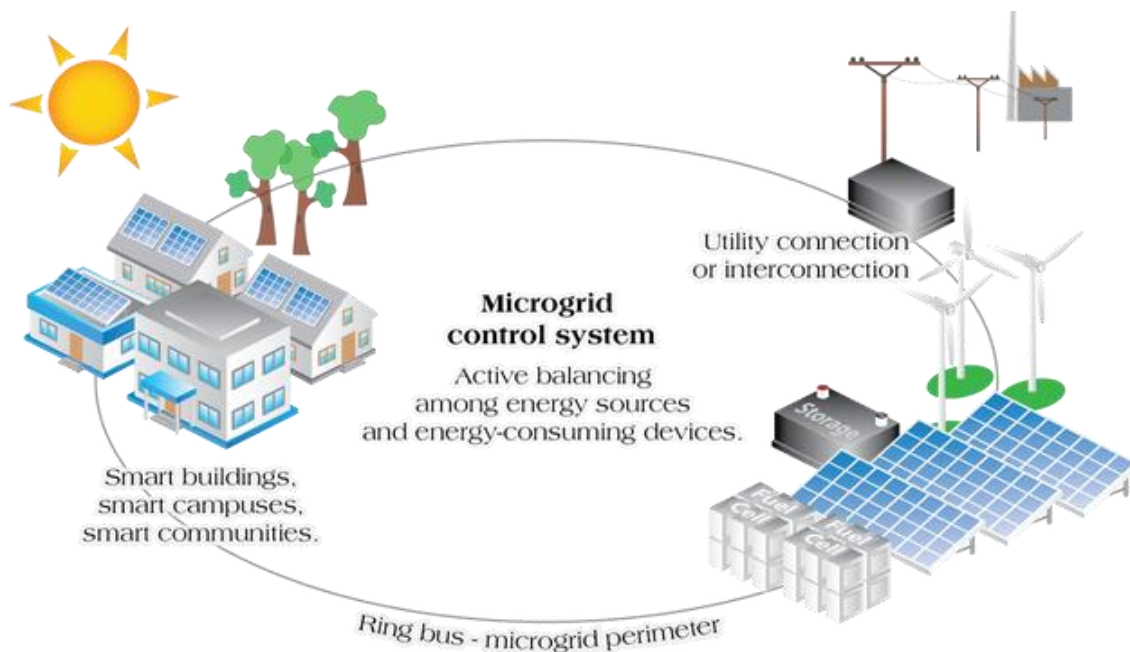


Рис.1.1 Схематичне зображення системи MicroGrid

MicroGrid, в спрощеному вигляді, розглядають як фізично розподілену структуру, яка характеризується наступними ознаками:

- виконанням основного завдання - забезпечення життєдіяльності або виробничого процесу;
- джерела живлення - розподілене (децентралізоване) виробництво електричної енергії, в тому числі з поновлюваних джерел енергії [6,7];
- територіальна обмеженість - зосередженість всіх електротехнічних пристроїв на визначеній користувачем площі;
- присутність людини - користувача або експерта (налагоджувача системи управління MicroGrid), який може внести корективи у функції управління окремих підсистем або всього об'єкта [8].

Прагнення до створення максимально комфортних умов для людини, призвело до високого ступеня насичення MicroGrid електротехнічними, електронними та іншими технічними пристроями і системами, контроль і регулювання робочих параметрів яких здійснюється спеціалізованими системами управління.

Система управління перетворювачами в MicroGrid - це інформаційно-інтелектуальна система, що інтегрує інформацію, яка надходить з різних гетерогенних складових MicroGrid, таких як: альтернативні джерела живлення, навантаження, датчики і яка характеризується різними типами фізичних даних. Така система створюється для контролю енергоресурсів з урахуванням побажань користувача.

Існує кілька видів MicroGrid для різних застосувань. У силу зміни ринків, технологій та регулювання типи MicroGrid продовжують розвиватися [5].

Військові MicroGrid. Здатність надійно включати сонячну фотоелектричну енергію та накопичувач енергії у військові енергетичні

системи є важливою метою для Міністерства оборони будь-якої країни. Залежність від дизельного палива у віддалених регіонах світу є слабким місцем у військових операціях, і наслідки можуть бути дорогими та смертельними через проблему транспортування палива через ворожі регіони. Крім того, Міністерство оборони США визнає зміну клімату рушієм збільшення нестабільності, що ускладнює ще більше транспортування палива у регіони.

MicroGrid спільноти можуть обслуговувати тисячі клієнтів і підтримувати проникнення місцевої енергії (електрика, опалення та охолодження). У MicroGrid спільнотах деякі будинки можуть мати будь-які поновлювані джерела, які можуть задовольнити їх попит, а також попит їхніх сусідів в тому ж співтоваристві. MicroGrid спільнота може також мати централізоване або кілька розподілених сховищ енергії. Такі мікромережі можуть бути в формі мікромереж змінного і постійного струму, з'єднаних разом через двонаправлений силовий електронний перетворювач.

Віддалені MicroGrid. Ці мікромережі ніколи не підключаються до Macrogrid і замість цього постійно працюють в острівному режимі через економічні проблеми або географічне положення. Як правило, «позамережеві» мікромережі будуються в районах, віддалених від будь-якої інфраструктури передачі і розподілу і, отже, не підключених до комунальної мережі. Дослідження показали, що експлуатація позамережевих мікромереж у віддалених районах або на островах, де переважають поновлювані джерела енергії, знижує нормовану вартість виробництва електроенергії протягом терміну реалізації таких проєктів мікромереж.

Великі віддалені райони можуть забезпечуватися декількома незалежними мікромережами, у кожній з яких свій власник (оператор).

Хоча такі мікромережі традиційно проектуються як енергонезалежні, переривчасті поновлювані джерела та їх несподівані і різкі зміни можуть викликати несподіваний дефіцит потужності або надмірну генерацію в цих мікромережах. Це негайно викличе неприпустиме відхилення напруги або частоти у MicroGrid. Щоб виправити такі ситуації, можна тимчасово з'єднати такі мікромережі з відповідною сусідньою мікромережею для обміну енергією і зменшення відхилень напруги і частоти. Це може бути досягнуто за допомогою перемикача на основі силової електроніки після належної синхронізації або прямого з'єднання двох силових електронних перетворювачів і після підтвердження стабільності нової системи. Визначення потреби в з'єднанні сусідніх мікромереж і пошук відповідної мікромережі для зв'язку може бути досягнуто за допомогою підходів до оптимізації або прийняття рішень. Також, впродовж останніх років набирають популярності такі типи систем MicroGrid:

Blockchain Microgrids дозволяють споживачам купувати та продавати електроенергію без участі електричних мереж, і це зараз стає реальністю. Цей демократичний тип мікромереж вже впроваджений у Нью-Йорку, Африці, Європі та Австралії, і електромережі намагаються наздогнати технології, які вони бачать як загрозу.

Автономні електромобілі EV забезпечують електромобілі без водія, які можуть отримувати електроенергію, коли в домашній системі або мікромережі немає енергії.

1.2. Керування електроспоживанням MicroGrid

Одним із сучасних підходів до розв'язання задачі керування електроспоживанням, є DSM (Demand-Side Management – керування за

попитом), який передбачає реалізацію наступних чотирьох напрямів керування [10-13, 41]:

- Load management – регулювання навантаження, яке включає в себе застосування «тарифного меню» та впровадження технічних засобів енергозбереження з метою зниження пікових навантажень [41];

- Energy efficiency – енергоефективність, тобто врахування екологічних факторів, здійснення адміністративних реформ, мотивуючі заходи та економічне заохочення споживачів, що використовують нетрадиційні та відновлювані джерела енергії [41];

- Energy conservation – енергозбереження, тобто заходи, що впроваджуються на промислових підприємствах щодо скорочення годин роботи, регулювання електричної потужності, збільшення продуктивності виробництва [41];

- Fuel substitution – заміна палива або використання інших видів палива, формування політики щодо заміни генеруючих об'єктів на більш ефективні та менш шкідливі з точки зору екології. При цьому розглядаються програми добровільної участі кінцевого споживача у керуванні електроспоживанням. Основними причинами використання в європейських країнах та США такої політики щодо керування енергетичними потоками стали енергетична криза, подорожчання енергетичних ресурсів, необхідність зменшення негативного впливу підприємств енергетичної галузі на довкілля. Важливою мотивацією залучення кінцевих споживачів електричної енергії стало усвідомлення необхідності проведення такої політики, висока ступінь підготовленості до можливих варіантів розвитку подій та можливість отримання економічного ефекту від впровадження методів DSM [41].

У рамках розглядання електротехнічної системи генерації та споживання MicroGrid найбільш перспективним напрямом є Load

Management [41]. Загальна система керування MicroGrid має таку задачу: постійна оцінка і розподіл пріоритетів навантажень, для того щоб гарантувати координоване управління режимами роботи та балансу енергетичних потоків.

Для новітніх MicroGrid, які мають високий ступень насиченості різномірним електротехнічним устаткуванням, важливим є забезпечення швидкісної обробки даних та вироблення керуючих впливів, а також реалізація керування з урахуванням вартісних чинників [14].

Під час розробки системи керування електроживленням розглядаються централізований, децентралізований або змішаний способи керування [8,9]. В будь-якому способі важливим є визначення критерію ефективності, який був би спільним для різних задач керування і дозволяв проводити комплексну оцінку стану системи електроспоживання MicroGrid. Достатньо універсальним в якості такого критерію є інтегральний вартісний параметр, який дає змогу оцінити витрати, необхідні для вироблення та споживання електричної енергії, а в подальшому – мінімізувати ці витрати [41].

Результатом розв'язання задачі оптимізації, де в якості критерію використовується вартість, є оптимальна функція керування системою в цілому (при використанні теорії загальної рівноваги) [15]. Формування цієї функції здійснюється шляхом поєднання математичних рівнянь, які описують технічні аспекти функціонування електротехнічних пристроїв, із рівняннями економічної теорії (зокрема, теорії загальної рівноваги на спільному ринку, де в якості товару розглядається вироблена та спожита електрична енергія) [41].

У загальному випадку MicroGrid містить різні типи навантажень та генераторів електричної енергії, які разом із акумуляторними батареями та

перетворювальними пристроями різних типів утворюють спільний локальний «ринок» електричної енергії [41]

Якщо в якості критерію оптимального керування обрано інтегральний вартісний показник, то для кожного електротехнічного пристрою вирішується задача мінімізації витрат без зниження ефективності, з якою він функціонує, та порушення режимів роботи [15]. Інакше кажучи, керування має забезпечувати виконання належної корисної роботи виконавчих пристроїв у заданих межах, при цьому не повинна погіршуватись комфортність споживача [41]

На рис.1.2 наведено модель загальної рівноваги, де продукцією виступає електрична енергія.

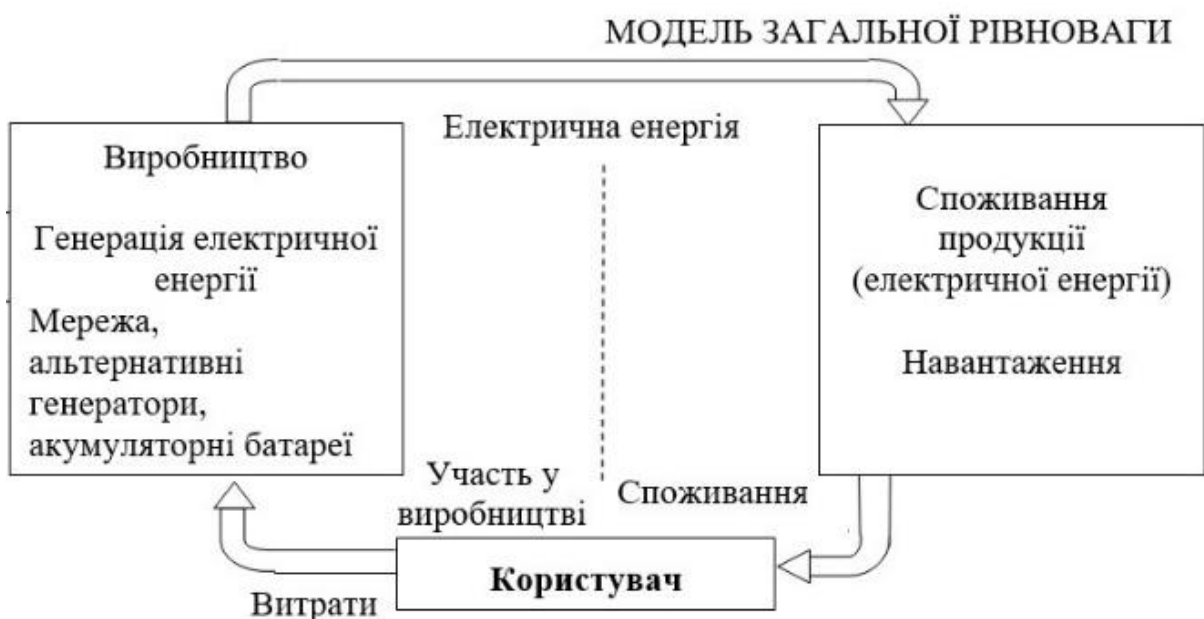


Рис.1.2 Модель загальної рівноваги виробництва та споживання електроенергії

1.3. Принципи автоматизації системи моніторингу параметрів

На теперішній час існує приблизно 300 найменувань протоколів передачі даних в автоматичних модулях. Звісно, для кожного протоколу

існують свої критерії. В таких модулях некоректні дані, які надходять з або до контролера, можуть слугувати причиною відмови всього пристрою. І в залежності від ситуації це може коштувати дорого. Основними критеріями, для таких протоколів є саме їх надійність та можливість оброблювати різні види помилок. Задачу, яку виконує система передачі даних розробка рішення передбачає наступні виконання наступних вимог:

- безпечна трансляція повідомлень,
- нагляд за цілісністю;
- збереження від збоїв,
- надмірність шляхом подвоєння вузлів, ліній, мереж;
- відгородження пошкоджених ділянок та їх відновлення.
- швидке знайдення зламаного вузла;
- управління на відстані за допомогою віддалених вказівок шляхом від'єднання збійних вузлів

Системи регулювання виробництвом підлягають удосконаленню кілька разів впродовж життєвого циклу. Як правило, під час освоєння підприємством нової продукції чи розширення ним виробництва, відбувається заміна тих датчиків, що існують, або доповнення їх точнішими. Під'єднуючи нові системи, до складу яких входять контролери або адресні датчики необхідно слідувати тому чи іншому протоколу передачі даних. Найкраще підійде для вирішення таких питань протокол з якомога меншими вимогами. Даний протокол матиме статус протоколу з вільною топологією. Такі маніпуляції повинні бути окреслені недвозначно, ясно і чітко і бути реалізовані без помилок так, щоб всілякі вузли і контролери могли кооперуватись одне з одним. [4].

1.4. Проблеми реалізації систем на базі інтернету речей

Через величезну кількість підключених датчиків і сервісних функцій, розгортання Інтернету речей (IoT) стає складним процесом, але необхідним.

При цьому комфорт користувача стає невід'ємною частиною екосистеми Інтернету речей, як і енергоефективність. Споживання електроенергії можна зменшити шляхом інтелектуального аналізу відповідних параметрів: температури, вологості, рівня CO₂, освітлення, опалення, безпеки, вентиляція, кондиціонування і т. д., враховувати історичні дані про попередні умови та стан середовища та індивідуальні налаштування користувача.

Багато серверних систем IoT складаються з одного або декількох веб-серверів, які використовуються для передачі потоків даних з датчиків та їх зберігання в хмарному сховищі. Потім ці збережені дані аналізуються за допомогою різних алгоритмів.

З огляду на складність характеристик програм IoT застосовуються декілька сценаріїв роботи системи, а саме: активний, бездіяльний, очікування, глибокий сон, аби зменшити енергоспоживання мікросхеми. Візьмемо для прикладу лічильник енергії: активний режим застосовується, коли лічильник працює нормально; режим сну використовується у випадках, коли лічильник живиться від акумуляторів під час відключення електроенергії; режим глибокого сну може додатково зменшити енергоспоживання режиму сну, вимикаючи більше функцій. Режими очікування та режиму сну безпосередньо управляються МК, і користувачам потрібно лише виконати конкретні інструкції для входу в ці режими або виходу з них.

