Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
(інститут)

Факультет інформаційних технологій
(факультет)

Кафедра Програмного забезпечення комп'ютерних систем
(повна назва)

ПОЯСНЮВАЛЬНА ЗАПИСКА
кваліфікаційної роботи ступеня

бакалавра
(назва освітньо-кваліфікаційного рівня)

студентки *Мудрук Вікторії Дмитрівни*
(ПІБ)

академічної групи *122-20-2*
(шифр)

напряму підготовки *122 Комп'ютерні науки*
(код і назва напряму підготовки)

на тему: *Розробка комп'ютерної системи агрегування даних агромоніторингу*

Керівники	**Прізвище, ініціали**	**Оцінка за шкалою**		**Підпис**
		рейтинговою	**інституційною**	
кваліфікаційної роботи	*проф. Лактіонов І.С.*			
розділів:				
спеціальний	*проф. Лактіонов І.С.*			
економічний	*доц. Касьяненко Л.В.*			
Рецензент				
Нормоконтролер	*доц. Гуліна І. Г.*			

Дніпро
2024

Міністерство освіти і науки України
НТУ «Дніпровська політехніка»

ЗАТВЕРДЖЕНО:
завідувач кафедри
програмного забезпечення комп'ютерних систем
(повна назва)

_____ М.О. Алексєєв
(підпис) (прізвище, ініціали)

« » _____ 2024 року

ЗАВДАННЯ
на кваліфікаційну роботу

бакалавра
(назва освітньо-кваліфікаційного рівня)

студентки *122-20-2* (група) *Мудрук Вікторії Дмитрівни* (прізвище та ініціали)

тема кваліфікаційної роботи *Розробка комп'ютерної системи агрегування даних агромоніторингу*

затверджена наказом ректора НТУ «ДП» від *23 . 05 .2024 р.* № *469 -с*

Розділ	Зміст виконання	Термін виконання
Спеціальний	***На основі матеріалів виробничої практики та інших науково-технічних джерел провести аналіз стану рішення проблеми та постановку задачі.*** ***Обґрунтувати вибір та здійснити реалізацію методів вирішення проблеми***	***10.06.2024 р.***
Економічний	***Провести розрахунок трудомісткості розробки програмного забезпечення, витрат на створення ПЗ й тривалості його розробки***	***14.06.2024 р.***

Завдання видав _____ *проф. Лактіонов І.С.*
(підпис) (посада, прізвище, ініціали)

Завдання прийняв до виконання _____ *Мудрук В.Д.*
(підпис) (прізвище, ініціали)

Дата видачі завдання: *14.01.2024 р.*

Термін подання кваліфікаційної роботи до ЕК: *17.06.2024 р.*

РЕФЕРАТ

Пояснювальна записка: 67 с., 18 рис., 4 дод., 6 табл., 21 джерел.

Об’єкт розробки: комп’ютерна система агрегування даних для автоматизації процесів збору, обробки, аналізу та візуалізації даних, отриманих з різних сенсорів і пристроїв, що встановлені на аграрних полях та фермах.

Мета кваліфікаційної роботи: підвищення ефективності сільськогосподарських робіт за рахунок оперативного та точного моніторингу стану рослин, ґрунту, а також інших параметрів навколишнього середовища що забезпечить українських фермерів інформацією для прийняття обґрунтованих рішень.

Основними методами, які використовувались у дослідженні, є: аналіз літературних джерел та технічної документації, визначення потреб та вимог українських фермерів для створення системи, моделювання, проектування та розробка архітектури системи, застосування технологій інтернету речей (IoT), зокрема, мікроконтролерів з Wi-Fi модулями для підключення до мережі Інтернет і відправлення даних на хмарне сховище.

У вступі розглядається сучасний стан проблеми та її аналіз, визначається мета дипломної роботи, її галузь використання, наводиться обґрунтування актуальності теми та конкретизується постановка завдання.

У першому розділі був проведений аналіз стану технологій точного землеробства, було визначено актуальність завдання та призначення розробки, зазначено вимоги щодо програмної реалізації, програмних засобів та технологій .На основі проведеного аналізу виявлено недоліки існуючих систем агротехнічного моніторингу, зокрема, масштабованість виключно для великих підприємств, що робить їх недоступними для господарств малого та середнього обсягу.

У другому розділі було проаналізовано функціональні можливості та технологічні аспекти розробленої системи, було детально описано використані технології, зокрема мікроконтролери та плата розробки, їх технічні характеристики та переваги у використанні для проекту, проаналізовано інтерфейс середовища розробки.

В економічному розділі визначено трудомісткість розробки програмного забезпечення, проведено розрахунок вартості робіт зі створення програми та встановлено необхідний час для її розробки.

Практичне завдання полягає у створенні системи моніторингу та контролю стану рослин, ґрунту та інших параметрів навколишнього середовища. Система дозволяє збирати, аналізувати та комп’ютеризовано контролювати дані в режимі реального часу. Висновки за результатами дослідження підкреслюють ефективність запропонованої системи Інтернету речей у моніторингу та контролі умов навколишнього середовища.

Список ключових слів: МІКРОКОНТРОЛЕР, АГРОТЕХНІЧНИЙ МОНІТОРИНГ, WI-FI, ДАВАЧ, ЗБІР ДАНИХ, ОБРОБКА ДАНИХ, ВІЗУАЛІЗАЦІЯ ДАНИХ, СХЕМА.

ABSTRACT

Explanatory note: 67 pages, 18 fig, 4 appendix, 6 table, 21 sources.

Object of development: a computer system of data aggregation to automate the processes of gathering, processing, visualization and analysis of agricultural data received from various sensors and devices installed on fields and farms.

The purpose of the qualification work: to increase the efficiency of agricultural work due to timely and accurate monitoring of the condition of plants, soil, as well as other environmental parameters, which will provide Ukrainian farmers with information for making justified decisions.

The main methods used in the research are: analysis of literary sources and technical documentation, determination of needs and requirements of Ukrainian farmers for the creation of a system, modeling, design and development of the system architecture, application of Internet of Things (IoT) technologies, in particular, microcontrollers with Wi-Fi modules for connecting to the Internet and sending data to cloud storage.

The introduction examines the current state of the problem and its analysis, defines the purpose of the thesis, the field of its use, provides a rationale for the topicality of the topic, and specifies the statement of the task.

In the first section, an analysis of the state of precision agriculture technologies was carried out, the relevance of the task and the purpose of the development were determined, the requirements for program implementation, software tools and technologies were specified. Based on the conducted analysis was found a number of concurrent systems downsides, such as inaccessibility to small and medium-sized farmsteads.

The second section analyzed the functionality and technological aspects of the developed system, described in detail the technologies used, in particular microcontrollers and the development board, their technical characteristics and advantages in use for the project, analyzed the interface of the development environment.

In the economic section, the labor intensity of software development is determined, the cost of work on creating the program is calculated, and the necessary time for its development is estimated.

The practical task is creating a system to monitor and control the condition of plants, soil and other environmental parameters. The presented system allows collecting, analyzing and managing the data in real time. Conclusions based on the results of the study emphasize the effectiveness of the proposed Internet of Things system in monitoring and controlling environmental conditions.

Key words list: MICROCONTROLLER, AGRICULTURAL MONITORING, WI-FI, TRANSMITTER, DATA COLLECTION, DATA PROCESSING, DATA VISUALIZATION, SCHEME.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

GPS – Global Positioning System

Wi-Fi – Wireless Fidelity

IoT – Internet of Things

MCU – MicroController Unit

IDE – Integrated Development Environment

HTTPS – Hypertext Transfer Protocol Secure

USB – Universal Serial Bus

UART – Universal Asynchronous Receiver-Transmitter

GPIO – General Purpose Input-Output

ВСТУП

Україна, як відомо, є однією з найбільших аграрних країн світу. Її родючі ґрунти та сприятливий клімат роблять її ідеальним місцем для вирощування різноманітних культур. Для забезпечення стійкого зростання українським аграріям необхідно впроваджувати новітні технології та методи. Одним із ключових напрямків розвитку агросектору є впровадження систем агромоніторингу.

Метою даної кваліфікаційної бакалаврської роботи є підвищення ефективності сільськогосподарських робіт за рахунок оперативного на точного моніторингу стану рослин, ґрунту, а також інших параметрів навколишнього середовища, що забезпечить українських фермерів інформацією для прийняття обґрунтованих рішень.

Системи агромоніторингу можуть допомогти українським фермерам:

- Оптимізувати використання ресурсів: фермери можуть точно знати, коли і де вносити добрива, пестициди та воду, що економить ресурси та мінімізує негативний вплив на довкілля.
- Підвищити врожайність: завчасне виявлення проблем, таких як хвороби, шкідники або дефіцит води, дозволяє фермерам вжити заходів для їх усунення та зберегти врожай.
- Покращити якість продукції: дані про стан посівів можуть бути використані для планування збирання врожаю та контролю якості продукції.
- Зменшити ризики: фермери можуть краще прогнозувати врожаї та планувати свої дії, що може допомогти їм зменшити ризики, пов'язані з мінливістю клімату та іншими факторами.
- Підвищити стійкість: системи агромоніторингу можуть допомогти фермерам краще адаптуватися до мінливих умов клімату та інших викликів.

Незважаючи на значний потенціал, використання систем агромоніторингу для господарств малого та середнього обсягу в Україні є низьким. Це пов'язано з низкою факторів, таких як:

- Висока вартість технологій: системи агромоніторингу можуть бути дорогими, що робить їх недоступними для багатьох фермерів.
- Складність впровадження: впровадження систем агромоніторингу може бути складним процесом, який потребує знань та досвіду.
- Нестача знань про переваги: багато фермерів не знають про переваги систем агромоніторингу або не впевнені, як їх використовувати.

Результатом виконання даної кваліфікаційної роботи є комп’ютерна система агрегування даних, яка допоможе аграріям оптимізувати використання ресурсів, підвищити врожайність та покращити якість продукції за рахунок оперативного та точного моніторингу стану навколишнього середовища.

РОЗДІЛ 1

АНАЛІЗ ПРЕДМЕТНОЇ ГАЛУЗІ ТА ПОСТАНОВКА ЗАДАЧІ

1.1. Загальні відомості з предметної галузі

Технологія «Точного землеробства» почала зароджуватись ще на початку 90-х років, коли технологія GPS стала активно використовуватись у цивільній промисловості. Проте, тільки в останні роки розвиток технологій дозволяє поширювати цю практику по всьому світу та робити її більш доступною для фермерів.

Точне землеробство – це комплексна система аграрного менеджменту, яка має на меті використання сучасних цифрових технологій на різних етапах вирощування сільськогосподарських культур. Такий підхід має багато переваг, у порівняні із традиційними технологіями, серед яких: краще розуміння місцевих типів ґрунтів, аналіз та покращення якості ґрунту, правильний підбір культур для посіву, своєчасна протидія хворобам та бур'янам та передбачення врожайності. Проте, найголовніша перевага яку надає технологія точного землеробства полягає у розумному використанні ресурсів. Керування часом та місцем поливу, та інформація яка саме ділянка сільськогосподарських угідь потребує застосування гербіцидів, фунгіцидів та поживних речовин призводить до значної економії ресурсів та запобіганню негативного впливу на навколишню середу [1].

Для запровадження технології точного землеробства на підприємстві, необхідно налаштувати систему агротехнічного моніторингу, що буде стабільно надавати актуальні дані [2]. До таких даних відносяться:

- Топологія: форма та рельєф земельної ділянки можуть сильно впливати на особливості догляду за посівами для розкриття повного потенціалу поля. Поля або його ділянки із великим нахилом більш схильні до ерозії або деградації ґрунту, а схили можуть призвести до нестачі вологи для

рослин. Окрім цього, натуральні елевації та впадини можуть призвести до недостатньої кількості сонячного світла на певних ділянках поля, що може бути шкідливо для культур. Також особливості схилів можуть впливати на локальний мікроклімат, створюючи зони з різними вітрами, вологістю та навіть температурою;

- Температура: один із найважливіших показників земельної ділянки, який напряму впливає на родючість та пристосованість культур. Знаючи річні коливання температури на своїх сільськогосподарських угіддях, фермери можуть прийняти обґрунтовані рішення стосовно вибору культури для посіву або заздалегідь спланувати зведення додаткових структур таких як теплиці;

- Вологість ґрунту: цей показник також напряму впливає на зростання посівів та їх майбутній врожай. Від вологості ґрунту залежать такі фізіологічні процеси рослин, як фотосинтез, поглинання поживних речовин та води з ґрунту, а також терморегуляція. При недостатній вологості рослини починають в’янути та в них підвищується сприйнятливість до шкідників, у той час, як надмірна вологість призводить до кисневого голодування коренів, через яке рослина перестає поглинати поживні речовини, рости, та може загнити. Окрім цього, важливо пам’ятати, що кожна культура має свій власний показник оптимальної вологості ґрунту. Саме тому фермерам важливо регулярно отримувати актуальні дані цього показника поля впродовж усього аграрного сезону;

- Вологість повітря: має ширший нормальний діапазон для оптимального росту культур (як правило, від 20% до 60%), проте цей показник також потребує постійного моніторингу, адже низька вологість повітря може призвести до в’янення та хвороб, у той час, як надвисока може стати причиною появи грибкових захворювань та шкідників;

- Освітленість: рослини потребують фотосинтезу для синтезування клітинних стінок, зростання та дозрівання плодів. Стабільний моніторинг

показника освітленості у полі дозволяє спланувати заходи з регулювання потреб рослин за допомогою контрольованого освітлення.

Схема на рис.1.1. наочно демонструє, як збираються, передаються, обробляються і використовуються дані в системах точного землеробства для покращення управління сільськогосподарськими процесами. Вона складається з чотирьох основних шарів: прикладний шар, сервісний шар, мережевий шар і сенсорний шар [3].

