Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Автоматики та комп'ютеризованих технологій_____
(повна назва)

Кафедра _Комп'ютерно-інтегрованих технологій, автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____перший (бакалаврський)_____

Розробка програмної системи для контролю параметрів культивування рослин у закритому просторі
(тема)

Виконав:
студент __4_ курсу, групи _ТРІТЗР-20-1_____
_____Анісімова Д. М._____
(прізвище, ініціали)

Спеціальність 172 Телекомунікації та радіотехніка_____
(код і повна назва спеціальності)
Тип програми ____освітньо-професійна_____
Освітня програма Інтелектуальні технології засобів радіоелектроніки_____
(повна назва освітньої програми)

Керівник доц. каф. КІТАР Жарікова І. В._____
(посада, прізвище, ініціали)

Допускається до захисту

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

2024 р.

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

04.06.2024 Анісімова Д.М.

ХАРКІВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ РАДІОЕЛЕКТРОНІКИ

Факультет АКТ
Кафедра КІТАР
Рівень вищої освіти перший (бакалаврський)
Спеціальність 172 Телекомунікації та радіотехніка
Тип програми освітньо-професійна
Освітня програма Інтелектуальні технології засобів радіоелектроніки
(шифр і назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«__»_____2024р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

студентові Анісімовій Дарині Максимівні
(шифр і назва)

1. Тема роботи: Розробка програмної системи для контролю параметрів культивування рослин у закритому просторі
Затверджена наказом університету від №479Ст від 20.05.2024
2. Термін подання студентом роботи до екзаменаційної комісії 05.06.2024 р.
3. Вихідні дані до роботи
3.1 Розробити програмну систему, призначену для контролю процесу культивування рослин;
3.2 Контрольовані параметри;
3.3 Мова програмування C#;
3.4 Використання фреймворку .NET для BackEnd та фреймворку Angular, який використовує мову програмування TypeScript для FrontEnd.
4. Перелік питань, що потрібно опрацювати в роботі:
4.1 Вступ;
4.2 Аналіз технічного завдання;
4.3 Функціонал програмної системи;
4.4 Розробка програмної системи;
4.5 Результат розробки програмної системи;
4.6 Висновки.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій Графічний демонстраційний матеріал в форматі PowerPoint(*.ppt) формату А4 –14 сторінок.

6. Консультанти розділів роботи

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

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Аналіз предметної галузі	01.05.2024	*виконано*
2	Створення специфікації програмної системи	03.05.2024	*виконано*
3	Проєктування програмної системи	06.05.2024	*виконано*
4	Розробка програмної системи	10.05.2024	*виконано*
5	Тестування програмної системи	15.05.2024	*виконано*
6	Оформлення пояснювальної записки	04.06.2024	*виконано*
7	Подання роботи на нормоконтроль	05.06.2024	*виконано*
8	Перевірка роботи у системі Unicheck	09.06.2024	*виконано*
9	Отримання відгуку та рецензії	12.06.2024	*виконано*
10	Подання роботи до ЕК	15.06.2024	*виконано*
11	Підготовка презентації та доповіді	17.06.2022	*виконано*

Дата видачі завдання 29 квітня 2024 р.

Студент Анісімова Д. М.
(підпис) (прізвище, ініціали)

Керівник роботи Жарікова І. В.
(підпис) (прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка: 85 стор., 1 табл., 26 рис., 2 дод., 20 джерел.

КУЛЬТИВУВАННЯ РОСЛИН, MQTT, HIVEMQ, .NET, C#, ANGULAR, TYPESCRIPT, MS SQL SERVER, .NET MAUI.

Об’єкт розробки – процес контролю ферми для налаштування параметрів мікроклімату та освітлення для вирощування овочів у теплиці.

Предмет розробки – програмне забезпечення для системи контролю ферми для вирощування овочів у теплиці.

Мета розробки – автоматизувати процес моніторингу та управління умовами культивування рослин з метою оптимізації їх росту та розвитку в закритих приміщеннях.

Метод вирішення – використання модуля ESP8266 для збирання даних від датчиків температури (DS18B20) та вологості (DHT22), передача даних через мережевий протокол MQTT до брокера HiveMQ, серверна частина реалізована на фреймворку .NET з мовою програмування C#, веб-частина розроблена за допомогою фреймворку Angular та мови програмування TypeScript, база даних MS SQL Server.

У результаті розробки було спроєктовано та реалізовано програмну систему, яка включає наступний функціонал: збір даних про температуру та вологість у приміщенні за допомогою датчиків; передача зібраних даних на сервер для обробки; зберігання даних у реляційній базі даних MS SQL Server; веб-інтерфейс для моніторингу та управління параметрами культивування рослин; мобільний застосунок, розроблений на основі технології .NET MAUI, для дистанційного контролю та управління умовами культивування.

ABSTRACT

Explanatory note: 85 pp., 1 table, 26 figs., 2 appendixes, 20 sources.

PLANT CULTIVATION, MQTT, HIVEMQ, .NET, C#, ANGULAR, TYPESCRIPT, MS SQL SERVER, .NET MAUI.

The object of development is the process of controlling the farm for setting microclimate parameters and lighting for growing vegetables in a greenhouse.

The subject of development is software for a farm control system for growing vegetables in a greenhouse.

The purpose of the development is to automate the process of monitoring and controlling the conditions of plant cultivation in order to optimize their growth and development indoors.

The solution method is the use of the ESP8266 module for collecting data from temperature (DS18B20) and humidity (DHT22) sensors, data transmission via the MQTT network protocol to the HiveMQ broker, the server part is implemented on the .NET framework with the C# programming language, the web part is developed using the Angular framework and the TypeScript programming language, and the MS SQL Server database.