Цілі: (1) забезпечення безпечного механізму управління побутовою технікою через мобільний або веб-додаток; (2) підвищення безпеки за рахунок сигналізації і шифрування даних бездротового зв'язку; (3)

виконання основних вимог до управління енергоспоживанням шляхом віддаленого моніторингу і відключення непотрібних активних пристроїв для економії енергії

1.5. Аналіз систем моніторингу та контролю параметрів

На сьогоднішній день в Україні збільшилась кількість технічних об'єктів. Це зумовило дуже сильний зріст попиту на технології та програмне забезпечення задля повної автоматизації, оскільки даний аспект дає змогу підвищити надійність, продуктивність та безпеку таких технічних об'єктів. Питання, яке розглядається в ході проектування автоматизованих систем, є попереднє обрання технічних, програмних та інструментальних методів для реалізації алгоритмів регулювання. Для функціонування автоматизованого модулю керування навантаженням брались до уваги контролери, на основі яких створюються автоматизовані системи керування, (Beckhoff, Siemens, Kooyo, WAGO). Дані постачальники поставляють на ринок системи, які є практичними, універсальними та надійними. Звісно, дані прилади можуть мати певні переваги та недоліки. На основі даних приладів вже збираються цілі системи, які вже можуть виконувати:

- 1) з'єднання виходів приладів (аналогові, цифрові) з давачами та керуючими ланками;
- 2) робота на різних протоколах зв'язку (RS - 232, USB, Ethernet);
- 3) програмування режимів роботи.

Апаратне устаткування більшості пристроїв функціонує на відомих операційних системах (MACOS, Windows). Ціна на пристрій, де середовище розробки програмного забезпечення є універсальним для багатьох виробників може бути вища [8]. Пристрої керування виконують

функції передачі відповідних вказівок розумному будинку. Через ці пристрої відбувається контроль стану приладів та віддаються команди. Керовані прилади здійснюють команди та передають її відповідним електроприладам. Датчикам надається інформація з довкілля, а шлюзи зв'язку підтримують зв'язок з відповідними пристроями керування, та з електроприладами, якими треба керувати не просто через подання напруги, відповідно до протоколів (RS 485, RS 232, LAN, Wi-Fi чи інфрачервоний зв'язок). Smart Bus – це система, яка може працювати з відокремленою логікою, тобто відсутня необхідність в центральному процесорі, і якщо у вас є один пристрій керування та один керований пристрій, то у результаті, ви вже володієте розумним будинком. У більшості випадків, такою базою є DDP панель та блок реле. А надалі вже можливо масштабувати до нескінченності увесь об'єкт (точніше до 50 000 пристроїв в системі). Пристрої керування виконують функції передачі відповідних команд. Через них відбувається контроль стану приладів та віддаються команди. Керовані прилади виконують команди та передають її відповідним електроприладам.

Висновки до першого розділу

В ході першого розділу було виконано огляд автоматизованих систем моніторингу та контролю параметрів, розглянуто види MicroGrid для різних застосувань. Також розглянуто проблеми реалізації систем Microgrid на базі інтернету речей.

РОЗДІЛ 2. ВИБІР ТА АНАЛІЗ КОМПОНЕНТІВ АВТОМАТИЗОВАНОЇ СИСТЕМИ

2.1. Модель автоматизації

Для виконання завдання магістерської дисертації в ролі автоматизованої системи виступатиме навчальна лабораторія. У приміщенні вже є певні інженерні системи, такі як: освітлення і електропостачання, вентиляція і т..і. Для проведення занять ці системи задовольняють мінімальні потреби. Якщо виокремити увагу на вентиляції, то вона функціонує сама по собі, без змоги якогось керування, контролю стану повітря. Щодо освітлення – також тільки два стани On/Off. Базове функціонування та зовсім ніякого комфорту. Через це виникла задача розробки системи, яка б імплементувала складові Інтернету Речей. Серед функцій, які система має виконувати:

- 1) регулювання процесу роботи функціональних ланок ;
- 2) підключення та відключення електромоторів;
- 3) виконувати сканування стану елементів;
- 4) сигналізувати про настання аварійної ситуації, як приклад за допомогою освітлення.

Автоматизовані системи покликані мінімізувати втручання людини. Нерідко буває так, що людина з певних причин забула вимкнути праску чи перекрити водопостачання або газ. Фінал таких ситуацій є непередбачуваним. Система, яка розглядається, зможе запобігти поганому перебігу таких випадків. Після чіткого виділення об'єктів автоматизації, необхідно виконати огляд всього устаткування і виокремити потрібну кількість вхідних та вихідних параметрів, щоб регулювати усі процеси.

Для коректного і повного регулювання використовуватимуться аналогові та цифрові входи-виходи [5].

2.2. Підсистеми моніторингу та контролю параметрів

Автоматизована система “SmartLab” з інтелектуальною обробкою даних задовольняє основним міркам з роботи у навчальному процесі. Навчальна аудиторія із системою «SmartLab» наведена на рис.2.1.



Рис.2.1 Навчальна аудиторія із системою “SmartLab”

“SmartLab” має наступні підсистеми :

1. Клімат-контроль та моніторинг стану повітря.
2. Контроль електроспоживання та освітлення.

3. Сповіщення працівників про настання аварійної ситуації у аудиторії.

Структурна схема, що наведена на рис.2.2 відображає взаємодію підсистем в загальній системі.

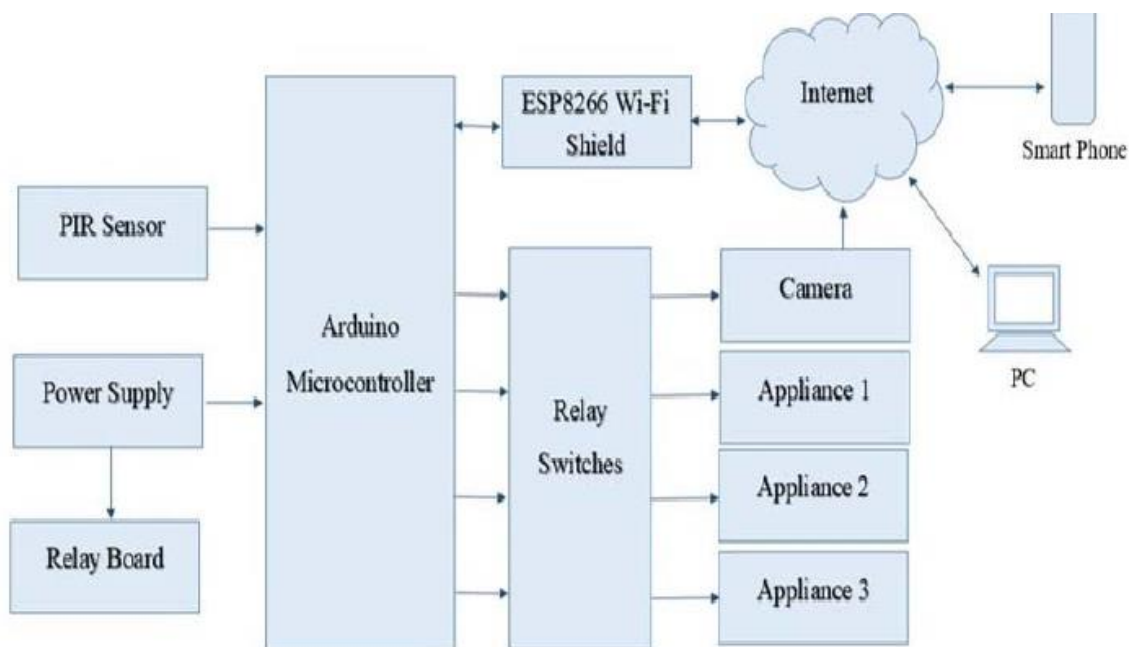


Рис.2.2 Структурна схема системи “SmartLab”

Створення окремих уніфікованих платформ дозволить гарантувати системі підтримку модульності. Віддалене керування, яке будуть підтримувати дані платформи, дозволить моніторити поточний стан аудиторії та, в разі потреби, вносити певні регулювання у систему, маючи при собі гаджет з браузером.

2.3. Підбір та обґрунтування виконавчих пристроїв підсистем

Наступним кроком необхідно вибрати мікропроцесор. Даний електронний компонент повинен мати малі габарити, відносно низьку

вартість та підтримувати різні протоколи безпроводної передачі даних. Проведений огляд представлено в табл.2.1.

Згідно з глибокого огляду моделей мікропроцесорів, можна зробити висновок, що розглянуті електронні компоненти володіють необхідним для автоматизованої системи устаткуванням. Проте, всі вони відмінні за розмірами та ціною. Для розробки «розумного приміщення» можна користуватись будь-яким з них.

Таблиця 2.1

Порівняння мікропроцесорів для реалізації системи “SmartLab”

Модель	WiFi/ Bluetooth	Ethernet	Розмір, мм	Ціна, грн
NodeMCU Esp8266	так/так	так	24,5x14	90
Arduino UNO R3 ATmega328P	так/так	так	68x54	375
Asus Tinker Board S	так/так	так	54 x 85.6	2000
Raspberry Pi Zero V1.3	так/так	так	65x30	910
STM32F411	так/так	так	10x10	485
Arduino MKR1000	так/так	так	61,5x25	875

Проте найбільш вдалим вибором, для автоматизованої системи «SmartLab» є процесор NodeMCU ESP8266. Його невеликі розміри та досить низька вартість відіграли ключову роль при відборі мікропроцесорів. На рис.2.3 зображена мікросхема NodeMCU ESP8266 з її виводами.

Характеристики модулю NodeMCU ESP8266 дозволяють користуватися ним для передачі даних з датчиків у хмарне сховище та

показу даних для користувача шляхом прямого доступу до користувацького інтерфейсу. При потребі, функціонал мікросхеми можна розширити за допомогою використання мікроконтролеру Arduino. NodeMCU - це плата розробника на базі чіпа ESP8266 (версія ESP12E), який є UART-Wi-Fi модулем з ультранизьким споживанням та дозволяє спростити розробку, тому що вона дозволяє заливати прошивку через USB. На додачу, плата поставляється з прошивкою NodeMCU, що дозволяє програмувати її за допомогою мови Lua або за допомогою середовища Arduino IDE [14].

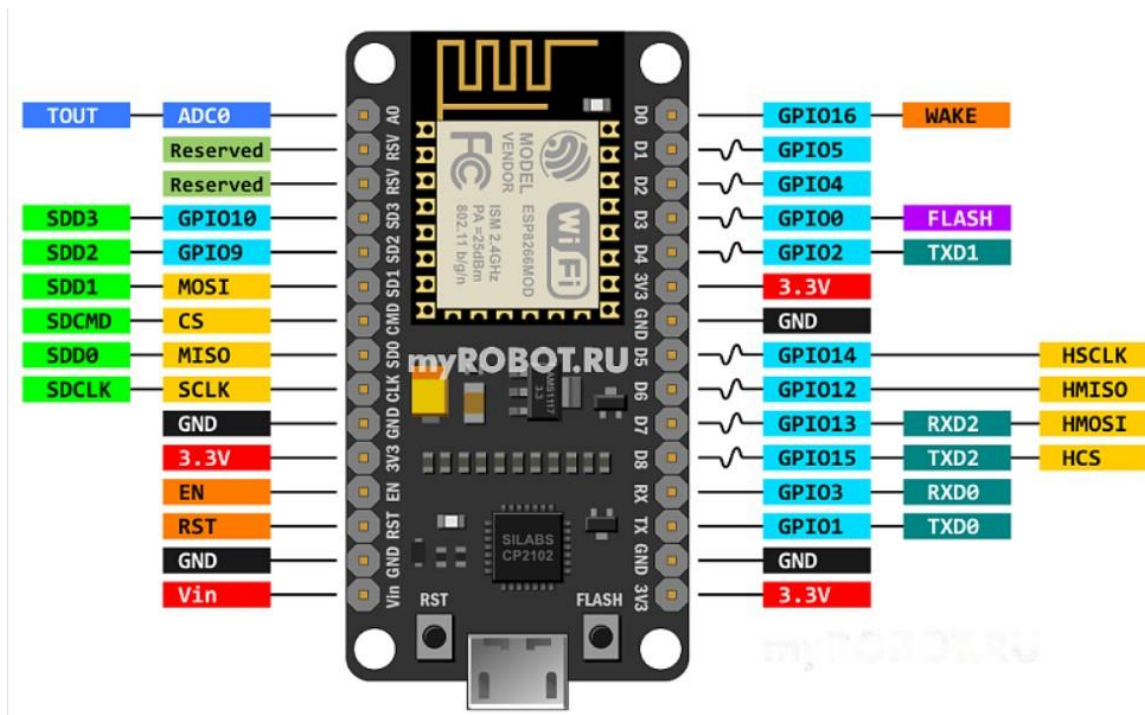


Рис.2.3 NodeMCU ESP8266 з позначеними виводами

Серед плюсів мікросхеми NodeMCU ESP8266 можна виокремити те, що при живленні від звичайних акумуляторів доступний режим енергозбереження, а також в ній реалізована підтримка протоколів безпеки WPA / WPA2, WiFi Direct (P2P), soft-AP[5].

Живлення ESP8266 реалізується через джерело напруги (3-3.6V). Під час вмикання та у часи застосування струм споживання може бути 300mA .

Мережевий адаптер MB102 Breadboard Power Supply Module 3.3V або джерело живлення від USB на мікросхемі AMS1117-3.3 можуть бути використані в якості живлення мікросхеми.

Наступним, необхідно було обрати датчик температури та вологості. Відповідні характеристики давачів наведено у табл. 2.2.

Серед датчиків, що були продемонстровані, необхідно звернути погляд на давачі DHT11 та DHT22. Дані компоненти широко використовуються для моніторингу температури та вологості. Маючи достатньо хороші характеристики та відносно низьку вартість було обрано датчик DHT11.

Таблиця 2.2

Параметри датчиків клімат-контролю

Модель	Точність	Діапазон значень температур і вологості	Напруга живлення, АС/DC, В	Ціна, грн
ESP-01S DHT11	$\pm 2\text{ }^{\circ}\text{C}$ $\pm 5\%$	0-50 $^{\circ}\text{C}$ 20-90%	DC 3,3-12	90
LIFESOS TP-3	-	-55 - +125 $^{\circ}\text{C}$ до 95%	DC 3	278
Ajax TS-101	-	-55 - +125 $^{\circ}\text{C}$ -	AC 12	156
DHT22	$\pm 0.5\text{ }^{\circ}\text{C}$ $\pm 2\%$	-40- +80 $^{\circ}\text{C}$ 0-100%	DC 3-6	210
DS18B20	$\pm 0.5\text{ }^{\circ}\text{C}$	-55- +125 $^{\circ}\text{C}$ -	DC 3-6	43
Sensor- AM2301	$\pm 0.5\text{ }^{\circ}\text{C}$ $\pm 2\%$	-40-80 $^{\circ}\text{C}$ 0-100%	DC 3.3-5.2	147

Схема підключення датчика температури наведена на рис.2.4.

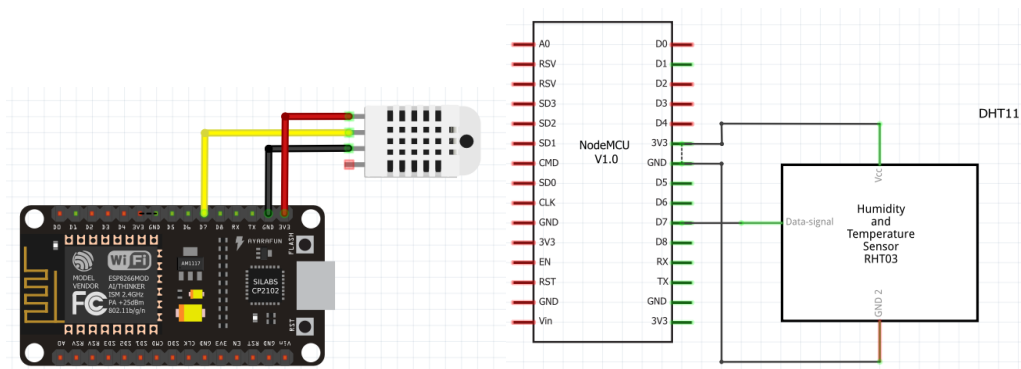


Рис.2.4 Підключення датчика температури до NodeMCU ESP8266

Автоматизована система повинна вимірювати температуру та за допомогою модулю системи прийняття рішень, до складу якого входить регулювання з прогнозуючою моделлю, підтримувати значення температури на певному проміжку. Дані дії можуть бути реалізовані за рахунок управління кондиціонером.

