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

Факультет електроніки
(повна назва інституту/факультету)

Кафедра мікроелектроніки
(повна назва кафедри)

«До захисту допущено»

Завідувач кафедри

_____ _____
(підпис) (ініціали, прізвище)

“___”_____20__ р.

Дипломна робота

на здобуття ступеня бакалавра

з напряму підготовки 6.050801 Мікро-та наноелектроніка
(код і назва)

на тему: «Система збору, збереження та відображення інформації про навколишнє середовище»_____

Виконав: студент 4 курсу, групи ДП-52
_____Яковенко Вадим Володимирович_____ _____
(прізвище, ім’я, по батькові) (підпис)

Керівник:_____доц. к.ф.м. наук_Заворотний В.Ф._____ _____
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Консультант з нормоконтролю проф., к.т.н., доц.. Орлов А.Т. _____

Консультант з інформаційних питань доц.. к.т.н., Діденко Ю.В. _____

Рецензент
(посада, науковий ступінь, вчене звання, прізвище та ініціали) (підпис)

Засвідчую, що у цій дипломній роботі немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

Завдання на дипломну роботу

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»

Факультет електроніки
(повна назва)

Кафедра мікроелектроніки
(повна назва)

Рівень вищої освіти – перший (бакалаврський)

Напрям підготовки 6.050801 «Мікро- та наноелектроніка»
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри

_____ _____
(підпис) (ініціали, прізвище)

«___»_____20__ р.

ЗАВДАННЯ

на дипломну робот студенту

групи ДП-52 Яковенку Вадиму Володимировичу
(прізвище, ім'я, по батькові)

1. Тема роботи: Система збору, збереження та відображення інформації про навколишнє середовище.

керівник роботи: Заворотний В.Ф., кандидат фізико-математичних наук, доцент

затверджені наказом по університету від «___»_____ 20__ р. №_____

2. Строк подання студентом роботи 04 червня 2019 р.

3. Вихідні дані роботи: Система моніторингу інформацію про навколишнє середовище, яка:

- виконує збір даних з навколишнього середовища;
- виконує збереження даних;
- забезпечує зручний перегляд даних;
- має функціонал критичних сповіщень показників;

4. Зміст дипломної роботи:

- Огляд та порівняння існуючих систем моніторингу навкоришнього середовища
- Вибір програмного забезпечення та компонентної бази для створення системи на основі уже встановлених вимог
- Реалізація системи

5. Перелік графічного (ілюстративного) матеріалу (із зазначенням обов'язкових креслеників, плакатів, презентацій тощо)

6. Консультанти розділів проекту (роботи)[1*]

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

7. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1.	Ознайомлення із принципом роботи цифровихдатчиків збору інформації про навколишнє середовище	01.03.2019	
2.	Розробка конструкції приладу та підбір компонентних складових пристрою	15.03.2019	
3.	Ознайомлення із принципами роботиі архітектурою контролера Arduino та створення клієнтської частини системи	15.04.2019	
4.	Ознайомлення із принципами роботи веб-додатків та створення серверної частини системи	15.05.2019	
5.	обговорення результатів та написання дипломної роботи	01.06.2019	

Студент _____ В.В.Яковенко
(підпис) (ініціали, прізвище)

Керівник проекту (роботи) _____ В.Ф. Заворотний
(підпис) (ініціали, прізвище)

РЕФЕРАТ

[1*] Консультантом не може бути зазначено керівника дипломного проекту (роботи)

Дипломна робота присвячена розробці системи для збору, збереження та відображення інформації про стан навколишнього середовища. Метою роботи є аналіз існуючих систем моніторингу інформації, та на основі поставлених вимог до них створення власної системи.

У роботі проведено детальний аналіз по вибору компонентної бази та програмного забезпечення системи. Клієнтська частина системи була розроблена на основі контролера Arduino UNO. Серверна частина системи була реалізована у вигляді веб-сервісу. Загальний обсяг роботи: 65 сторінок, 16 ілюстрацій, 1 таблиця, 13 посилань , 5 додатків.

Ключові слова: система моніторингу, збір інформації, відображення інформації, зберігання інформації, навколишнє середовище, контролер, веи-сервіс.

ABSTRACT

The thesis is devoted to the development of a system for collecting, storing and displaying information on the state of the environment. The purpose of the work is to analyze existing information monitoring systems and, based on the requirements set for them, to create their own system.

In this work, a detailed analysis was carried out on the choice of component base and software system. The client part of the system was developed on the basis of the Arduino UNO controller. The server part of the system was implemented as a web service. Total volume of work: 65 pages, 16 illustrations, 1 table, 13 references, 5 annexes.

Keywords: monitoring system, information gathering, information display, storage of information, environment, controller, vei-service.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ОС - операційна система

БД - база даних

ПК - персональний комп’ютер

GSM - Global System for Mobile Communications (глобальна система мобільного зв'язку)

СУБД - система управління базами даних

API - Application Programming Interface (інтерфейс програмування застосунків, інтерфейс прикладного програмування)

DNS - Domain Name System (служба імен доменів)

ЕОМ - електронно обчислювальна машина

ОС - операційна система

IP - Internet Protocol (Інтернет протокол)

FTP - File Transfer Protocol (протокол передачі файлів)

ШІМ - широтно-імпульсна модуляція

TCP - Transmission Control Protocol (протокол управлінням передачі даних)

HTTP - Hypertext Transfer Protocol (протокол передачі гіпертексту)

IP-адреса - Internet Protocol address (ідентифікатор мережевого рівня)

URL - Uniform Resource Locator (єдиний вказівник на ресурс)

JSON - JavaScript Object Notation (текстовий формат обміну даними між комп'ютерами)

ВСТУП

Саме слово "моніторинг" походить від латинського "monitor" - нагадувати, спостерігати, наглядати, що визначає внутрішній зміст цієї діяльності - отримувати інформацію для відповідного реагування.

Моніторинг довкілля - це система постійного спостереження, збирання, оброблення, передавання, збереження та аналізу інформації про стан довкілля, прогнозування його змін і розроблення науково обґрунтованих рекомендацій для прийняття рішень про запобігання негативним змінам стану довкілля та дотримання вимог екологічної безпеки.

При автоматичному контролі відбувається отримання і обробка інформації про стан об'єкта і зовнішніх умов для виявлення подій, що визначають управлінські дії. Подією може бути будь-який якісний результат: поява деталі з розмірами, що виходять за допустимі межі, коротке замикання, вихід температури за встановлене значення, аварія обладнання та інші.

Бакалаврська робота присвячена розробці системи, яка б дозволила збирати інформацію про стан навколишнього середовища, зберігати та відображати її, за допомогою сучасних методів представлення інформації, у зручному для користувача вигляді.

Під час проектування системи не було знайдено готових та дешевих рішень, які б дозволяти проводити моніторинг стану навколишнього стану в зручному вигляді тому було прийнято рішення розробити її самому.

Система складається з переносного пристрою та серверної частини. Етапи проектних рішень системи можна розділити на три частини:

1. Етап збору та та передачі інформації для подальшого збереження. На даному етапі головним компонентом системи є контроллер Arduino та WiFi модуль ESP8266 .

2. Етап отримання та збереження інформації. Головна складова етапу є веб сервер та реляційна база даних.
3. Етап відображення інформації передбачає використання веб серверу для отримання інформації з БД для подальшої її відображення на веб сторінці та відображення інформації за допомогою API соціального мессенджера.

В кінцевому результаті, створену систему, можна буде використовувати в багатьох сферах людської діяльності, починаючи від сільського господарства, для моніторингу клімату теплиць чи температурного режиму інкубатора, закінчуючи будівлями з кластерами серверів для відстеження показників кімнати.

1 МОНІТОРИНГ ІНФОРМАЦІЇ ПРО НАВКОЛИШНЄ СЕРЕДОВИЩЕ

Моніторинг є системою, яка виконує постійне спостереження за явищами і процесами, що відбуваються в навколишньому середовищі і суспільстві, результати якого служать для керування прийняття наступного рішення на основі отриманих раніше даних.[1]

Системи моніторингу можуть включати в себе:

- системи, які виконують збір, спостереження, обробку, зберігання,
- передачу та аналіз даних про оточуюче середовище;
- системи для передбачення розвитку небезпечних ситуацій на основі зібраних раніше даних;
- системи для виявлення відхилень параметрів оточення від нормативних вимог або критеріїв безпеки;

Найбільш важливою частиною моніторингу є своєчасне виявлення небезпечного розвитку процесів та підготовка рішеньшень, які спрямовані на ліквідацію чи мінімізацію негативних наслідків внаслідок взаємодії об'єкта з середовищем;

Моніторинг має базуватися на принципах системності, комплексності, безперервності й варіантності, внаслідок чого можна досягти:

- вивчення основних компонентів навколишнього середовища у взаємодії;
- розробки прогнозів навколишнього середовища.
- отримання необхідної кількості інформації у вигляді безперервних спостережень;

Загальним моніторингом навколишнього середовища називаються певні спостереження в місцях, що об’єднані в єдину інформаційну мережу та

дають змогу, спираючись на уже зібрані дані, спрогнозувати стан оточуючого середовища та розробляти рішення, спрямовані на відновлення допустимих значень показників до допустимих меж.

Основною частиною автоматичного моніторингу навколишнього середовища є системи автоматизованого спостереження та контролю за даним середовищем. Вони зазвичай являють собою центральну ЕОМ, яка призначена для збору та виконання аналізу даних з робочих станцій.[3]

1.1 Огляд існуючих систем моніторингу про навколишнє середовище

В даний час на ринку немає широкого представлення систем для збору, збереження та відображення інформації про навколишнє середовище. Однак все ж, вдалося знайти кілька представлених систем. Порівняльну характеристику можна переглянути в таблиці 1.1.1

Таблиця 1.1.1 порівняльна характеристика систем моніторингу

Назва	Multibox	Simvolt	Tera
Набір датчиків	температура, вологість повітря	температура, вологість повітря, освітленість	температура, вологість повітря
Передача даних	кабель езернет	кабель езернет	GSM
Збереження даних	PostgreSQL	MySQL	PostgreSQL
Відображення інформації	веб-браузер	програма на ПК	екран датчика, програма на ПК
Критичні сповіщення	-	SMS - сповіщення	SMS - сповіщення
Ціна	450$	520$	600$

Кожна із порівняльних систем має переваги та недоліки. Так, серед різноманіття набору датчиків, переважає система Simvolt, серед способів передачі даних - Tera, оскільки не потребує додаткових кабелів для з'єднання. Збереження даних в кожній із систем аналогічне. Відображення зафіксованих даних найкраще виконує система Multibox, оскільки використовуючи веб-браузер, дані можна переглянути зі смартфона, планшета або ж ПК, при цьому не потрібно встановлювати ніякі додаткові програми. Функцію критичних сповіщень за допомогою SMS підтримують системи Simvolt та Terra. Аналізуючи ціновий діапазон систем найдешевшою являється система Multibox, найдорожчою Tera.

1.2 Висновки

В даному розділі розглянуто в загальному вигляді поняття моніторинг, з'ясовано які вимоги висуваються до систем моніторингу та чого можна досягди отримуючи регулярні дані із системи моніторингу.

Зроблено огляд та порівняльну характеристику декількох, представлених на ринку систем моніторингу інформації. Виходячи із їхньої порівняльної характеристики можна встановити, кращим варіантом із розглянутих варіантів являється система Simvolt.

2 АНАЛІЗ ВИХІДНИХ ДАНИХ

Змістом дипломної роботи є створення системи для забезпечення збору, зберігання та відображення інформації про навколишнє середовище. Отримана система повинна відповідати ряду вимог для її роботи та зручного користування.

Функціональні вимоги:

Оскільки система складається з двох складових, то розглянемо функціональні вимоги для кожної складової окремо.