As a result of the development, a software system was designed and implemented, which includes the following functionality: collecting data on temperature and humidity in the room using sensors; transferring the collected data to the server for processing; storing data in a relational MS SQL Server database; a web interface for monitoring and controlling plant cultivation parameters; a mobile application developed on the basis of .NET MAUI technology for remote monitoring and control of cultivation conditions.

ЗМІСТ

Перелік умовних скорочень 9
Вступ 10
1 Аналіз технічного завдання 12
1.1 Аналіз предметної галузі 12
1.2 Бізнес-аналіз розроблюваної програмної системи 18
1.3 Потреби клієнтів та ринку 19
2 Функціонал програмної системи 20
2.1 Концепція програмної системи 20
2.2 Аналіз технічних вимог до програмної системи 21
2.3 Обмеження та винятки, що потрібно враховувати при впровадженні системи контролю умов вирощування овочів 22
2.4 Проєктування програмної системи 22
2.5 Робоче середовище 23
2.6 Розрахунок впливу температури на ріст рослин 24
2.6.1 Модель Гаусса і ріст рослин 24
2.6.2 Розрахунок швидкості росту рослин 25
2.7 Вибір принципу автоматичного управління температурою у теплиці 26
3 Розробка програмної системи 28
3.1 Проєктування архітектури програмної системи 28
3.2 Вибір інтегрованого середовища розробки 29
3.3 Збір даних з датчиків 30
3.3.1 Ініціалізація та налаштування датчиків 30
3.3.2 Підготовка сервісів для датчиків 31
3.3.3 Основний цикл симуляції датчиків 31
3.4 Обробка та зберігання даних 33
3.5 Розробка мобільного застосунку 34
3.6 Безпека системи 36
3.7 Аналіз питань охорони праці під час роботи з програмною системою 37

4 Результат розробки програмної системи 38
4.1 Візуалізація даних 38
4.1.1 Розробка веб-інтерфейсу 38
4.1.2 Архітектура веб-інтерфейсу 49
4.1.3 Реалізація реального часу 50
4.2 Розрахунки параметрів програмної системи 50
Висновки 53
Перелік джерел посилання 55
Додаток А Лістинг коду програми 58
Додаток Б Демонстраційний матеріал 84

ПЕРЕЛІК УМОВНИХ СКОРОЧЕНЬ

КІТАР – комп’ютерно-інтегровані технології, автоматизація та робототехніка;

ПІД – пропорційно-інтегрально-диференціальні;

ТАУ – теорія автоматичного управління;

ХНУРЕ – Харківський національний університет радіоелектроніки;

ЦСР – цілі сталого розвитку;

ER – Entity-relationship;

IOT – Internet of Things;

MEMC – Microelectromechanical Systems;

MOEMC – Microoptoelectromechanical Systems;

MQTT – Message Queue Telemetry Transport;

SPA – Single Page Application;

SQL – Structured Query Language;

UML – Unified Modeling Language.

ВСТУП

У сучасному світі тенденція до автоматизації виробничих процесів стає все більш актуальною. Зокрема, у галузі сільського господарства зростає попит на розвиток технологій, спрямованих на оптимізацію процесів культивування рослин. Завдяки автоматизації, можна досягти більшої ефективності та точності в управлінні параметрами середовища для росту рослин у закритому просторі.

Розробка програмної системи для контролю параметрів культивування рослин у закритому просторі відіграє важливу роль у підвищенні врожайності та якості продукції. Ця система не лише забезпечує постійний моніторинг і регулювання умов росту, але й відкриває нові можливості для використання інноваційних підходів у сільському господарстві.

Враховуючи швидкий розвиток технологій, поява нових матеріалів та сенсорів, а також зростаючу потребу в екологічно чистих та ефективних методах вирощування рослин, розробка автоматизованих систем керування параметрами середовища у сільському господарстві стає невід'ємною складовою успішного агротехнічного процесу.

Мета роботи – розробка програмного забезпечення для контролю параметрів мікроклімату у теплицях агропідприємства для підвищення врожайності.

Об’єкт розробки – процес контролю ферми для налаштування параметрів мікроклімату та освітлення для вирощування овочів у теплиці.

Предмет розробки – програмне забезпечення для системи контролю ферми для вирощування овочів у теплиці.

Для досягнення поставленої мети необхідно вирішити такі завдання:

– проаналізувати предметну область;

– розробити програмну систему, що складається з 4 частин: веб-застосунку, емулятора IOT-пристроїв, серверної частини та мобільний застосунок;

– впровадити механізми обробки та збереження даних з IOT-пристроїв для

аналізу параметрів культивування;

– оформити кваліфікаційну роботу згідно ДСТУ 3008:2015 [1], а також з методичними вказівками з підготовки й оформлення кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 172 Телекомунікації та радіотехніка за освітньою програмою «Інтелектуальні технології засобів радіоелектроніки» [2].

За допомогою розроблюваної системи користувачі зможуть створювати власні шаблони для оптимізації процесів вирощування рослин, які в подальшому можуть бути оцінені та випробувані іншими фермерами.

Дана робота проводилась у рамках програми двостороннього міжнародного співробітництва між Харківським національним університетом радіоелектроніки, Навчальним закладом Кельнський університетом (Universität zu Köln) та Вищою технічною школою (Technische Hochschule Köln) за «Програмою професійного навчання в межах академічної мобільності студентів факультету Автоматики і комп’ютеризованих технологій Харківського національного університету радіоелектроніки», в ході інноваційного дослідження «InnoBioDiv» взаємодії рослин і середовища в мінливому кліматі, що поєднує біологію та сучасні технології Інтернету речей. Проведені дослідження відповідають цілям сталого розвитку (ЦСР): ЦСР 9, ЦСР 12, ЦСР 13 і ЦСР 17.