Також до складу платформи включено датчики якості повітря, які наведено у табл.2.3.

Таблиця 2.3

Порівняння модулів моніторингу стану повітря

Модель	Тип	Вимірювальні речовини	Ціна, грн
MH-Z19B	Інфрачервоний	CO2 - 0 до 0.5%	766
Модуль датчиків якості повітря CCS811 + SI7021 + BMP280	Цифровий	1. летючі органічні сполуки (TVOC): від 0 до 1 187 ч. на мільярд 2. eCO2: від 400 до 8 192 ч. на мільйон 3. Тиск, температура, вологість	367
BME280	Цифровий	Тиск, температура, вологість, висота над рівнем моря	78
MQ-7	Аналоговий	Чадний газ, 20ppm-2000ppm carbon monoxide	37
PM2.5 PMS1003	Лазерний	Кількість часток пилу	587

Серед розглянутих датчиків варто звернути увагу на датчик BME280 - це комбінований цифровий датчик вологості, тиску та температури, заснований на перевірених технологіях принципи зондування. Модуль датчика розміщений у надзвичайно компактній металевій кришці LGA розмір площі всього $2,5 \times 2,5$ мм² при висоті 0,93 мм. Його невеликі розміри та мала потужність споживання дозволяють застосовувати в пристроях, що працюють від батареї, таких як слухавки, модулі GPS або годинники. BME280 є зареєстрованим та продуктивним, сумісним із цифровим Bosch Sensortec BMP280 датчик тиску BME280 досягає високої продуктивності у всіх додатках, що вимагають вологості та тиску вимірювання. Ці нові додатки управління автоматизацією будинку, навігаційна система вдома, фітнес яка також вдосконалення GPS вимагають одночасно високої точності та низького рівня TCO.

Датчик вологості забезпечує надзвичайно швидкий час відгуку для програм швидкого розуміння контексту і високу загальну точність у широкому діапазоні температур.

Датчик тиску є абсолютним барометричним датчиком тиску з надзвичайно високою точністю і роздільною здатністю і значно нижчий рівень шуму, ніж Bosch Sensortec BMP180.

Вбудований датчик температури оптимізований для найнижчого шуму та найвищої роздільної здатності. Його вихід використовується для температурної компенсації датчиків тиску та вологості, а також можливим є використання для оцінки температури довкілля.

Датчик забезпечує інтерфейси SPI та I2C і може подаватися з використанням напруги від 1,71 до 3,6 В для датчика живлення VDD і від 1,2 до 3,6 В для інтерфейсу живлення VDDIO. Вимірювання можуть ініціювати користувач або виконуватись через рівні проміжки часу. Коли датчик вимкнено, споживання струму падає до 0,1 мкА.

Керування живленням. BME280 має два окремі штирі джерела живлення:

- VDD є основним джерелом живлення для всіх внутрішніх аналогових та цифрових функціональних блоків
- VDDIO - це окремий штифт живлення, який застосовується для живлення цифрового інтерфейсу

Вбудований генератор перезавантаження (POR); він скидає логічну частину та значення регістру після того, як VDD і VDDIO досягають мінімальних рівнів. Немає лімітів щодо нахилу та послідовності підйому рівні VDD та VDDIO. Після включення датчик переходить у режим сну.

Забороняється тримати будь-який інтерфейсний штифт (SDI, SDO, SCK або CSB) на логічно високому рівні, коли VDDIO вимкнений. Така конфігурація може назавжди пошкодити пристрій через надмірний потік струму через діоди захисту від ESD.

Якщо постачається VDDIO, а VDD - ні, інтерфейсні штифти підтримуються на високому рівні Z. Отже, автобус може вже вільно застосовуватися до встановлення подачі VDD BME280.

Скинути датчик можна за допомогою перемикання рівня VDD або написання команди м'якого скидання.

Регулювання електроспоживання в автоматизованій системі здійснюється за рахунок передачі даних через Wi-Fi мережу до ESP8266. Блок керування має у своєму складі власне NodeMCU ESP8266, транзисторний модуль IRF520 та в якості навантаження світлодіодну стрічку. Платформа регулювання та керування навантаженням наведена на рис.2.5.



Рис.2.5 Платформа керування навантаженням

Також, під час розроблення було прийнято рішення застосовувати модуль MOSFET-транзистора IRF520.

Модуль IRF520 використовується для того, щоб підключити навантаження до одного з пінів контролера, що виконує свою роботу на постійному струмі.

Різниця між цим модулем та модулем реле полягає у високій швидкодії і довговічності першого. Цей модуль може регулювати навантаження в ШІМ (PWM) режимі. Тобто є можливість регулювати за допомогою MOSFET-транзистора у ШІМ-режимі яскравість світлодіодної стрічки, а також іншими виконавчими пристроями.

Оскільки в даному модулі транзистора відсутні механічні контакти, то у зіставленні з електромагнітним реле можна помітити вищий рівень надійності та довговічності.

На рис.2.6 зображено електричну схему модуля IRF520.

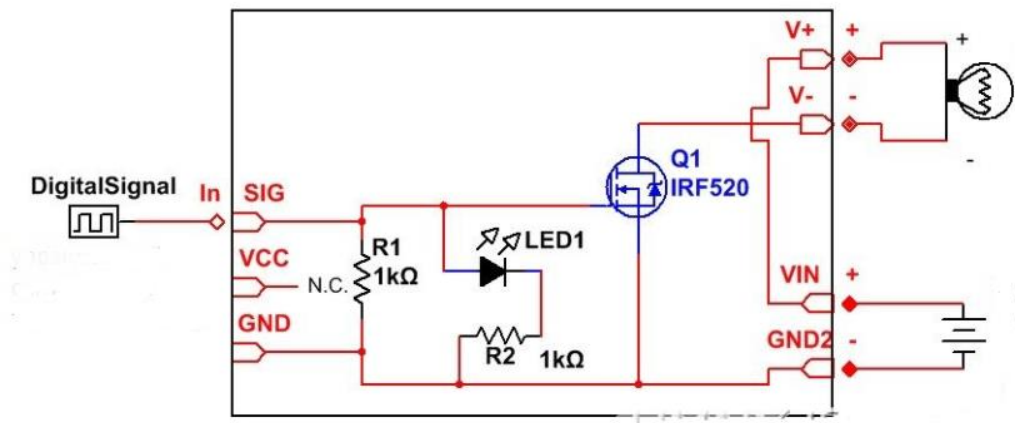


Рис.2.6 Модуль IRF520

Керуюча напруга даного модулю складає 3.3...5В, а напруга на виході 0...24В. Маючи невеликі розміри (33х24 мм) та вагу 10 г даний модуль ідеально підходить для використання у схемі.

В цілях пониження споживання електроенергії створено алгоритм, де присутні декілька режимів роботи.

Регулювання всієї автоматизованої системи «SmartLab» здійснюється за рахунок сценарного підходу.

В даній системі можна виділити декілька робочих рівнів:

- виконавчі ланки: датчики, мікросхеми, електроніка;
- транспортний рівень – включає в себе протоколи зв'язку для обміну даними;
- рівень хмарного середовища; до його складу входять два підрівні: Back-end (тут відбувається обробка та прийняття рішень) та Front-end (тут відбувається контакт із самим користувачем)

Отже, концепція Internet of Things дає змогу приймати та обробляти дані і вже потім приймати рішення.

Сервіс для роботи з клієнтами Серверу додатків дає змогу для:

- індивідуальних налаштувань кожного користувача з функцією збереження даних;
- поділу користувачів на ролі, що дає змогу виокремити їх права;
- надати серверам права адміністратора;
- отримання віддаленого керування, використовуючи при цьому звичайний браузер.

В якості транспортного протоколу використовується MQTT 3.1.1. Даний протокол дає змогу передавати інформацію між рівнем виконавчих ланок та рівнем хмарного середовища (M2M) [8]. На рис.2.7 наведена ієрархія інформаційної взаємодії у системі.

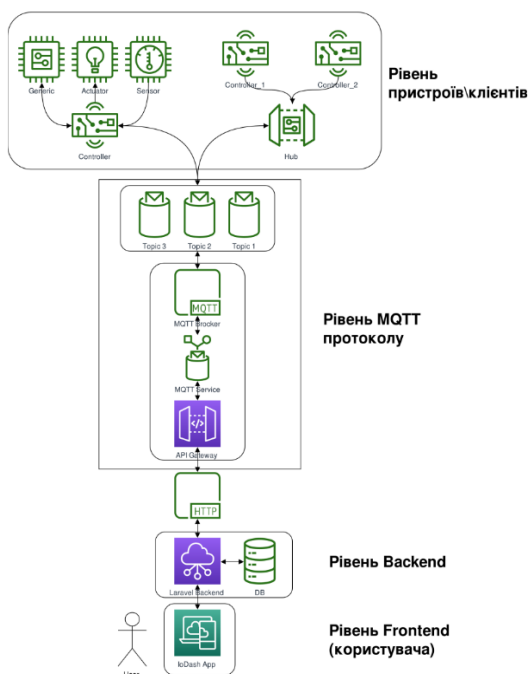


Рис.2.7 Ієрархія інформаційної взаємодії у системі «SmartLab»

Висновки до другого розділу

В ході виконання другого розділу було розглянуто компоненти IoT для автоматизованої системи «SmartLab», виконано порівняння та підібрано елементи схем для системи.

РОЗДІЛ 3. КЕРУВАННЯ СИСТЕМОЮ МОНІТОРИНГУ ЗА ДОПОМОГОЮ MPC

Регулювання з прогнозуючими моделями (MPC - Model Predictive Control) - це вдосконалений метод керування системою, який використовується для регулювання процесів, одночасно задовольняючи набір обмежень. Він використовується у переробній промисловості на хімічних та нафтопереробних заводах з 1980-х років. Впродовж останніх років даний метод знаходить своє застосування у моделях балансування енергетичних систем [16] та в силовій електроніці [17]. В основі керування з прогнозуючими моделями лежать динамічні моделі процесу, частіше за все лінійні емпіричні моделі, отримані шляхом ідентифікації системи. Головною перевагою MPC є той факт, що він дозволяє оптимізувати поточний часовий інтервал, одночасно враховуючи майбутні часові інтервали. Це досягається за рахунок оптимізації кінцевого часового горизонту, але лише за рахунок реалізації поточного часового інтервалу, а потім повторною оптимізацією, неодноразово, тим самим відрізняючись від лінійно-квадратичного регулятора (LQR). Також MPC має можливість передбачати майбутні події і може, відповідно, приймати деякі рішення. ПІД-регулятори не мають такої передбачувальної здатності. MPC майже універсально реалізоване в якості цифрового керування, хоча є дослідження щодо досягнення більш швидкого часу відгуку за допомогою спеціально розроблених аналогових схем [18].

Ідея MPC не обмежується певним описом системи, вона включає у себе обчислення та реалізацію, яка залежить від представлення моделі. Припустимо система, яка описується у просторі станів виглядає наступним чином:

$$x(k) = Ax(k-1) + Bu(k-1) \quad (3.1)$$

$$y(k) = Cx(k) \quad (3.2)$$

Для нульових початкових умов еквівалентна передача матричного вигляду є:

$$y(z) = P(z)u(z) \quad (3.3)$$

де

$$P(z) = C(zI - A)^{-1}B \quad (3.4)$$

Оскільки розглядаються стабільні системи, коли A інвертована, то (3.4) можна розширити до послідовності Неймана:

$$P(z) = \sum_{i=1}^{\infty} CA^i Bz^{-i-1} \quad (3.5)$$

$$\triangleq \sum_{i=1}^{\infty} H_i z^{-i} \quad (3.6)$$

де H_i - коефіцієнти імпульсної характеристики, величини яких зникають, коли $i \rightarrow \infty$ [19]. Таким чином, у часі домену виконується модель усіченого імпульсного відгуку:

$$y(k) = \sum_{i=1}^n H_i u(k-i) \quad (3.7)$$

і з визначеннями

$$\bar{H}_i = H_i - H_{i-1} \quad (3.8)$$

$$H_i = \sum_{j=1}^i \bar{H}_j \quad (3.9)$$

модель усіченого кроку відповіді

$$y(k) = \sum_{i=1}^{n-1} H_i \Delta u(k-i) + H_n u(k-n) \quad (3.10)$$

де

$$\Delta u(k) = u(k) - u(k-1) \quad (3.11)$$

і H_i - коефіцієнти крокової реакції. У залежності від структури затримки в часі системи, крок матриці коефіцієнта відгуку може бути нульовим або мати нульові елементи [19].

Класичним прикладом MPC є динамічне матричне керування (DMC).

3.1. Dynamic Matrix Control (DMC)

Маніпульовані змінні обрані для мінімізації квадратичної задачі:

$$\min_{\Delta u(k) \dots \Delta u(k+m-1)} \sum_{l=1}^p \left\| \hat{y}(k+l|k) - r(k+l) \right\|_r^2 + \left\| \Delta u(k+l-1) \right\|_B^2 \quad (3.12)$$

$$\begin{aligned} \hat{y}(k+l|k) = & \sum_{i=1}^l H_i \Delta u(k+l-i) + \sum_{i=l+1}^{n-1} H_i \Delta u(k+l-i) + \\ & + H_n u(k+l-n) + \hat{d}(k+l|k) \end{aligned} \quad (3.13)$$

$$\hat{d}(k+l|k) = \hat{d}(k|k) = y_m(k) - \sum_{i=1}^{n-1} H_i \Delta u(k-i) + H_n u(k-l-n) \quad (3.14)$$

$$\begin{aligned} \sum_{l=1}^p C_y^j \hat{y}(k+l|k) + C_u^j u(k+l-1) + c^j & \leq 0; \\ j = 1, n_c \end{aligned} \quad (3.15)$$

де $\hat{y}(k+l|k)$ - прогнозоване значення y в момент часу $k+l$ на основі на інформації, яка доступна на час k ;

$\hat{d}(k+l|k)$ - прогнозоване значення адитивних збурень на виході процесу в момент часу $k+l$ на основі інформації, доступної на час k ;

$y_m(k)$ - вимірювання y в момент часу k ;

$H_i, i=1, n$ - модель кроку відповіді матричного коефіцієнта;

n - порядок усічення, n_c - кількість обмежень, p - довжина горизонту (в загальному випадку $p \gg n$), m - кількість опрацьованих змінних в майбутньому $\Delta u(k+l)=0 \quad \forall l \geq m; m < p$);

$$\|x\|_Q^2 = x^T Q x ;$$

C_{yl}^i, C_{ul}^j, c^j - матриці констант;

Прогнозування результату (3.13) включає три доданки з правої сторони. Перший доданок включає теперішні і всі майбутні рухи маніпульованих змінних, які слід визначити так, щоб розв'язати (3.12). Другий доданок включає лише минулі значення маніпульованих змінних і відомий у час k . Третій член - це передбачуване збурення d , яке отримане з (3.7). $\hat{d}(k+l|k)$ передбачена константа для майбутніх часів ($l \geq 0$). У момент k це дорівнює різниці між виміряним результатом $y(k)$ і результатом, передбаченим з моделлю. Рівняння (3.12) - (3.15) визначають квадратичну програму що вирішується в режимі реального часу на кожному часовому проміжку.

Алгоритм буде автоматично коригувати дії всіх інших маніпульованих змінних для компенсації нештатних ситуацій якомога краще. У несподіваних надзвичайних ситуаціях, де системам з традиційною фіксованою логікою буде важко впоратися, MPC буде тримати процес безпечно працюючи далеко від усіх обмежень або дозволяючи користувачу вимкнути його. Саме такий алгоритм і було реалізовано для контролю стану повітря у системі «SmartLab».