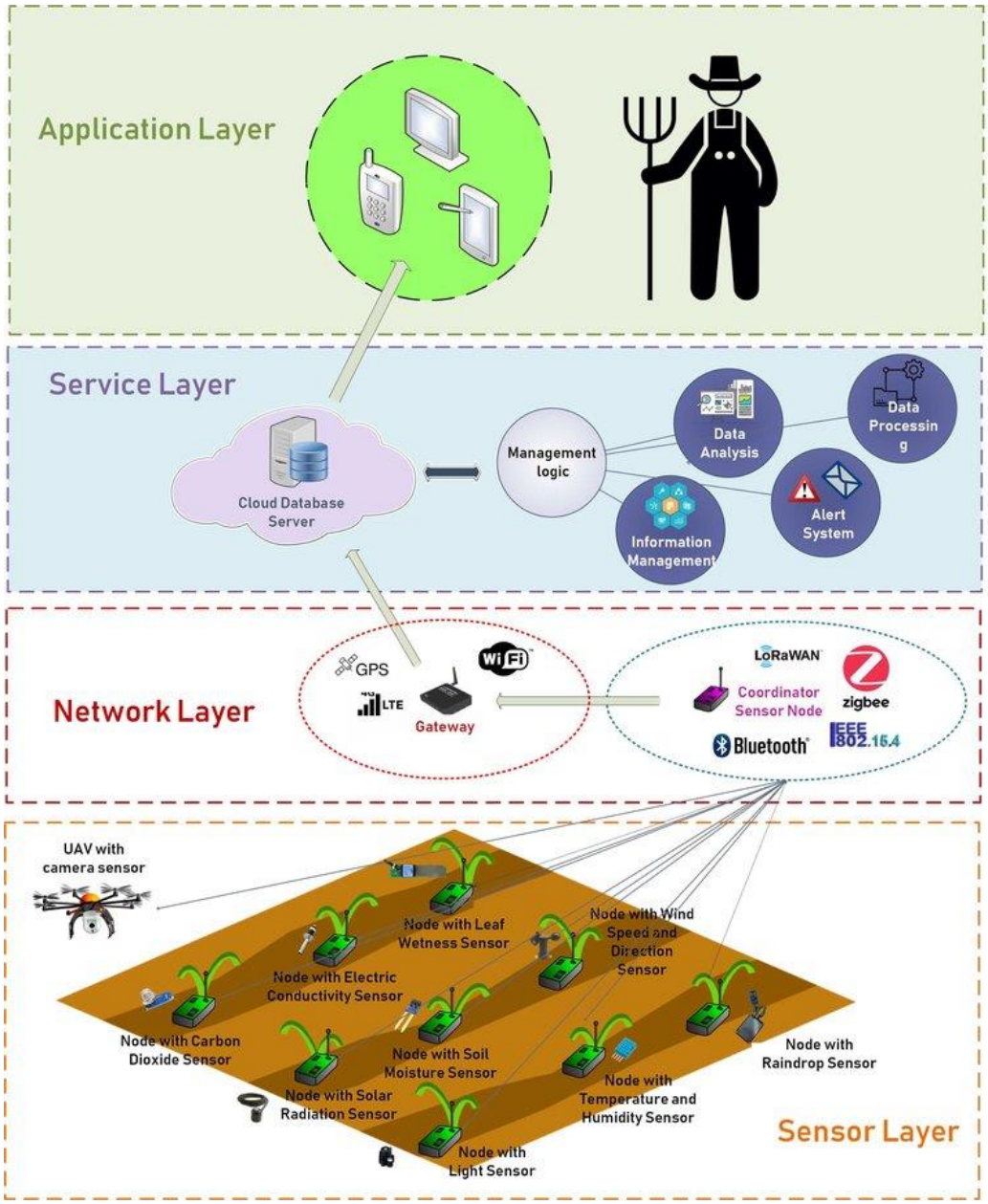


Рис.1.1. Схема багатошарової архітектури системи точного землеробства [3]

Сучасний ринок комп’ютерних технологій пропонує різні апаратні рішення для збору цих даних, розподілених на наступні просторові рівні:

- Наземні: до них відносяться датчики та сенсори, що безпосередньо знаходяться на полі та збирають фізичні показники з земельної ділянки, такі як датчики вологості ґрунту та повітря, температури та освітленості. Також до наземних засобів збирання даних відноситься розумна польова техніка та екіпіровка, обладнана системою GPS та численними датчиками для знаходження проблемних ділянок, що потребують уваги агрономів;
- Повітряні: безпілотні літальні апарати (дрони) можуть бути обладнані спеціальними багатоспектральними камерами з високою роздільною здатністю (до 15 см на піксель), що дозволяють проводити швидкий та ефективний скаутинг та моніторинг полів. Окрім здобування даних, дрони мають можливість контролювати певні показники земельної ділянки, ефективно розпорошуючи необхідні агрохімікати на проблемні ділянки, а також займатися посівом та зрошенням полів;
- Супутникові: використовуються для більш масштабних рівнів моніторингу та аналізу сільськогосподарських угідь, через низьку роздільну здатність вихідного зображення (приблизно 3 м на піксель). Завдяки регулярному потоку даних з супутників, агрономи можуть знаходити певні закономірності у поведінці культур, а також робити прогнози врожайності, розповсюдження хвороб та погодних умов.

На ринку інформаційних технологій наявні компанії, які надають комплексні технічні рішення для агропідприємств. Вони допомагають із закупівлею та налаштуванням техніки, а також надають аналітику, як частина свого сервісу. Нижче приведені приклади компаній, що надають такі послуги:

- John Deere: ця компанія спеціалізується на розробці та продажу сільськогосподарської техніки. Особливістю техніки цього виробника є включення високоточного GPS датчика та інших датчиків до сенсорів, які вбудовують процес агротехнічного моніторингу безпосередньо у польові

роботи. Всі дані автоматично вносяться та можуть бути переглянуті та проаналізовані на організаційній платформі John Deere Operation Center, яка має веб-інтерфейс та мобільні додатки для зручності;

- Climate FieldView: ця компанія надає комплексне рішення із допомоги фермерам оптимізувати свої аграрні процеси та збільшити ефективність полів. Для цього вони надають аналітику та корисні поради своїм клієнтам під час планування посіву, періоду вегетації та збирання врожаю. Дані вони отримують з власних супутникових знімків відносно високої роздільної здатності, а також базуючись на архівних даних клієнта (у разі, якщо вони в нього були). Окрім цього, компанія пропонує клієнту інноваційне апаратне рішення: система FieldView Drive – це девайс, що сумісний із більшістю моделей сівалок, посівних комплексів, обприскувачів, а також комбайнів, та перетворює їх на повноцінні агротехнічні монітори, які можна відстежувати через технологію GPS та які автоматично вносять дані до системи. Надалі, всю аналітику, поради, мапи та дані клієнт та його команда можуть продивлятися у програмному продукті Climate FieldView Cab;
- Intelinair AGMRI: функціональна система агротехнічного моніторингу та аналізу. Клієнти сервісу AGMRI отримують цифрову версію своїх полів, із повною інформацією стосовно топографії місцевості, термального розподілу, індексу вегетації, а також загального аграрного скаутингу. Цифрова версія полів створюється за допомогою сукупності регулярних супутникових знімків, зображень з літаків та дронів. Пізніше дані оброблюються за допомогою штучного інтелекту, що дозволяє прогнозувати потенціальні проблеми та допомагає приймати вірні своєчасні рішення під час аграрного сезону. Сервіс гарантує підтримку та надання корисних аналітичних даних не тільки під час аграрного сезону, а й до його початку та після його завершення. Окрім цього, AGMRI зберігає архівні дані полів користувача для подальшого прогнозування та прийняття рішень, що будуть підтверджені чинними даними.

Важливою особливістю системи Intelinair AGMRI є широка можливість експорту аналітичних даних в інші системи. Наприклад, усі дані про цифрові поля можна експортувати у застосунок John Deere Operations Center, де команді агрономів буде легше координувати дії команди. Також AGMRI за допомогою штучного інтелекту генерує оптимальні маршрути та алгоритми роботи польової техніки, такої як сівалки при посіві або комбайни при зборі врожаю, та імпортувати ці маршрути напряму до програмного забезпечення сумісної машини;

- Culver Aerospace: сервіс з надання послуг агротехнічного моніторингу із використанням безпілотних літальних апаратів (БПЛА). Компанія адаптує основні принципи Точного Землеробства, надаючи своїм клієнтам своїм клієнтам сучасні функціональні дрони, що володіють наступними можливостями:

  - Організація банку землі клієнта, полів та зон посівів, та оцінка ефективності засадження земель;
  - Отримання візуальної оцінки стану посівів, знаходження проблемних ділянок;
  - Детальний аналіз здоров’я рослин із використанням багатоспектральних камер;
  - Планування внесення до полів добрив та інших агрохімікатів;
  - Мінімізація витрат врожаю та ефективне планування бюджету.

У висновку можна сказати що Culver Aerospace повністю виконує основні правила технології Точного Землеробства: використання новітніх технологій для збільшення врожайності та ефективності використання ресурсів.

На підставі проведеного аналізу було встановлено наступні недоліки у наявних системах агротехнічного моніторингу:

- Масштабованість виключно під великі підприємства;

Господарства малого та середнього обсягу не мають достатньо коштів для користування великими аналітичними системами які пропонує сучасний ринок інформаційних технологій, до того ж такі підприємства не зможуть отримати вигоди з такої глибокої та масштабної аналітики, яка пропонується.

- Неактуальність систем в нашій місцевості.

Деякі з наведених платформ недоступні в Україні, або їх алгоритми не враховують місцевих умов, таких як особливості ґрунту, ландшафту, клімату та флори.

1.2. Призначення розробки та галузь застосування

Метою розробки системи агротехнічного моніторингу є надання власникам українських сільськогосподарських підприємств можливості запровадити технологію Точного Землеробства. Розробка системи має на меті автоматизацію збору агротехнічних даних з полів фермерів із подальшою її відправленням на віддалений сервер для візуалізації та аналізу.

Напрями сільського господарства, де використовується система моніторингу стану навколишнього середовища:

- Відстеження стану посівів: своєчасне виявлення та оцінка проблем зі здоров’ям посаджених культур дозволяє агрономам зупинити процес розповсюдження хвороби, щоб запобігти втратам врожаю;
- Точне землеробство: агротехнічний моніторинг допомагає агрономам впровадити практики точного землеробства, які спрямовані на максимізацію продуктивності посівів при мінімізації впливу на навколишнє середовище. Завдяки точному аналізу фермери знатимуть які ділянки полів потребують певних ресурсів, що призводить до більш ефективного їх використання;
- Прийняття рішень на основі наявних даних: маючи можливість збирати та аналізувати як поточні, так і архівні дані, агрономи можуть

приймати обґрунтовані рішення на основі наявних даних, що зазвичай призводить до підвищення врожайності посівів та зниженню загальних втрат.

1.3. Підстава для розробки

В кінці навчання, студент виконує кваліфікаційну роботу (проект). Тема роботи узгоджується з керівником проекту, випускаючою кафедрою.

Підставою для розробки кваліфікаційної роботи на тему «Розробка комп’ютерної системи агрегування даних агромоніторингу» є наказ по Національному технічному університету «Дніпровська політехніка» №469-с від 23.05.2024.

1.4. Постановка завдання

Метою кваліфікаційної роботи є розробка комп’ютерної системи агрегування даних агротехнічного моніторингу з різних джерел, щоб забезпечити українських фермерів інформацією для прийняття обґрунтованих рішень щодо управління своїм господарством.

Основна ідея полягає в створені доступної системи для представників різних сегментів агробізнесу, та в адаптації системи під українські особливості місцевості. Для досягнення мети зробити систему доступною, обираються лише необхідні датчики та сенсори, з яких складається функціональна частина, та запроваджується масштабованість, щоб таку систему було зручно застосувати на господарстві будь-якого розміру. Задля кращої адаптації системи до вимог українських агрономів, планується тісна співпраця із локальними експертами в області землеробства.

1.5. Вимоги до програми або програмного виробу

1.5.1. Вимоги до функціональних характеристик

Для комп’ютерної системи агрегації агротехнічних моніторингових даних встановлюються наступні вимоги до функціональних характеристик:

- Перевірка працездатності усіх датчиків: Система повинна надавати можливість провести тест усіх наявних сенсорів та датчиків на предмет їх працездатності для запобігання обробки та відправлення пошкоджених даних;
- Модульність: Система повинна підтримувати різну конфігурацію датчиків та сенсорів для агротехнічного моніторингу, щоб кожен користувач міг змогу зібрати систему під потреби свого господарства;
- Легкість у підключенні та налаштуванні: Система повинна бути достатньо зрозумілою та простою в користуванні, щоб не потребувати технічного спеціаліста для налаштування та використання;
- Збір агротехнічних даних: Система повинна мати змогу збирати зі своїх датчиків та сенсорів такі дані як:
    - Температура повітря;
    - Вологість ґрунту;
    - Вологість повітря;
    - Кількість опадів;
    - Освітлення.
- Відправлення агротехнічних даних на віддалений сервер: Система повинна мати змогу налаштовувати з’єднання із віддаленим сервером через межу Інтернет, та відправляти дані на хмарне сховище даних.

1.5.2. Вимоги до інформаційної безпеки

Для інформаційної системи, яка має змогу звертатися до мережі Інтернет, та зберігає у внутрішньому сховищі приватні дані, встановлюються наступні

вимоги до інформаційної безпеки:

- Захист локально збережених даних: Для запобігання несанкціонованого доступу до приватних даних, усі дані, що зберігаються у внутрішній пам’яті мікроконтролера, мають бути зашифровані. Окрім цього, сама апаратна складова системи повинна знаходитись під захистом від викрадення або пошкодження;
- Безпека передачі даних мережею: Для запобігання різних видів мережевих атак під час передачі даних мережею необхідно використовувати зашифровані протоколи, наприклад, HTTPS, під час комунікації із віддаленим сервером.

1.5.3. Вимоги до складу та параметрів технічних засобів

До системи, встановлюються наступні вимоги до складу та параметрів технічних засобів:

- Мікроконтролер: Для роботи системи необхідний Wi-Fi модуль NodeMCU на базі чіпа ESP8266;
- Доступ до мережі Інтернет: Для відправлення даних на сервер, системі необхідно мати широкосмуговий доступ до мережі Інтернет;
- Датчик вологості та температури: Для вимірювання вологості повітря та температури системою необхідний датчик вологості та температури ASAIR DHT11;
- Датчик вологості ґрунту: Для вимірювання вологості ґрунту системі необхідний сумісний з платою датчик вологості ґрунту;
- Датчик освітленості: Для вимірювання освітленості навколишньої середи системі необхідний датчик освітленості;
- Залежно від збірки, системі необхідна макетна плата, або заздалегідь спроектований корпус, або проводи-з’єднувачі із необхідними формфакторами.