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Аналіз предметної галузі

У квітні 2023 року 4 університети, а саме: Харківський національний університет радіоелектроніки, Харківський національний університет імені В. Н. Каразіна, Technology Arts Sciences TH Koln та Universitat zu Koln об’єдналися для досягнення однієї мети – розробка стратегій адаптації сільськогосподарських культур та екосистем до змін клімату. Це включає збереження середовища існування, забезпечення продовольчої безпеки, збереження ресурсів, а також ефективне використання води та поживних речовин у сільському господарстві. На проєкті студентам був продемонстрований FarmBot [5]. FarmBot – це роботизована система для автоматизованого вирощування рослин, призначена для використання в малих сільськогосподарських господарствах, домашніх садах або освітніх проєктах. FarmBot являє собою відкриту апаратно-програмну платформу, яка використовує роботизовану руку, встановлену на рейках, щоб висаджувати, поливати, моніторити та доглядати за рослинами (рис. 1.1). Основні особливості FarmBot:

– автоматизація: система може виконувати різні завдання з вирощування рослин, такі як посадка насіння, полив, контроль за станом рослин та видалення бур’янів;

– програмування: користувачі можуть програмувати робота для виконання специфічних завдань за допомогою спеціального програмного забезпечення;

– відкритий код: проєкт FarmBot є відкритим і доступним для модифікації, що дозволяє ентузіастам і розробникам адаптувати та вдосконалювати систему під свої потреби;

– інтеграція з IoT: FarmBot може підключатися до Інтернету, що дозволяє віддалено керувати роботом та отримувати дані про стан рослин і умови

вирощування.

Рисунок 1.1 – FarmBot з українсько-німецького проєкту

За допомогою FarmBot студенти могли дистанційно поливати свої рослини та контролювати норму води, забезпечуючи оптимальну кількість для вирощування рослин. Виконувались досліди, наскільки кількість води та її синхронне постачання до рослини впливає на її зростання (рис. 1.2). Саме цей проєкт наштовхнув на думки про те, як важливо мати автоматизовані системи

контролю, які б полегшували процес вирощування рослин фермерам та іншим користувачам.

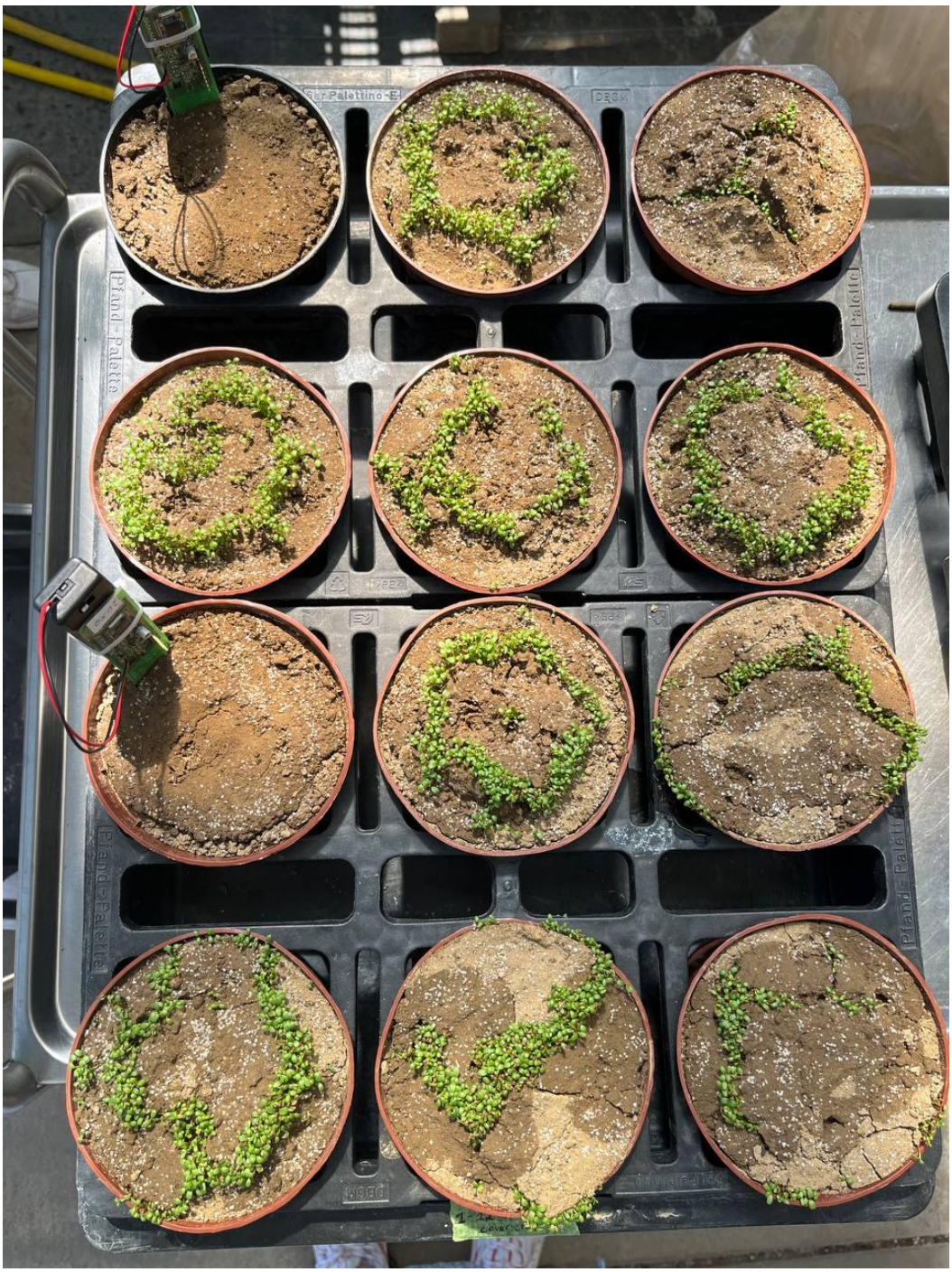

Рисунок 1.2 – Приклад, як кількість води впливає на зростання рослин

З розвитком міських і вертикальних систем землеробства зростає інтерес до методів кімнатного вирощування рослин. Це вимагає розробки ефективних автоматизованих систем контролю параметрів культивування рослин для

забезпечення оптимальних умов росту рослин.