3.2. Лінійні МРС

Моделі, які застосовуються у МРС, покликані показати поведінку складних динамічних систем. Алгоритм керування МРС, як правило, не потребує для забезпечення адекватного регулювання простих систем, які часто добре контролюються звичайними ПІД-регуляторами. Загальні динамічні характеристики, складні для ПІД-регуляторів, включають великі затримки часу та динаміку високого порядку.

Моделі МРС передбачають зміну залежних змінних модельованої системи, які будуть спричинені змінами незалежних змінних. У навчальній аудиторії незалежними змінними, які можуть регулюватися регулятором, є або задані значення регуляторних ПІД-регуляторів (тиск, температура тощо) або кінцевий елемент керування (клапани, заслінки тощо). В якості збурень можуть виступати незалежні змінні, які не в змозі регулювати контролер. В якості залежних змінних в цих процесах виступають інші вимірювання, які показують або цілі керування, або обмеження процесу.

Поточні вимірювання процесу, поточний динамічний стан процесу, моделі МРС і цільові змінні процесу та обмеження використовує МРС для розрахунку майбутніх змін у залежних змінних. Ці зміни розраховуються для утримання залежних змінних близько до бажаних значень, одночасно виконуючи обмеження як на незалежні, так і на залежні змінні. Зазвичай, МРС відправляє тільки першу зміну в кожну незалежну змінну, яка має бути реалізована, та виконує обчислення знову, коли необхідна наступна зміна.

На рис.3.1 зображена базова структура системи, яка керується моделлю МРС.

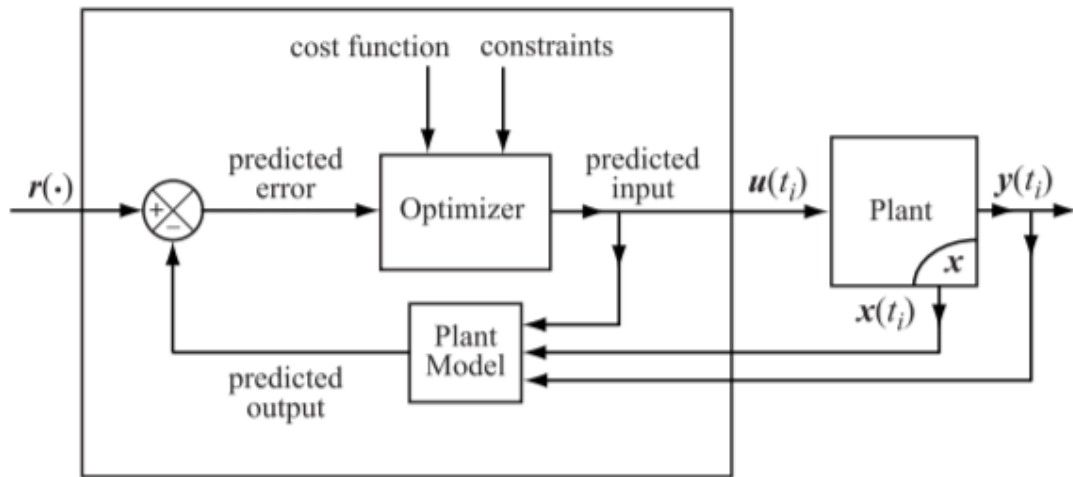


Рис.3.1 Модель системи керованої MPC

Хоча багато реальних процесів є нелінійними, їх часто можна вважати приблизно лінійними в малому робочому діапазоні. Більшість додатків з механізмом зворотнього зв'язку застосовують лінійні підходи MPC, що дає змогу компенсувати помилки прогнозування шляхом структурної невідповідності між моделлю та процесом. У регуляторах з прогнозуючою моделлю, які складаються лише з лінійних моделей, принцип суперпозиції лінійної алгебри дозволяє додавати ефект змін у декількох незалежних змінних для прогнозування реакції залежних змінних. Це спрощує проблему регулювання до набору прямих матричних алгебраїчних обчислень, які є швидкими та надійними.

Коли лінійні моделі недостатньо точні для відображення реальних нелінійностей процесу, можна використовувати кілька підходів. У деяких випадках змінні процесу можуть трансформуватися до або після лінійної моделі MPC для зменшення нелінійності. Процесом можна керувати за допомогою нелінійної MPC, яка використовує нелінійну модель безпосередньо в процесі регулювання. Нелінійна модель може бути у формі емпіричних даних (наприклад, штучних нейронних мереж) або високоточної динамічної моделі, заснованої на основних балансах маси та

енергії. Нелінійна модель може бути лінеаризована, щоб отримати фільтр Калмана або вказати модель для лінійної MPC.

Алгоритмічне дослідження Ель-Герві, Будмена та Ель Камела показує, що використання підходу подвійного режиму може забезпечити значне зменшення у обчисленнях в реальному часі, зберігаючи порівняльну продуктивність із незмінною реалізацією. Запропонований алгоритм паралельно розв'язує N задач опуклої оптимізації на основі обміну інформацією між регуляторами.

MPC базується на ітеративній, скінченно-горизонтальній оптимізації моделі процесу. У момент часу t фіксується поточний стан системи та обчислюється стратегія регулювання, що мінімізує витрати (за допомогою алгоритму числової мінімізації) для відносно короткого часового горизонту в майбутньому: $[t, t + T]$. На рис.3.2 зображено графік дискретної моделі MPC.

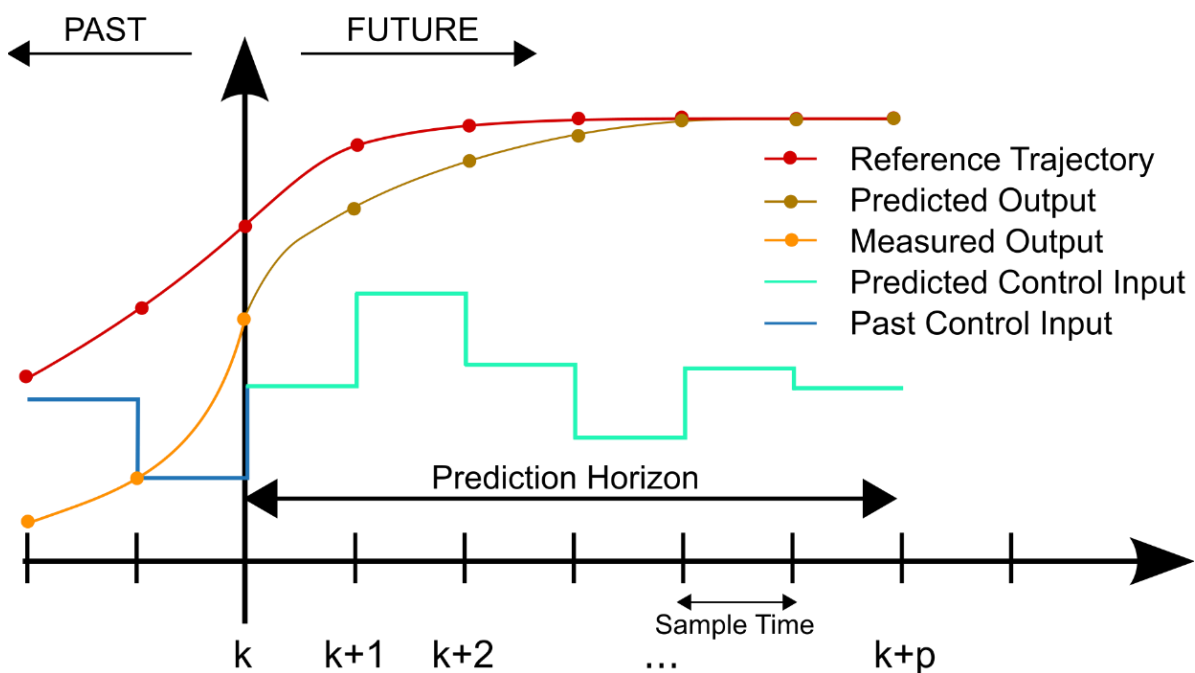


Рис.3.2 Дискретна модель MPC

Зокрема, розрахунок у реальному часі або на льоту використовується для розрахунку траєкторій стану, що впливають із поточного стану, і для пошуку (за допомогою рішення рівнянь Ейлера – Лагранжа) стратегії регулювання, що мінімізує витрати, до часу $t+T$. Застосовується лише перший крок стратегії регулювання, потім знову фіксується стан процесу та обчислення повторюється, починаючи з нового поточного стану, отримуючи нове регулювання та новий прогнозований шлях цього стану. Горизонт прогнозування постійно зміщується вперед, і з цієї причини MPC також називають відступаючим горизонтом регулювання. Хоча такий підхід не є оптимальним, на практиці він дав дуже хороші результати. Було проведено багато академічних досліджень для пошуку швидких методів розв’язання рівнянь типу Ейлера – Лагранжа, для розуміння властивостей глобальної стійкості локальної оптимізації MPC та загалом для вдосконалення методу MPC [22].

MPC - це багатоваріантний алгоритм регулювання, який використовує:

- внутрішню динамічну модель процесу
- функцію витрат J протягом відступаючого горизонту
- алгоритм оптимізації, що мінімізує функцію витрат J , використовуючи вхід регулювання u

Приклад квадратичної функції витрат для оптимізації наведено:

$$J = \sum_{i=1}^N w_{x_i} (r_i - x_i)^2 + \sum_{i=1}^N w_{u_i} \Delta u_i^2 \quad (3.16)$$

без порушення обмежень (низькі/високі межі) з:

$x_i : i^{th}$ контрольована змінна (наприклад, виміряна температура)

$r_i : i^{th}$ контрольна змінна (наприклад, необхідна температура)

$u_i : i^{th}$ маніпульована змінна (наприклад, регулюючий клапан)

w_{x_i} ваговий коефіцієнт, що відображає відносну важливість x_i

w_{u_i} ваговий коефіцієнт, що зменшує відносно великі зміни у u_i

3.3. Нелінійні MPC

Нелінійна модель прогнозуючого керування (NMPC - Nonlinear Model Predictive Control) - це варіант модельного прогнозного контролю (MPC), яка характеризується використанням нелінійних системних моделей у прогнозуванні. Як і в лінійних MPC, NMPC вимагає ітеративного вирішення задач оптимального регулювання на кінцевому горизонті прогнозування.

Чисельне вирішення задачі оптимального регулювання NMPC, як правило, базується на прямих оптимальних методах регулювання, що використовують Ньютонівські схеми оптимізації, в одному з таких варіантів як:

- пряма одиночна стрільба (це метод для розв'язку крайової задачі зведенням її до розв'язання задачі початкових значень)
- прямі багаторазові методи стрільби
- метод прямої колокації (метод числового розв'язання звичайних диференціальних рівнянь, диференціальних рівнянь з частковими похідними та інтегральних рівнянь) [8].

Алгоритми NMPC зазвичай використовують той факт, що послідовні задачі оптимального регулювання подібні одна до одної. Це дозволяє ефективно ініціалізувати процедуру вирішення задачі Ньютона за допомогою відповідного зміщення припущення від раніше розрахованого оптимального рішення, заощаджуючи значну кількість обчислювального

часу. Подібність наступних задач навіть більше використовується за допомогою цих алгоритмів (або "ітерацій у режимі реального часу"), які ніколи не намагаються повторити будь-яку проблему оптимізації до конвергенції, а замість цього виконують лише кілька ітерацій для вирішення останньої задачі NMPC перед переходом до наступної, яка належним чином ініціалізується [20].

У той час як програми NMPC раніше використовувались в основному в переробній та хімічній промисловості із порівняно низькими частотами дискретизації, NMPC все частіше застосовується з вдосконаленням апаратного забезпечення та обчислювальних алгоритмів регулятора, наприклад, попереднє кондиціонування [21], для застосування з високою частотою дискретизації, наприклад, в автомобільній промисловості, або навіть коли стани розподілені в просторі [22]. Нещодавно, як аерокосмічну програму, NMPC застосовували для відстеження оптимальних траєкторій слідування місцевості в режимі реального часу [23].

3.4. Explicit MPC

Explicit MPC (eMPC) дозволяє швидко оцінити закон регулювання для деяких систем, на відміну від MPC реального часу. eMPC працює на техніці параметричного програмування, де рішення проблеми управління MPC, сформульоване як задача оптимізації, попередньо обчислюється в автономному режимі [24]. Це офлайн-рішення, тобто закон регулювання, часто має форму кусково-лінійної функції (PWA), отже, регулятор eMPC зберігає коефіцієнти PWA для кожної підмножини (контрольної області) простору станів, де PWA константа, а також коефіцієнти деяких параметричних відображень усіх регіонів. Кожна область виявляється

геометрично опуклим багатогранником для лінійного MPC, який зазвичай параметризується коефіцієнтами для його граней, що вимагає аналізу точності квантування [25]. Потім отримання оптимальної дії регулювання зводиться до першого визначення області, що містить поточний стан та другої простої оцінки PWA з використанням коефіцієнтів PWA, збережених для всіх областей. Якщо загальна кількість регіонів невелика, використання eMPC не вимагає значних обчислювальних витрат (порівняно з MPC у режимі реального часу) і підходить для систем регулювання зі швидкою динамікою [26]. Серйозним недоліком eMPC є експоненціальне зростання загальної кількості контрольних областей щодо деяких ключових параметрів керованої системи, наприклад, кількості станів, тим самим різко збільшуючи вимоги до пам'яті регулятора і роблячи перший крок оцінки PWA, - тобто пошук поточної контрольної області є досить обчислювально витратним.

Висновки до третього розділу

В рамках третього розділу було розглянуто методологію MPC (Model Predictive Control). У своєму складі вона має такі алгоритми, як Linear MPC, Nonlinear MPC та Explicit MPC. Для вирішення задачі контролю температури у системі прийняття рішень було обрано Linear MPC, оскільки вона оптимально справляється з завданням регулювання таких залежностей.

РОЗДІЛ 4. РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ

4.1. Опис Back-End частини

Back-End частина була виконана на основі системи з прийняття рішень та, як і всі сервери має свій клієнтський API. На цей API може підключатись будь-яка клієнтська частина (Front-End). Загальну архітектуру наведено на рис. 4.1.

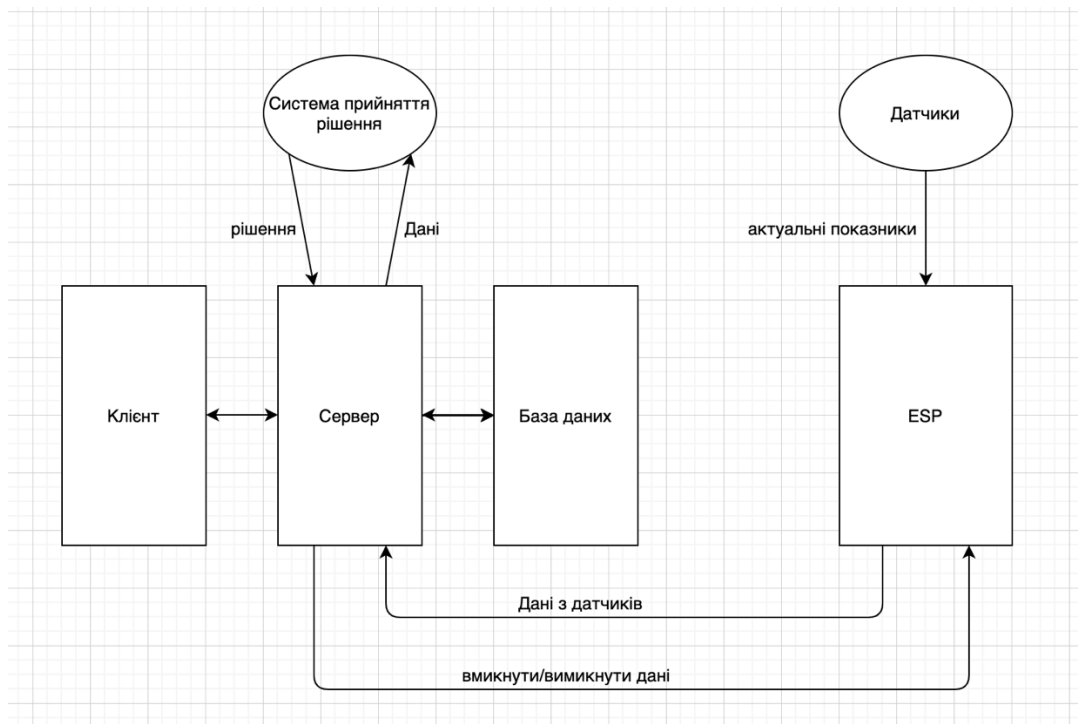


Рис. 4.1 Загальна архітектура взаємодії системи

За проміжок часу τ (на ESP8266 було встановлено 10 сек) прибувають дані про температуру, вологість, атмосферний тиск, реальний час. Як базу даних було вибрано хмарне сховище Firebase від Google. Дане сховище має можливість зберігати інформацію, яка надходить з датчиків лабораторії, у форматі JSON. Функціонал системи такий, що він вільно може працювати як серверською частиною (Node.js) так і з клієнтською. Основний сервіс - хмарна СУБД класу NoSQL, що дозволяє розробникам

додатків зберігати і синхронізувати дані між декількома клієнтами [35, 36, 37].