Для переносного пристрою висуваються наступні вимоги:

1. Живлення за допомогою блоку живлення 5 В або портативного зарядного пристрою
2. Передача інформації за стандартом IEEE 802.11 передачі цифрових потоків даних по радіоканалах
3. Стійкість корпусу приладу до впливів навколишнього середовища.

Для серверної частини системи висуваються наступні вимоги:

1. ЕОМ під управління ОС Linux
2. Статична IP адреса

Вимоги користувачів:

1. Зручний та інтуїтивний інтерфейс
2. Простота встановлення системи
3. Наявність мобільної версії для відображення даних
4. Наявність десктопної версії для відображення даних
5. При відображенні інформації підтримка розповсюджених браузерів (Firefox, Chrome, Opera, Safari, IE, Microsoft Edge)
6. Наявність системи сповіщення критичних показників

Системні вимоги:

Для переносного пристрою:

1. Контроллер для забезпечення фіксування показників датчиків
2. WI-FI модуль
3. Датчик температури та вологості
4. Датчик освітленості

Для серверної частини:

1. Сервер:
    - ОС: Linux (Debian 7, SUSE Linux Enterprise Server 11 SP3 & 12, Ubuntu 16.04 LTS, 18.04 LTS)
    - RAM: від 1 GB
    - CPU: 1 GHz, 1 Core
    - HDD: 2GB
2. База даних: MySQL або PostgreSQL
3. Веб-сервер: Apache2 або Nginx (php-fpm)
4. PHP 7.0+
5. Веб браузер:
    - IE11+
    - Microsoft Edge
    - Firefox 14+
    - Chrome 18+
    - Safari 7+
    - Opera 21+

Вимоги до зовнішніх інтерфейсів:

Для переносно пристрою

1. Відкрите з’єднання за допомогою порту USB тип B
2. Відкрите з’єднання за за стандартом IEEE 802.11 передачі цифрових потоків даних по радіоканалах

Для серверної частини системи:

1. Відкрите з’єднання для протоколів HTTP
2. Відкрите з’єднання для протоколів HTTPS

2.1 Висновки

Під час написання даного розділу було проведено аналіз вимог необхідних для реалізації системи. Під час аналізу було виділено такі групи вимог: функціональні, користувачів, системні та зовнішніх інтерфейсів. Аналіз був проведений для подальшого полегшення вибору складових і побудови системи, яка відповідатиме вхідним вимогам.

3 ОБГРУНТУВАННЯ ВИБОРУ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ТА КОМПОНЕНТНОЇ БАЗИ ДЛЯ СТВОРЕННЯ СИСТЕМИ

Виходячи з аналізу вхідних даних, які були розглянуті в попередньому розділі, потрібно проаналізувати доступне програмне забезпечення та компонентну базу для створення системи.

3.1. Вибір об’єкта керування

Контроллер являє собою мініатюрний комп’ютер з набором входів та виходів, який працює за раніше написаною програмою. Мікросхема-контроллер обов’язково присутня в комп’ютерній миші, смартфоні, плеєрі та майже у всіх інших сучасних електронних пристроях. Контролер є досить універсальною річчю. До його входів можна підключити як звичайні кнопки так і датчики різних конфігурацій та модулі бездротового зв’язку. Завдання контролера полягає у вимірюванні електричної напруги на його входах і подачі електричної напруги, відповідно заданої програми, на вихід.

3.1.1 Raspberry Pi

Raspberry Pi представляє із себе одноплатний комп’ютер, тобто різні частини комп’ютера, які зазвичай розміщені на різних платах, тут розташовані на одній (рис. 3.1.1.1). Дана плата має відносно невеликий розмір - приблизно 8,5 x 10,5. Raspberry Pi - недорога платформа - середня ціна за одну складає близько $35, а для новішої версії A+ близько $20. Продаж даних платформ почався в 2012 році. Сьогодні це одна з найбільш популярних платформ в своїй сфері. Продано уже близько 4 млн екземплярів Raspberry Pi.

Рисунок 3.1.1.1 - Одноплатний комп’ютер Raspberry Pi B

Raspberry Pi випускається в двох версіях - A і B. Версія B на сьогоднішній день більш популярна. Порівняння версій і основні характеристики Raspberry Pi:

- Процесор ARM11, Broadcom BCM2835, 700 МГц;
- Оперативна пам’ять - 256 Мб у A, 512 у B
- USB входи/виходи - 1 у A, 2 у B;
- SD вхід;
- RCA вихід;
- HDMI вихід;
- Ethernet вхід/вихід - присутній тільки у B версії;
- Audio вихід;
- GPIO контакти;

Для початку роботи з даною мікро- ЕВМ необхідна SD-картка, із завантаженою операційною системою. Рекомендується використовувати карту з об’ємом пам’яті від 4 до 32 ГБ. Для організації інтерфейсу з користувачем необхідний монітор, з роз’ємом HDMI, DVI або RCA, кабель з

даними інтерфейсом, USB-клавіатура і USB-миша. Безумовно, Raspberry Pi є повнофункціональним комп'ютером. Він володіє всіма атрибутами справжнього комп'ютера: виділеним процесором, пам'яттю і графічним драйвером для виведення через HDMI. На ньому навіть працює спеціальна версія операційної системи Linux. Тому на Raspberry Pi легко встановити більшість програм для Linux. Варто трохи попрацювати - і Raspberry Pi можна використовувати як повноцінний медіа-сервер або емулятор відеоігор. Хоча в Pi і відсутня внутрішнє сховище даних, на цьому комп'ютері можна використовувати смарт-карти в якості флеш-пам'яті, яка обслуговує всю систему. Таким чином, можна швидко завантажувати для налагодження різні версії операційної системи або програмних оновлень. Оскільки цей пристрій забезпечує незалежне з'єднання по мережі, його можна налаштовувати і для доступу по SSH, або посилати на нього файли по протоколу FTP.

Raspberry Pi для роботи потрібно постійна напруга 5V, більш того, робота даної мікро- ЕОМ завершується програмним процесом - як у звичайного комп'ютера. Raspberry Pi складно переносити з місця на місце, так як ви не зможете просто вставити в нього дві батарейки АА. Для роботи цього комп'ютера необхідно забезпечити безперебійне живлення, а також підключити додаткове обладнання, яке гарантує подачу постійного струму.

У Raspberry Pi є вбудований Ethernet-порт, який забезпечує легкий доступ до будь-якої мережі і практично не вимагає настройки. Провести бездротовий Інтернет на Raspberry Pi також не складає труднощів: для цього достатньо придбати USB-адаптер для Wi-Fi і встановити відповідний драйвер. Як тільки це зроблено, можна приступати до використання операційної системи для підключення до веб-серверів, обробленню HTML або просто що-небудь писати в Інтернеті. Також Raspberry Pi можна використовувати як для створення віртуальної приватної мережі, так і в якості

сервера друку.[2]

3.1.2 Контроллер Arduino

Порівнюючи плати Arduino і Raspberry Pi має сенс відразу сказати, що плати Arduino - це мікроконтролери, а не повноцінні комп'ютери. На них немає операційної системи як такої, Arduino просто виконує код, інтерпретується прошивкою. В даному випадку, відсутні базові інструменти, що надаються операційною системою, але, з іншого боку, таке безпосереднє виконання нескладного коду протікає простіше, а при роботі не виникає ніяких затримок, яких помилок, пов'язаних з операційною системою.

Основне призначення плати Arduino - взаємодія з сенсорами і пристроями, тому Arduino відмінно підходить для апаратних проектів, де потрібно просто реагувати на різні сигнали сенсорів і ручне введення. Може здатися, що в цьому немає нічого особливого, проте на ділі Arduino - складна вивірена система, значно полегшує управління пристроями. Вона відмінно підходить саме для організації взаємодії інших пристроїв і виконавчих механізмів, де операційна система просто не потрібно, так як мова йде просто про отримання сигналів з сенсорів і реагуванні на них.

Як приклад характеристик наведемо основні характеристики найбільш популярною і універсальної серії контролерів Arduino - Uno:

- Мікроконтролер ATmega328;
- Робоча напруга 5 В;
- Рекомендована вхідна напруга 7-12 В (граничне 6-20 В);
- 14 контактів для цифрового сигналу (6 з яких можуть використовуватися як виходи ШІМ);
- 6 Аналогових входів;
- Постійний струм через вхід / вихід 40 мА;

- Постійний струм для виходу 3.3 В 50 мА;
- Flash-пам'ять 32 Кб (ATmega328);
- ОЗУ 2 КБ (ATmega328);
- EEPROM 1 Кб (ATmega328);
- Тактова частота 16 МГц.
- Розміри 68,6x53,4 мм

Arduino Uno може працювати як від USB підключення, так і від зовнішнього джерела: батарейки або звичайної електричної мережі. Джерело визначається автоматично. Платформа може працювати при наявності напруги від 6 до 20В. Однак при напрузі менше 7 В робота може бути нестійкою, а напруга більше 12 В може призвести до перегріву і пошкодження. Отже діапазон напруг для нормальної роботи має бути: 7-12 В.

На Arduino доступні наступні контакти для доступу до живлення:

- Vin надає той же вольтаж, що використовується для живлення платформи. При підключенні через USB буде дорівнює 5 В.
- 5V надає 5 В незалежно від вхідної напруги. На цій напрузі працює процесор. Максимальний допустимий струм, який можна отримати з даного контакту - 800 мА.
- 3.3V надає 3,3 В. Максимальний допустимий струм, який можна отримати з даного контакту - 50 мА.
- GND - земля.

Arduino починає виконувати код відразу після включення і припиняє роботу відразу, як тільки відключити плату від джерела живлення. щоб розширити функціонал пристрою з Arduino, необхідно підключити периферійні пристрої безпосередньо до контактів самої плати Arduino (рис. 3.1.2.1), або до плат розширення для неї. Для Arduino існують сотні

різноманітних модулів, кожен з яких призначений для вирішення специфічного завдання, може взаємодіяти з тими чи іншими сенсорами, а також з іншими модулями, які разом утворюють повноцінний керуючий блок.[4]

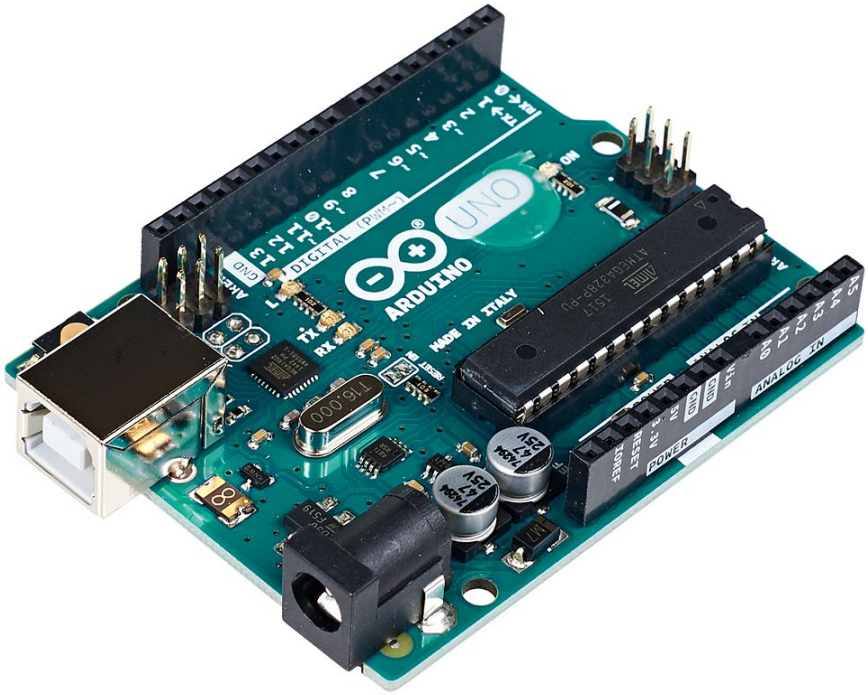

Рисунок 3.1.2.1 - Контролер Arduino UNO

На жаль, система Arduino без додаткових модифікацій не пристосована для роботи по мережі інтернет. Щоб встановити надійне з'єднання вона вимагає або підключення додаткових модулів до плати, або використання модифікацій платформи зі відразу вбудованими необхідними модулями. Як вже говорилося вище, Arduino може мати велику кількість сумісних модулів.