1.5.4. Вимоги до інформаційної та програмної сумісності

Для системи, що відправляє дані на хмарне сховище, встановлюються наступні інформаційні сумісності:

- Сумісність із хмарним сховищем даних ThingSpeak;
- Сумісність із засобами формування та відправки HTTP запитів;
- Функціонал із синтаксичного розбору відповідей інтернет-пакетів;
- Базові функції забезпечення мережевої безпеки;

На основі проведеного аналізу встановили необхідність розробки приладу із вимірювання показників навколишнього середовища для формування статистичних даних агротехнічного моніторингу і забезпечення українських аграріїв можливістю проводити аналіз для прийняття обґрунтованих рішень у сфері управління своїми земельними ділянками та посівами.

РОЗДІЛ 2
ПРОЕКТУВАННЯ ТА РОЗРОБКА ПРОГРАМНОГО ПРОДУКТУ

2.1. Функціональне призначення програми

Призначенням розробленої інформаційної системи є керування встановленими сенсорами та датчиками, для проведення автоматизованого зчитування та агрегування даних навколишнього середовища, а також завантаження їх на хмарний сервіс IoT платформи ThingSpeak для подальшого аналізу та прийняття рішень.

Основний функціонал включає в себе:

- Перевірку працездатності під’єднаних сенсорів та датчиків;
- Збір даних сенсорів про стан ґрунту та рослин;
- Збір даних сенсорів про умови навколишнього середовища у реальному часі;
- Комунікація із користувачем та вивід інформації за допомогою дисплею;
- Передача даних на сервер хмарного сховища у структурованому форматі;
- Можливість надання віддаленої комунікації із користувачем за допомогою мережі Інтернет.

2.2. Опис застосованих математичних методів

Під час розробки даного програмного продукту використовувались лише базові арифметичні та логічні дії. Різні математичні та логічні функції виконуються компонентами приладу у процесі зчитування та отримання даних для обробки та конвертації інформації у зручні та більш значущі одиниці та

представлення, тож інформаційна система отримує значення у готовому для завантаження вигляді.

2.3. Опис використаних технологій та програмних продуктів

2.3.1. ESP

Espressif Systems – компанія, що займається розробкою та продажом лінійки чіпів та плат розробки низького споживання на ринку Інтернету Речей (IoT). Перший продукт компанії – чіп ESP8266 отримав широку відомість за рахунок своєї доступності, низької ціни, легкості у використанні та потужних характеристик. Компанія Espressif в останні роки розширила лінійку товарів, дозволяючи розробникам обирати чіп та конфігурації плат в залежності від потреб та складності розроблюваного проекту. Для розуміння подальшого вибору моделі, наведемо розбір популярних лінійок чіпів ESP MCU:

- ESP8266: Перший продукт компанії. Чіп має дуже низьку ціну, включає у себе Wi-Fi модуль для виконання простих дій у бездротовій мережі. Мікроконтролер має 17 пінів GPIO (General purpose input-output), працює на тактовій частоті 80-160 МГц та має малий об’єм пам’яті (ROM відсутній, 160KB SRAM, 32KB Cache та всього лише 768B RTC), що унеможливлює його використання на більш масштабних проектах [4];
- ESP32: Наступник серії ESP8266, що враховує обмеження та проблеми останнього. ESP32 є більш потужним, двоядерним чіпом, що працює на частоті 240 МГц, та володіє значно більшим обсягом пам’яті. Окрім покращеного модуля Wi-Fi, ESP32 оснащений модулем Bluetooth Low Energy, що розширює спектр використання чіпу, дозволяючи працювати над задачами у яких необхідне підключення по мережі Bluetooth або менше енергоспоживання [4];
- ESP32-S: Похідна модель від лінійки ESP32, яка фокусується на максимально можливому зменшені енергоспоживання, при цьому зберігаючи

весь функціонал та можливість Wi-Fi та Bluetooth підключення (Зменшений лише обсяг пам’яті) [5];

- ESP32-C Series: Бюджетний варіант чіпу ESP32, має значно нижчу вартість, але пропонує менше обчислювальної сили. Навіть при значно меншому обсязі пам’яті, чіп зберігає можливість під’єднання до мереж, як і представники інших лінійок. Підходить для проектів із меншими вимогами до обчислювальної потужності та при необхідності легшої доступності мікроконтролеру [5].

Врахувавши усі моделі мікроконтролерів від компанії Espressif Systems, було прийняте рішення використовувати саме оригінальний чіп ESP8266 у складі плати розробника NodeMCU. Нижче наведене обґрунтування такого рішення:

- Вартість: Чіп ESP8266 залишається найдоступнішим чіпом у всій лінійці. Використання цієї недорогої моделі значно полегшить процес прототипування та репродукції, у разі виникнення необхідності;
- Легкість у використанні: ESP8266 підтримує розробку у середовищі Arduino IDE та потребує мінімум сторонніх бібліотек, що робить його значно простішим у вивченні та використанні;
- Підтримка спільноти: Через широку популярність даної моделі мікроконтролера, у вільному доступі можна знайти велику кількість ресурсів таких як бібліотеки, форуми та рекомендації стосовно розробки, що значно полегшує та покращує процес створення інформаційної системи;
- Достатня потужність: Не дивлячись на обмеження ESP8266 у пам’яті та обчислювальних можливостях, було проаналізовано, що можливостей мікроконтролера вистачає для підтримання розроблюваного програмного забезпечення.

2.3.2. NodeMCU ESP8266

NodeMCU (Node MicroController Unit) - це плата розробки на базі

описаного у попередньому підрозділі чіпу ESP8266, що завоювала широку популярність у галузі прототипування та розробки Інтернету Речей [6]. Плата NodeMCU спрощує користування мікроконтролером наступними способами:

- Порт USB: Потужний та простий канал зв’язку між системами розробки та чіпом. Дозволяє одночасно заживляти та програмувати мікроконтролер з персонального комп’ютера, а також надає змогу відлагодження програм за допомогою послідовного монітора;
- Сумісні із макетною платою піни: Наявний у платі стандарт розташування пінів дозволяє легко та швидко приєднувати до плати різноманітні компоненти, такі як сенсори, давачі, пристрої виведення інформації та інші. Вигляд плати показано на рисунку 2.1;
- Регуляція напруги: Електрична схема плати забезпечує безперебійне та безпечне живлення плати різними способами;
- Попередньо встановлена прошивка (Firmware): Плати NodeMCU поставляються із встановленою прошивкою, що дозволяє одразу використовувати її у проектах Інтернету Речей та програмувати, використовуючи мову Lua або мову C у середовищі програмування Arduino IDE.

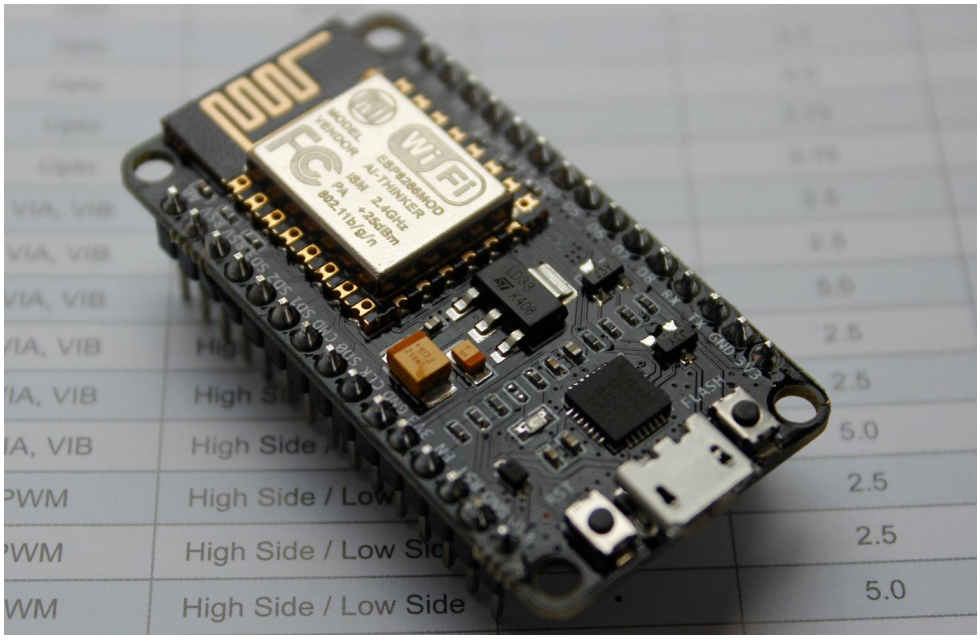


Рис. 2.1. Вигляд плати NodeMCU [7]

Нижче наведений перелік технічних характеристик плати та мікроконтролеру.

Таблиця 2.1

Загальні характеристики NodeMCU V3 ESP8266

Мікроконтролер	ESP8266
Розрядність	32-bit
Тактова частота	160 MHz
Напруга логічних рівнів	3.3 V
Вхідна напруга живлення	4.5-9 V
Кількість загальних портів вводу-виводу (GPIO)	17
Максимальна сила струму (I) пінів вводу-виводу	15 mA
ШІМ порти (PWM pins)	10
АЦП порти (ADC)	1 (read-only)
Розрядність АЦП	10-bit
Flash-пам’ять	4 Mb
EEPROM-пам’ять	Відсутня, симулюється з Flash memory
ОЗУ (RAM)	80 Kb
RTS (Real Time Clock) memory	768 B
Wi-Fi протоколи	802.11 b/g/n із вбудованими протоколами TCP, UDP та IP
Габарити	48x26 мм

Відповідно до зображеної на рисунку 2.2 схеми [8], плата має такі піни:

- ADC0: Пін для проведення аналогового зчитування. У самому чіпі підтримується діапазон зчитуваних значень від 0 до 1В, проте із

використанням плати NodeMCU діапазон розширюється від 0 до 3.3В, завдяки вбудованому розділювачу напруги у платі. АЦП має розрядність 10 біт, тож уявляє подану напругу у вигляді значення від 0 до 1023;

- VIN (Voltage in): Отримання напруги від зовнішнього джерела живлення. Діапазон робочої напруги від 4.5В до 9В. При використанні даного піна необхідно запевнитись у стабільності значення напруги, що передається, адже перепади та перевищення значень діапазону можуть пошкодити плату або чіп

- GND: Пін заземлення, використовується для підключення електричних компонентів до плати, та замикання їх у загальну схему;

- 3.3V: Пін що видає напругу 3.3В від стабілізатора на платі. Ця напруга зазвичай використовується для живлення електронних компонентів із низькою напругою. Багато конкурентів плати NodeMCU використовують стандарт вихідної напруги 5В, що ускладнює складання схеми за рахунок додавання резисторів та адапторів напруги, адже більшість сучасних електронних компонентів у галузі IoT розраховані на робочу напругу 3.3В. Однак, деякі компоненти потребують більшої напруги, у такому разі необхідно використовувати сторонні джерела живлення;

- GPIO (Загальні порти вводу-виводу): Цифрові піни входу\виходу, для яких значення логічного рівня одиниці дорівнює - 3.3В, а нуля – 0 В. Максимальний струм виходу дорівнює 15мА. Усі піни окрім деяких (Необхідно консультуватися із документацією від виробника) мають вбудований підтягуючий резистор, який за замовчуванням замикає пін і встановлює його значення на 0 при відсутності вхідного сигналу;

- PWM (Широко-імпульсна модуляція): До десяти портів (Необхідно консультуватися із документацією) дозволяють виводити значення у вигляді 8-бітного аналогового ШІМ сигналу;

- I2C (Inter-Integrated Cicruit): Протокол комунікації, що використовується для зв’язку мікроконтролеру із периферійними девайсами, використовуючи два канали зв’язку – SDA (Serial Data) та SCL (Serial Clock).

Сам чіп ESP8266 апаратно не підтримує модулі I2C, однак плата NodeMCU додає їх програмну підтримку. Зазвичай у вигляді пінів SDA та SLC використовуються піни GPIO 4 та GPIO 5 відповідно;

- UART (Universal Asynchronous Receiver Transmitter): Модуль UART дозволяє проводити послідовну комунікацію мікроконтролера із іншими девайсами. Модуль використовує два піна: TX 1 (Transmit) та RX 3 (Receive).

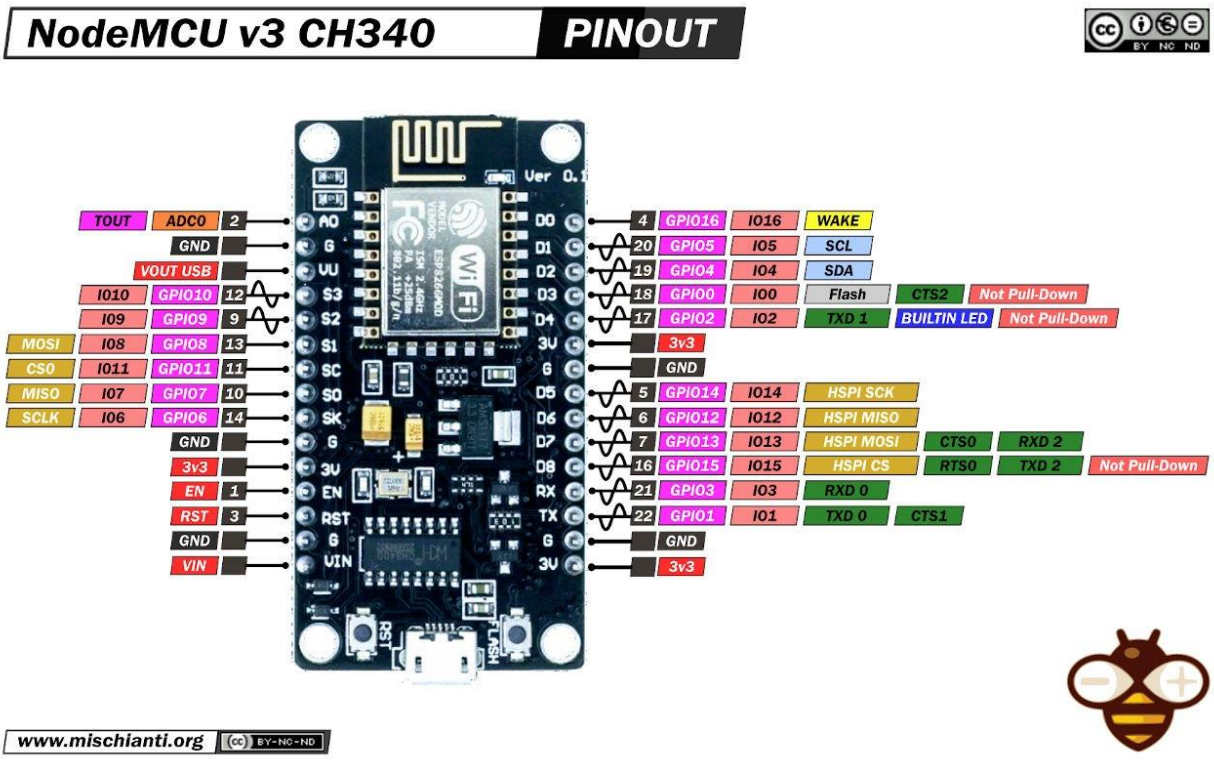


Рис. 2.2. Схема пінів плати NodeMCU V3

2.3.3. Резистивний давач вологості DHT11

Сенсор DHT11 широко використовується у аматорській сфері та у освіті, через легкість у використанні, доступність та компактний розмір. У модулі поєднані можливості давача вологості повітря та температури, що

робить його корисним при створенні систем моніторингу навколишнього середовища.