Підтримані особливості інтеграції з додатками під операційні системи Android і iOS, реалізовано API для додатків на JavaScript, Java, Objective-C і Node.js, також можливо працювати безпосередньо з базою даних в стилі REST з ряду JavaScript-фреймворків, включаючи AngularJS, React, Vue.js, Ember.js і Backbone.js [38]. Передбачено API для шифрування даних [39]. Згідно зі статистикою, на сьогоднішній день, Firebase від Google використовується частіше за всі інші хмарні платформи.

Для встановлення налаштувань та отримання актуальних налаштувань, а також для передачі даних на Front-end було створено API, який робить можливою взаємодію між клієнтом та сервером.

В роботі була використана база даних в режимі реального часу (Realtime Database) від Firebase. Даний сервіс має свій API для розробників додатків, який дає змогу синхронізовувати роботу між клієнтами [39, 40].

Загальний вигляд інтерфейсу користувача Realtime Database наведено на рис. 4.2.

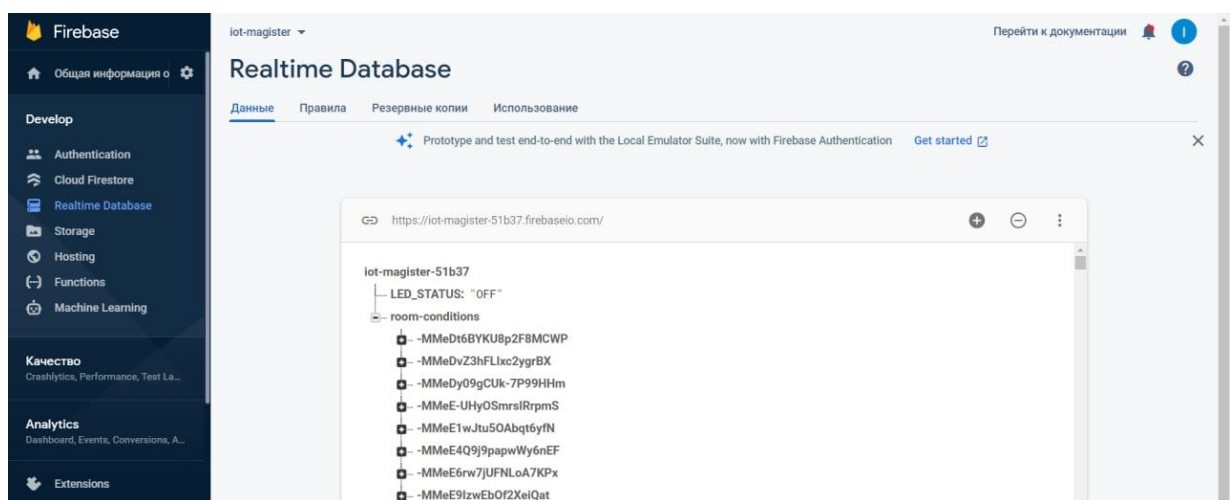


Рис.4.2 Загальний вигляд користувацького інтерфейсу Realtime Database

Якщо розгорнути одне JSON-повідомлення, то можна побачити, всі параметри (температура, вологість, тиск, активність, час), які NodeMCU ESP8266 відправляє по мережі до Realtime Database. Також в режимі реального часу реалізована можливість керування навантаженням. В данному випадку використовувалась світлодіодна стрічка. Шляхом зміни значення поля “LED_STATUS” на “On” або “Off” відбувалось увімкнення або вимкнення навантаження відповідно. На рис.4.3 зображено вигляд JSON-повідомлення, яке відправляє NodeMCU ESP8266.

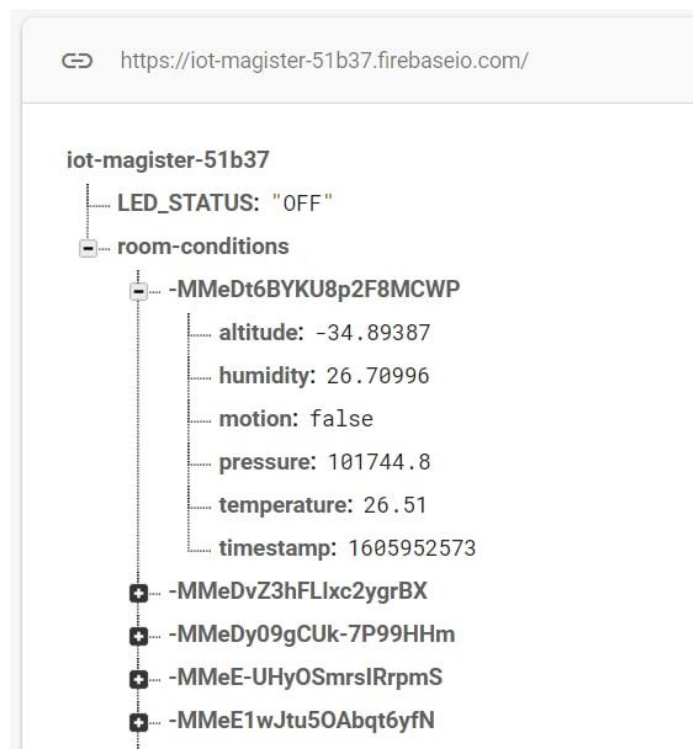


Рис.4.3 Вигляд JSON-повідомлення у Realtime Database

Для того, щоб передавати точний час використовувався NTP. NTP означає Network Time Protocol (протокол мережевого часу). Це стандартний інтернет-протокол для синхронізації годинника комп'ютера з певним еталоном в мережі.

Даний протокол може застосовуватися для синхронізації всіх мережевих пристроїв з всесвітнім координованим часом (UTC) з точністю

до декількох мілісекунд (50 мілісекунд в загальнодоступному Інтернеті і менше 5 мілісекунд в середовищі LAN).

Всесвітній координований час (UTC) - це всесвітній стандарт часу, тісно пов'язаний з GMT (середнім часом за Гринвічем). UTC не змінюється, воно однаково у всьому світі.

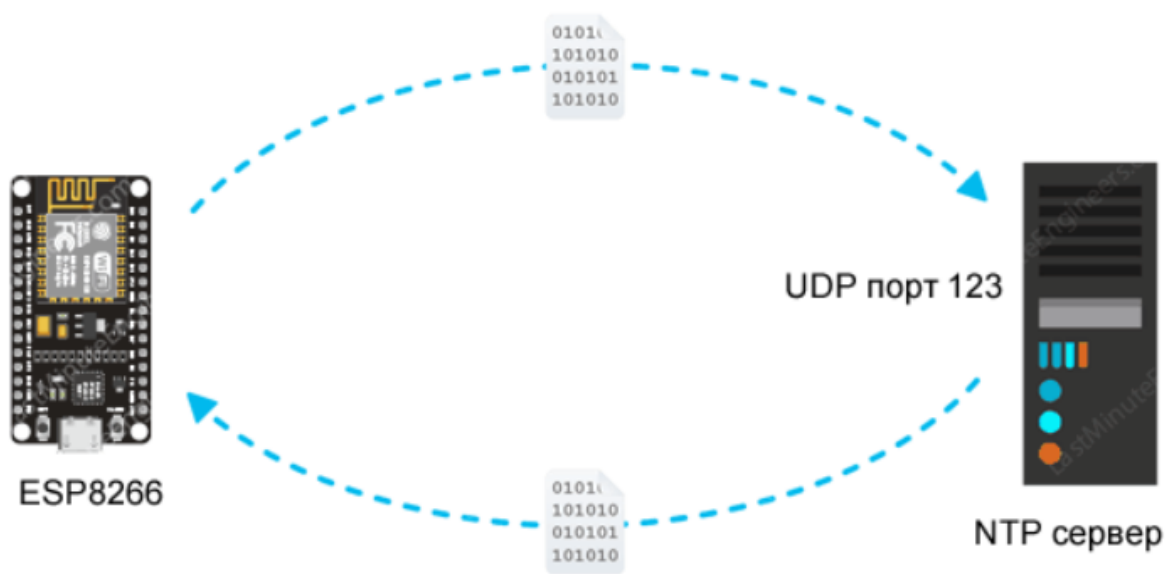


Рис.4.4 Робота з NTP-сервером: передача пакета запиту та пакета мітки часу

NTP встановлює годинник комп'ютерів в форматі UTC, зміщення місцевого часового поясу або зсув літнього часу застосовується вже клієнтом. Таким чином, клієнти можуть синхронізуватися з серверами незалежно від місця розташування і різницю часових поясів.

NTP може працювати різними способами. Найбільш поширена конфігурація - робота в режимі клієнт-сервер. Основний принцип роботи наступний:

- клієнтський пристрій, ESP8266 під'єднується до сервера через протокол користувальницьких датаграм (UDP) через порт 123;
- потім клієнт надсилає пакет запиту на сервер NTP;

- у відповідь на цей запит сервер NTP відправляє пакет з міткою часу;
- пакет з міткою часу містить безліч інформації, такої як мітка часу UNIX, точність, затримка або часовий пояс;
- потім клієнт може проаналізувати поточні значення дати і часу.

4.2 Інструменти та реалізація Back-End частини

В якості середовища, де можна отримати вільний доступ з будь-якого пристрою може бути використане web-середовище.

Back-End частину було розроблено за допомогою Node.js.:

```
const express = require('express')
const cors = require('cors')
const router = require('./post.route')
const bodyParser = require('body-parser')

const app = express()
const PORT = process.env.PORT || 3002

app.use(cors())
app.use(bodyParser.urlencoded({ extended: true }))
app.use(bodyParser.json())
app.use(express.static('dist'))
app.use('/', router)

app.listen(PORT, function () {
  console.log('Server is running on Port:', PORT)
})
```

У Firebase config-файлі було прописане підключення до RealTime Database:

```
module.exports = {
  apiKey: 'AlzaSyChET9qKbZmPTd4oYCuApLqUv8_n_3hgCQ',
  authDomain: 'iot-magister-51b37.firebaseio.com',
  databaseURL: 'https://iot-magister-51b37.firebaseio.com',
  projectId: 'iot-magister-51b37',
  storageBucket: 'iot-magister-51b37.appspot.com',
  messagingSenderId: '735856257117',
  appId: '1:735856257117:web:8b6f59146dd25f8a5005b4'
}
```


Для того, щоб можна було в режимі реального часу передавати дані, реагувати на зміни між базою даних та сервером, був використаний сокетний підхід.

Розрахунок середніх значень температури та вологості приміщення вказано у наступному фрагменті коду:

```
const store = firebase.firestore()
const db = firebase.database()
const router = express.Router()

db.ref('room-conditions').orderByKey().limitToLast(1).on('value', async snapshot => {
  try {
    await getAverage(store, snapshot)
  } catch (e) {
    console.log(e)
  }
})
```

де getAverage – функція підрахунку та оновлення «середнього» у базі даних:

```
async function getAverage (store, snapshot) {
  await store.collection('average').get().then(docs => {
    docs.forEach(doc => {
      let snapshotValue = {}
      snapshot.forEach(item => {
        snapshotValue = item.val()
      })
      store.collection('average').doc(doc.data().key).set({
        value: (+doc.data().value * (doc.data().count) + snapshotValue[doc.data().key]) /
(doc.data().count + 1),
        count: doc.data().count + 1,
        key: doc.data().key
      })
    })
  })
}

module.exports = getAverage
```

Для прийняття рішення було створено модуль на основі МРС (див. розділ 3), який на базі даних тренувань аналізує дані за тривалий проміжок часу та на їх основі приймає рішення щодо стабілізації температури відносно деякого значення або ж аварійного відключення навантаження. Модель МРС, яка використовувалась в роботі була проаналізована у середовищі Matlab. На рис.4.5 зображено графіки реальних та

прогнозованих температур разом з графіками моделей керування навантаженням.

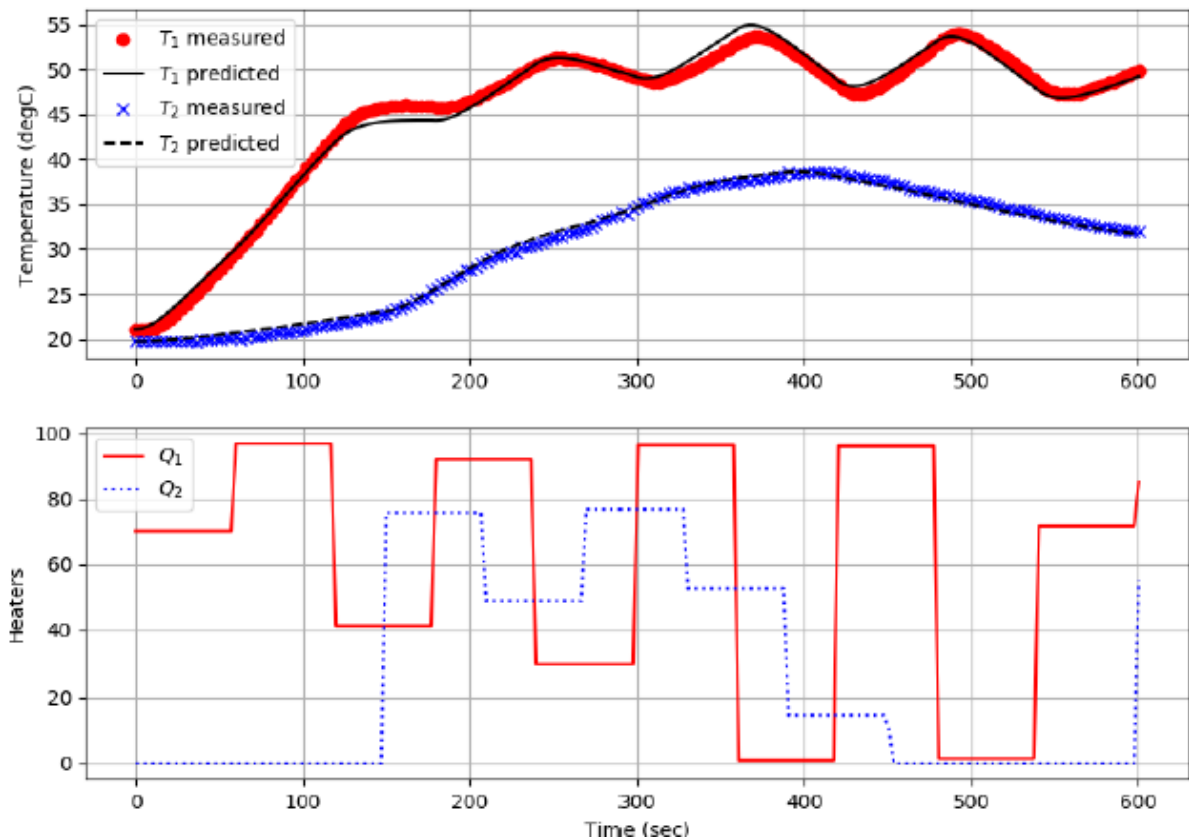


Рис.4.5 Графіки виміряних та прогнозованих температур за допомогою MPC

4.3. Опис реалізації Front-End частини - користувацького інтерфейсу

Як **Front-End** фреймворк було здійснено вибір на користь Vue.js. Використовуючи Vue.js можна розробляти сучасні великі веб-додатки, які працюють без перезавантаження сторінки та надають легку взаємодію користувача та системи. Його мета бути швидким, простим і масштабованим. React js відповідає тільки за користувацький інтерфейс. Дана особливість відповідає шаблону MVC (модель-вид-контролер). Може бути застосоване у комбінації з іншими JavaScript бібліотеками.