3.1.3 Обгрунтування використання контролера Arduino

Основна задача в цьому проекті полягала в обробці сигналів із сенсорів і подальша передача її на Веб сервер. Платформа Arduino проста в обслуговуванні і не вимоглива в живленні. У середовищі Arduino IDE, врахувавши те, що нам потрібно автономна робота платформи, можна

скласти лістинг програми, який дозволить використовувати пристрій, при цьому не виключаючи і абсолютно не втручаючись в його роботу.

Наведемо аргументи на користь платформи Arduino в порівнянні з мікро- ЕОМ Raspberry Pi. Основними плюсами даного одноплатного контролера в порівнянні з Arduino є продуктивність, багатозадачність зручність роботи з інтернетом, наявність вибору мови програмування, робота з відео, звуком і комп'ютерним зором. Недоліками - швидкість реакції в швидкодіючих проектах і коротка тривалість роботи від акумулятора. З переваг даного мікрокомп'ютера нам може знадобитися зручність роботи з даними з інтернет протоколами, але інші переваги в даній роботі нам абсолютно не критичні. До того ж, з огляду на сотні мА, які Raspberry Pi споживає при своїй роботі на нормальне функціонування, доцільність використання даного мікрокомп'ютера сходить нанівець. До того ж, Raspberry Pi для ефективної взаємодії з різними пристроями вимагає спеціального програмного забезпечення. Використання даної мікро- ЕОМ виправдане при вирішенні таких завдань, які було б логічно виконувати на персональному комп'ютері.

Отже кращим вибором для даної системи буде вибір контролера Arduino UNO.

3.2 Вибір об’єкта передачі даних

Відповідно до вихідних даних та обраного в попередньому пункті контролера об’єкт передачі даних повинен відповідати двом головним вимогам: підтримка протоколу передачі даних Wi-Fi 802.11n, інтерфейс для взаємодії з контролером Arduino UNO.

Провівши аналіз доступних на ринку модулів, було знайдено тільки один, який задовольняє поставлені вище вимоги - модуль ESP8266.

3.2.1. Модуль ESP8266

ESP8266 - це мікроконтролер, розроблений в 2014 році і випускається компанією Espressif Systems - китайська компанія з Шанхаю. Він являє собою мережеве рішення з Wi-Fi-трансивером на борту плюс можливість виконання записуються в його пам'ять додатків.

Існує безліч модифікацій плат, іменованих зазвичай від ESP-01 до ESP-12. Зображення модифікацій модулів можна переглянути на (рис. 3.2.1.1).

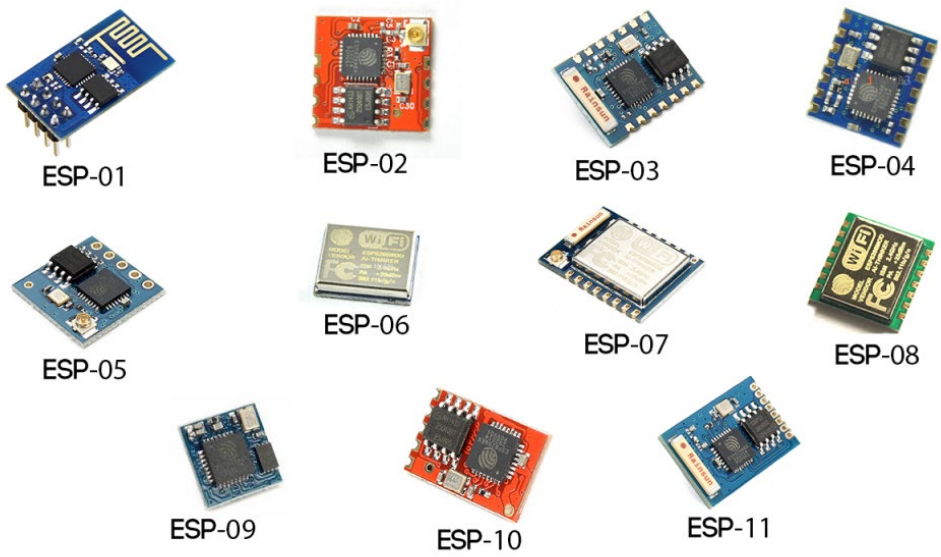

Рисунок 3.2.1.1 - Модифікації модуля ESP8266

Специфікація ESP8266:

- Напруга живлення: 3,3 В
- Енергоспоживання: 10 мкА ... 170 мА
- Флеш-пам'ять: до 16 мб максимум (зазвичай 512 кб)
- Процесор: Tensilica L106, 32 біт
- Швидкість процесора: 80 ... 160 МГц
- ОЗУ: 32 кб + 80 кб
- Порти введення-виведення загального призначення: 17
- АЦП: 1 введення з роздільною здатністю 1024

- Підтримка 802.11: b/g/n/d/e/i/k/r
- Максимальне число підключень TCP: 5 [5]

  Виходячи із даних специфікації видно, що споживання енергії змінюється в дуже широкому діапазоні - при передачі на повній потужності воно становить 170 міліампер, а в режимі сні - всього 10 мікроампер.

ESP8266 розроблений так, що він може використовувати підключений до нього модуль пам'яті і це зазвичай флеш-пам'ять. Це цілком достатньо для випадків коли додаток записує в пам'ять свої настройки або веде будь-якої лог даних, але якщо ваше додаток записує свої дані занадто швидко, пам'ять незабаром перестане працювати, оскільки кількість циклів перезапису пам’яті становить 10000.

3.3 Вибір датчиків для збору інформації

Спираючись на вихідні дані, для створення системи необхідно підібрати датчики, які б забезпечили вимірювання параметрів навколишнього середовища (температуру, вологість повітря, освітленість);

Після проведення дослідження можливих варіантів датчиків, були вибрані комбінований датчик температури та вологості DHT22 та датчик освітленості BH1750.

3.3.1 Датчик вологості та температури DHT22

Датчик складається із двох частин ємнісного датчика температури та гігрометра. Перший використовується для вимірювання температури, другий - для вологості повітря. Також в в датчик вмонтований простий аналого - цифровий перетворювач який дозволяє зчитувати з датчика цифровий сигнал. Будову датчика можна переглянути на (рис. 3.3.1.1).

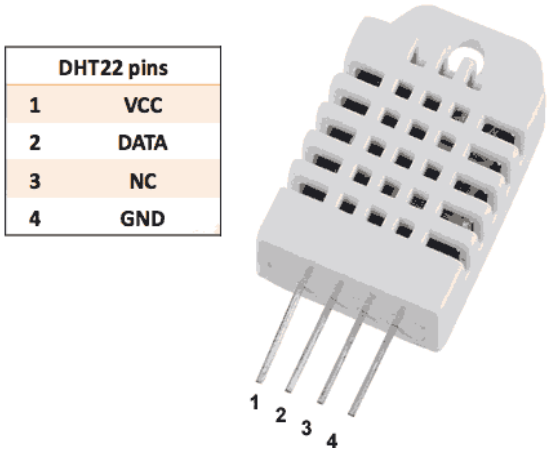

Рисунок 3.3.1.1 - Цифровий датчик вологості та температури DHT22

Датчик має наступні технічні характеристики:

- Струм споживання - 2,5 мА (максимальне значення при перетворені даних);
- Живлення 3 - 5 В.
- Діапазон вимірювальної вологості від 0% до 100%;
- Похибка вимірювання вологості від 2% до 5%;
- Діапазон вимірювання температури від -40 до 80 °С;
- Похибка вимірювання температури до 2%;
- Датчик здатний робити одне вимірювання за 2 секунди. Частота - до 0,5 Гц;
- Габаритні розміри: 15,1 мм довжина, 25 мм ширина, 5,5 мм висота [6]

3.3.2 Датчик освітленості BH1750

Датчик типу BH11750FVI - один з найпоширеніших в сучасній електроніці та електротехніці. Його перевага в цифровому інтерфейсі, який дозволяє вимірювати освітленість у Люксах і дає можливість працювати з

мікроконтролером по протоколу I2C. Малий розмір датчика робить його незамінним в сучасній побутовій електроніці.

Як чутливий елемент датчика BH11750FVI виступає фотодіод (bh1750), приймач оптичного випромінювання, який уловлює світло і перетворює його в електричний сигнал.

Датчик має наступні технічні характеристики:

- Напруга живлення: 5 В;
- Інтерфейс: I2C;
- АЦП: 16 Біт;
- Точність: 1 Люкс;
- Чуттєвість: 65536 градацій;
- Калібровка: не потрібна;
- Габаритні розміри: 19x13x2 мм; [7]

Будову датчика можна переглянути на (рис. 3.3.2.1).

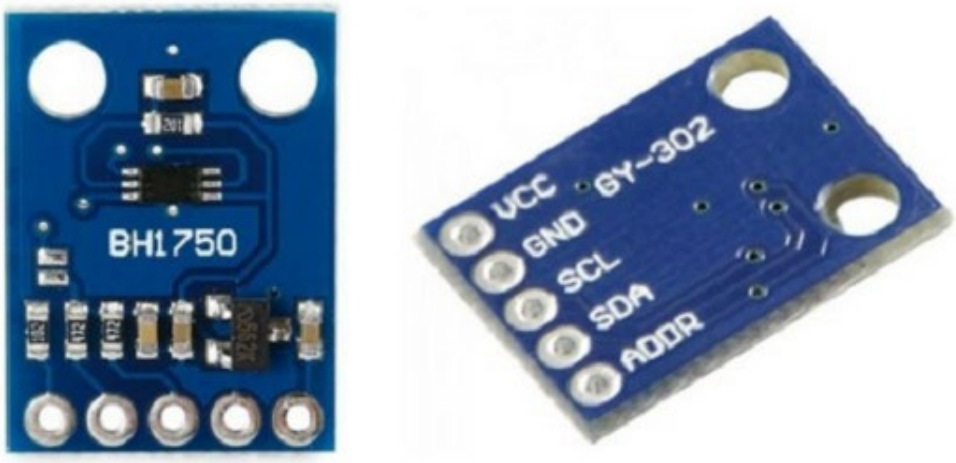

Рисунок 3.3.2.1 - Цифровий датчик освітленості BH1750

3.4 Вибір програмного забезпечення серверної частини системи

Для вибору складових даної системи, коротко, розглянемо будову мережі Інтернет, термінологію та основні її складові .

Мережа Інтернет представляє собою велику кількість комп'ютерів, які з'єднані один з одним кабелями, а також радіоканалами, супутниковими каналами та іншими методами зв'язку. Однак, як відомо, одних лише проводів або радіохвиль для передачі інформації недостатньо: сторона, яка віддає інформацію та сторона, яка приймає інформацію потрібно дотримуватися ряду домовленостей, які дозволяють чітко регламентувати передачу даних і гарантувати, що ця передача пройде без спотворень. Такий набір правил називається протоколом передачі.

Для різних цілей існують різні протоколи. Розглянемо той який будемо використовувати, який в свою чергу використовується у Web - програмуванні. Таким для нас є протокол TCP, а точніше HTTP який базується на TCP. Протокол HTTP саме використовується браузерами і веб-серверами.