Прогрес у галузі мікроконтролерів і вбудованих систем, зростання можливостей мікроконтролерів і розвиток вбудованих систем дозволяють створювати компактні та ефективні рішення для автоматичного контролю параметрів навколишнього середовища. Це дає можливість розробляти нові модулі для управління умовами вирощування рослин.

Одним із прикладів є автоматизована теплиця компанії «ЕКО ТЕС» (рисунок 1.3) [6]. Теплиця має блок управління, вбудований безпосередньо в саму теплицю, зв’язок між датчиками та блоком управління відбувається за допомогою дротових та локальних мереж. Теплиця може працювати в холодний період на найрізноманітніших типах палива, обігрівання відбувається за допомогою:

– печі на дровах;

– інфрачервоного підігріву;

– електричного нагрівачу.

Освітлення відбувається лампами різного спектру, що забезпечує найкращі ілюмінаційні умови для різних етапів росту рослин.

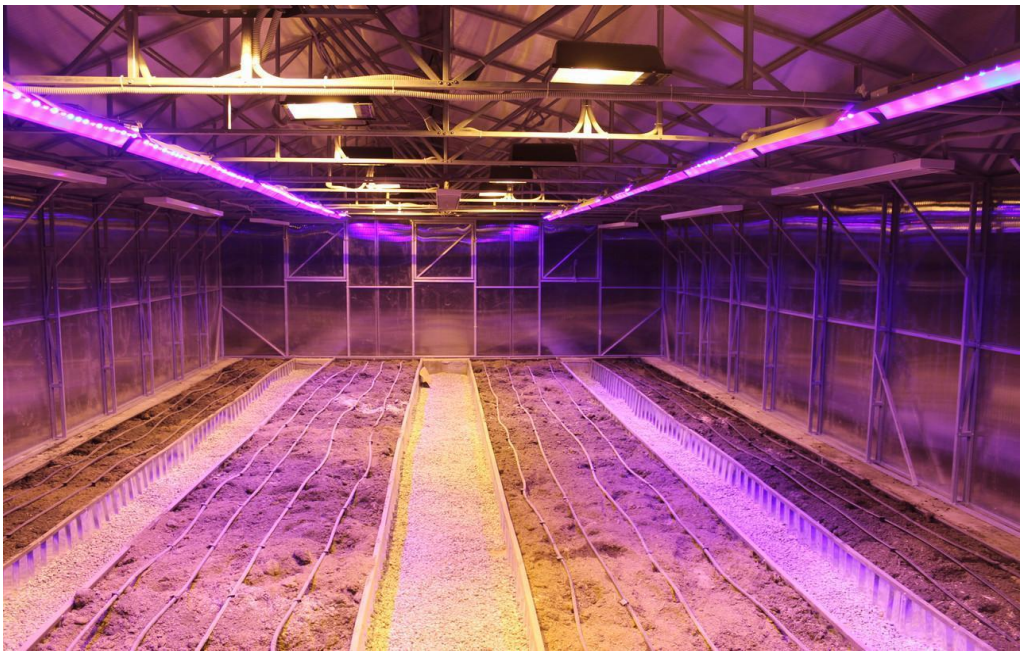

Рисунок 1.3 – Приклад промислової автоматичної теплиці від «ЕКО ТЕС» [6]

Ще за приклад можна взяти гроубокс [7]. Гроубокс – це спеціальна ємність для вирощування рослин, яка регулює мікроклімат і підтримує сприятливі для рослин умови довкілля, включно зі світлом, вологістю, температурою і балансом поживних речовин. Гроубокс від GrowBoxUA [8] має доволі широкий набір функцій (рисунок 1.4). У виробі вже наявна система канальної вентиляції з вугільним фільтром, а також вбудований вентилятор обдуву для циркуляції повітря всередині гроубокса.

Рисунок 1.4 – Гроубокс стелс GrowBoxUA 120вт [8]

Ще за приклад можна взяти фітотрон [9]. Фітотрон – це спеціальна установка або приміщення, яке створює контрольовані умови для вирощування рослин. У фітотроні можна регулювати такі параметри, як світло, температура, вологість і склад повітря, що дозволяє створювати оптимальні умови для росту та розвитку рослин незалежно від зовнішніх кліматичних умов.

Фітотрони (рис. 1.5) використовуються у наукових дослідженнях, агрономії та біотехнології для вивчення впливу різних факторів на ріст рослин, розробки нових методів культивування та селекції рослин, а також для комерційного вирощування рослин у контрольованих умовах.

Рисунок 1.5 – Фітотрон [9]

На основі порівняння можливостей різних типів тепличних систем можна краще зрозуміти, які функції можуть бути додані до розроблюваної системи для підтримки абіотичних факторів.