Принцип роботи даного резистивного давача полягає в тому, що опір чутливого елемента змінюється у відповідності до зміни вологості. У свою чергу, зміна опору проявляє себе у вигляді зміни сили електричного сигналу. Підвищення рівня вологості часто спричиняє посилення електропровідності чутливої плівки, водночас знижуючи питомий опір системи. Такі зміни опору можуть відбуватися в межах від 1 кОм до 100 кОм [9].

На рис. 2.3 наведена типова структура давача. Металевий провідник у формі гребінця друкується на керамічній підкладці, потім його випалюють для формування електродів. Далі поверх електродів осаджується полімерний матеріал для утворення чутливої до вологи плівки.

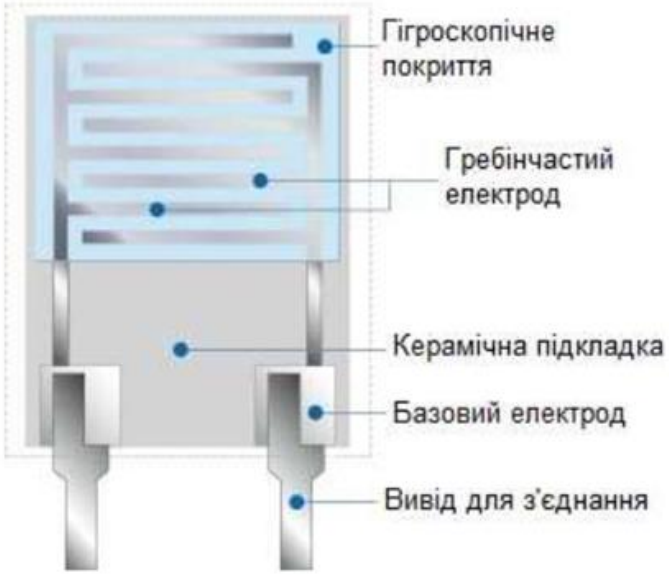


Рис. 2.3. Структура резистивного давача вологості та температури DHT11

Нижче наведений перелік технічних характеристик давача вологості та температури.

Таблиця 2.2

Характеристики сенсору DHT11

Робоча напруга	Від 3В до 5В
Максимальна сила струму	2.5мА
Діапазон вимірювання вологості	20 - 80%
Середня похибка вимірювання вологості	± 5%
Діапазон вимірювання температури	0 – 50 °C
Середня похибка вимірювання температури	± 2 °C
Габарити	15.5мм x 12мм x 5.5мм

2.3.4. GPS модуль GY-NEO6MV2

GY-NEO6MV2 – це універсальний та надійний давач даних по Системі Глобального Розташування (GPS). Модуль використовує високоефективний чіп GPS NEO-6M, що має змогу отримувати інформацію про місцезнаходження із високим ступенем точності та частотою оновлення даних [10] . Ця особливість дуже корисна при створенні дронів або робототехнічних систем, систем відстежування транспорту та моніторингу навколишньої середи. Таким чином, GPS модуль GY-NEO6MV2 є незамінимим при роботі над проектами у галузі Точного Землеробства.

Розглянемо основні особливості модуля:

- Точність та надійність даних: Чіп NEO-6M підтримує до 50 одночасно підключених каналів відстеження, та володіє інноваційною системою отримування сигналу, щоб гарантувати отримання точного сигналу навіть у складних умовах та віддаленій місцевості;
- Висока частота роботи: Модуль оновлює дані із частотою 5 Гц, що дозволяє відстеження місцезнаходження у режимі реального часу;

- Широка сумісність: Комунікація із модулем проходить по стандарту UART, що уможливлює його інтеграцію у платформи які працюють на мікроконтролерах різних виробників, таких як Arduino, Raspberry Pi та ESP;
- Енергонезалежна пам’ять: Чіп має вбудований об’єм енергонезалежної пам’яті, що може бути використана для зберігання налаштувань;
- Компактний дизайн: Полегшує інтеграцію модулю у схеми із обмеженим вільним простором;
- Зовнішня антена: Модуль потребує підключення до зовнішньої антени для отримання сигналу GPS, зазвичай вона йде у комплекті із модулем. Модуль у комплекті із антеною зображений на рис. 2.4.

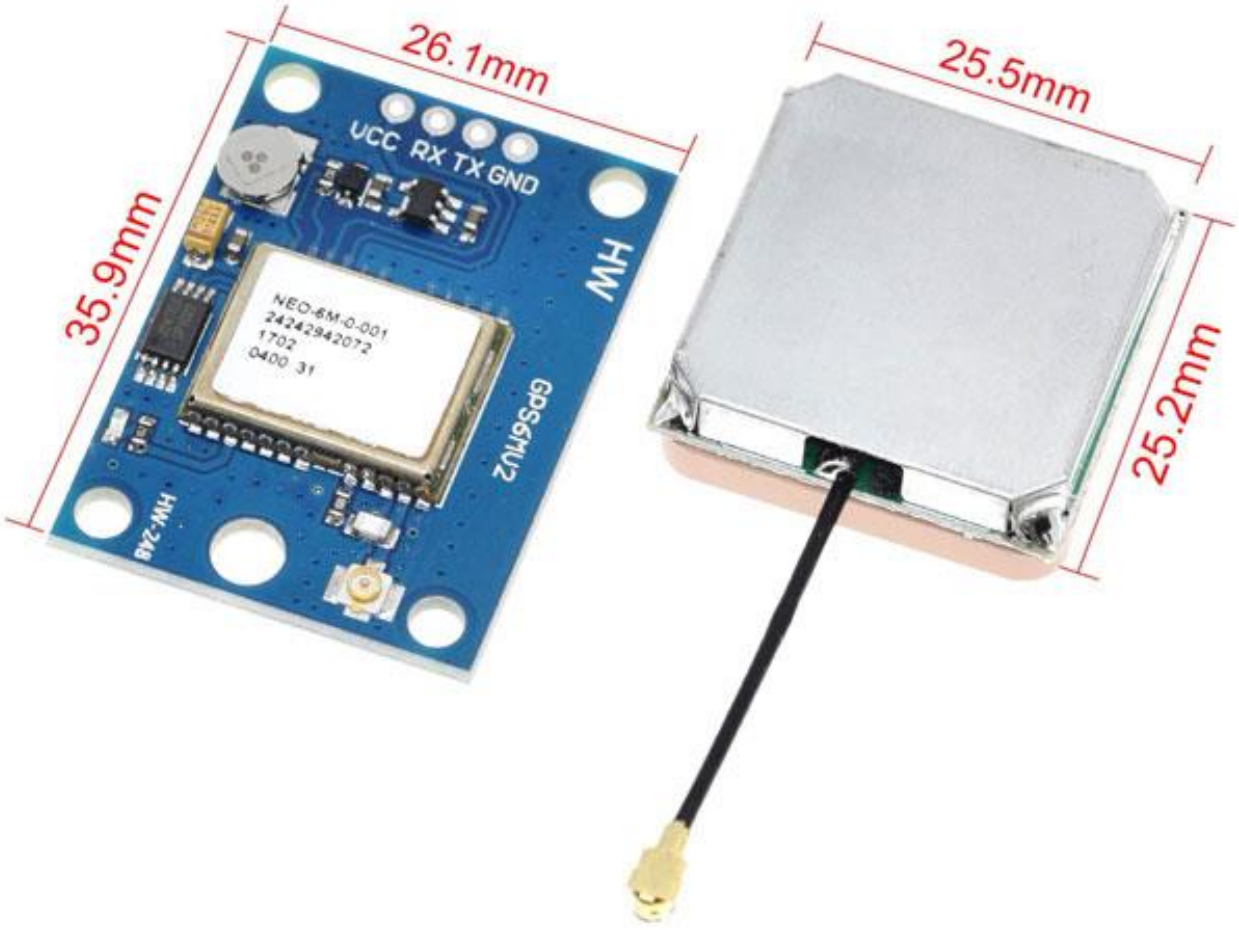


Рис.2.4. Вигляд модуля разом із зовнішньою антеною GPS

Нижче наведений перелік технічних характеристик GPS модуля.

Таблиця 2.3

Характеристики модуля GY-NEO6MV2

Чіп	U-blox NEO-6M GPS
Робоча напруга	3.3В – 5В
Споживання електроенергії	50мА – 80мА
Точність даних GPS – горизонтальні	До 2.5 метрів
Точність даних GPS – вертикальні	До 5 метрів
Частота оновлення	5 Гц

2.3.5. Сенсор GUVA-S12SD

GUVA-S12SD є аналоговим сенсором ультрафіолетового випромінювання. Його особливістю є відсутність необхідності комунікації через інтерфейси типу I2C. Замість цього, сенсор самостійно генерує струм відповідний до інтенсивності отриманого UV індексу. Сенсор дуже простий у впроваджені у схему та володіє компактним дизайном. Не дивлячись на обмежений діапазон вимірювання ультрафіолетового випромінювання, сенсор підходить для проектів у сфері Точного Землеробства, та може надати фермерам та аграріям корисні дані. Сенсор зображений на рисунку 2.5.

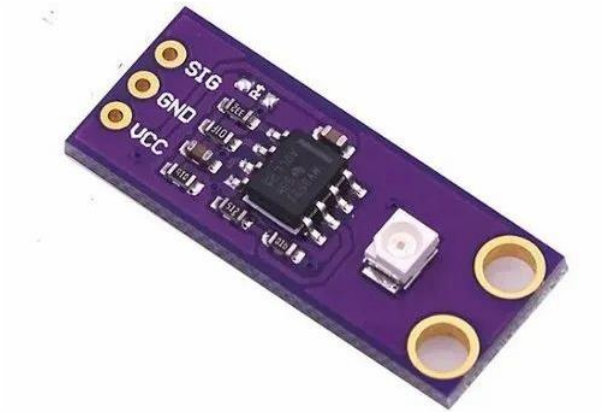


Рис. 2.5. Сенсор GUVA-S12SD

Нижче наведений перелік технічних характеристик сенсора ультрафіолетового випромінювання [11].

Таблиця 2.4

Характеристики сенсора GUVA-S12SD

Діапазон робочої напруги	Від 2.5В до 5В
Споживання струму	5мА
Діапазон сприйняття UV хвиль	Від 240нм до 370нм
Вихідна напруга	Від 0В до 1В
Час відгуку	0.5 секунд
Точність вимірювання	± 1 UV-індекс

2.3.6. OLED SSD1306

За умови польової роботи приладу, доцільно виводити частину даних для локального користувача (за необхідності). Для цього можна використовувати OLED (Organic Light-Emitting Diode) дисплей. Дисплеї цього типу відомі низьким споживанням енергії, високою контрастністю та доступною ціною. Керувати екраном дозволяє вбудована схема SSD1306 за допомогою сторонніх бібліотек. Не дивлячись на обмеження у вигляді низької роздільної здатності екрану, його достатньо для виводу базової інформації, необхідної для користувача [12].

Приклад виведення інформації дисплеєм зображений на рис. 2.6.

Рис. 2.6. 1.3-дюймовий I2C OLED SSD1306 екран із роздільною здатністю 128x64

Нижче наведений перелік технічних характеристик дисплея.

Таблиця 2.5

Характеристики модуля дисплея OLED SSD1306

Діагональ екрану	0.96”
Розмір екрану	22 мм x 11 мм
Роздільна здатність	128 x 64 точок
Діапазон робочої напруги	Від 3В до 5В
Габарити модуля	27 мм x 27 мм x 3.5 мм

2.3.7. Давач вологості ґрунту DFRobot Gravity SEN0193

DFRobot Gravity – це аналоговий ємнісний давач вологості ґрунту. Він зручний у використанні, надійний та володіє значною точністю даних, що повертаються.

Більшість запропонованих на ринку давачів вологості ґрунту є опірними. Принципом роботи таких сенсорів є вплив вологості ґрунту на його опір, який вимірюється двома електродами, що занурені у землю. Через знаходження у землі, такі давачі схильні до корозії та деградації, втрати точності вимірювання та функціоналу із часом. На відміну від цього типу сенсорів, DFRobot Gravity є ємнісним, та використовує зміну показника електричної ємності ґрунту в залежності від його вологості [13]. Через відсутність необхідності вимірювання опору, електроди покриті антикорозійним шаром (Рис. 2.7), що дозволяє цьому виду давачів запобігати деградації чутливих елементів та подовжує строк служби компонента.

Рис. 2.7. Вигляд давача DFRobot Gravity

Нижче наведений перелік технічних характеристик давача вологості ґрунту.

Таблиця 2.6

Характеристики давача DFRobot Gravity

Діапазон робочої напруги	3.3В – 5.5В
Діапазон вихідної напруги	0В – 3.0В

Продовження таблиці 2.6

Робоча сила струму	5мА
Інтерфейс підключення	PH2.0-3P
Габарити давача	98 мм х 23 мм
Вага	15 г

2.3.8. Бібліотеки

2.3.8.1. Adafruit_GFX та Adafruit_SSD1306

Компанія Adafruit Industries надає велику кількість бібліотек із відкритим вихідним кодом, які спрощують комунікацію мікроконтролеру із електронними компонентами.