Будь-який комп'ютер, який підключений до Інтернету і бажає обмінюватися інформацією повинен мати деяке унікальне імя, або IP-адресу. Середньостатистична IP-адреса виглядає наступним чином: 127.12.232.56. Зазвичай людям досить незручно працювати з IP-адресами напряму. Досить простіше запам'ятовувати символьне ім'я, ніж набір чисел. Щоб полегшити роботу з Інтернетом, було придумано систему DNS. Загальносвітова DNS представляє собою розподілену базу даних, яка здатна перетворювати доменні імена машин в їх IP-адреси. При використанні DNS будь-який комп'ютер в мережі інтернет може мати не тільки IP-адресу, а й символьне ім'я, яке може виглядати наступним чином: www.example.com.

Сервер - певний комп'ютер в мережі інтернет, який дає змогу іншим комп'ютерам використовувати себе в як "посередника" під час передачі

інформації. Сервер являється машиною, яка може мати декілька IP - адресів, ніби це незалежні системи незалежних систем.

Порт - деякий ідентифікатор програму, що має на меті отримувати з Інтернету дані. Будь - яка програма, яка прагне передати дані іншій, повинна знати номер порту, за яким закріплена інша. Зазвичай кожному сервісу призначається фіксований номер порту. Традиційно веб-серверу виділяється порт з номером 80.

Мережевий сервіс - це програма, яка працює на сервері і займається обслуговування різного роду користувачів, які можуть до неї підключитися. Типові приклади - веб-сервер, а також FTP- і telnet- сервери.

Провайдер - організація, що має в наявності певну кількість модемних входів, до яких можуть підключатися користувачі для доступу в Інтернет.

Хостинг-провайдер - організація, яка має можливість створювати та продавати хости клієнтам за певну плату.[8]

3.4.1 Огляд популярних веб-серверів

За даними статистичних даних проекту W3techs, більш 80% веб-додатків і сайтів працюють на серверах з відкритим вихідним кодом. Не даремно саме з «домінуючими мережами» частіше говорять про відкриті вихідні коди, коли заходить решту Linux на ринку.

Розглянемо найбільш розповсюджені та популярні, представлені, веб-сервери.

3.4.1.1 Apache

Apache являється відкритим веб сервером, завдяки чому можна використовувати на різних операційних системах таких як Windows, Novell NetWare або UNIX. Даний веб-сервер являється найбільш популярним, частка його використання серед усіх інших веб-серверів становить близько 55%.

Веб-сервер було створено в 1995 року, незалежними розробниками співтовариства «Apache Group». Apache - самостійна, некомерційна розробка, яка вільно розповсюджується. Підтримує багато можливостей, більшість з яких реалізовано скомпільованими модулями, що розширюють його функціонал. Даному веб-серверу доступні інтерфейси, які дозволяють підтримувати мови програмування Perl, Python, і PHP. Сюди ж включений модуль для архівування, який розроблений щоб зменшити розмір даних, які передаються. Має функцію віртуального хостингу, яка дозволяє одним встановленням обслуговувати декілька різних сайтів. Сервер використовують для передачі статичних та динамічних Web-сторінок в мережі інтернет, використовуючи протокол HTTP.[9]

Даний веб-сервером було зіграно визначальну роль на початковому етапі розвитку Інтернету і навіть зараз він продовжує бути найпопулярнішим веб-сервером.

3.4.1.2 Nginx

Nginx представляє собою HTTP-сервер, зворотній проксі-сервер, поштовий проксі-сервер, а також TCP/UDP проксі-сервер загального призначення. Даний сервер було розроблено 2004 року Ігорем Сисоєвим. З того часу він почав набирати популярність завдяки своїй легковажності і можливості до простого масштабування на не дуже потужних машинах.

Nginx чудово себе поводить при віддачі статичного контенту і спроектованим таким чином щоб передавати динамічні запити іншому програмному забезпеченню, для подальної обробки. Ще однією його перевагою є можливість одночасної роботи із 10000 з’єднаннями. Тому на даний веб-сервер досить часто покладається вибір завдяки його ефективному використанні ресурсів та швидким відгукам під навантаженням, а також через можливість його використання як проксі-сервер та веб-сервер. На даний момент відсоток використань даного веб-сервера становить близько 15%.[9]

3.4.2 Огляд баз даних для збереження інформації

Людська діяльність, яка була спрямована для знаходження методів структурування інформації, привела до створення спеціальних програмних комплексних рішень - системам управління базами даних (СУБД).

Особливістю СУБД являється наявність процедур. Дані процедури забезпечують введення, зберігання та опис структури інформації. Системи, що забезпечували дані функції із даними стали називати банками даних, а потім базами даних (БД).

За способом організації даних бази даних можна розділити на наступні види:

- Ієрархічний. Дані представляються у вигляді дерева, яке складається з об'єктів різних рівнів. Між ними розміщені зв'язки, які поєднують предоків із нащадками.
- Реляційний. Збереження даних відбувається у вигляді таблиць. Найбільш вживані СУБД використовують саме реляційну модель даних.
- Об'єктно-орієнтований. Представлення даних відбувається у вигляді об’єктів.
- Мережевий. Схожий до ієрархічного, але кожен об'єкт може мати більше одного предка.

Оскільки реляційні бази даних найбільш вживані, розглянемо кілька популярних варіантів.

MySQL - являється вільною реляційною системою управління базами даних. Її розробкою та підтримкою займається корпорація Oracle. MySQL використовується для малих і середніх додатків. Дану СУБД використовують у вигляді сервера, до якого звертаються локальні або віддалені клієнти.

MySQL є досить гнучкою СУБД, оскільки забезпечує підтримку великої кількості типів таблиць: користувачі можуть вибрати як таблиці типу

MyISAM, що підтримують повнотекстовий пошук, так і таблиці InnoDB, що підтримують транзакції на рівні окремих записів.[10]

Настуною розглянемо PostgreSQL, це не просто реляційна, а об'єктно-реляційною СУБД. Це дає їй деякі переваги над іншими SQL базами даних з відкритим вихідним кодом, такими як MySQL, MariaDB і Firebird.

Фундаментальна характеристика об'єктно-реляційної бази даних - це підтримка об'єктів і їх поведінки, включаючи типи даних, функції, операції, домени і індекси. Це робить Postgres досить гнучким і надійним. Він підтримує велику кількість типів даних порівняно з MySQL.

Дана СУБД створена з використанням об'єктно-реляційної моделі, та підтримує складні структури і широкий спектр вбудованих і обумовлених користувачем типів даних.

3.4.2.1 Обґрунтування вибору бази даних

Розглянемо в даному розділі було представлено що представляють із себе бази даних та коротко представлено 3 найбільш популярних реляційних баз даних. Кожна з них має свої переваги та недоліки. Відштовхуючих від вихідних вимог ми можемо обрати між MySQL та PostgreSQL. Враховуючи той факт, що система, яка розробляється в даній роботі, не потребує багато ресурсів, залишимо вибір на MySQL. Також, до переваг, в користь вибору бази даних MySQL, можна віднести той факт, що дана БД стандартно підтримується більшістю хостингів.

3.5 Вибір методів відображення даних

Після отримання даних про навколишнє середовище із датчиків, та збереження їх в базу даних, постає завдання, яке полягає у відображенні зафіксованої інформації. Дані можна представити у табличному вигляді, за

допомогою графіків або напряму відображати останній зафіксований результат.

В даний час людство досить сильно пов’язане з мережею Інтернет тому було б чудово представляти інформацію у вигляді веб-сторінки. Веб-сторінка являє собою інформаційний ресурс мережі Інтернет або документ, доступ до якого можна здійснити за допомогою веб-браузера. Зазвичай веб-сторінка - текстовий файл у форматі HTML, який задає каркас сторінки, підключений файл зі стилями сторінки у форматі CSS, а також може бути підключений файл в форматі JS, в якому записаний алгоритм для забезпечення інтерактивності сторінки. Даний файл може містити в собі посилання на файли в інших форматах (текст відео, аудіо, зображення, мультимедіа, веб-служби та інше), а також гіперпосилання, завдяки яким, можна перейти на інші веб-сторінки. Інформація, яка відображається на сторінці називається контентом.

Веб-сторіноки, які об'єднані спільною темою і дизайном, та пов'язані одна з одною посиланнями, створюють веб-сайт. В залежності від складності сайту сторінки можуть перебувати на одному або декількох веб-серверах, які, в свою чергу, можуть бути розташованими в одному дата-центрі або віддалено один від одного, часто в різних країнах.[11]

Останнім часом широкого розповсюдження і використання почали набувати соціальні месенджери. Месенджер являє собою програму, мобільний додаток або веб-сервіс, який забезпечує миттєвий обмін повідомленнями між користувачами. Дані додатки розробляються різними компаніями та встановлюються на смартфон або комп’ютер. В залежності від функціоналу кожної із програм, в них можна обмінюватися текстовими повідомленнями, вставляти в переписку стікери, надсилати файли різних форматів, створювати групи для спілкування та навіть спілкуватися за допомогою відеозв’язку. Деякі месенджери, на своїй платформі дозволяють створювати ботів - програми, які імітують поведінку людини, за раніше

написаним програмним сценарієм. В Україні до популярних месенджерів можна віднести Viber, Telegram, Skype, Facebook messenges.[12]

Отже для відображення данних будемо використовувати описані вище два методи - веб-сторінка та бот месенджера. На веб-сторінці будемо відображати дату останнього оновлення даних, окремо останні показники датчиків, а також статистику показників за минулий день у графічному вигляді. В якості месенджера виберемо Telegram так як він один із найрозповсюдженіших та має найкращу документацію по розробці бота. За допомогою нього будемо відображати останні актуальні показники із датчиків.

3.6 Висновки

В даному розділі було розглянуто компонентну базу та програмне забезпечення, яке необхідне для створення системи. В результаті було прийнято використовувати наступні рішення:

- контролер: Arduino
- модуль передачі даних: ESP8266
- датчики: BH1750 - (датчик освітленості) DHT22 - (датчик вологості та температури)
- веб-сервер: Apache
- база даних: MySQL
- відображення даних: веб-сторінка, соціальний месенджер

Розгляд проводився серед найбільш популярних рішень з урахуванням вихідних даних системи та доступності компонентів.

4 РЕАЛІЗАЦІЯ СИСТЕМИ ЗБОРУ, ЗБЕРІГАННЯ ТА ВІДОБРАЖЕННЯ ІНФОРМАЦІЇ ПРО НАВКОЛИШНЄ СЕРЕДОВИЩЕ

Розробка системи вимагає чіткого та структурованого підходу до її створення. Функціональну схему системи можна переглянути на (рис. 4.1). В даній частині роботи буде розглянутий поетапний процес створення системи.