1.2 Бізнес-аналіз розроблюваної програмної системи

Розширення ринкових можливостей сільськогосподарських підприємств відкриває великі перспективи для збільшення обсягів виробництва та розширення асортименту продукції. Забезпечуючи стабільну якість та врожайність сільськогосподарських культур, компанії можуть зайняти більш активну позицію на ринку та залучити нових клієнтів. Підвищення конкурентоспроможності сільгоспвиробників дозволяє компаніям знизити собівартість продукції та підвищити продуктивність праці.

Це відкриває ще більше можливостей для співпраці з науковими установами та виробниками техніки, дозволяючи впроваджувати передові рішення та підвищувати ефективність виробництва.

Головна мета системи є збільшити ефективність росту рослин, а для цього потрібно забезпечити надійний моніторинг температури та вологості, а також освітлення. Потрібно ще врахувати, щоб система була стабільною та працювала без перебоїв.

Основною метою є збільшення врожайності та покращення якості вирощуваної продукції, програмна система повинна забезпечувати оптимальні умови для росту рослин, щоб досягти максимального виходу продукції від кожного квадратного метра землі.

Програмна система повинна спростити процес контролю та управління умовами вирощування рослин, щоб зменшити час та ресурси, потрібні для їх підтримки. Це дозволить збільшити ефективність виробництва та знизити витрати.

Програмна система повинна бути гнучкою та масштабованою, щоб вона легко адаптувалась до потреб різних розмірів та типів сільськогосподарських ферм. Це дозволить підприємствам масштабувати своє виробництво відповідно до змін потреб ринку та їхніх власних стратегій розвитку.

Існує ризик того, що нові системи автоматизації можуть бути не повністю сумісними з існуючими системами або не інтегруватися з ними належним чином.

Це може ускладнити використання та обслуговування системи або спричинити нестабільність.

Використання мереж IoT для збору даних про вирощування сільськогосподарських культур може наражати систему на ризики безпеки даних. Це може включати можливість несанкціонованого доступу до даних і атак на систему, що може вплинути на ефективність і безпеку системи.

Регулятивні вимоги та стандарти, пов’язані з вирощуванням сільськогосподарських культур і використанням сучасних технологій, можуть змінитися та ускладнити впровадження нових систем. Наслідки призводять до додаткових витрат і перешкод для компанії.

1.3 Потреби клієнтів та ринку

Фермери та агропідприємства шукають рішення, які дозволять їм швидко реагувати на зміну умов вирощування всередині та поза межами своїх культур. Наприклад, це системи, які можуть автоматично реагувати на зміни температури, вологості або освітлення, щоб забезпечити оптимальні умови для росту рослин і отримання максимального врожаю.

Клієнти на ринку сільськогосподарської техніки та технологій шукають рішення для зменшення ризиків і витрат, пов’язаних із вирощуванням сільськогосподарських культур. Наприклад, системи, які дозволяють ефективно використовувати ресурси, уникати втрат врожаю та мінімізувати вплив негативних факторів на посіви.

2 ФУНКЦІОНАЛ ПРОГРАМНОЇ СИСТЕМИ

2.1 Концепція програмної системи

Програмна система розробляється, щоб допомогти користувачам керувати тепличними операціями та підвищувати врожайність за допомогою шаблонів, які користувачі можуть створювати самі, а потім інші користувачі можуть оцінювати та тестувати на своїх фермах шаблони.

Головна функціональність програмної системи включає:

– створення ферми шляхом додавання опису самої ферми;

– створення шаблону для вирощування овочів;

– створення використання шаблону для вирощування рослини на фермі;

– відстеження температури та вологості на фермі в режимі реального часу та відображення цих даних на графіці.

Розглянемо передумови для впровадження системи сільськогосподарського контролю з використанням програмної системи.

Припущення для впровадження системи сільськогосподарського контролю передбачають:

– наявність фізичної інфраструктури, тому що впровадження системи сільськогосподарського контролю потребує наявної теплиці або структури для вирощування овочів, яку можна оснастити необхідними датчиками IoT;

– наявність живлення та доступу до Інтернету, бо датчикам і програмним системам IoT потрібне постійне живлення та доступ до Інтернету для передачі даних;

– знання та навички програмування, тому що розробка систем програмного забезпечення та їх інтеграція з датчиками IoT вимагає знань програмування та розуміння маніпулювання даними датчиків;

– наявність фінансових ресурсів для витрат на обладнання, розробку програмного забезпечення та підтримку системи.

Успішне впровадження системи сільськогосподарського контролю включає:

– підтримку та прийняття фермерами, ефективність та успіх системи залежить від готовності фермерів використовувати нову технологію та розуміння її переваг, тому важливо, щоб фермери хотіли вивчити програмне забезпечення та використовувати його;

– надійне і доступне підключення до Інтернету, для безперебійної роботи системи, передачами даних між датчиками IoT і програмними системами;

– технічна підтримка та оновлення для вирішення потенційних проблем і забезпечення безперебійної роботи є критично важливими для забезпечення стабільної роботи системи.

2.2 Аналіз технічних вимог до програмної системи

Необхідно розробити і впровадити програмну систему, що складається з чотирьох частин частин: BackEnd, FrontEnd, IoT і мобільний додаток.

Частина BackEnd повинна мати end-points для створення, редагування та видалення таких даних: користувачі, ферми, шаблони для вирощування овочів, використання шаблону для вирощування.

Частина FrontEnd повинна мати зручний інтерфейс, який підтримує дві мови – українську та англійську, функціональність для створення, редагування та видалення таких даних: користувачі, ферми, шаблони для вирощування овочів, використання шаблону для вирощування. Відображення інформації про користувачів, ферми, шаблони для вирощування овочів, використання шаблону для вирощування.