Бібліотека Adafruit_GFX дозволяє легко ініціалізувати взаємодію програми із компонентом через інтерфейси I2C та SPI, та надає функції виводу графічної інформації на дисплей [14]:

- void drawPixel();

- void drawLine();

- void drawRect();

- void fillRect();

- void drawCircle();

- void fillCircle();

- void drawTriangle();

- void fillTriangle().

Також бібліотека надає функції рендерингу тексту:

- void setCursor();

- void setTextColor();

- void setTextSize();

- void setTextWrap().

Окрім цього, Adafruit_GFX дозволяє працювати із зображеннями розширення BMP. Для стабільної та передбачуваної роботи бібліотеки, потребується також бібліотека-компаньйон, розроблена спеціально під певну модель дисплею. У нашому випадку, це бібліотека Adafruit_SSD1306, що містить необхідні конфігурації та значення для роботи із I2C OLED SSD1306 дисплеєм.

2.3.8.2. Wire

Вбудована бібліотека, яка створює зв'язок між мікроконтролером та I2C, забезпечуючи отримання та надсилання інформації портами SDA та SCL [15].

2.3.8.3. SimpleDHT

Спрощує процес налаштування сенсорів DHT11 та DHT22 та зчитування даних з них. Методи цієї бібліотеки ініціалізують давач DHT у програмі, оброблює помилки зчитування та надає доступ до результатів зчитування [16].

2.3.8.4. ESP8266WiFi та ThingSpeak

Через ці бібліотеки пристрій надсилає дані до хмарного середовища ThingSpeak. ESP8266WiFi є вбудованою у прошивку мікропроцесору, проводить сканування на наявність доступних WiFi мереж, забезпечує під'єднання до них, а також може налаштувати ESP8266 як точку доступу бездротової мережі, дозволяючи іншим девайсам під'єднуватись до нього.

Бібліотека ThingSpeak, розроблена компанією MathWorks, тривіалізує відправку HTTP запити до серверів ThingSpeak для зчитування та запису

даних. Вона самостійно форматує дані та формує запит до каналу на основі чисельних даних.

2.3.8.5. TinyGPS++ та SoftwareSerial

Бібліотека TinyGPS++ використовується разом із модулем GY-NEO6MV2, та розроблена для розбору даних, отриманих від NMEA (Національної асоціації морської електроніки) у результаті GPS запитів [17].

Однак, GPS модуль GY-NEO6MV2 обмінюється даними із мікроконтролером шляхом Універсального Асинхронного Приймача/Передача (UART), під час тестування розроблюваної схеми послідовний монітор використовується персональним комп'ютером. Задля уникнення конфліктів, використовується бібліотека SoftwareSerial, яка програмно емулює послідовні канал передачі даних на основі цифрових пінів плати.

2.4. Опис структури системи та алгоритмів її функціонування

Узагальнена блок-схема роботи системи зображена на рис.2.8.

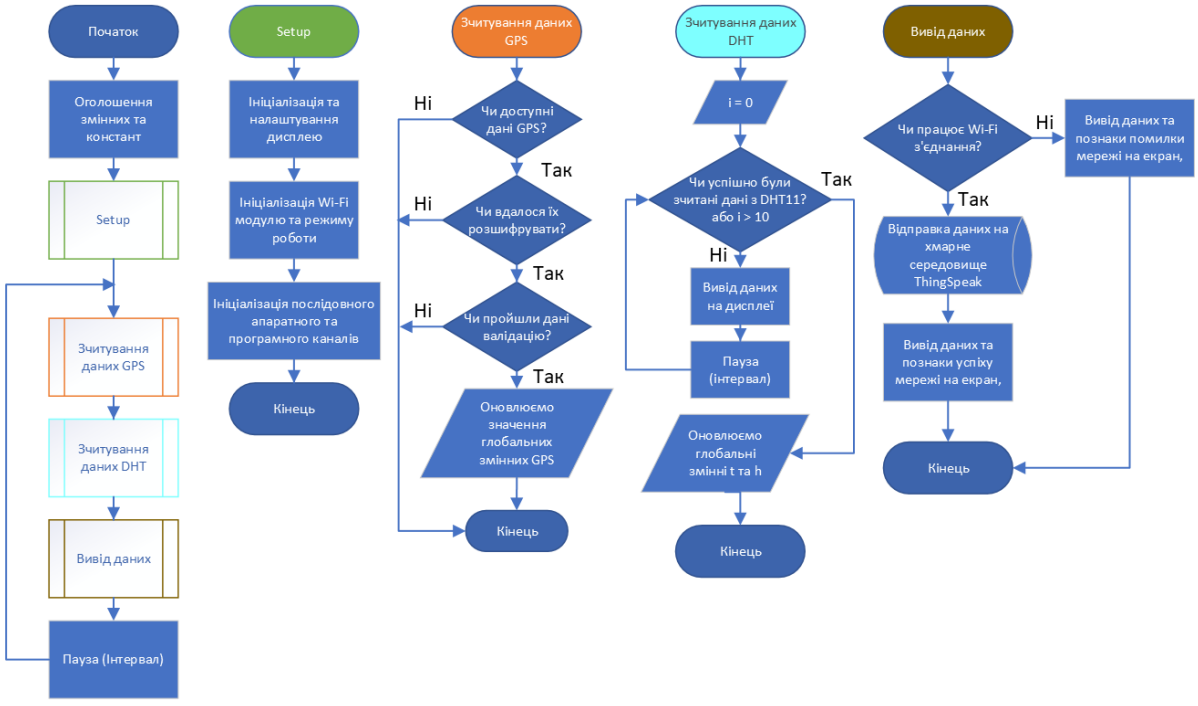


Рис. 2.8. Блок-схема роботи системи

2.5. Обґрунтування та організація вхідних та вихідних даних програми

Вхідними даними для розробленої системи є дані отримані із сенсорів, тобто показники навколишнього середовища (температура повітря, вологість повітря, вологість ґрунту).

Вихідними даними для розробленої системи є дані, що виводяться на екран локальному користувачу (рис.2.9) та графіки, що передались на ThingSpeak (рис.2.10) , які відображають динаміку змін показників за певний проміжок часу.

Рис. 2.9. Виведені дані на екран

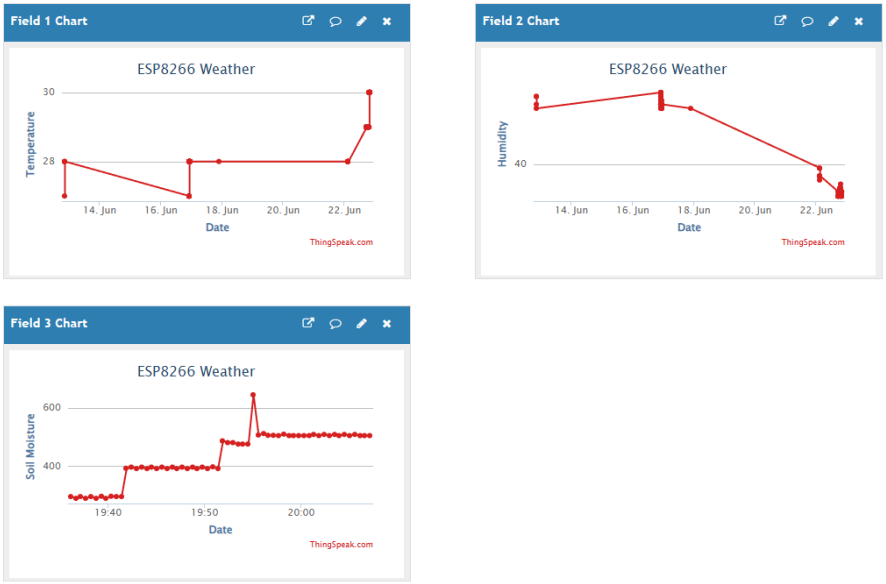


Рис. 2.10. Візуалізовані дані на платформі ThingSpeak

2.6. Опис розробленої системи

2.6.1. Використані технічні засоби

Для роботи розробленої системи достатньо мінімальних комп'ютерних ресурсів. Для використання необхідно мати: персональний комп’ютер, мишу та клавіатуру.

2.6.2. Використані програмні засоби

2.6.2.1. Arduino IDE

Arduino IDE (Integrated Development Environment) – це зручне програмне забезпечення, призначене для кодування, компіляції та завантаження програм на Arduino-сумісні плати, а також на плати інших виробників. Інтегроване середовище розробки характеризується складною функціональністю, включаючи редагування і компіляцію вихідного коду, створення програмного забезпечення, створення бази даних тощо. Це програмне забезпечення дає змогу комп’ютеру взаємодіяти з мікроконтролером для передачі даних та прошивки коду.

Програми, які створюються в Arduino IDE, називаються скетчами. Створення скетчів відбувається в текстовому редакторі, а збереження здійснюється у файлах з розширенням .ino [18]. Вбудований текстовий редактор оснащений стандартними інструментами для роботи з текстом: копіювання, вставка, пошук та заміна. У спеціальному вікні повідомлень користувачі можуть отримувати інформацію про події та помилки, пов'язані з написанням та експортом коду. Внизу праворуч вікна програми можна побачити інформацію про підключену плату: її модель та номер порту.

Інтерфейс інтегрованого середовища розробки Arduino має такі основні елементи: головне меню, панель інструментів, текстовий редактор, панель інструментів імпорту залежностей, консоль і область повідомлень. На рис. 2.11 зображено зовнішній вигляд програми:

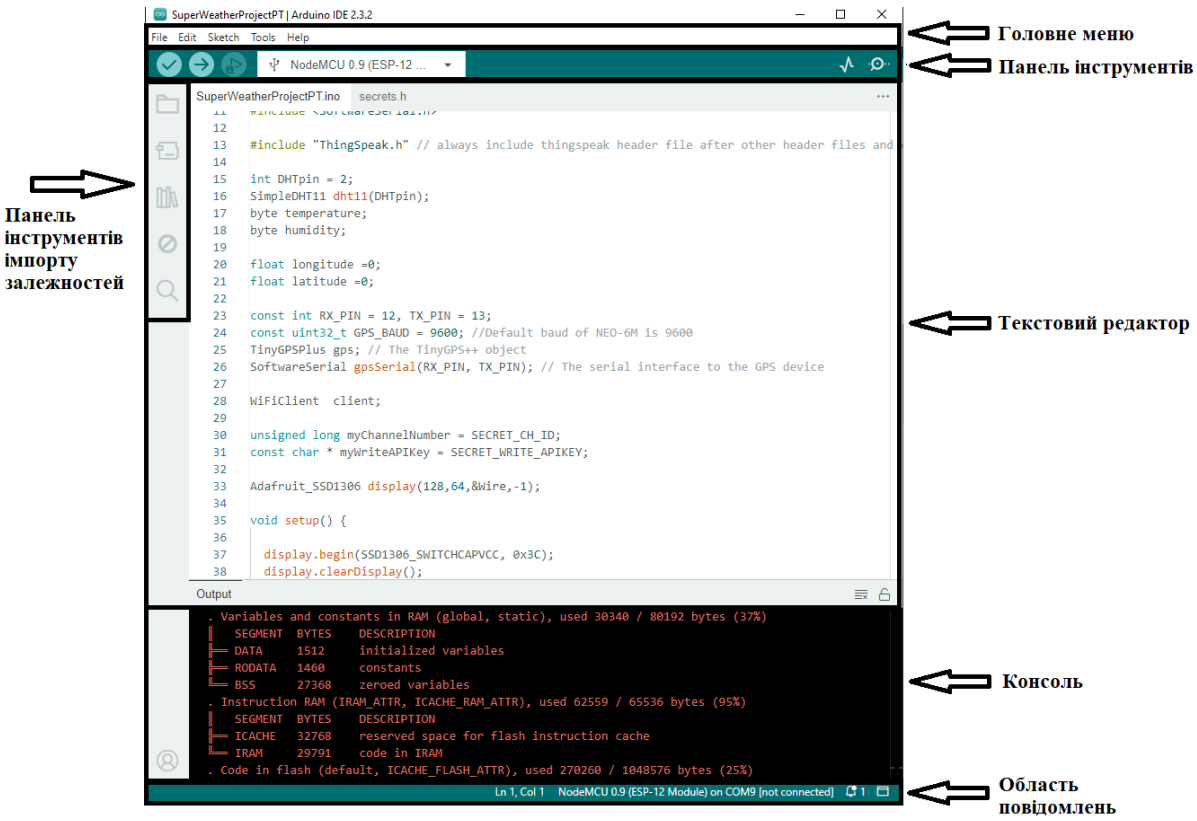


Рис. 2.11. Інтерфейс програми Arduino IDE

Панель інструментів містить шість кнопок (рис. 2.12) [19]:

Рис. 2.12. Панель інструментів

- Компіляція (Verify): інструмент призначений для перевірки синтаксичної правильності написаного коду, у разі виявлення помилок синтаксису, вони відображаються в області повідомлень;

- Завантаження (Upload): інструмент компілює написаний код і завантажує її на підключений мікроконтролер;

- Відлагодження (Debugger): інструмент, за допомогою якого можна встановлювати точки зупинки, покроково виконувати код, переглядати та змінювати значення змінних, а також отримувати доступ до стану регістрів і пам’яті мікроконтролера;

- Вибір моделі та порту (Select board & port): інструмент, що дозволяє обрати модель мікроконтролера та відповідний COM-порт для програмування;

- Побудування графіків (Open serial plotter): відкриває інструмент для візуалізації даних, що надходять через серійний порт, у вигляді графіків;

- Монітор послідовного порту (Open serial monitor): відкриває інструмент для перегляду і відправлення даних через серійний порт.