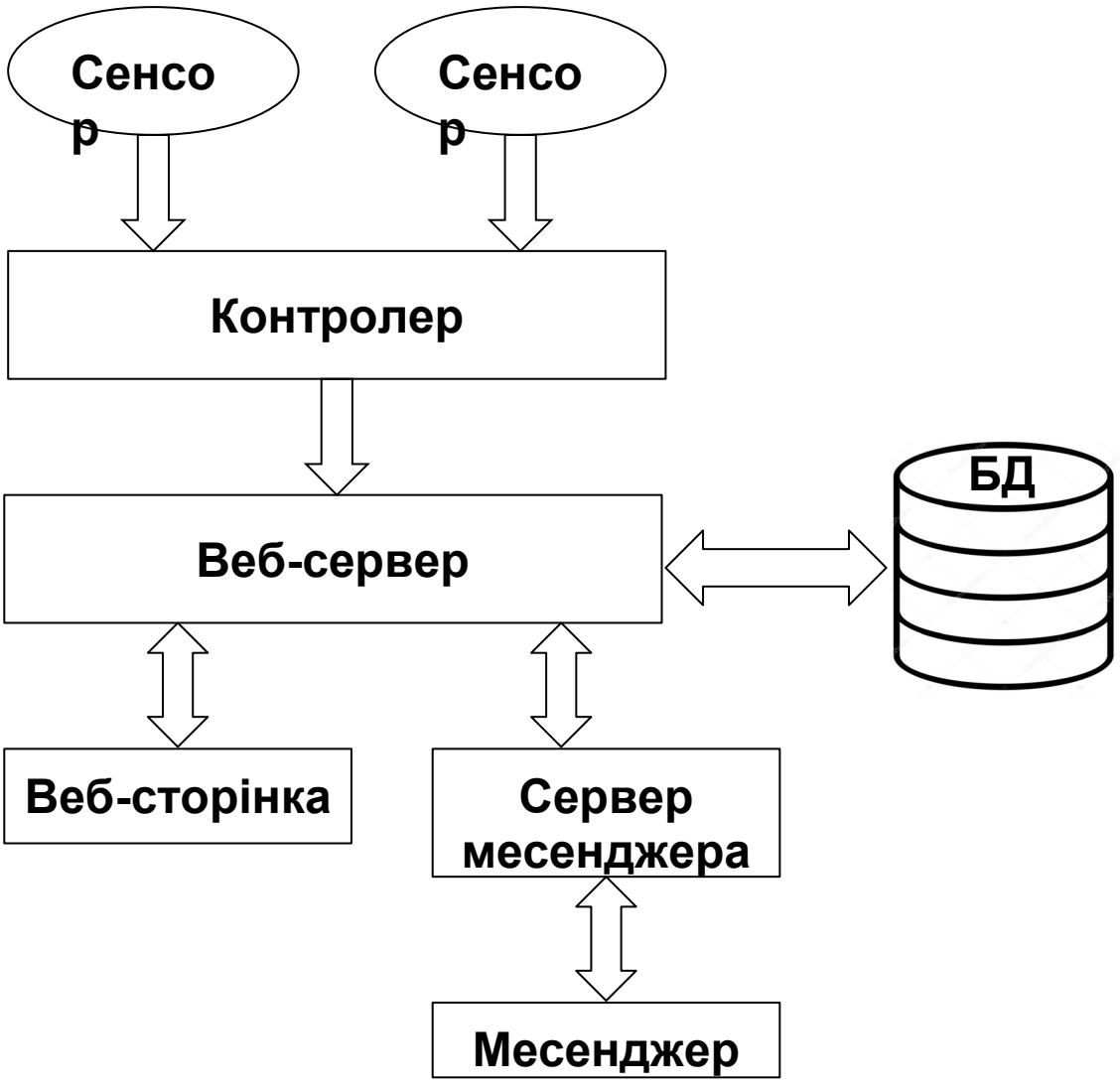

Рисунок 4.1 - Функціональна схема системи

4.1 Реалізація збору інформації

В етапі збору інфрмації приймаюсть участь контролер Arduino, датчик вологості повіттря DHT22 та датчик освітленості BH1750.

Датчик DHT22 має чотири контакти з яких використовується тільки три. Перший контакт відповідає за живлення датчика, другий - вивід даних, третій - не використовується, четвертий - земля. При підключені датчика між виходами живлення та виводом даних потрібно розмістити підтягуючий резистор номіналом 10 кОм. Схему підключення даного датчика до контролера Arduino, можна переглянути на (рис. 4.1.1).

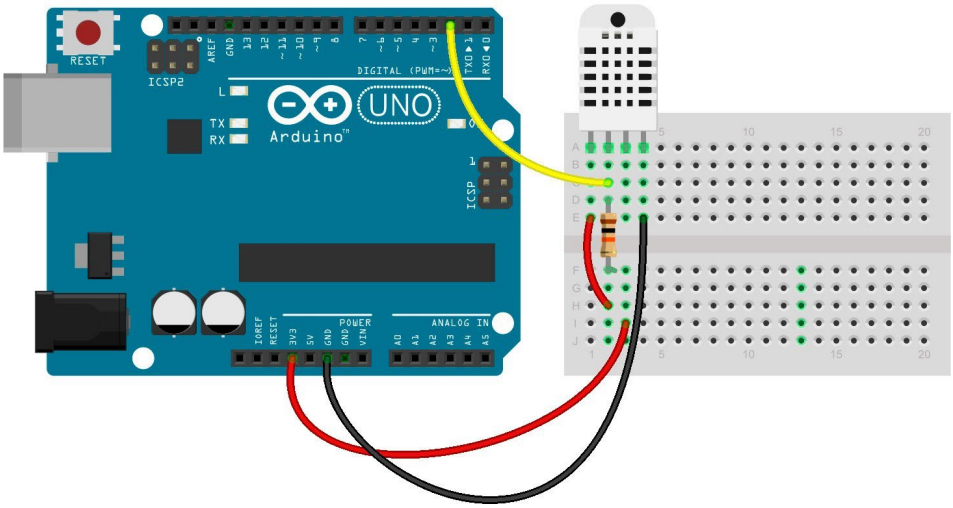

Рисунок 4.1.1 - Схема підключення датчика DHT22 до контролера Arduino

Після підключення датчика до контролера.ю необхідно запрограмувати контролер для зняття показників із датчика. Для цього використовується програма Arduino IDE. Детальний лістинг програми із коментарями можна переглянути в додатку А.

Далі необхідно до контролера підключити датчик освітленості BH1750. Даний датчик підключається та працює по шині I2C. I2C є послідовним протоколом зв'язку, тому дані передаються по біту по одному проводу. Даний протокол зв’язку є синхронним, тому вихід бітів синхронізується з

дискретизацією бітів за допомогою синхронізуючого сигналу, спільного між головним і підлеглим сигналами. Сигнал синхронізації завжди контролюється головним сигналом. Датчик для підключення ма 4 контакти. Перший - VCC (живлення 4,5 В), другий - ADDR (I2C slave-address), третій - GND (земля), четвертий - SDA (шина даних інтерфейсу I2C), п’ятий - SCL (шина тактування інтерфейсу I2C). Інтерфейс I2C в контролері Arduino реалізовано в аналогових виходах A4 (під’єднується шина даних), A5 (під’єднується шина тактування). Схему підключення датчика до контролера можна переглянути на (рис. 4.1.2).

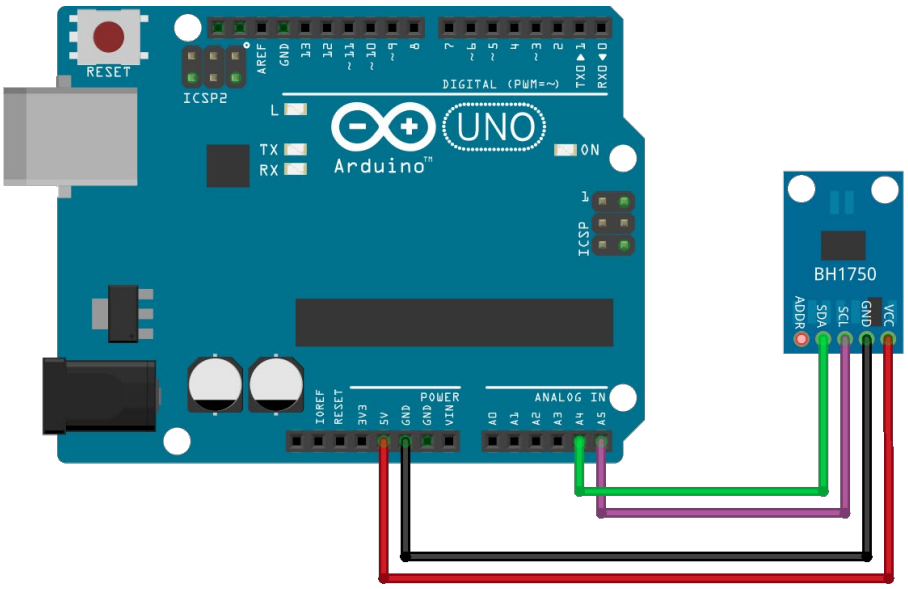

Рисунок 4.1.2 - Схема підключення датчика BH1750 до контролера Arduino

Наступний етап програмування контролера для зняття показників із датчика. Детальний лістинг програми із коментарями можна переглянути в додатку Б.

4.2 Реалізація передачі інформації

Після етапу збору інформації слідує етап передачі зібраних даних із датчиків на веб-сервер. На даному етапі, із компонетів системи приймає участь контролер Arduino, Wi-Fi модуль ESP8266 веб-сервер.

Для початку необхідно приєднати передавач до контролера. Кожен модуль із сімейства ESP8266 має 8 контактів, які повторюються в кожній із конфігурацій:

- GND - земля
- TX - підключення до RX контакту програматора та завантаження програми
- GPIO-2 - вхід/вихід загального призначення
- CH_EN - включення мікросхеми в режим енергозбереження
- GPIO-0 - вхід/вихід загального призначення
- RESET - перезавантаження модуля
- RX - підключення до TX контакту програматора та завантаження програми
- VCC - живлення 3,3 В

Підключення ESP8266 до контролера відбувається наступним чином:

до контаку живлення Arduino 3,3 В під’єднуються контакти VCC і CH_EN, земля контролера - до землі модуля, TX - до цифрового входу 0 Arduino, RX - до цифрового входу 1 Arduino. Перед підключенням потрібно врахувати то факт що напруга живленя ESP8266 не повинна перевищувати 3,6 В, в той час як напруга на платі Arduino становить 5 В, тому з’єднання мікроконтролерів необхідно проводити, використовуючи навантажувальні резистори. Детальну схему підключення можна переглянути на (рис.4.2.1).

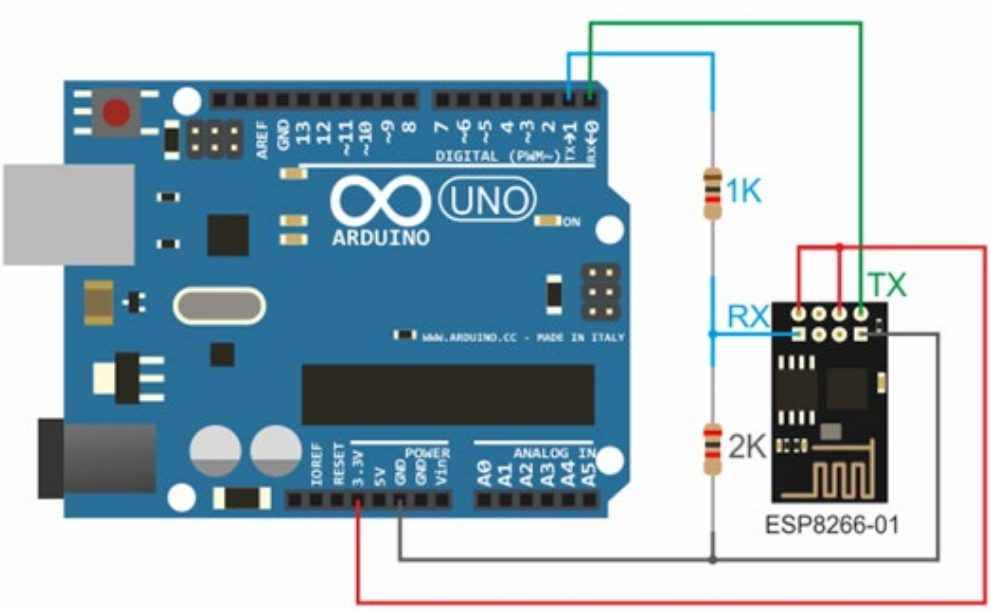

Рисунок 4.2.1 - Схема підключення модуля ESP8266 до контролера Arduino UNO

Після з’єднання між собою контролера та Wi-Fi модуля йде етап програмування, який представлений в додатку В. Після програмування частина системи, яка складається із датчиків, контролера та Wi-Fi модуля буде здатна фіксувати дані про навколишнє середовище та посилати їх на веб-сервер.