IoT-частина має вимірювати температуру і вологість повітря. Дані з вимірювання температури та вологості повітря передаються в реальному часі та відображаються на графіку. Додатковий функціонал буде доданий в майбутніх випусках на основі відгуків клієнтів.

Мобільний додаток має давати змогу адміністратору переглядати

інформацію про свій профіль користувача, список ферм, шаблони для вирощування рослин. Додатковий функціонал буде доданий в майбутніх випусках на основі відгуків клієнтів.

2.3 Обмеження та винятки, що потрібно враховувати при впровадженні системи контролю умов вирощування овочів

Фізичні обмеження можуть бути пов'язані з розміром теплиці та кліматичними особливостями регіонів. Технічні обмеження стосуються інтеграції автоматизованого модуля з існуючими сільськогосподарськими системами та доступності простого у використанні інтерфейсу. Фінансові обмеження включають витрати на обладнання, обслуговування та підтримку системи. Організаційні обмеження можуть виникати з внутрішнього прийняття та підтримки ферми, а також співпраці з постачальниками обладнання. Винятки стосуються екстремальних природних явищ та технічних проблем, які можуть впливати на надійність системи контролю умов вирощування в теплиці.

2.4 Проєктування програмної системи

Перед початком реалізації програмної системи було побудовано ER-діаграму бази даних [10] (рис. 2.1). Модель зв’язку сутності (ER Modeling) – це графічний підхід до проєктування бази даних. ER-діаграми найчастіше використовуються для проєктування або налагодження реляційних баз даних у сферах розробки програмного забезпечення, бізнес-інформаційних систем, освіти та досліджень.

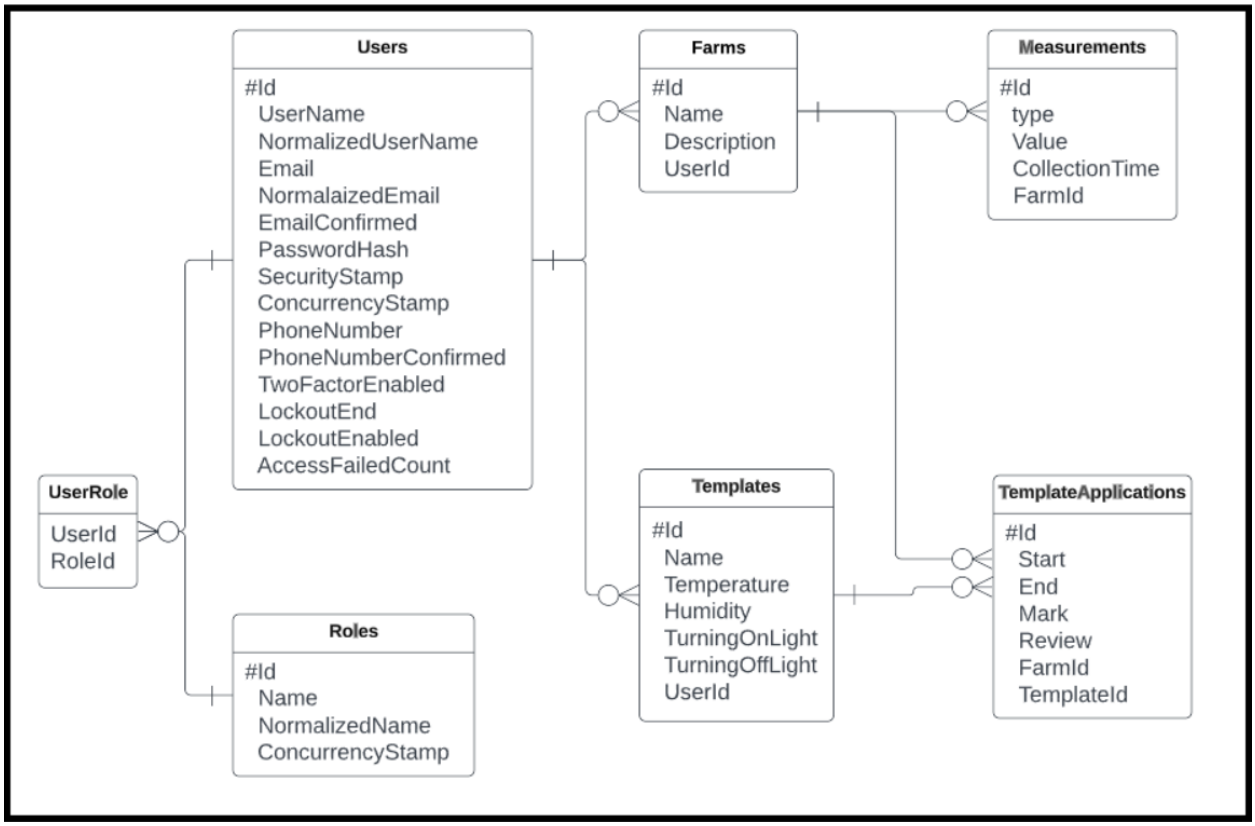

Рисунок 2.1 – ER-діаграма бази даних

На діаграмі продемонстровано відношення між таблицями бази даних, а саме: один користувач може мати безліч ролей, один користувач може мати безліч ферм, створювати безліч шаблонів, одна ферма може мати безліч вимірів даних датчиків та мати безліч використань шаблонів.

2.5 Робоче середовище

Під час розробки та наявністю практичних знань для управління базою даних було обрано реляційну базу даних, а саме MS SQL Server [11]. Для серверної частини програми застосувала фреймворк .NET [12] з мовою програмування C#, Веб частина застосунку була розроблена за допомогою фреймворку Angular [13], який використовує мову програмування TypeScript [14].