Важливою частиною розробки програмного забезпечення мікроконтролерів є підбір правильних бібліотек та прошивок плат. Панель інструментів імпорту залежностей має 5 кнопок (рис. 2.13):

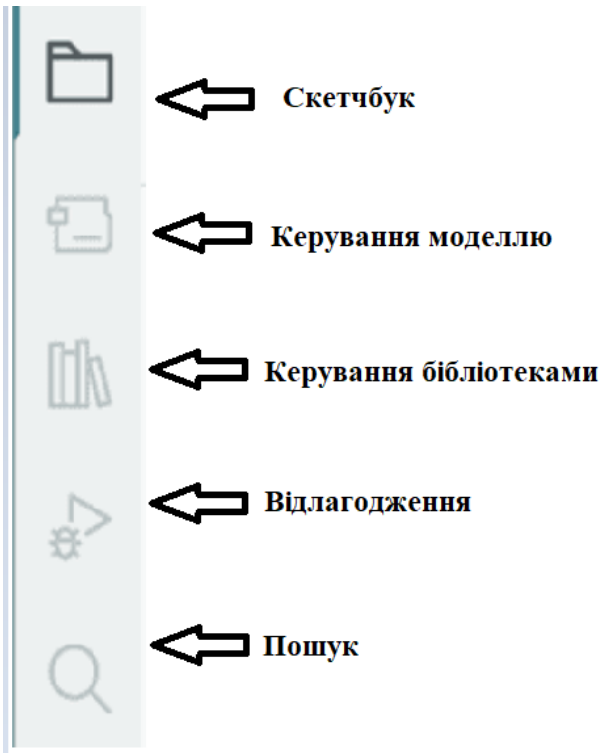


Рис. 2.13. Панель інструментів імпорту залежностей

- Скетчбук (Sketchbook): це місце, де зберігаються файли з кодом. Скетчі Arduino зберігаються з розширенням .ino і обов'язково повинні знаходитись в окремій папці з такою ж самою назвою;

- Керування моделлю (Board manager): за допомогою цього інструменту можна переглядати та встановлювати пакети плат. Пакет плати містить "інструкції" для компіляції коду для плат, що входять до цього пакета;

- Керування бібліотеками (Library manager): за допомогою цього інструменту можна переглядати та встановлювати бібліотеки. Бібліотеки розширюють функціональність програм, надаючи їм додаткові можливості для роботи з апаратними засобами, обробки даних та іншого;

- Відлагодження (Debugger): інструмент, за допомогою якого можна встановлювати точки зупинки, покроково виконувати код, переглядати та змінювати значення змінних, а також отримувати доступ до стану регістрів і пам’яті мікроконтролера;

- Пошук (Search): інструмент дозволяє шукати текстові рядки або фрагменти коду у відкритому файлі.

Загалом Інтерфейс середовища розробки відрізняється простотою та зручністю використання, що робить його доступним для користувачів будь-якого рівня підготовки.

2.6.2.2. ThingSpeak

Для відображення результатів моніторингу стану навколишнього середовища було обрано популярну хмарну IoT платформу ThingSpeak.

ThingSpeak – це безкоштовна платформа, яка дозволяє збирати, аналізувати та візуалізувати дані з датчиків та інших пристроїв IoT у режимі реального часу [20].

ThingSpeak підтримує різні види HTTP запитів: запис, зчитування значень поля, зчитування значень усіх полів, зчитування оновлених значень та

інші. Деякі аргументи запиту коригуються користувачем перед відправкою (ідентифікатор потрібно поля, кількість записів, що повернуться).

В рамках даного проекту був створений канал "ESP8266 Weather" який має два інформаційні поля: "Temperature", "Humidity" та "Soil Moisture" (рис.2.14).

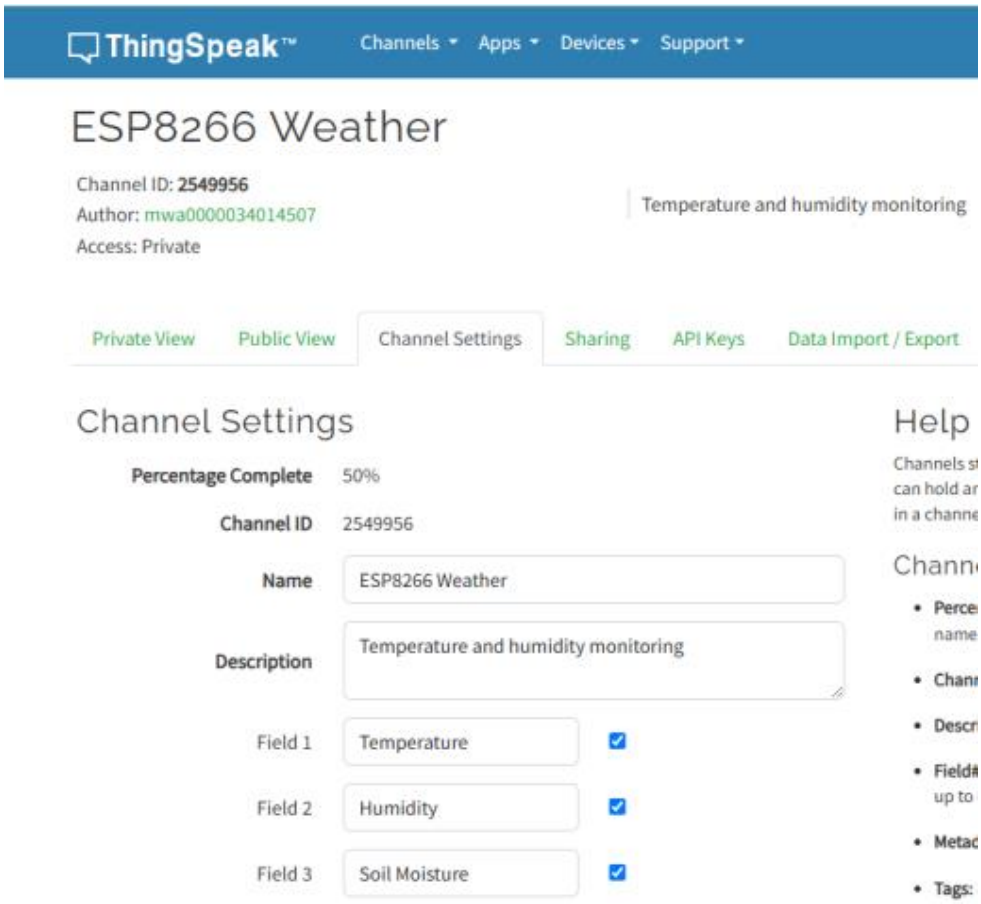


Рис. 2.14. Канал в ThingSpeak для проектованої системи

Для того, щоб передавати дані на канал ThingSpeak, необхідно отримати його API ключ (рис.2.15, 2.16).

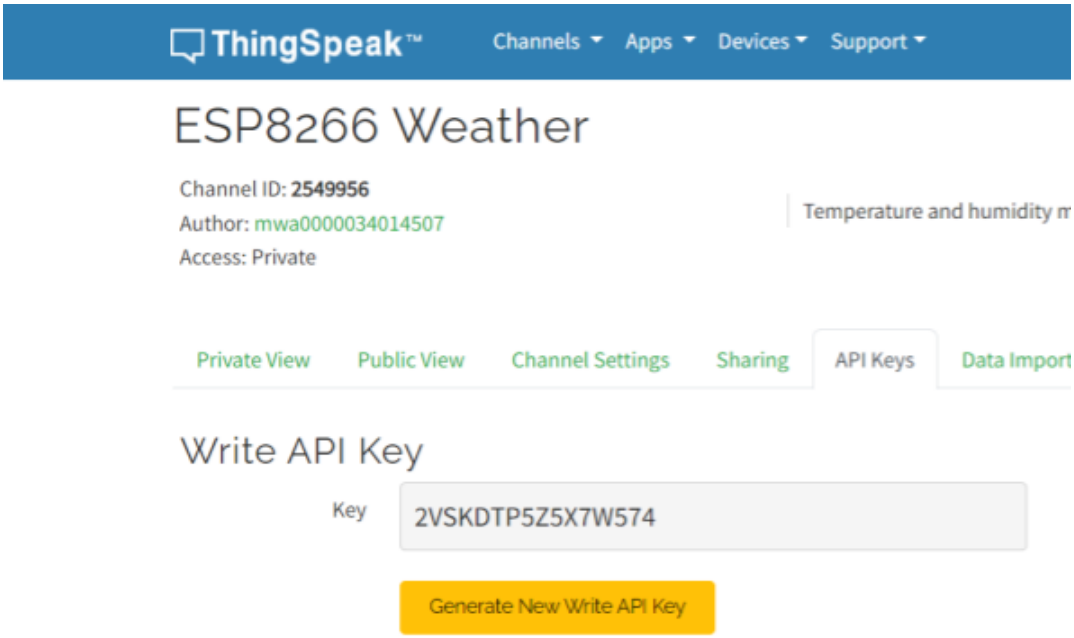


Рис. 2.15. API ключ запису

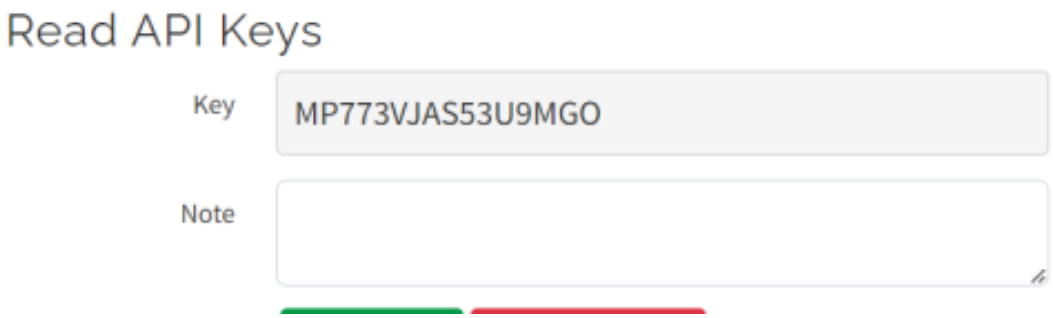


Рис. 2.16. API ключ зчитування

2.6.2.3. Proteus

Proteus – це комплексне програмне забезпечення, що використовується для моделювання, проектування та аналізу електронних схем [21].

При виконанні кваліфікаційної роботи було використано програму Proteus для моделювання електронної схеми приладу, щоб запобігти помилкам при з'єднанні, виходу із ладу певного компоненту та для планування використання пінів перед написанням програми. На рис. 2.17 зображено вигляд побудованої схеми.

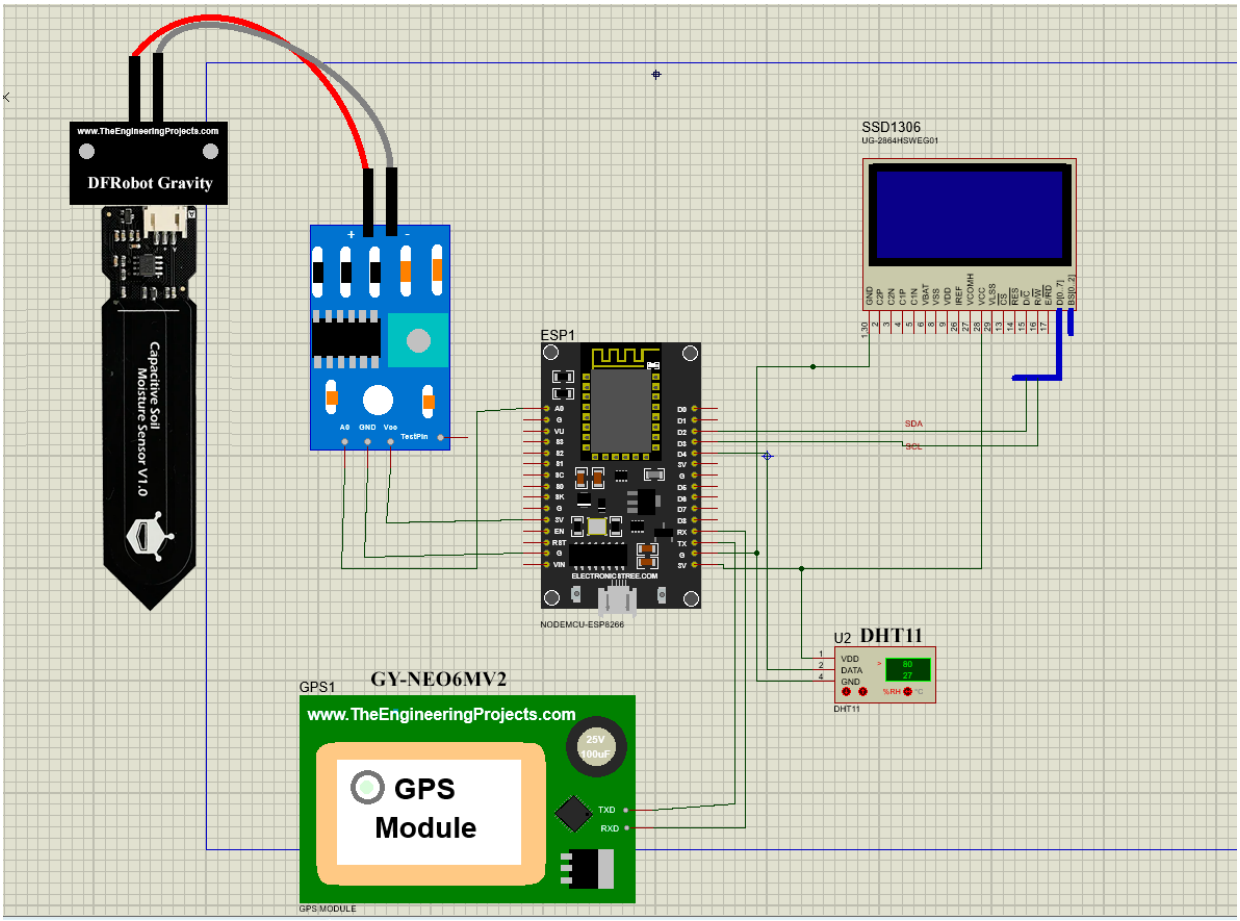


Рис. 2.17. Побудована схема

2.6.3. Виклик та завантаження програми

Для завантаження та виклику комп’ютерної системи необхідно попередньо налаштувати програмний та апаратний компоненти. Для налаштування програмного компоненту, необхідно під'єднати плату розробки до Персонального Комп'ютера із встановленим інтегрованим середовищем розробки Arduino IDE та прошити скетч у пам'ять пристрою. Налаштування апаратного компоненту полягає у зборі електронної схеми шляхом під'єднання відповідних пінів плати до компонентів а також підключення зовнішнього джерела живлення.