Відсилання даних відбувається у вигляді GET запиту на сервер по раніше заданому маршруті та параметрами на сервері. URL запиту формується програмно в залежності від отриманих із датчиків даних та має має наступний вигляд:

http://diplomastation.000webhostapp.com/api/set?temperature=23&humidity=73&light=253&key=fi13MMWTqKC5gpNNZIz2 де:

http://diplomastation.000webhostapp.com - адреса хоста на якому розміщений веб-сервер для обробки результатів із датчиків;

/api/set - частина URL, яка вказує на те, який програмний код буде виконаний для обробки даних запитів;

? - ідентифікатор який показує, що в наступній частині URL передаються дані;

temperature=23&humidity=73&light=253&key=fi13MMWTqKC5gpNNIz2 - частина URL, яка відповідає за передачу інформації. Вона має наступні параметри:

temperature - температура повітря отримана із датчика DHT22;

humidity - вологість повітря отримана із датчика DHT22;

light - освітленість отримана із датчика BH1750;

key - спеціальний секретний ключ, для запобігання фіксування даних на сервері зі сторонніх пристроїв;[13]

4.3 Створення портативного пристрою

Після коректного з’єднання всіх компонентів портативного пристрою між со між собою, та програмування контролера портативний пристрій готовий до використання. В ході його створення до пристрою був добавлений корпус, і результаті чого він матиме вигляд закінченого приладу (рис. 4.3.1).

Розроблений пристрій не є досить вибагливим до живлення. В якості джерела живлення може бути використаний блок живлення від 5 В до 12 В. Оскільки Wi-Fi модуль та датчики живляться від портів контролера, то додаткового живлення інші компоненти пристрою не потребують.

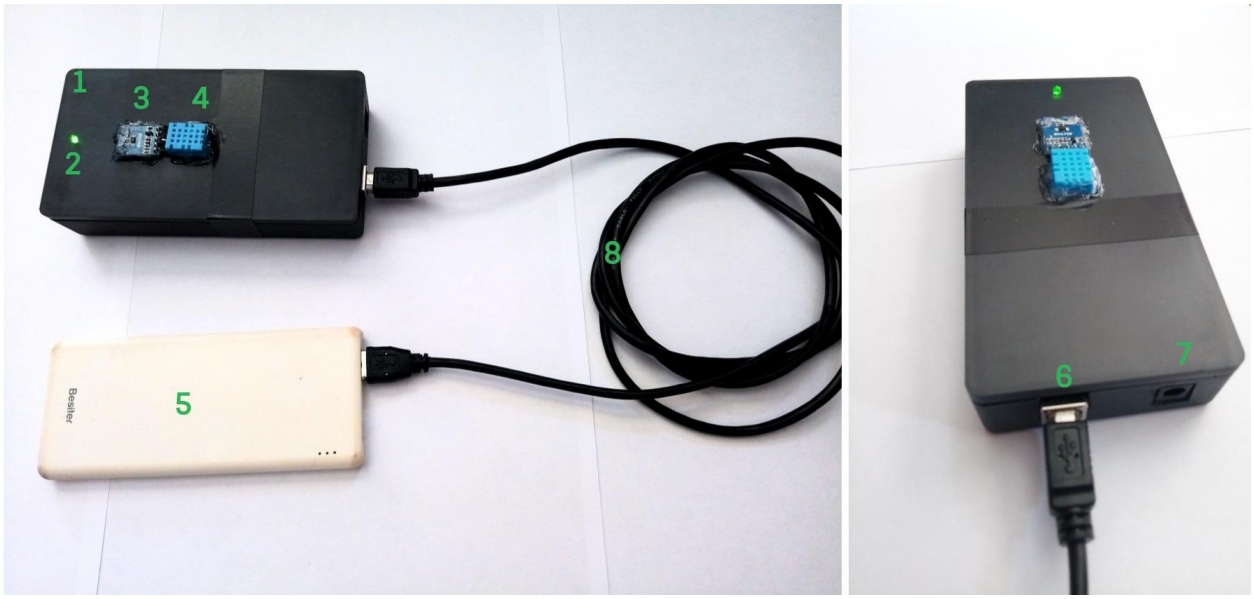

Рисунок 4.3.1 - Реалізація портативного пристрою збору інформації

(1 - корпус, 2 - індикатор роботи, 3 - датчик освітленості BH1750, 3 - датчик температури та вологості повітря DHT22, 4 - елемент живлення, 5 - вхід USB тип B для програмування контролера або живлення пристрою, 6 - вхід живлення, 7 - кабель живлення)

4.4 Реалізація збереження інформації

В даному етапі роботи системи приймає участь веб-сервер та база даних. В якості серверної мови програмування будемо використовувати PHP, так як вона є досить популярною та головним чином спрямована на написання веб-додатків. В якості БД буде використовуватися СУБД MySQL.

Для початку необхідно в створити базу даних, в якій будуть зберігатися дані із датчиків. Для цього необхідно запустити СУБД та виконати команду:

```
mysql> CREATE DATABASE databaseName ;
```

Після створення бази даних необхідно переключитися на її використання та створити в ній таблицю для збереження даних. Для створення таблиці можна використати наступну команду:

```
mysql> CREATE TABLE `indicators` (
   `id` INT NOT NULL AUTO_INCREMENT ,
   `temperature` FLOAT NOT NULL ,
   `humidity` FLOAT NOT NULL ,
   `light` FLOAT NOT NULL ,
   `created_at` TIMESTAMP NOT NULL , PRIMARY KEY (`id`)
) ENGINE = InnoDB CHARSET=utf8mb4 COLLATE utf8mb4_general_ci;
```

В результаті буде створена таблиця, яка готова до використання та має наступні колонки:

id - унікальний ідентифікатор запису типу int;

temperature - поле для збереження показника температури типу float;

humidity - поле для збереження показника вологості повітря типу float;

light - поле для збереження показника освітленості типу float;

created_at - поле для збереження дати та часу оримання даних;

Перейдемо до етапу прийому даних. На на маршрут /api/set приходять GET запити з даними від контролера. Для їх обробки використаємо певну послідовність дій. Спочатку дістанемо з файлу з налаштуваннями таємний ключ та перевіримо його чи він присутній в запиті та чи співпадає з ключем із налаштувань. Якщо співпадає зберігаємо дані в БД, якщо ні - пропускаємо запит і чекаємо наступний. Програмну реалізацію описаних дій можна переглянути в додатку Г.

4.5 Реалізація відображення інформації

Даний етап ставить перед собою завдання представити зібрані результати з датчиків перед користувачем. Для відображення даних буде

використано два різних способа. Відображення даних на веб сторінці та отримання останніх актуальних даних з використанням соціального месенджера.

4.5.1 Відображення інформації на Веб сторінці

Веб-сторінка є гіпертекстовим ресурсом мережі Інтернет, який створений з допомогою мови розмітки HTML. Програмою, як дозволяє інтерпретувати HTML - файли є вею-браузер. Дані файли являть собою звичайні текстові файли з певними інструкціями - тегами, які дозволяють задавати форматування тексту. Зазвичай HTML - файли знаходяться або динамічно формуються на веб-сервері, в залежності від типу сторінки (динамічна, статична). Механізм відображення веб-сторінки є наступним:

1. Браузер відкриває з’єднання з веб-сервером
2. Браузер відправляє серверу запит на отримання веб-сторінки
3. Сервер формує формує відповідь та закриває з’єднання.
4. Браузер опрацьовує відповідь сервера та відображає контент сторінки.

Даний механізм можна переглянути на (рис. 4.4.1.1).

Зазвичай для представлення даних на веб-сторінці самої HTML розмітки не достатньо. Тому для більш детального контролю зовнішнім виглядом використовують технологію CSS (Cascading Style Sheets – каскадні таблиці стилів).

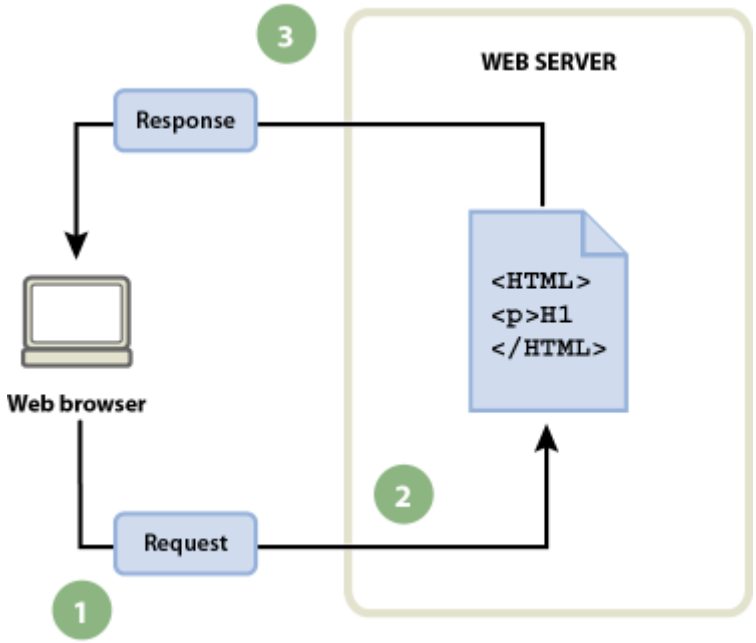

Рисунок 4.4.1.1 - Схема відображення веб-сторінки

В даному випадку веб-сторінка буде динамічною - в різні моменти часу інформація на сторінці буде різна. Так як на мова програмування, що використовується на сервері PHP, то формувати сторінку будемо з її допомогою. Для початку необхідно отримати дані з бази даних. Для цього необхідно виконати наступний запит до БД: "SELECT * FROM indicators".

Після отримання даних із БД, їх необхідно підставити у раніше підготовлений HTML шаблон, див додаток Д. Підставивши дані, постає завдання у відображені даних в графічному вигляді. Для побудови графіків буде використана мова програмування JavaScript, так як саме її здатний інтерпретувати веб-браузер. Для спрощення процесу побудови графіків буде використано бібліотеку ChartJS. Після завершення формування HTML шаблона, він відправляється веб-браузеру в якості відповіді веб-сервера (рис. 4.5.1.2).

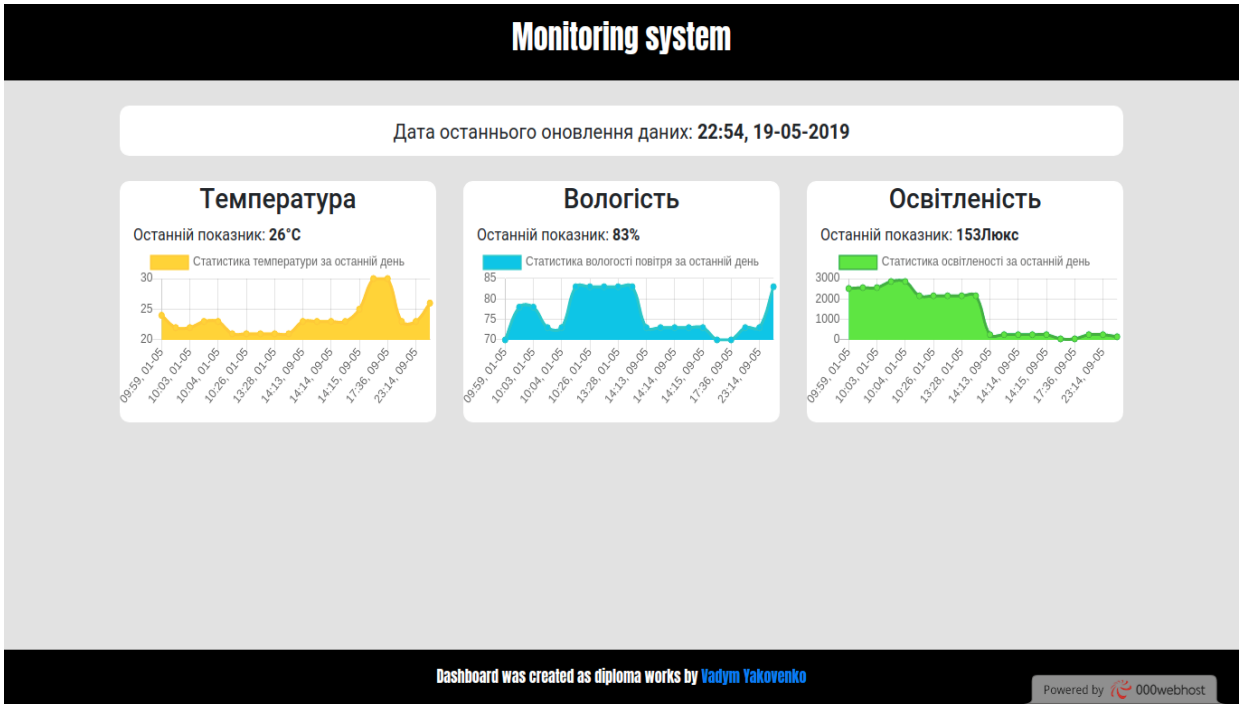

Рисунок 4.5.1.2 - Веб-сторінка для відображення зібраних даних системою

4.5.2 Відображення інформації за допомогою Telegram bot

Наступним видом відображення інформації, який використовуватиме система, є відображення інформації показників за допомогою соціального месенджера. В даному випадку буде використано сервіс обміну повідомленнями Telegram.