Наступним кроком після верстки, обробки даних з Back-End було створено сторінку користувацького інтерфейсу, показану на рис 4.6.

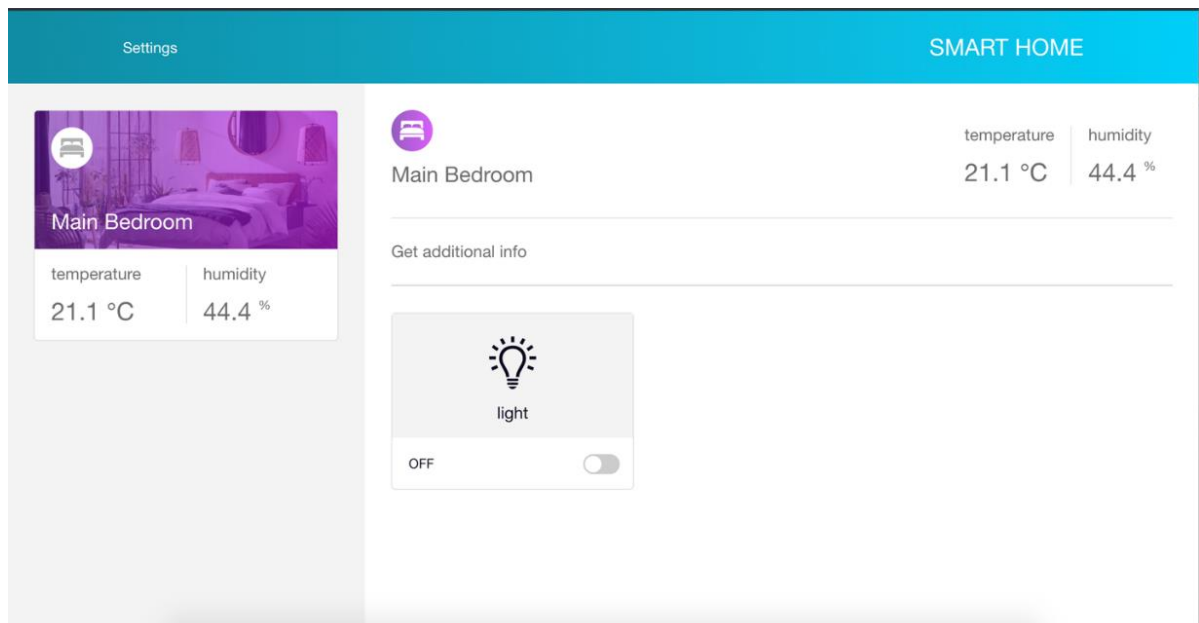


Рис 4.6. Головна сторінка користувацького інтерфейсу Front-End

У своєму складі програма містить сторінку налаштувань системи, де можна обрати, як часто будуть оновлюватись дані на головному екрані. Знаходиться ця сторінка за ендпоінтом налаштувань (“/settings”).

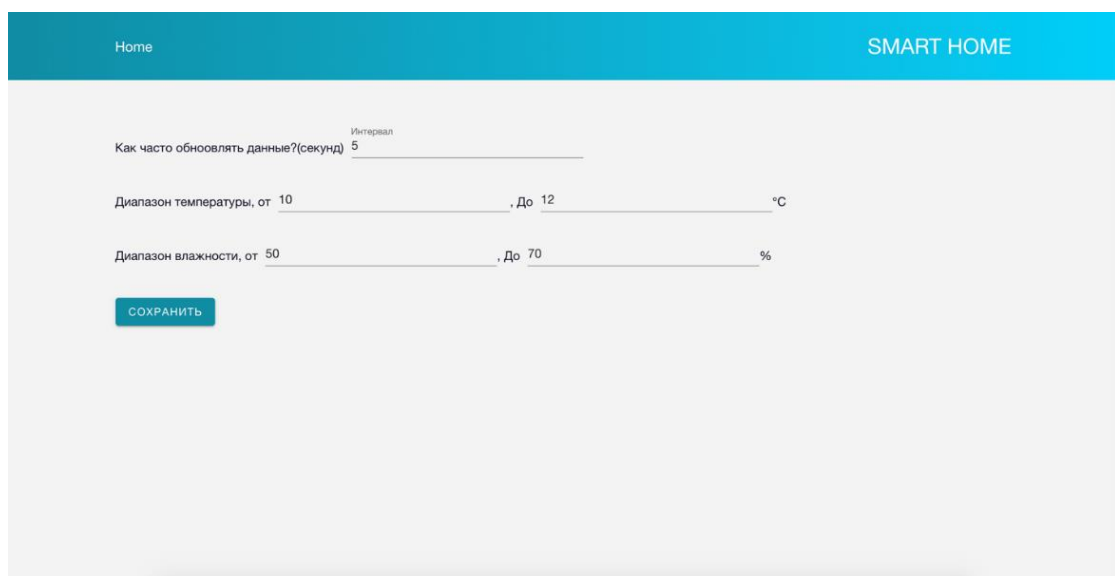


Рис 4.7. Сторінка налаштувань системи

Додаткова інформація про числові показники датчиків температури, вологості та тиску має місце на головній сторінці. Також за кожним отриманим значенням закріплений час, коли саме були прийняті показники. Розширений головний екран можна побачити на рис. 4.8

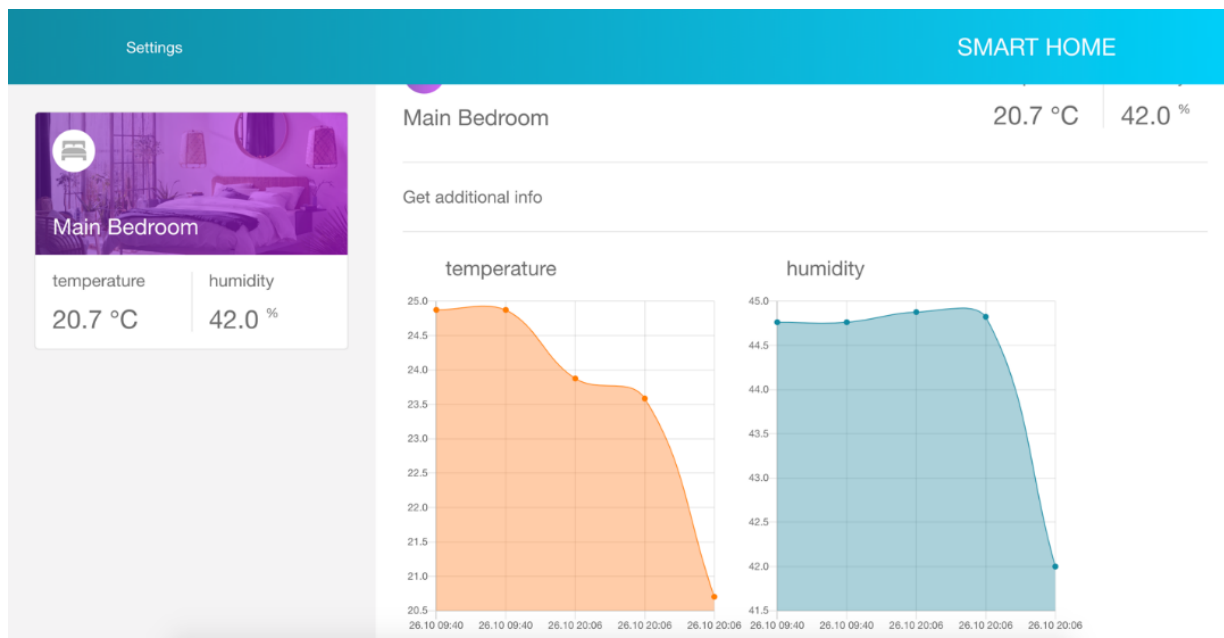


Рис 4.8. Розширений головний екран

Код розробленого програмного забезпечення знаходиться у відкритому доступі на Github за посиланням: <https://github.com/savefimvasil/iot-magister>

Висновки до четвертого розділу

В якості розробки фронтенд забезпечення було використано фреймворк Vue.js. Даний фреймворк використовує мову JavaScript, HTML та CSS. За сервер, в якості Back-End частини було обрано Node.js, де знаходиться логіка обробки даних які приходять з ESP8266 до Realtime Database у Firebase. Даний сервер вміє оброблювати як користувацькі побажання, так і приймати власні рішення щодо підключення/відключення навантаження.

В ході досліджень було виявлено, що зі збільшенням кількості вхідних даних підвищується точність прийняття рішення.

РОЗДІЛ 5. РОЗРОБЛЕННЯ СТАРТАП – ПРОЕКТУ

Стартапи створюються для того, щоб знаходити рішення для тих проблем і завдань, які з плином часу неможливо вирішити без застосування здобутків технічного прогресу. Новітні високотехнологічні розробки мають переслідувати одну мету: робити легшим життя користувачів шляхом спрощування будь-яких дій в їхньому повсякденному житті.

Сприяти руху технічного прогресу вперед – не є задачею того, хто відкриває стартап. Справді, коли мова йде про стартапи, мало ймовірно що буде згадана нова перспективна компанія, яка займається розробкою фільтруючих наноматеріалів.

Проте цілком можливим у межах однієї невеликої команди є створення унікального програмного продукту, який забезпечить його користувачам інноваційні послуги, і який раптом може стати популярним і затребуваним у видимому майбутньому. Аналогічно, не є надзвичайно складним і створення унікального поєднання «заліза» та програмного комплексу, що забезпечує його роботу.

В рамках магістерської дисертації буде розглянуто перший етап розробки стартап-проекту, а саме маркетинговий аналіз стартап-проекту системи контролю та моніторингу стану повітря.

5.1. Опис ідеї проекту (технології)

В першу чергу було розроблено зміст ідеї та можливі базові потенційні ринки, на яких варто шукати потенційних клієнтів даного проекту. В табл. 5.1 продемонстровано напрямки використання та вигоди для користувача.

Таблиця 5.1

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Зміст ідеї: розробка програмно-апаратної системи “SmartLab” для моніторингу та контролю умов перебування людей у приміщеннях	1. Інформаційні технології	Можливість використання продукту у режимі реального часу та в автономному режимі
	2. Медицина	Експрес-оцінка стану приміщення, де знаходяться пацієнти
	3. Сфера безпеки	Детектування надзвичайних явищ у оселях

Було встановлено 3 напрямки використання ідеї та вигоди для користувача.

Для оцінки бізнес-ідеї, потрібно проаналізувати її потенційні техніко-економічні переваги (різницю з аналогами та заміниками, що вже існують) порівняно із тим, що пропонують конкуренти. З цією метою було обрано 4 конкуренти. В табл. 5.2 показано попередньо встановлені сильні, слабкі та нейтральні характеристики ідеї проекту.

Таблиця 5.2.

№ п/п	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабка сторона)	N (нейтральна сторона)	S (сильна сторона)
		Мій проект	GatorTech MicroGrid	TERVA	MavHome			

1.	Система збору даних	Детектування переміщення людини за часові інтервали і використання МРС за допомогою датчиків	Спостереження за аварійними режимами за допомогою датчиків	Спостереження за людиною за допомогою телефону	Спостереження за людиною за допомогою датчиків без МРС			+
2.	Електронна база даних користувача	Ведеться з повною історією. Є можливість її перегляду	Не ведеться	Ведеться без запису (у режимі онлайн)	Не ведеться			+
3.	Відображення даних для користувача	У реальному часі, без перезавантаження	Із перезавантаженням	Із перезавантаженням	Без перезавантаження використовуючі застарілі технології			+
4.	Детектування аномалій	Прогнозування за допомогою МРС	Не має	Не має	Не має			+

Можна зробити висновок, що ідея є спроможною для конкуренції та володіє унікальними рисами для ринку.

5.2. Технологічний аудит ідеї проекту

На подальшому етапі проведена розробка технологічного аудиту ринку для того, щоб визначити доступність технологій для розробки програмного забезпечення (табл. 5.3).

Таблиця 5.3

№ п/п	Ідея проекту	Технології її реалізації	Наявність технологій	Доступність технологій
1	Створення системи моніторингу «SmartLab» для контролю перебування людей у приміщенні	Розробка, дослідження, програмування;	Не наявна	Доступна
2	Відображення результатів, оптимізація та розробка веб-додатку для відображення даних	Розробка, дослідження, програмування;	Не наявна	Доступна
Обрана технологія реалізації ідеї проекту: самостійна розробка на основі дослідження				

Як наслідок, стало зрозуміло те, що технічно реалізацію продукту реально зробити, але потрібно створити відповідні технології, щоб досягнути поставленої цілі.

5.3. Аналіз ринкових можливостей запуску стартап-проекту

У ході проведення аналізу було визначено можливості, доступні для використання протягом ринкового впровадження проекту, та ринкові загрози, які можуть стати на заваді реалізації проекту. Це дає змогу спроектувати напрями розвитку проекту, враховуючи стан ринкового середовища, потреби потенційних клієнтів та пропозиції конкурентних проектів (табл. 5.4).

Таблиця 5.4

№ п/п	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	400

2	Загальний обсяг продаж, грн/ум.од	10000
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу (вказати характер обмежень)	Немає
5	Специфічні вимоги до стандартизації та сертифікації	Є
6	Середня норма рентабельності в галузі (або по ринку), %	73

Бачимо, що відбувається зростання динаміки ринку, загальний обсяг продажів є значним, і як наслідок, у розвитку ідеї даного проекту є перспективи, він є доцільним. Потім виділяються потенційні групи клієнтів, характеристики, які їм властиві, та складається орієнтовний перелік вимог до товару для кожної з груп (табл. 5.5).

Таблиця 5.5

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Спостереження за станом MicroGrid; передвчасне попередження про проблеми;	Побутовий користувач; Навчальні заклади; Військові та рятувальні служби; Медичні клініки;	Експлуатація як у динаміці так і у статиці, в різних, як складних, так і у умовах спокою;	- до продукції: Точність; Надійність; Дешевизна; Якість; - до компанії-постачальника: Точність; Брендинг та відомість; Гарантійність

Наступним кроком в межах аналізу було виділено цільову аудиторію та її вимоги. Далі, після визначення потенційних груп клієнтів виконується аналіз ринкового середовища: формуються таблиці факторів (табл. 5.6, 5.7), що позитивно впливають на ринкове впровадження проекту, та факторів, що мають негативний вплив.

Таблиця 5.6

№ п/п	Фактор	Зміст загрози	Можлива реакція компанії
1	Перехват передачі інформації	Складність персоналізації інформації	Налагодження системи захисту інформації; розширення серверів для зберігання інформації
2	Конкуренція		
3	Зберігання інформації		

Таблиця 5.7

№ п/п	Фактор	Зміст можливості	Можлива реакція компанії
1	Достовірність і надійність інформації	Переваги при передачі інформації	Маркетинг у цих напрямках для рекомендуванню себе, як компанії, на ринку;
2	Безпомилковість		

Як наслідок маємо той факт, що загрози і можливості є реальними для фізичного подолання. Потім виконується аналіз пропозиції: досліджуються загальні риси конкуренції на ринку (табл. 5.8).

Таблиця 5.8

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
1. Тип конкуренції: чиста	В кого краще - в того купують	Покращення товару та сфери обслуговування
2. За рівнем конкурентної боротьби: локальна	Належить до повсякденного ринку збуту;	Розширення функціоналу та орієнтації користувачів

3. За галузевою ознакою: міжгалузева	Притаманна різним галузям застосування;	Розширення функціоналу та галузей застосування
4. Конкуренція за видами товарів: товарно-родова та товарно-видова	Належить до аналізаторів поведінки людини	Розширення функціоналу пристрою
5. За характером конкурентних переваг: цінова та нецінова	Чим дешевше – тим привабливіше; Чим краще – тим рентабельніше;	Покращення цінової політики та якості товару
6. За інтенсивністю: не марочна	Не жорстка конкуренція	Агресивні та не агресивні форми піару

Етапом, що завершує ринковий аналіз можливостей впровадження проекту є проведення SWOT-аналізу (табл. 5.9) - матриці аналізу сильних (Strength) та слабких (Weak) сторін, загроз (Troubles) та можливостей (Opportunities).