2.6.4. Опис інтерфейсу користувача

Графічний інтерфейс не використовується в цій системі, адже вона орієнтована на автоматичний збір даних про навколишнє середовище. Відсутність інтерфейсу користувача в цій системі компенсується можливістю інтеграції з іншими програмними комплексами або API для доступу до даних та керування системою.

РОЗДІЛ 3
ЕКОНОМІЧНИЙ РОЗДІЛ

3.1 Розрахунок трудомісткості та вартості розробки програмного продукту

Початкові дані:

1. Передбачуване число операторів програми: 700;
2. Коефіцієнт складності програми: 1,5;
3. Коефіцієнт корекції програми в ході розробки: 0,5;
4. Годинна заробітна плата розробника – 292,76 грн/год.

За статистикою платформи найму “DOU”, на початок 2023 року медіана заробітної плати на посаді Embedded Software Developer працівника рівня Junior складає 1225$ на місяць [1]. У середньому, графік роботи розробника складається з 21 робочого дня по 8 годин. Отримуємо погодинну ставку, як 1225$/(21 день * 8 годин) = 7,29$ / година. Станом на початок червня 2024 року, за офіційним курсом валют НБУ, один долар США дорівнює 40,15 грн [2]. Отже, розрахована погодинна ставка еквівалентна 292,76 грн. на годину.

5. Коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі: 1,2;
6. Коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності: 1.1;
7. Вартість машино-години ЕОМ: 5,5 грн/год;

Для розробки системи агротехнічного моніторингу використовувався персональний комп’ютер із широкосмуговим інтернет-підключенням. Вартість машино-години ЕОМ складається з вартості амортизації та обслуговування обладнання, вартості електроенергії та інтернет-підключення. Виходячи з характеристик ПК та монітору, дізнаємось, що приблизне споживання електроенергії за годину становить 300 Вт за годину. Згідно із тарифним планом постачальника електроенергії Yasno, 1 кВт/год коштує 4,32

грн [3]. Таким чином рахуємо вартість машино-години, як 0,3 кВт/год * 4,32 грн/кВт = 1,3 грн/год. Щомісячна плата за підключення до інтернету від провайдера Fregat становить 225 грн. Отже, вартість робочої години інтернету = 225 грн / 168 год = 1,34 грн/год.

Також, необхідно врахувати використання мікроконтролерів та сенсорів на етапі розробки та тестування, які піддаються зношуванню та потребують своєчасної заміни та обслуговування. Електроенергією, що споживається мікропроцесором та сенсорами нехтуємо, через її незначну кількість.

Загальну вартість машино-години ЕОМ, враховуючи вартість амортизації та обслуговування робочої машини та додаткових компонентів, встановимо, як 5,5 грн/год.

Оцінка трудомісткості розробки програмного забезпечення ускладняється через творчий характер роботи програмістів. Для цього існує ряд моделей з різним ступенем точності.

Трудомісткість розробки ПЗ можна розрахувати за формулою:

$$t = t_o + t_u + t_a + t_n + t_{отл} + t_д \text{, людино-годин,} \qquad (3.1)$$

де t_o- витрати праці на підготовку й опис поставленої задачі (приймається 50 людино-годин);

t_u - витрати праці на дослідження алгоритму рішення задачі;

t_a- витрати праці на розробку блок-схеми алгоритму;

t_n-витрати праці на програмування по готовій блок-схемі;

$t_{отл}$-витрати праці на налагодження програми на ЕОМ;

$t_д$ - витрати праці на підготовку документації.

Складові витрати праці визначаються через умовне число операторів у програмному забезпеченні, яке розробляється.

Умовне число операторів (підпрограм):

$$Q = q \cdot C \cdot (1 + p) \qquad (3.2)$$

де q - передбачуване число операторів (700);

C - коефіцієнт складності програми (1,5);

p - коефіцієнт корекції програми в ході її розробки (0,5).

Отже, умовне число операторів в програмі має значення:

$$Q = 700 \cdot 1{,}5 \cdot (1 + 0{,}5) = 1575$$

Витрати праці на вивчення опису задачі t_u визначається з урахуванням уточнення опису і кваліфікації програміста:

$$t_u = \frac{Q \cdot B}{(75..85) \cdot k} \text{ , людино-годин} \qquad (3.3)$$

де B - коефіцієнт збільшення витрат праці внаслідок недостатнього опису задачі, враховуючи роботу із сторонніми джерелами даних він становить 1.2;

k - коефіцієнт кваліфікації програміста, обумовлений від стажу роботи з даної спеціальності. Оцінюється в значення 1.1.

Підставивши значення у формулу (3.3), отримуємо результат:

$$t_u = \frac{1575 \cdot 1{,}2}{75 \cdot 1} = 22{,}9 \text{ людино-годин}$$

Витрати праці на розробку алгоритму рішення задачі визначаються за формулою:

$$t_a = \frac{Q}{(20 \ldots 25) \cdot k} \text{ , людино-годин} \qquad (3.4)$$

де Q – умовне число операторів програми (1575);

k – коефіцієнт кваліфікації програміста (1.1).

Підставивши відповідні значення в формулу (3.4), отримуємо:

$$t_a = \frac{1575}{21 \cdot 1,1} = 68,18 \text{ людино-годин.}$$

Витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{Q}{(20 \ldots 25) \cdot k} \text{ , людино-годин} \qquad (3.5)$$

Підставимо значення у формулу (3.5) та отримаємо витрати на складання програми по готовій блок-схемі:

$$t_n = \frac{1575}{20 \cdot 1,1} = 71,59 \text{ людино-годин.}$$

Витрати праці на налагодження програми на ЕОМ:

- за умови автономного налагодження одного завдання:

$$t_{отл} = \frac{Q}{(4..5) \cdot k} \text{ , людино-годин} \qquad (3.6)$$

$$t_{отл} = \frac{1575}{4 \cdot 1,1} = 357,95 \text{ людино-годин.}$$

- за умови комплексного налагодження завдання:

$$t_{отл}^{k} = 1{,}5 \cdot t_{отл} \text{ , людино-годин} \quad (3.7)$$

$t^{k}_{отл}$= 1,5 · 357,95 = 536,93 людино-годин.

Витрати праці на підготовку документації визначаються за формулою:

$$t_{д} = t_{др} + t_{до} \text{ , людино-годин} \quad (3.8)$$

де $t_{др}$-трудомісткість підготовки матеріалів і рукопису:

$$t_{др} = \frac{Q}{(15..20) \cdot k} \text{ , людино-годин} \quad (3.9)$$

$t_{до}$ - трудомісткість редагування, печатки й оформлення документації:

$$t_{до} = 0{,}75 \cdot t_{др} \text{ , людино-годин.} \quad (3.10)$$

Підставляючи відповідні значення, отримаємо:

$t_{др}$ = = $\frac{1575}{15 \cdot 1{,}1}$ = 95,45 людино-годин.

$t_{до}$ = 0,75 · 174,01 = 71,59 людино-годин.

$t_{д}$ = 95,45 + 71,59 = 167,04 людино-годин.

Повертаючись до формули (3.1), отримаємо повну оцінку трудомісткості розробки програмного забезпечення:

t = 50 + 22,9 + 68,18 + 71,59 + 357,95 + 167,04 = 737,66 людино-години.

3.2. Розрахунок витрат на створення програмного забезпечення

Витрати на створення ПЗ $К_{ПО}$ включають витрати на заробітну плату виконавця програми $З_{ЗП}$ і витрат машинного часу, необхідного на налагодження програми на ЕОМ:

$$К_{ПО} = З_{ЗП} + З_{МВ}, \text{ грн} \quad (3.11)$$

Заробітна плата виконавців $З_{ЗП}$ визначається за формулою:

$$З_{ЗП} = t \cdot С_{ПР}, \text{ грн} \quad (3.12)$$

де: t - загальна трудомісткість, людино-годин;

$С_{ПР}$ - середня годинна заробітна плата програміста, грн/година

З урахуванням того, що середня годинна зарплата програміста становить 292,76 грн / год, отримуємо:

$$З_{ЗП} = 737{,}66 \cdot 292{,}76 = 215\,957{,}34 \text{ грн.}$$

Вартість машинного часу, необхідного для налагодження програми на ЕОМ, визначається за формулою:

$$З_{мв} = t_{отл} \cdot С_{мч}, \text{ грн,} \quad (3.13)$$

де $t_{отл}$ - трудомісткість налагодження програми на ЕОМ, год;

$С_{мч}$ - вартість машино-години ЕОМ, грн/год (5,5 грн/год).

Підставивши в формулу (3.3) відповідні значення, визначимо вартість необхідного для налагодження машинного часу:

$$З_{мв} = 357,95 \cdot 5,5 = 1968,73 \text{ грн.}$$

Звідси витрати на створення програмного продукту:

$$К_{ПО} = \ 215\ 957,\ 34 + 1968,73 = 217\ 926,06 \text{ грн.}$$

Очікуваний період створення ПЗ:

$$T = \frac{t}{B_k \cdot F_p}, \text{ міс.} \qquad (3.14)$$

де B_k - число виконавців (дорівнює 1);

F_p - місячний фонд робочого часу (при 40 годинному робочому тижні F_p=176 годин).

Очікуваний період створення ПЗ:

$$T = \frac{737,66}{1 \cdot 176} \approx \ 4,2 \text{ міс.}$$

Висновок: Розробка системи агротехнічного моніторингу потребує 737,66 людино-години. З урахуванням курсу долару НБУ та медіани заробітної плати розробника ПЗ в Embedded напряму на момент написання роботи, для розробки знадобиться 217 926,06 грн. Запланований термін процесу розробки складає приблизно 4,2 місяця при роботі над проектом одного спеціаліста рівня Junior із тривалістю робочого часу не більше 40 годин на тиждень.

ВИСНОВКИ

У даній бакалаврській роботі було розроблено комп'ютерну систему для агрегування даних агромоніторингу з використанням сучасних апаратно-програмних засобів. Вибір технологій, таких як мікроконтролери NodeMCU ESP8266, різноманітні сенсори для моніторингу навколишнього середовища та хмарний сервіс ThingSpeak, був обґрунтований їх доступністю, простотою використання та високою функціональністю.

Під час виконання роботи були виконані наступні завдання:

- Проведено аналіз існуючих рішень та виділені основні функції систем агромоніторингу;
- Вибрано та використано відповідні давачі для збору даних та мікроконтролер;
- Реалізовано програмний код у середовищі Arduino IDE для збирання, передачі та обробки даних;
- Використано хмарний сервіс ThingSpeak для зберігання та візуалізації зібраних даних.

Розроблена система дозволяє ефективно збирати, обробляти та аналізувати дані агромоніторингу, що сприяє прийняттю обгрунтованих рішень для підвищення продуктивності та ефективності сільськогосподарської діяльності. Використання хмарних технологій забезпечує доступ до даних в режимі реального часу, що є важливим для оперативного управління аграрними процесами.

Економічна частина роботи містить розрахунок трудомісткості та вартості розробки програмного забезпечення. Було визначено, що загальна трудомісткість проекту склала 737,66 людино-годин, а загальні витрати на створення програмного продукту 217926,06 грн.

Результати роботи підтверджують доцільність використання обраних технологій для створення систем агромоніторингу та їх високу ефективність у вирішенні поставлених завдань.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Alo Sen, Rahul Roy, Satya Ranjan Dash // Smart Farming Using Machine Learning and IoT, Chapter 2 (08 March 2021): / DOI https://doi.org/10.1002/9781119769231.ch2 (дата звернення 15.05.2024)

2. Precision Agriculture Technology for Crop Farming by Qin Zhang 2016, 374c. (дата звернення 15.05.2024)

3. An Architecture model for Smart Farming: / URL https://www.researchgate.net/publication/335362251_An_Architecture_model_for_Smart_Farming (дата звернення 15.05.2024)

4. ESP32 vs ESP8266 – Pros and Cons: / URL https://makeradvisor.com/esp32-vs-esp8266/ (дата звернення 20.05.2024)

5. Comparison table for ESP8266/ ESP32/ ESP32-S/ ESP32-C: / URL https://gist.github.com/fabianoriccardi/cbb474c94a8659209e61e3194b20eb61 (дата звернення 20.05.2024)

6. NodeMCU ESP8266 Specifications, Overview and Setting up: / URL https://www.make-it.ca/nodemcu-details-specifications/ (дата звернення 20.05.2024)

7. Appearance NodeMCU: / URL https://projecthub.arduino.cc/PatelDarshil/getting-started-with-nodemcu-esp8266-on-arduino-ide-b193c3 (дата звернення 20.05.2024)

8. NodeMCU v3 high resolution pinout and specs: / URL https://mischianti.org/nodemcu-v3-high-resolution-pinout-and-specs/ (дата звернення 20.05.2024)

9. Interfacing DHT11 and DHT22 sensors: / URL https://lastminuteengineers.com/dht11-dht22-arduino-tutorial/ (дата звернення 29.05.2024)

10. Interface ublox NEO-6M GPS Module with Arduino: / URL https://lastminuteengineers.com/neo6m-gps-arduino-tutorial/ (дата звернення 29.05.2024)

11. Датчик ультрафіолету на GUVA-S12SD: / URL https://arduino.ua/prod1430-datchik-yltrafioleta-na-guva-s12sd (дата звернення 29.05.2024)

12. Interface OLED Graphic Display Module: / URL https://lastminuteengineers.com/oled-display-esp8266-tutorial/ (дата звернення 29.05.2024)

13. Capacitive soil moisture SKU SEN0193: / URL https://wiki.dfrobot.com/Capacitive_Soil_Moisture_Sensor_SKU_SEN0193 (дата звернення 29.05.2024)

14. Adafruit GFX Graphics Library: / URL https://learn.adafruit.com/adafruit-gfx-graphics-library/overview (дата звернення 02.06.2024)

15. Wire library: / URL https://www.arduino.cc/reference/en/language/functions/communication/wire/ (дата звернення 02.06.2024)

16. SimpleDHT Arduino library: / URL https://github.com/winlinvip/SimpleDHT (дата звернення 02.06.2024)

17. TinyGPS++ Arduino library: / URL https://github.com/mikalhart/TinyGPSPlus/tree/master (дата звернення 02.06.2024)

18. About Arduino IDE software: / URL https://www.javatpoint.com/arduino-ide (дата звернення 09.06.2024)

19. Getting started with Arduino IDE 2: / URL https://docs.arduino.cc/software/ide-v2/tutorials/getting-started-ide-v2/ (дата звернення 09.06.2024)