Перед початком використання сервісу Telegram необхідно в сервісі створити самого бота. Для цього в пошуку сервісу необхідно знайти аккаунт @BotFather та дати йому команду “/newbot”, після цього буде створений власний бот та видано його токен - унікальний ідентифікатор, який дає доступ для керування ботом, методами API сервісу Telegram. Після створення бота, необхідно в сервісі зареєструвати “вебхук” - адреса, веб сервера, на яку сервіс буде надсилати запити з інформацією про дії користувача. Встановивши “вебхук” сервіс Telegram готовий до використання.

Процес отримання інформації про навколишнє середовище, за допомогою соціального месенджера складатиметься з шести послідовних етапів (рис. 4.4.2.1).

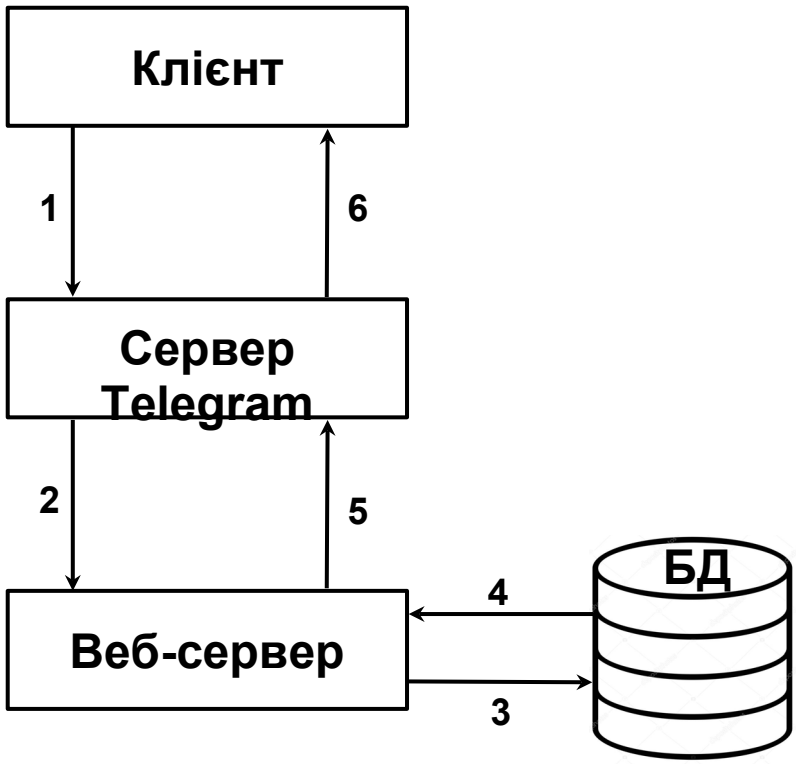

Рисунок 4.5.2.1 - Схема отримання інформації про навколишнє середовище з використанням соціального месенджера Telegram

На першому етапі користувач використовуючи клієнт Telegram (мобільний або десктопний додаток) відсилає команду. Сервер Telegram отримує запит від клієнта, формує дані для відправки на встановлений по “вебхуку” веб-сервер та відправляє їх. Дані відправляються в форматі JSON, містять інформацію про користувача, який зробив запит і інформацію про зроблену команду.

На третьому етапі, веб сервер отримує запит, аналізує інформацію введену користувачем та приймає вибір щодо подальших дій. Якщо була задана коректна команда то веб-сервер формує і робить запит до бази даних для отримання останніх актуальних даних із датчиків, якщо не коректна -

нічого не робить й очікує наступну команду. Після цього, якщо користувач ввів вірні дані, веб-сервер отримує із БД дані останніх показників датчиків, формує запит для відповіді до серверу Telegram відповідно до правил API, та робить сам запит. На останньому, шостому, етапі сервер месенджера приймає дані від веб-сервера, аналізує їх коректність і вразі відсутності помилок валідації відправляє сповіщення клієнту.

Після успішного проходження усіх шести стадій виконнання запиту, користувач матиме змогу успішно переглянути інформацію про навколишнє середовище, яку він запросив у Telegram бота. Чат між користувачем та ботом можна переглянути на (рис. 4.5.2.2).

Рисунок 4.5.2.2 - Результат взаємодії користувача та Telegram bot

4.5.3 Реалізація системи сповіщень

Важливою складовою системи є її частина, яка відповідає за критичні сповіщення показників. У інформаційних технологіях система оповіщення являє собою комбінацію програмного забезпечення та апаратних засобів, що забезпечує засоби доставки повідомлень до користувачів системи. Такі системи є важливим аспектом сучасних веб-додатків.

Оскільки однією із складових системи є соціальний месенджер Telegram, то для відправки сповіщень можна використати його.

Система сповіщень буде працювати за певним алгоритмом. На початковому етапі веб-сервер отримує інформацію із контролера. Після збереження інформації в БД, відбувається порівняння отриманих даних із раніше встановленими граничними показниками. Для цього їх потрібно дістати з БД. Якщо показники датчиків виходять за встановлені межі, необхідно відправити сповіщення.

Для надсилання сповіщень буде використовуватися API сервісу Telegram. Для надсилання сповіщення в автоматичному режимі, з використання бота, необхідно прив’язати Telegram акаунт користувача до системи. Процес прив’язки являє собою виконання послідовності дій для отримання унікального “chat_id” із сервісу Telegram. Даний ідентифікатор являє собою унікальний ключ, що дозволяє по ньому надсилати повідомлення користувачеві. Після прив’язки все готове до надсилань сповіщень. В разі виконання умови на надсилання сповіщення, веб-сервер формує та робить запит із сповіщенням до сервісу Telegram, який в свою чергу надсилає сповіщення користувачеві на його клієнти (мобільний додаток, десктопний додаток чи веб-версію клієнтів). Приклад сповіщення можна переглянути на (рис. 4.5.3.1).

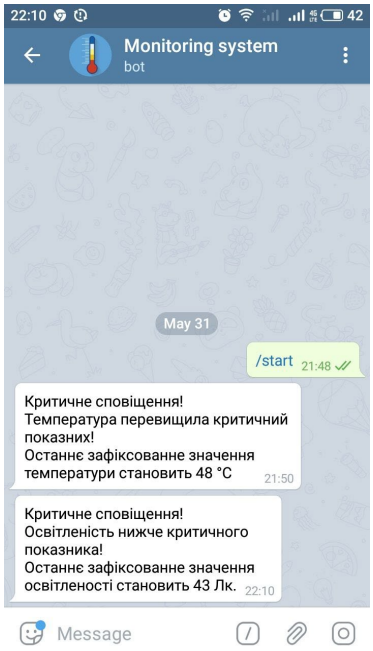

Рисунок 4.5.3.1 - Відображення критичних сповіщень в мобільному додатку

Розглянемо, раніше згаданий, процес привязки Telegram профілю та встановлення допустимих меж показників. Для даних завдань створена окрема веб-сторінка, див рис 4.4.3.2. Для початку, користувачеві необхідно перейти на дану сторінку та натиснути на кнопку “Приєднати Telegram”, відбудеться перехід на наступну сторінку з пропозицією відкрити клієнт месенджера. Прийнявши пропозицію, на пристрої користувача буде відкрито клієнт Telegram. Після цих операції у клієнті необхідно натиснути кнопку “START” і профіль буде автоматично прив’язаний до системи. Коли користувач, в клієнті Telegram, натискає кнопку “START”, з клієнта надсилається запит до сервісу Telegram, фіксує той факт, що користувач бажає приймати повідомлення від бота. Далі сервер Telegram надсилає запит з інформацією про користувача на раніше зареєстровану адресу вебхука. Веб сервер отримує даний запит з даними, та зберігає із них “chat_id” в БД. На цьому завершується прив’язка Telegram профілю до системи. В разі успішної прив’язки каналу розсилки, після оновлення сторінки налаштувань, частина сторінки, що відповідає за прив’язку месенджера, відображатися не буде.

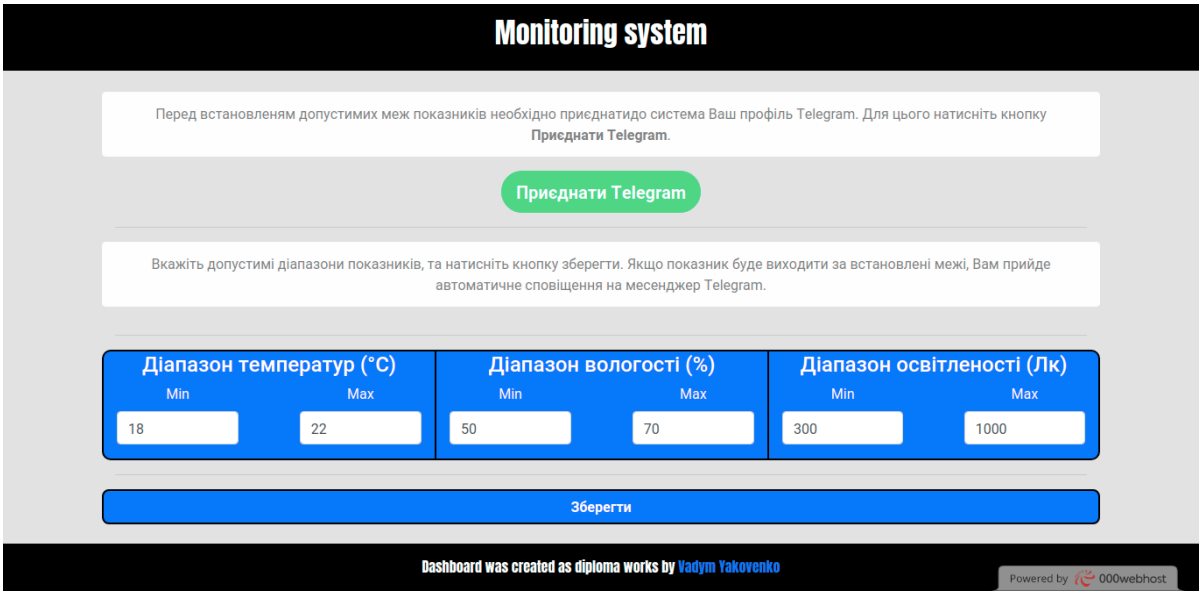

Рисунок 4.5.3.2 - Веб-сторінка для налаштувань автоматичних сповіщень.

Після успішної прив’язки сервісу для розсилки сповіщень, необхідно встановити допустимий діапазон значень показників датчиків, виходячи за які, будуть надсилатися критичні сповіщення. Для встановлення діапазону, користувачеві необхідно заповнити форму та натиснути кнопку зберегти. Дані з форми будуть відправлені на веб-сервер, який в свою чергу збереже дані в базу даних для подальшого використання.