Таблиця 5.9

Сильні сторони:	Слабкі сторони:
Навчання системи за допомогою MPC; Робота програми на усіх пристроях що містять веб-браузер; Використання сучасних технологій Google Firebase; Обробка великих даних (Big Data)	Не захищені дані; Необхідно багато серверів для зберігання Big Data.
Можливості:	Загрози:
Покращення безпомилковості інформації; Надійність захищеності інформації при її передачі;	Передача інформації; Зберігання інформації; Захист інформації;

5.4. Розроблення маркетингової програми стартап-проекту

Початковим кроком у розробці ринкової стратегії є вибір стратегії ринку, тобто обрання стратегії конкурентної поведінки та опис цільових груп потенційних споживачів. (табл. 5.10).

Таблиця 5.10

№ п/п	Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки*
1	З погляду аналізатора – ні, з погляду поєднання MicroGrid і MPC – так	Відбуватиметься пошук нових споживачів, розширюючи функціонал і потенціал продукту, а також існуючі клієнти у конкурентів самовільно будуть використовувати більш кращий продукт	Ні, не буде, так як це зменшить клієнтську базу	Помірна, місцями агресивна

Перший крок: формування маркетингової концепції товару, який буде отриманий споживачем. Щоб це зробити, необхідно скласти разом результати попереднього аналізу конкурентоспроможності товару. (табл. 5.11).

Таблиця 5.11

№ п/п	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами (існуючі або такі, що потрібно створити)
-------	---------	----------------------------	--

1	«Спостереження у Real-Time-Database»; MPC Швидка обробка даних; Прогнозування аномальних явищ;	Реалізація ідеї виявлення аномальної поведінки людини характеризується невисокою вартістю	Створення надійного бренду; Постійний розвиток та апгрейд системи та компанії у всіх напрямках; Дотримуватися схеми ціна – якість; Розширення планів щодо користування сервісом
---	---	---	--

Подальшим кроком є окреслення меж, в яких буде розташовуватись ціна. Потрібно орієнтуватись на ці межі при формуванні ціни потенційного товару (фінальне формування ціни відбувається протягом фінансово-економічного аналізу проекту), в яке входить аналіз ціни на товари-аналоги або товари субститути, а ще аналіз рівня доходів цільової групи споживачів (табл. 5.12).

Таблиця 5.12

№ п/п	Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
1	180-200% від ціни нашого продукту	180-210% від ціни нашого продукту	20000 - 400000 грн зі 100 проданих од.	6000/13000 грн

Подальшим етапом є вибір оптимальної системи збуту, в його рамках приймається рішення (табл. 5.13).

Таблиця 5.13

№ п/п	Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
	Задоволення	Збут товару	Усі можливі	

1	потреб користувача за автоматичним та розумним виявленням аномалій у системі	та задоволення запитуваних потреб клієнтів	канали збуту (глибока)	Власна
2		Збут та реклама товару та задоволення запитуваних потреб клієнтів	Усі можливі канали збуту (глибока)	Залучена

Висновки до п'ятого розділу

В ході проведеного дослідження, стало зрозуміло, що проект, який перебуває на етапі розробки, має перспективи для ринкової комерціалізації. Зростання рівня попиту на аналогічні послуги робить покупку даного програмного забезпечення масовою, і в той же час створює високий рівень конкуренції при виході на ринок, де динаміка ринку є певною мірою сприятливою для проекту, що розроблюється.

Даний проект «SmartLab» є перспективним для впровадження, зважаючи на потенційні групи клієнтів. Ними першочергово є: учбові навчальні заклади, медичні та експериментальні клініки, військові служби та прості рядові користувачі. Входження на ринок може обмежуватися такими бар'єрами як : відсутність масового виробника, високий рівень конкуренції з великими фірмами аналогічних продуктів. Проте, за умови ведення агресивної боротьби в конкурентному середовищі, у проекту є значні шанси та перспективи зарекомендувати бренд, і в майбутньому завоювати місце на ринковій економіці. Наступна імплементація проекту є досить перспективним кроком.

ВИСНОВКИ

Остаточним результатом створення проекту та написання цієї дипломної роботи стало вивчення та дослідження відомих способів застосування Інтернету речей, проведення аналізу алгоритмів прийняття рішення в роботі систем споживання енергії.

Було надано пропозицію щодо конструювання web-системи контролю енергопостачання для розумної лабораторії, яка відповідатиме за керування станом електроприладів, які під'єднано до системи енергопостачання, а також за зберігання всього об'єму показників в хмарному середовищі та надання доступу до клієнтських додатків.

З метою забезпечити гарантію безпеки системи та зберігання ключів було представлено пропозицію використати хмарний сервер, а також таблиці бази даних. Для втілення цих цілей до використання виділено нині загальноприйнятий набір web-технологій: JavaScript, CSS, Vue.js, Nuxt, Node.js та Google Firebase. Розроблена нами послуга може функціонувати на кожній з сучасних платформ.

Велика перевага цього сервісу є в тому, що він характеризується низькими, за сучасним розумінням, технічними вимогами до комп'ютерів користувачів, які є необхідними для завантаження та роботи сервісу як на стороні клієнта, так і сервера.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Петергеря Ю.С., Жуйков В. Я., Терещенко Т.О., «Інтелектуальні системи забезпечення енергозбереження житлових будинків» Навчальний посібник. – К.: Медіа-ПРЕС, 2008 – 256 с.
2. Moskovitz D. «Demand-Side Management in China's restructured power industry» /Zhaoguang Hu, David Moskovitz, Jianping Zhao// Energy sector management assistance program. 2005. 148 p.
3. «H. Jiayi, J. Chuanwen, and X. Rong, "A review on distributed energy resources and MicroGrid," Renewable and Sustainable Energy Reviews, vol. 12, no. 9, pp. 2472-2483, 2008/12/01/ 2008.
4. Duy Long Ha «Realtimes dynamic optimization for demand-side load management» / Duy Long Ha, St'ephane Ploix, Eric Zamaï, Mireille Jacomino // International journal of management science and engineering management. Vol. 3 (2008) №4 P.243-252
5. Electropedia. International Electrotechnical Commission. 2017-12-15. Retrieved 2020-10-06.
6. А. В. Кириленко, Ю. И. Якименко, В. Я. Жуйков, С. П. Денесюк Преобразователи параметров электроэнергии в smart системах энергетики// труды института электродинамики.-спец.выпуск. К:2010г.с17.
7. О. В. Кириленко, Ю. С. Петергеря, Т. О. Терещенко, В. Я. Жуйков. Інтелектуальні системи керування потоками електроенергії у локальних об'єктах. – К.: “Аверс”, 2005.
8. Oviatt S. L. Multimodal Interfaces // The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, Jacko J. and Sears A. (Eds.). Mahwah, NJ: Lawrence Erlbaum Assoc. 2003. P. 286–304. <http://microgridprojects.com/types-of-microgrids>

9. «Model demand side management regulations», ABPS Infra, May, 2010. 17p
10. Duy Long Ha «Realtimes dynamic optimization for demand-side load management» / Duy Long Ha, St'ephane Ploix, Eric Zamai, Mireile Jacomino // International journal of management science and engineering management. Vol. 3 (2008) №4 P.243-252
11. Бараннік В.О. «Стратегія та практика управління паливно-енергетичним комплексом. Досвід України» [Електронний ресурс]/ В.О. Бараннік, М.Г. Земляний. – режим доступу <http://www.db.niss.gov.ua/docs/energy/58.htm>. 2.10.2013.
12. Єрмілов С. Ф. «Державна політика енергоефективності в українському та європейському контексті» [Електронний ресурс] / С.Ф. Єрмілов. – режим доступу: http://www.esco-ecosys.narod.ru/2011_2/art044.pdf. – 2.10.2013.
13. Bill Moran «Microgrid load management and control strategies» 2016 IEEE/PES Transmission and Distribution Conference and Exposition (T&D), 2016.
14. Интриллигатор М. «Математические методы оптимизации и экономическая теория» / Пер. с англ. – М.:Прогресс 1975. – 606 с.
15. Chandola V., Banerjee A., Kumar V. Anomaly detection: A Survey // ACM Computing Surveys. 2009. Vol. 41, no. 3. Article 15. DOI: 10.1145/1541880.1541882
16. Michèle Arnold, Göran Andersson. "Model Predictive Control of energy storage including uncertain forecasts" https://www.psc-central.org/uploads/tx_ethpublications/fp292.pdf
17. Tobias Geyer: Model predictive control of high power converters and industrial drives, Wiley, London, ISBN 978-1-119-01090-6, Nov. 2016.

18. Vichik, Sergey; Borrelli, Francesco (2014). "Solving linear and quadratic programs with an analog circuit". *Computers & Chemical Engineering*. 70: 160–171. doi:10.1016/j.compchemeng.2014.01.011.
19. Zafiriou, E. and M. Morari (1985a). Digital controllers for SISO systems. A review and a new algorithm. *Int. J. Control*, 42, 855-876.
20. Jump up to:a b Wang, Liuping (2009). *Model Predictive Control System Design and Implementation Using MATLAB®*. Springer Science & Business Media. pp. xii.
21. Al-Gherwi, Walid; Budman, Hector; Elkamel, Ali (3 July 2012). "A robust distributed model predictive control based on a dual-mode approach". *Computers and Chemical Engineering*. 50 (2013): 130–138. doi:10.1016/j.compchemeng.2012.11.002.
22. Michael Nikolaou, *Model predictive controllers: A critical synthesis of theory and industrial needs*, *Advances in Chemical Engineering*, Academic Press, 2001, Volume 26, Pages 131-204
23. An excellent overview of the state of the art (in 2008) is given in the proceedings of the two large international workshops on NMPC, by Zheng and Allgower (2000) and by Findeisen, Allgöwer, and Biegler (2006).
24. J.D. Hedengren; R. Asgharzadeh Shishavan; K.M. Powell; T.F. Edgar (2014). "Nonlinear modeling, estimation and predictive control in APMonitor". *Computers & Chemical Engineering*. 70 (5): 133–148. doi:10.1016/j.compchemeng.2014.04.013.
25. Ohtsuka, Toshiyuki (2004). "A continuation/GMRES method for fast computation of nonlinear receding horizon control". *Automatica*. 40 (4): 563–574. doi:10.1016/j.automatica.2003.11.005.
26. Knyazev, Andrew; Malyshev, Alexander (2016). "Sparse preconditioning for model predictive control". 2016 American Control

Conference (ACC). pp. 4494–4499. arXiv:1512.00375.
doi:10.1109/ACC.2016.7526060. ISBN 978-1-4673-8682-1. S2CID 2077492.

27. M.R. García; C. Vilas; L.O. Santos; A.A. Alonso (2012). "A Robust Multi-Model Predictive Controller for Distributed Parameter Systems" (PDF). *Journal of Process Control*. 22 (1): 60–71. doi:10.1016/j.jprocont.2011.10.008.

28. R. Kamyar; E. Taheri (2014). "Aircraft Optimal Terrain/Threat-Based Trajectory Planning and Control". *Journal of Guidance, Control, and Dynamics*. 37 (2): 466–483. Bibcode:2014JGCD...37..466K. doi:10.2514/1.61339.

29. Bemporad, Alberto; Morari, Manfred; Dua, Vivek; Pistikopoulos, Efstratios N. (2002). "The explicit linear quadratic regulator for constrained systems". *Automatica*. 38 (1): 3–20. doi:10.1016/s0005-1098(01)00174-1.

30. Knyazev, Andrew; Zhu, Peizhen; Di Cairano, Stefano (2015). "Explicit model predictive control accuracy analysis". 2015 54th IEEE Conference on Decision and Control (CDC). pp. 2389–2394. arXiv:1509.02840. Bibcode:2015arXiv150902840K. doi:10.1109/CDC.2015.7402565. ISBN 978-1-4799-7886-1. S2CID 6850073.

31. Klaučo, Martin; Kalúz, Martin; Kvasnica, Michal (2017). "Real-time implementation of an explicit MPC-based reference governor for control of a magnetic levitation system". *Control Engineering Practice*. 60: 99–105. doi:10.1016/j.conengprac.2017.01.001.

32. Scokaert, P.O.M.; Mayne, D.Q. (1998). "Min-max feedback model predictive control for constrained linear systems". *IEEE Transactions on Automatic Control*. 43 (8): 1136–1142. doi:10.1109/9.704989

33. I.A., Smith L.S. A Neural Network Approach to Time Series Forecasting // *Proceedings of the World Congress on Engineering, London, 2009, Vol 2 P. 1292 - 1296.*

34. Pradhan R.P., Kumar R. Forecasting Exchange Rate in India: An Application of Artificial Neural Network Model // Journal of Mathematics Research. 2010, Vol. 2, No. 4. P. 111 - 117.
35. Melendez, Steven Sometimes You're Just One Hop From Something Huge (May 27, 2014).
36. Farr, Christina Firebase's scalable backend makes it '10 times easier' to build apps (February 13, 2013).
37. Marshall, Matt Firebase is building a Dropbox for developers (August 29, 2013). Firebase Integrations. , Inc..
38. Darrow, Barb Firebase secures its real-time back-end service (DEC. 18, 2012).
39. Farr, Christina (February 13, 2013). Firebase's scalable backend makes it '10 times easier' to build apps. VentureBeat.
40. Marshall, Matt (August 29, 2013). Firebase is building a Dropbox for developers. VentureBeat. <https://radioprogram.ru/post/910>
41. Пікож А.В. «Прогнозування електроспоживання на базі нейромереж»

ABSTRACT

With the development of home automation systems, and in a more global context - intelligent control systems for power consumption, including distributed generation in the presence of alternative renewable power sources, more and more attention is paid to the development of those researches in the direction of monitoring the internal environment. parameters of microclimate, biotelemetric indicators of the person as the user or the system operator. These, as well as many other tasks related to the creation and operation of energy information infrastructure, are solved within modern Microgrid systems, defined as integrated power systems for the distribution of energy resources of different types of electrical loads, acting together as a single system - autonomously, in parallel with the existing power grid, either in the "island" or separate from the existing network modes.

The concept of the Internet of Things (IoT - Internet of Things) has been successfully used to coordinate modes of work and information connections between devices in order to be a part of such systems. set of interconnected technologies:

- 1) measurement, transmission, processing and storage of data;
- 2) robotics;
- 3) artificial intelligence, including methods of machine guidance, fuzzy logic, artificial neural networks;
- 4) processing and storage of large data (Big Data).

Wide stiffening of the built-in systems and technologies of collecting and transfer of the information in combination with settings and algorithms of data processing promote increasing popularity and expansion of spheres of a stop of the Internet revision even on first levels.

As a result of designing, the existing ways of using the Internet of Things are studied, the analysis of decision-making algorithms for energy consumption control systems is carried out.

A web-based power supply control system in a smart laboratory has been proposed, which will allow controlling the status of electrical appliances connected to the power supply system and will store all indicators in the cloud environment and provide access for customer applications. It is proposed to use a cloud server and database tables to store keys, which ensures the security of the system. A standard modern set of web-technologies has been chosen for implementation: JavaScript, CSS, Vue.js, Nuxt, Node.js, Google Firebase. The general structure of a system is shown in Fig.1.