2.6 Розрахунок впливу температури на ріст рослин

Температура є одним із ключових факторів, що впливає на швидкість росту рослин. Різні рослини мають різні оптимальні температурні діапазони, у межах яких їхній ріст є максимальним. За межами цих діапазонів швидкість росту рослин зменшується. Для моделювання впливу температури на ріст рослин використовується рівняння, яке описує залежність швидкості росту від температури [15]. Залежність швидкості росту рослин від температури можна описати за допомогою моделі Гауса (нормального розподілу). Ця модель враховує, що швидкість росту рослин є максимальною за певної оптимальної температури та зменшується у міру відхилення від цієї температури. Використання нормального розподілу обґрунтоване тим, що більшість біологічних процесів мають оптимальні умови, за яких вони протікають найефективніше.

Нормальний розподіл описує, як значення випадкової величини розподіляються навколо середнього значення, і використовується в багатьох галузях науки для моделювання природних явищ. У контексті росту рослин це означає, що швидкість росту максимальна за оптимальної температури та зменшується симетрично у міру відхилення від цієї температури.

2.6.1 Модель Гаусса і ріст рослин

Нормальний розподіл (або Гауссовий розподіл) має форму дзвону та описується наступною формулою:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, \tag{2.1}$$

де x – випадкова величина (у нашому випадку це температура);

μ – середнє значення (оптимальна температура для росту рослин);

σ – стандартне відхилення, яке визначає ширину розподілу.

У нашій моделі ми хочемо описати швидкість росту рослин $R(T)$ як функцію температури T, з максимальною швидкістю росту за оптимальної температури T_{opt}. Формула виглядає так:

$$R(T) = R_{opt} \cdot e^{-\left(\frac{(T-T_{opt})^2}{2\sigma^2}\right)}, \quad (2.2)$$

де $R(T)$ – швидкість росту за температури T;

R_{opt} – максимальна швидкість росту за оптимальної температури T_{opt};

T – поточна температура;

T_{opt} – оптимальна температура для росту;

σ – стандартне відхилення температури, що визначає, як швидкість росту зменшується за умови відхилення від оптимальної температури.

2.6.2 Розрахунок швидкості росту рослин

Припустимо, що для конкретного виду рослин оптимальна температура для росту становить 22,5°C максимальна швидкість росту R_{opt} дорівнює 1 умовній одиниці, а стандартне відхилення температури σ становить 2°C. Визначимо швидкість росту за температури 25°C.

$$R(25) = 1 \cdot e^{-\left(\frac{(25-22,5)^2}{2 \cdot 2^2}\right)} = e^{-\left(\frac{2,5^2}{8}\right)} = e^{-\left(\frac{6,25}{8}\right)} = e^{-0,78125} \approx 0,457. \quad (2.3)$$

Таким чином, за температури 25°C швидкість росту рослин становитиме приблизно 45,7% від максимальної швидкості росту за оптимальної температури.

2.7 Вибір принципу автоматичного управління температурою у теплиці

Автоматичне управління є важливою складовою програмної системи для контролю параметрів культивування рослин у закритому просторі. Воно дозволяє забезпечити оптимальні умови для росту рослин шляхом регулювання різних параметрів, таких як температура, вологість, освітлення та інші.

У розроблюваній програмній системі для контролю параметрів культивування рослин автоматичне управління може бути реалізовано на базі таких алгоритмів, як ПІД-регулятори (пропорційно-інтегрально-диференціальні регулятори) [16]. ПІД-регулятор забезпечує ефективне управління шляхом розрахунку трьох складових:

– пропорційна складова (P) – пропорційна до помилки між заданим значенням параметра та його поточним значенням;

– інтегральна складова (I) – враховує накопичену помилку за певний період часу;

– диференціальна складова (D) – враховує швидкість зміни помилки.

Математично ПІД-регулятор описується наступною формулою:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de(t)}{dt}, \qquad (2.4)$$

де $u(t)$ – керуючий сигнал;

$e(t)$ – помилка;

K_p, K_i, K_d – коефіцієнти пропорційної, інтегральної та диференціальної складових відповідно.

Для реалізації ПІД-регулятора для контролю температури у програмній системі можна використати наступний алгоритм:

– зчитування поточної температури з датчика DS18B20;

– обчислення помилки між поточною температурою та заданою температурою;

– обчислення пропорційної, інтегральної та диференціальної складових на

основі поточної помилки;

– формування керуючого сигналу для обігрівача або системи охолодження;

– корекція температури відповідно до керуючого сигналу.

Такий підхід дозволяє забезпечити точне та стабільне управління температурою у теплиці, що є критично важливим для оптимального росту рослин. Застосування теорії автоматичного управління у розробці програмної системи для контролю параметрів культивування рослин дозволяє значно підвищити ефективність і точність управління умовами росту рослин. Використання сучасних алгоритмів регулювання, таких як ПІД-регулятори, забезпечує оптимальні умови для вирощування рослин у закритих приміщеннях, що сприяє підвищенню врожайності та якості продукції.

3 РОЗРОБКА ПРОГРАМНОЇ СИСТЕМИ

3.1 Проєктування архітектури програмної системи

Під час розробки програмної системи та з урахуванням практичних знань з управління базами даних було обрано реляційну базу даних MS SQL Server. Для серверної частини програми використовувався фреймворк .NET з мовою програмування C#. Був використаний модуль ESP8266 Wi-Fi, який використовував доступ до Інтернету та мережевий протокол MQTT для надсилання даних брокеру, серверна частина якого підписана на отримання даних від датчика.

Модуль був підключений до датчика DS18B20 для вимірювання температури та датчика DHT22 для вимірювання вологості. У якості брокера використовувався HiveMQ cloud. Веб-частина застосунку була створена за допомогою фреймворку Angular, який використовує мову програмування TypeScript. На рисунку 3.1 наведено UML-діаграму взаємодії.