20. ThingSpeak for Internet of Things Projects: / URL https://thingspeak.com/ (дата звернення 13.06.2024)

21. Програма для проектування електронних схем Proteus: / URL https://uk.wikipedia.org/wiki/Proteus_Design (дата звернення 13.06.2024)

Додаток А

ЛІСТИНГ ПРОГРАМИ

Лістинг файлу WeatherMonitorScript.ino:

```
// library for I2C connection via SDA and SCL pins

#include <Wire.h>

// General graphics library for OLED screens

#include <Adafruit_GFX.h>

// Support library for specific model of OLED screen

#include <Adafruit_SSD1306.h>

// DHT control library

#include <SimpleDHT.h>

// Built-in board networking management library

#include <ESP8266WiFi.h>

// passwords and SSID data file

#include "secrets.h"

// GY-NEO6MV2 control library

#include <TinyGPS++.h>

// Software serial port emulation for communication with GPS module

#include <SoftwareSerial.h>

// HTTP requests handler library

#include "ThingSpeak.h" // always include thingspeak header file after other
header files and custom macros


#define analogPin A0;


// Change this value to modify interval between data reads and ThingSkeak
updates

const int iterationTime = 30;


// Variables for calibrating Soil Moisture output

const int AirValue = 650;

const int WaterValue = 260;
```

```
int intervals = (AirValue - WaterValue)/3;

// Agritech data values
int soilMoistureValue = 0;
byte temperature;
byte humidity;
float longitude =0;
float latitude =0;

// Defining comunication pins
int DHTpin = 2;
const int RX_PIN = 12, TX_PIN = 13;

// Initializing sensor and utility modules
SimpleDHT11 dht11(DHTpin);
const uint32_t GPS_BAUD = 9600; //Default baud of NEO-6M is 9600
TinyGPSPlus gps; // The TinyGPS++ object
SoftwareSerial gpsSerial(RX_PIN, TX_PIN); // The serial interface to the GPS
device
Adafruit_SSD1306 display(128,64,&Wire,-1);
WiFiClient  client;


// Setting values to sensor's channel variables
unsigned long myChannelNumber = SECRET_CH_ID;
const char * myWriteAPIKey = SECRET_WRITE_APIKEY;


void setup() {
  // Preparing the OLED screen
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(WHITE);
```

```
  display.setCursor(0,0);

  // Attune wireless connection
  WiFi.mode(WIFI_STA);
  // Initialize ThingSpeak
  ThingSpeak.begin(client);

  // SETING UP GPS SERIAL
  Serial.begin(9600);
  gpsSerial.begin(GPS_BAUD);

}

void loop() {

  readDHTData();

  readSoilMoisture();

  readGPS();

  handlerWiFi();

  countDownTillRepeat();

}

// ACTUAL METHODS

void readDHTData(){
  // Partially clean the screen
  display.fillRect(0,0,128,15, BLACK);
  display.setCursor(0,0);
```

```
  // Trying to read DHT values
  if ((dht11.read(&temperature, &humidity, NULL)) != SimpleDHTErrSuccess) {
      display.println("Data reading failed!!");
    } else {
        // Printing values on screen for local user to see
      display.print("Temperature: ");
      display.print((int)temperature);
      display.write(0xF7);
      display.println("C");
      display.print("Humidity: ");
      display.print((int)humidity);
      display.println("%");
    }
  display.display();
}



void readSoilMoisture(){
  soilMoistureValue = analogRead(A0);
  // Partially clean the screen
  display.fillRect(0,16,128,7, BLACK);
  display.setCursor(0,16);
  // Printing values on screen for local user to see
  display.print("SM: ");display.print(soilMoistureValue);
  if(soilMoistureValue > WaterValue && soilMoistureValue < (WaterValue +
intervals))
    {
      display.println(" - Very Wet");
    }
  else if(soilMoistureValue > (WaterValue + intervals) && soilMoistureValue <
(AirValue - intervals))
    {
      display.println(" - Wet");
    }
```

```
  else if(soilMoistureValue < AirValue && soilMoistureValue > (AirValue -
intervals))
    {
      display.println(" - Dry");
    }
  display.print(soilMoistureValue);
}


void readGPS(){
    // Partially clean the screen
  display.fillRect(0,24,128,7, BLACK);
  display.setCursor(0,24);
  if (gpsSerial.available() > 0) {
    if (gps.encode(gpsSerial.read())) {
      if (gps.location.isValid()) {
        // Printing values on screen for local user to see
        display.print(F("Lat: "));
        display.print(gps.location.lat());
        latitude = gps.location.lat();
        display.print(F(" Lon: "));
        display.print(gps.location.lng());
        longitude = gps.location.lng();
        display.print(F(" Alt: "));
        if (gps.altitude.isValid())
          display.println(gps.altitude.meters());
        else
          display.println(F("INVALID"));
      } else {
        display.println(F("location: INVALID"));
      }
    }
  }
  display.display();
}
```

```
void handlerWiFi(){
   // Connect or reconnect to WiFi
  if(WiFi.status() != WL_CONNECTED){
    int i = 0;
    while(WiFi.status() != WL_CONNECTED && i < 10){
      WiFi.begin(SECRET_SSID, SECRET_PASS);
      display.setCursor(0,48);
      display.print("Connecting");
      // Maximum of 10 tries
      for(int j = 0; j < i; j++){
        // Visualizing the number of connection attempts
        display.print(".");
      }
      display.display();
      delay(5000);
      i++;
    }
    // Connection happened:
    display.setTextColor(BLACK);
    display.setCursor(0,48);
    display.println("Connecting...");
    display.setTextColor(WHITE);
    display.setCursor(0,48);
    if(WiFi.status() != WL_CONNECTED){
      display.println("No Internet Connection");
      display.display();
      return;
    }
    // Printing Network info for local user
    display.print("Wi-Fi: ");
    display.println(SECRET_SSID);
    display.display();
```

```
  } else { // If connection already established
    display.setTextColor(WHITE);
    display.setCursor(0,48);
    display.print("Wi-Fi: ");
    display.println(SECRET_SSID);
    display.display();
  }

    // Preparing values for HTTP request
  ThingSpeak.setField(1, temperature);
  ThingSpeak.setField(2, humidity);
  ThingSpeak.setField(3, soilMoistureValue);
  ThingSpeak.setField(4, latitude);
  ThingSpeak.setField(5, longitude);

    // Assembling HTTP request
 int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);

    // Partially clean the screen
  display.fillRect(120, 56, 8, 8, BLACK);
  display.setCursor(120,56);
  // Output success\failure marker at corner of the screen
  if(x == 200){
    display.println("S"); //SUCCESS
  }
  else{
    display.println("E"); //ERROR
  }
  display.display();
}

// Visualizing time interval before next iteration
void countDownTillRepeat(){
```

```
for(int i = iterationTime; i > 0; i--){
    display.setCursor(0,56);
    display.setTextColor(BLACK);
    display.print("Repeating in: ");
    display.println(i+1);
    display.setCursor(0,56);
    display.setTextColor(WHITE);
    display.print("Repeating in: ");
    display.println(i);
    display.display();
    delay(1000);
  }
  // CLEAR LAST ROW BEFORE REPEATING
  display.setCursor(0,56);
  display.setTextColor(BLACK);
  display.print("Repeating in: 1");
  display.setTextColor(WHITE);
  display.display();
}
```

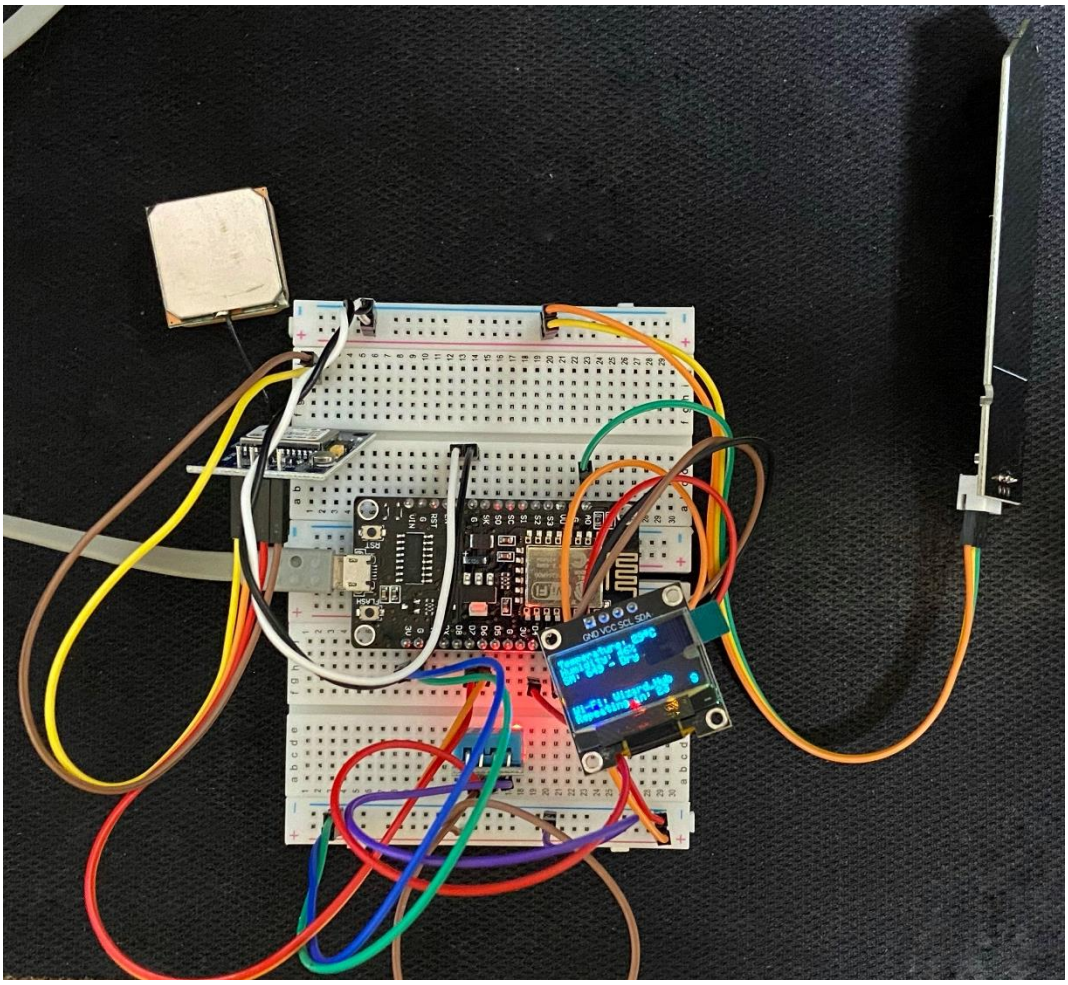
Лістинг файлу secrets.h:

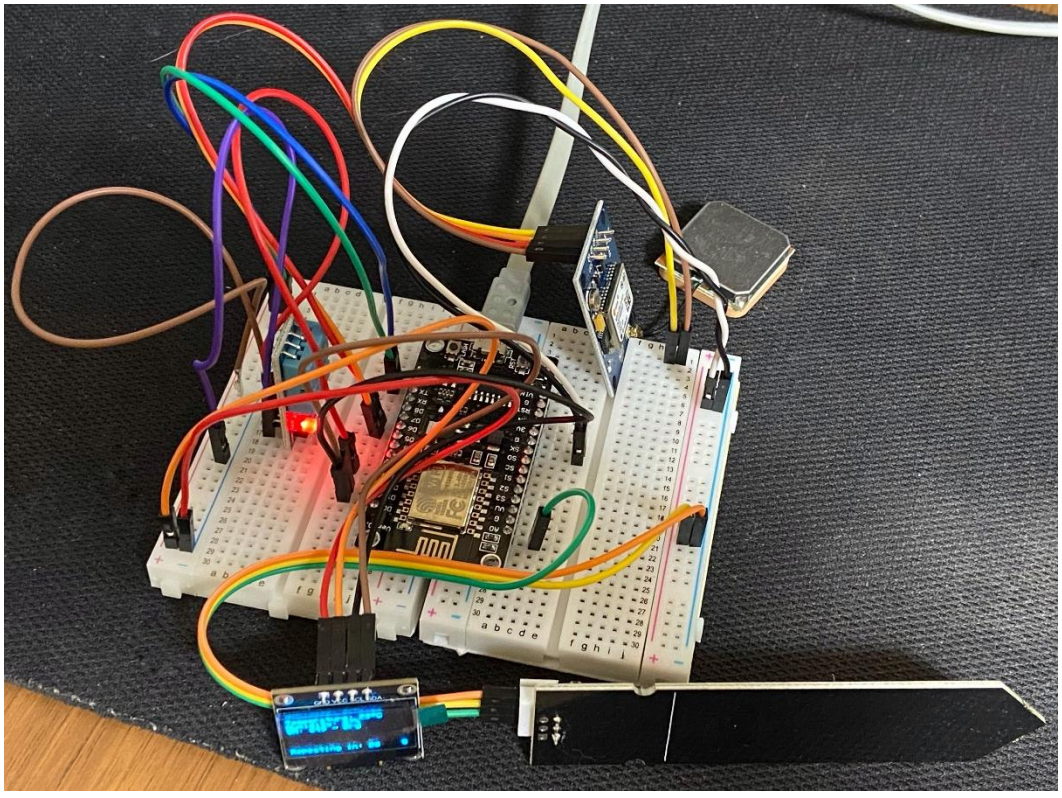
```
#define SECRET_SSID "Wizard_Hub"
#define SECRET_PASS "=D034>sOi7A"

#define SECRET_CH_ID 2549956
#define SECRET_WRITE_APIKEY "2VSKDTP5Z5X7W574"
```

Додаток Б

ЗОВНІШНІЙ ВИГЛЯД ПРИЛАДУ





Додаток В

ВІДГУК КЕРІВНИКА ЕКОНОМІЧНОГО РОЗДІЛУ

Додаток Г

ПЕРЕЛІК ФАЙЛІВ НА ДИСКУ

Ім’я файлу	**Опис**
Пояснювальні документи	
	Пояснювальна записка до дипломного проекту. Документ Word.
	Пояснювальна записка до дипломного проекту в форматі PDF
Програма	
	Архів. Містить коди програми
Презентація	
	Презентація дипломного проекту