4.6 Висновки

В даному розділі описано детальний, поетапний процес створення та функціонування системи збору збереження та відображення інформації про навколишнє середовище. Отримана, в результаті створення, система складається з двох частин: портативного приладу та серверної частини. До складу портативного приладу входить мікроконтролер, набір датчиків для фіксації даних та модуль для передачі даних. Серверну частину можна безпосередньо розділити ще на дві складові. До першої складової входить веб-сервер, для прийому запитів від контролера та база даних для збереження інформації. Другою складовою являється сторонній сервіс Telegram,

призначений для відображення актуальних показників датчиків та для розсилки критичних повідомлень.

ВИСНОВКИ

В ході виконання дипломної роботи було проаналізовано процес моніторингу інформації та розроблено систему для збору, збереження та відображення інформації про навколишнє середовище.

В першому розділі роботи було розглянуто розглянуто в загальному вигляді поняття моніторингу, з’ясовано, які вимоги висуваються до систем моніторингу та чого можна досягди отримуючи регулярні дані із системи моніторингу. Також зроблено огляд та порівняльну характеристику декількох, представлених на ринку систем моніторингу інформації та визначено кращий варіант із них.

У другому розділі роботи сформовано вимоги до системи, які висувають користувачі. Дані вимоги, відповідно до характеру, було розподілено на чотири групи: користувацькі, функціональні, зовнішніх інтерфейсів та системні .

В третьому розділі розглянуто процес вибору компонентної бази та програмного забезпечення, яке необхідне для створення системи. Розгляд проводився серед найбільш популярних рішень з урахуванням вихідних даних системи та доступності компонентів.

В четвертому розділі було описано детальний, поетапний процес створення та функціонування системи. Отримана, в результаті створення, система складається з двох частин: портативного пристрою та серверної частини. До складу портативного приладу входить мікроконтролер, набір датчиків, для фіксації даних, та модуль для передачі даних. До серверної частини входить веб-сервер, для прийому запитів від контролера, база даних для збереження інформації та сторонній сервіс Telegram, призначений для відображення актуальних показників датчиків та для розсилки критичних повідомлень.

Результуюча система може мати досить широкий спектр використання. Вона може встановлюватися в складських приміщеннях, тепличних комплексах, інкубаторах та багатьох інших місцях, які потребують моніторингу температури, вологості повітря та освітленості. Завдяки сучасній реалізації відображення даних, показники можна буде переглянути як на персональному комп’ютері так і на смартфоні. Зручна функція критичних сповіщень дозволить вчасно приймати рішення для запобігання негативних наслідків.

ПЕРЕЛІК ПОСИЛАНЬ

1. Power System Monitoring and Control Wiley - IEEE/Hassan Bevrani, Masayuki Watanabe, Yasunori Mitani - John Wiley & Sons, 2014 - 288c.:
2. Офіційний сайт Raspberry Pi [Електронний ресурс] – Режим доступу: https://www.raspberrypi.org/ – Дата доступа: 10.05.2017
3. George E.P. Box. Time Series Analysis: Forecasting and Control 5th Edition / George E.P. Box, Gwilym M. Jenkins, Gregory C. Reinsel, Greta M. Ljung. – USA: John Wiley & Sons P, 2016. – С. 250-252.
4. Офіційний сайт Arduino [Електроннийесурс] – Режим доступу: https://store.arduino.cc/arduino-uno-rev3 – Дата доступа: 12.05.2017
5. Офіційний сайт ESPRESSIF [Електронний ресурс] – Режим доступу: https://www.espressif.com/products/hardware/esp8266ex/overview/ – Дата доступу: 13.05.2017
6. Документація датчика DHT22 [Електронний ресурс] – Режим доступу: https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf – Дата доступу: 9.05.2017
7. Документація датчика BH1750 [Електронний ресурс] – Режим доступу: https://www.mouser.com/ds/2/348/bh1750fvi-e-186247.pdf – Дата доступу: 9.05.2017
8. PHP7 / Д.В.Котеров, И.В. Симдянов. - СПБ.:БХВ-Петербург, 2016 - 1088с.:
9. WebServer [Електронний ресурс] – Режим доступу: https://news.netcraft.com/archives/2019/04/22/april-2019-web-server-survey.html – Дата доступу: 15.05.2017
10. Введение в реляционные базы данных / В. В. Кириллов, Г. Ю. Громов. — СПб.: БХВ-Петербург, 2009. — 464 с.:

11. HTML5, CSS3 и JavaScript. Исчерпывающее руководство / Дженнифер Роббинс; [пер. с англ. М. А. Райтман]. — 4-е издание. — М. : Эксмо, 2014. — 528 с.

12. Офіційний сайт Telegram APIs [Електронний ресурс] – Режим доступу: https://core.telegram.org/

13. Офіційний сайт PHP [Електронний ресурс] – Режим доступу: https://php.net/

ДОДАТОК А

Лістинг програми для зняття показників із датчика DHT22:

```
#include "DHT.h"
#define DHTPIN 2    // пін  виводу інформації

//вибір використовуваного датчика
#define DHTTYPE DHT22

//инициализация датчика
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(9600);
  dht.begin();
}

//головний цикл
void loop() {
  // зчитування температури і вологості
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  // перевірка правильності даних
  if (isnan(t) || isnan(h)) {
    Serial.println("Error reading from DHT");
  } else {
    Serial.print("Humidity: ");
    Serial.print(h);
    Serial.print(" %t");
    Serial.print('\n');
    Serial.print("Temperature: ");
    Serial.print(t);
    Serial.println(" *C");
  }

  delay(3000); //затримка 3 сек

}
```

ДОДАТОК Б

Лістинг програми для зняття показників із датчика BH1750:

```
// підключення бібліотеки I2C:
#include <Wire.h>
// підключення бібліотеки датчика BH1750:
#include <BH1750.h>
//оголошення об'єкта lightMeter:
BH1750 lightMeter;
void setup() {
 Serial.begin(9600);
 //ініціалізація датчика BH1750
 lightMeter.begin();
}
void loop() {
 //зчитування показників із датчика
 uint16_t lux = lightMeter.readLightLevel();
 //вивід результату
 Serial.print("Light: ");
 Serial.println(String(lux) + " lx");
 delay(3000); //затримка 3 сек
}
```

ДОДАТОК В

Лістинг коду роботи WiFi модуля:

```
#include <SoftwareSerial.h>
#define RX 10
#define TX 11
String AP = "TOP7_GUEST";      // CHANGE ME
String PASS = "19guestpleaseletmein"; // CHANGE ME
String HOST = "diplomastation.000webhostapp.com";
String PORT = "80";
int countTrueCommand;
int countTimeCommand;
boolean found = false;
int valSensor = 1;
SoftwareSerial esp8266(RX,TX);
void setup() {
 Serial.begin(9600);
 esp8266.begin(115200);
 sendCommand("AT",5,"OK");
 sendCommand("AT+CWMODE_CUR=1",5,"OK");
 sendCommand("AT+CWJAP=\""+ AP +"\",\""+ PASS +"\"",20,"OK");
Serial.println("Setup done");
}
void loop() {
 String getData = "GET
api/set?temperature=23&humidity=73&light=253&key=fi13MMWTqKC5gpNNZlz2";
  sendCommand("AT",5,"OK1");
 sendCommand("AT+CIPMUX=1",5,"OK");
 sendCommand("AT+CIPSTART=0,\"TCP\",\""+ HOST +"\","+ PORT,15,"OK");
 sendCommand("AT+CIPSEND=0," +String(getData.length()+4),4,">");
 esp8266.println(getData);delay(1500);countTrueCommand++;
```

```
 sendCommand("AT+CIPCLOSE=0",5,"OK");
}

void sendCommand(String command, int maxTime, char readReplay[]) {
 Serial.print(countTrueCommand);
 Serial.print(". at command => ");
 Serial.print(command);
 Serial.print(" ");
 while(countTimeCommand < (maxTime*1)) {
  esp8266.println(command);//at+cipsend
  if(esp8266.find(readReplay)) {
   found = true;
   break;
  }
  countTimeCommand++;
 }
 if(found == true)  {
  Serial.println("OYI");
  countTrueCommand++;
  countTimeCommand = 0;
 }

 if(found == false)  {
  Serial.println("Fail");
  countTrueCommand = 0;
  countTimeCommand = 0;
 }
 found = false;
}
```

ДОДАТОК Г

Лістинг коду збереження даних:

```
<?php
        class ApiController
        {
          public function set()
          {
            $key = Config::get('app.api_key');
            if( isset($_GET['key']) && $_GET['key'] == $key ){
              return Indicator::create([
                'temperature' => $_GET['temperature'],
                'humidity' => $_GET['humidity'],
                'light' => $_GET['light'],
              ]);
            }
            return false;
          }
        }
class Indicator
{
  private const TABLE = 'indicators';
  public static function create(array $params)
  {
    $params['created_at'] = Carbon::now(Config::get('app.timezone'));
    $db = DB::getInstance();
    $result = $db->insert(self::TABLE, $params);
    var_dump($result);
  }
}
```

ДОДАТОК Д

Лістинг коду веб-сторінки:

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="description" content="">
    <title>Diploma Y.V.V.</title>
    <link rel="canonical" href="https://getbootstrap.com/docs/4.3/examples/cover/">
    <link href="https://fonts.googleapis.com/css?family=Anton" rel="stylesheet">
    <link href="css/bootstrap.css" rel="stylesheet" >
    <link href="css/style.css" rel="stylesheet" >
  </head>
  <body class="text-center">
    <header> <h2>Diploma meteostation</h2> </header>
    <main class="content">
      <div class="container">
        <div class="row">
          <div class="col-md-12">
            <div class="update-container">
                Дата останнього оновлення даних: <strong> <?=$lastUpdate ?
                $lastUpdate : '-'?></strong>
            </div>
          </div>
        </div>
        <div class="row">
          <div class="col-md-4">
            <div class="chart-container">
              <h3>Температура</h3>
              <div class="last-value">
                Останній показник: <strong>20°C</strong>
              </div>
```

```
                    <div class="chart">
                        <canvas id="temp"></canvas>
                    </div>
                </div>
            </div>
            <div class="col-md-4">
                <div class="chart-container">
                    <h3>Вологість</h3>
                    <div class="last-value">
                        Останній показник: <strong>50%</strong>
                    </div>
                    <div class="chart"><canvas id="humidity"></canvas></div>
                </div>
            </div>
            <div class="col-md-4">
                <div class="chart-container">
                    <h3>Освітленість</h3>
                    <div class="last-value">
                        Останній показник: <strong>5000Л</strong>
                    </div>
                    <div class="chart"><canvas id="ligth"></canvas> </div>
                </div>
            </div>
        </div>
    </main>
    <footer>
        Dashboard was created as diploma works by <a
href="https://www.instagram.com/vadyakovenko/">Vadym Yakovenko</a>
    </footer>
    <script src="https://cdnjs.cloudflare.com/ajax/libs/Chart.js/2.8.0/Chart.bundle.js"></script>
    <script src="js/main.js"></script>
    <script>
        var data = {
```

```
            lables: [
                <?php foreach($indicators as $indicator):?>
                    '<?=$indicator['created_at']?>',
                <?php endforeach?>
            ],
            temperature: [
                <?php foreach($indicators as $indicator):?>
                    '<?=$indicator['temperature']?>',
                <?php endforeach?>
            ],
            humidity: [
                <?php foreach($indicators as $indicator):?>
                    '<?=$indicator['humidity']?>',
                <?php endforeach?>
            ],
            light: [
                <?php foreach($indicators as $indicator):?>
                    '<?=$indicator['light']?>',
                <?php endforeach?>
            ],
        };
        runCharts(data);
    </script>
  </body>
</html>
```