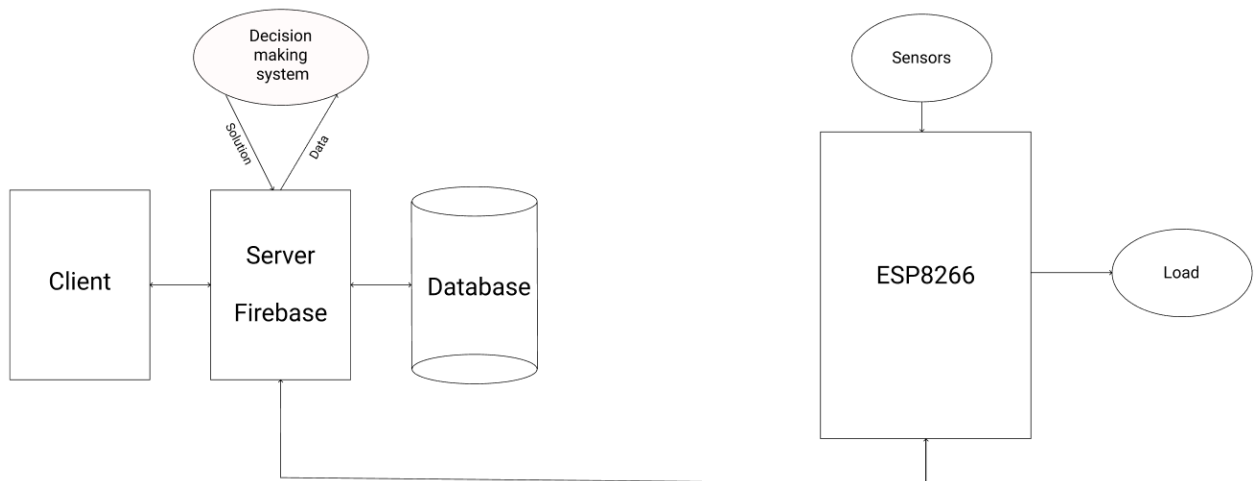


Fig.1 General structure of a system

The developed service works on all modern platforms, and has low, by modern standards, technical requirements for loading and deployment on the client and server side.

```

// post.route.js

const express = require('express')
const path = require('path')
const config = require('./config')
const firebase = require('firebase')
const getAverage = require('./helpers/getAverage')
const detectNormal = require('./helpers/detectNormal')

firebase.initializeApp(config)

const store = firebase.firestore()
const db = firebase.database()
const router = express.Router()

function setIntervallImmediately (func, interval) {
  func()
  return setInterval(func, interval)
}

function toggleLight () {
  const ref = db.ref('LED_STATUS')
  ref.once('value').then(snapshot => {
    if (snapshot.val() === 'ON') {
      let state = 'OFF'

      const myInterval = setIntervallImmediately(_ => {
        ref.set(state)
        state = state === 'OFF' ? 'ON' : 'OFF'
      }, 1000)

      // clear interval after 4.5 seconds
      setTimeout(_ => {
        clearInterval(myInterval)
        ref.set('OFF')
      }, 4500)
    }
  })
}

db.ref('room-conditions').orderByKey().limitToLast(1).on('value', async snapshot => {
  snapshot.forEach(item => {
    const t = item.val().temperature
    const h = item.val().humidity

    if (!detectNormal('temperature', t)) {
      toggleLight()
    } else {
      // normal
      console.log('normal temp')
    }

    if (!detectNormal('humidity', h)) {
      toggleLight()
    } else {
      // normal
      console.log('normal humidity')
    }
  })
})

```



```

    try {
      await store.collection('settings').doc('settings').get().then(el => {
        // todo set timeout period in ESP
        console.log('timeout', el.data().timeout)
      })

      await getAverage(store, snapshot)
    } catch (e) {
      console.log(e)
    }
  })

  router.get('*', (req, res) => {
    res.sendFile(path.join(__dirname, './dist', 'index.html'))
  })

  router.route('/settings').post(function (req, res) {
    try {
      const body = req.body
      store.collection('settings').doc('settings').set({
        timeout: body.timeout,
        temperature: body.temperature,
        humidity: body.humidity
      })
      res.json({ status: 200, message: 'success' })
    } catch (e) {
      console.log(e)
      res.json({ status: 500, message: 'unable to set timeout' })
    }
  })

  router.route('/get-settings').post(function (req, res) {
    try {
      store.collection('settings').doc('settings').get().then(el => {
        res.json(el.data())
      })
    } catch (e) {
      res.json({ status: 500, message: 'unable to get timeout' })
    }
  })

  router.route('/get-last').post(function (req, res) {
    try {
      db.ref('room-conditions').orderByKey().limitToLast(100).once('value', snapshot => {
        res.json(snapshot.val())
      })
    } catch (e) {
      res.json({ status: 500, message: 'unable to get timeout' })
    }
  })

  module.exports = router

// kmeans.js

'use strict'
const kmeans = function (data, nClusters) {
  if (typeof data === 'undefined') throw new Error('Sample data is missing.')

```

```

if (typeof nClusters === 'undefined') nClusters = 2
const self = {}

let nfeatures = null
let nSamples = null

const centroids = []
const labels = []
let inertia = 0

// check data
nfeatures = data.length
nSamples = data[0].length
for (let f = 0; f < nfeatures; f++) {
  if (nSamples !== data[f].length) { throw new Error('nSamples are not the same for all features.') }
}

// get data boundaries
const boundaries = []
for (let f = 0; f < nfeatures; f++) {
  boundaries[f] = { min: Infinity, max: -Infinity }
  for (let s = 0; s < nSamples; s++) {
    if (data[f][s] < boundaries[f].min) { boundaries[f].min = data[f][s] }
    if (data[f][s] > boundaries[f].max) { boundaries[f].max = data[f][s] }
  }
}

// start centroids
for (let c = 0; c < nClusters; c++) {
  centroids[c] = []
  for (let f = 0; f < nfeatures; f++) {
    centroids[c][f] = Math.random() * (boundaries[f].max - boundaries[f].min) + boundaries[f].min
  }
}

const step = function () {
  // reset inertia
  inertia = 0

  // pick closest centroid
  for (let s = 0; s < nSamples; s++) {
    let distance = Infinity
    let label = 0

    for (let c = 0; c < nClusters; c++) {
      let _distance = 0

      for (let f = 0; f < nfeatures; f++) { _distance += (data[f][s] - centroids[c][f]) * (data[f][s] - centroids[c][f]) }
      if (_distance < distance) {
        distance = _distance
        label = c
      }
    }

    inertia += distance
    labels[s] = label
  }

  // move centroids towards center of their own cluster
  for (let c = 0; c < nClusters; c++) {
    for (let f = 0; f < nfeatures; f++) {

```

```

    let meanSum = 0
    let meanCount = 0

    for (let s = 0; s < nSamples; s++) {
      if (labels[s] === c) {
        meanSum += data[f][s]
        meanCount++
      }
    }

    centroids[c][f] = meanSum / meanCount
  }
}

return {
  centroids: centroids,
  labels: labels,
  inertia: inertia
}
}

self.step = step

// eslint-disable-next-line no-unused-vars
self.predict = function (maxIterations) {
  if (typeof maxIterations === 'undefined' || !Number.isInteger(maxIterations)) maxIterations = 1000

  let previousCentroids = null
  for (let i = 0; i < maxIterations; i++) {
    previousCentroids = centroids.toString()
    step()
    if (centroids.toString() === previousCentroids) break
  }
  return {
    centroids: centroids,
    labels: labels,
    inertia: inertia
  }
}

return self
}

if (typeof exports !== 'undefined') {
  if (typeof module !== 'undefined' && module.exports) {
    exports = module.exports = kmeans
  }
  exports.kmeans = kmeans
}

// pages/index.vue

<template>
  <div class="wrapper">
    <BaseLeftBar class="wrapper__left-bar" :menu="coloredMenu" @set-active-card="setInfo" />
    <BaseRightBar class="wrapper__right-bar" :activeRoom="menu[activeRoom]" />
  </div>
</template>

<script>
import axios from 'axios'

```

```

import { mapGetters, mapActions } from 'vuex'

export default {
  data () {
    return {
      activeRoom: 0,
      menu: [
        {
          title: 'SmartLab',
          image: 'smartlab.png',
          icon: 'experiment',
          temperature: {
            label: 'temperature',
            value: '29'
          },
          humidity: {
            label: 'humidity',
            value: '16'
          }
        }
      ],
      interval: null,
      humidity: 1,
      ref: null
    }
  },
  computed: {
    ...mapGetters({ getSettings: 'settings/getSettings' }),
    coloredMenu () {
      const menu = [...this.menu]
      let index = 0
      const colors = ['purple', 'green', 'orange', 'blue']
      menu.forEach((item) => {
        item.color = colors[index]
        index++
        if (index > colors.length - 1) index = 0
      })
      return menu
    }
  },
  async mounted () {
    this.ref = this.firebase.database().ref('room-conditions').orderByKey().limitToLast(1)
    this.ref.on('value', this.firebaseCallback)
  },
  beforeDestroy () {
    // clearInterval(this.interval)
    this.ref.off('value', this.firebaseCallback)
  },
  methods: {
    ...mapActions({ setGraphInfo: 'temperature/setGraphInfo', clearGraphInfo: 'temperature/clearGraphInfo' }),
    firebaseCallback (snapshot) {
      snapshot.forEach(item => {
        this.menu[0].temperature.value = item.val().temperature.toFixed(1)
        this.menu[0].humidity.value = item.val().humidity.toFixed(1)

        this.setGraphInfo(item.val())
      })
    },
    setInfo (index) {
      this.activeRoom = index
    }
  },

```

```

    async changeSettings () {
      const url = window.location.href.includes('localhost') ? 'http://localhost:3002/settings' : '/settings'
      try {
        await axios.post(url, { timeout: 1 })
      } catch (e) {
        console.log(e)
      }
    }
  }
}
</script>

```

//chart.js

```

import { Line, mixins } from 'vue-chartjs'
const { reactiveProp } = mixins

```

```

export default {
  extends: Line,
  mixins: [reactiveProp],
  props: ['options', 'color'],
  data () {
    return {
      chart: null
    }
  },
  watch: {
    chartData: {
      deep: true,
      handler (val) {
        this.$data._chart.update()
      }
    }
  },
  mounted () {
    this.renderChartMethod()
  },
  methods: {
    renderChartMethod () {
      this.renderChart({
        labels: this.chartData.labels,
        datasets: [
          {
            pointBackgroundColor: this.color === 'red' ? 'rgba(255,126,0,1)' : this.color === 'blue' ?
'rgba(19,139,163,1)' : 'rgba(35,163,57,1)',
            pointBorderColor: this.color === 'red' ? 'rgba(255,126,0,1)' : this.color === 'blue' ? 'rgba(19,139,163,1)' :
'rgba(35,163,57,1)',
            borderWidth: 1,
            responsive: false,
            borderColor: this.color === 'red' ? 'rgba(255,126,0,1)' : this.color === 'blue' ? 'rgba(19,139,163,1)' :
'rgba(35,163,57,1)',
            fontSize: 18,
            backgroundColor: this.color === 'red' ? 'rgba(255,126,0,0.4)' : this.color === 'blue' ? 'rgba(19,139,163,0.4)'
: 'rgba(35,163,57,0.4)',
            data: this.chartData.values
          }
        ]
      }, {
        legend: {
          display: false
        },
        responsive: false,
        animation: {
          tension: {
            duration: 100,

```

```

        easing: 'linear',
        from: 0.1,
        to: 0,
        loop: true
      }
    }
  })
}
}
}
//BaseltemCard
<template>
  <div class="item-card">
    <div class="item-card__image">
      <component :is="card.icon" />
      <h5 class="item-card__title">
        {{ card.name }}
      </h5>
    </div>
    <div class="item-card__wrapper">
      <span>{{ card.enabled ? 'ON': 'OFF' }}</span>
      <BaseToggle :defaultValue="isChecked" @setIsActive="changeActive" />
    </div>
  </div>
</template>

<script>
import light from '~/static/icons/light.svg'

export default {
  name: 'BaseltemCard',
  components: {
    light
  },
  props: {
    card: Object
  },
  data () {
    return {
      check: null,
      isChecked: !!this.card.enabled,
      isLoading: false,
      ref: null
    }
  },
  watch: {
    isChecked (val) {
      console.log(val)
    }
  },
  mounted () {
    this.ref = this.firebase.database().ref('LED_STATUS')
    this.ref.once('value').then(snapshot => {
      this.isChecked = snapshot.val() !== 'OFF'
    })

    this.ref.on('value', this.firebaseCallback)
  },
  beforeDestroy () {
    // clearInterval(this.interval)
    this.ref.off('value', this.firebaseCallback)
  },
  methods: {

```

```

changeActive (val) {
  const ref = this.firebase.database().ref('LED_STATUS')
  ref.once('value').then(snapshot => {
    this.check = val
    if (snapshot.val() === 'OFF') {
      ref.set('ON')
    } else {
      ref.set('OFF')
    }
  })
},

firebaseCallback (snapshot) {
  this.isChecked = snapshot.val() !== 'OFF'
}
}
}
</script>
//pages/learn.vue
<template>
<div>
  <client-only>
    <div id="myScatterPlot" style="width: 640px; height: 520px;" />
  </client-only>
  <button @click="nextStep">
    learn
  </button>
</div>
</template>

<script>
import axios from 'axios'
import kmeans from '~/api/kmeans/kmeans.js'

export default {
  name: 'Learn',
  data () {
    return {
      values: [],
      data: null,
      cluster: null,
      Plotly: null
    }
  },
  async mounted () {
    try {
      this.Plotly = require('plotly.js')
      await this.getLast()
      this.refresh()
    } catch (e) {
      console.log(e)
    }
  },
  methods: {
    async getLast () {
      const url = window.location.href.includes('localhost') ? 'http://localhost:3002/get-last' : '/get-last'
      try {
        const { data } = await axios.post(url)
        Object.keys(data).forEach(item => {
          this.values.push({ temperature: data[item].temperature, humidity: data[item].humidity })
        })
      } catch (e) {

```

```

    console.log(e)
  }
},
getData () {
  // generate focal points
  const min = 10
  const max = 30
  const hMin = 30
  const hMax = 80
  const nPoints = 2
  const points = []

  for (let i = 0; i < nPoints; i++) {
    points[i] = []
    points[i][0] = Math.random() * (max - min) + min
    points[i][1] = Math.random() * (hMax - hMin) + hMin
  }

  // generate sample data
  const nSamples = 100
  const samplesX = []
  const samplesY = []

  for (let j = 0; j < points.length; j++) {
    for (let i = 0; i < nSamples; i++) {
      samplesX[i + j * nSamples] = gaussian(points[j][0])
      samplesY[i + j * nSamples] = gaussian(points[j][1])
      if (isNaN(samplesX[i + j * nSamples]) || isNaN(samplesY[i + j * nSamples])) i--
    }
  }

  function gaussian (mean, stdev) {
    let u1, u2, v1, v2, s
    if (mean === undefined) {
      mean = 0.0
    }
    if (stdev === undefined) {
      stdev = 1.0
    }
    if (gaussian.v2 === null) {
      do {
        u1 = Math.random()
        u2 = Math.random()

        v1 = 2 * u1 - 1
        v2 = 2 * u2 - 1
        s = v1 * v1 + v2 * v2
      } while (s === 0 || s >= 1)

      gaussian.v2 = v2 * Math.sqrt(-2 * Math.log(s) / s)
      return stdev * v1 * Math.sqrt(-2 * Math.log(s) / s) + mean
    }

    v2 = gaussian.v2
    gaussian.v2 = null
    return stdev * v2 + mean
  }

  return [samplesX, samplesY]
},

```



```

drawPlot (data) {
  const series = []
  for (let i = 0; i < data.length; i++) {
    series[i] = {
      x: data[i][0],
      y: data[i][1],
      mode: 'markers',
      name: 'cluster ' + (i + 1)
    }
  }
}

const layout = {
  title: 'K-Means Clusters',
  height: 520,
  width: 640
}

this.Plotly.newPlot('myScatterPlot', series, layout, { staticPlot: true })
},

refresh () {
  // this.data = this.getData()
  this.data = [], []
  this.values.forEach(item => {
    this.data[0].push(item.temperature)
    this.data[1].push(item.humidity)
  })
  this.cluster = kmeans(this.data)
  this.drawPlot([this.data])
},

nextStep () {
  const series = []
  const clusterData = this.cluster.step()

  for (let c = 0; c < clusterData.centroids.length; c++) {
    series[c] = [], []
  }

  // populate series
  for (let s = 0; s < clusterData.labels.length; s++) {
    series[clusterData.labels[s]][0].push(this.data[0][s])
    series[clusterData.labels[s]][1].push(this.data[1][s])
  }

  this.drawPlot(series)
}
}
}
</script>

<style lang="scss">

</style>

```