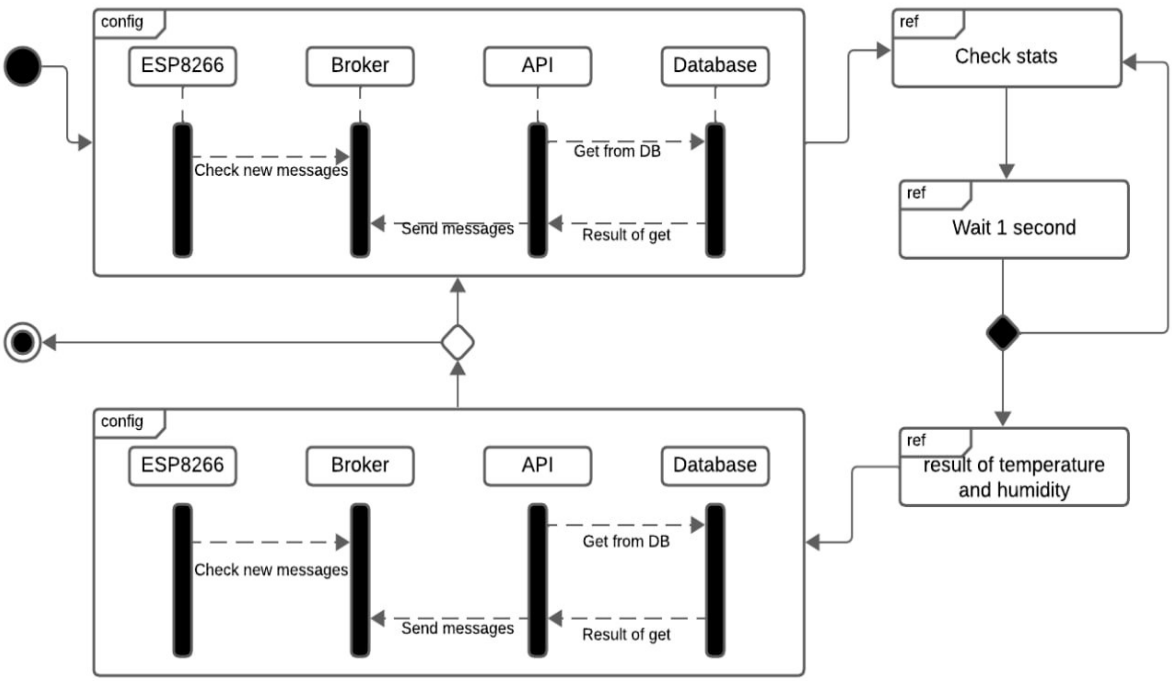

Рисунок 3.1 – UML діаграма взаємодії компонентів програмної системи

Діаграма взаємодії UML, представлена на рисунку 3.1, описує процес взаємодії компонентів системи для збору та перевірки даних про температуру та вологість. Процес починається з початкової точки. Далі виконується конфігурація системи, яка складається з чотирьох компонентів: ESP8266, Broker, API та Database.

Спочатку модуль ESP8266 перевіряє нові повідомлення. Після цього Broker отримує ці повідомлення від ESP8266 і надсилає їх до API. API, у свою чергу, отримує повідомлення від Broker і робить запит до бази даних (Database). База даних повертає результат запиту до API.

Після отримання результату від бази даних процес переходить до блоку "Check stats", де перевіряються статистичні дані. Після цього процес переходить до блоку "Wait 1 second", де очікує одну секунду перед тим, як продовжити.

Після очікування процес переходить до наступного блоку, який отримує результат температури та вологості. Далі виконується друга конфігурація системи, аналогічна першій. Вона також складається з компонентів ESP8266, Broker, API та Database. ESP8266 перевіряє нові повідомлення, Broker отримує ці повідомлення та надсилає їх до API, який робить запит до бази даних, а Database повертає результат запиту. Процес завершується кінцевою точкою.

3.2 Вибір інтегрованого середовища розробки

Visual Studio є інтегрованим середовищем розробки (IDE), яке використовувалося для розробки проєкту. Visual Studio забезпечує зручний інтерфейс для написання, відлагодження та тестування коду для різних частин системи, зокрема:

– серверна частина: Visual Studio використовувалось для написання коду на C# з використанням фреймворку .NET;

– робота з базою даних: Visual Studio використали для управління MS SQL Server, включаючи створення та модифікацію бази даних, написання SQL-запитів та скриптів;

– інтеграція з іншими компонентами: Visual Studio дозволяє інтегрувати код для роботи з модулем ESP8266 WiFi та протоколом MQTT.

Таким чином, Visual Studio є ключовим інструментом у проєкті для розробки, тестування та відлагодження різних компонентів системи.

3.3 Збір даних з датчиків

3.3.1 Ініціалізація та налаштування датчиків

На рисунку 3.2 наведено приклад, на якому визначаємо початкові параметри для симуляції вимірювання даних.

```
9     var start = new DateTime(2024, 5, 15);
10    var step = TimeSpan.FromMinutes(1);
11
12    var devices = new List<Farm>()
13    {
14        new(5),
15        new(6)
16
17    };
18
19    var measurementConfigs = new List<MeasurementConfig>()
20    {
21        new MeasurementConfig(MeasurementType.Temperature, 15, 35),
22        new MeasurementConfig(MeasurementType.Humidity, 30, 70)
23    };
```

Рисунок 3.2 – Приклад ініціалізації та налаштування

Пояснення до частини коду з рисунку 3.2:

– “start” встановлює початковий час для симуляції;

– “step” крок часу, з яким будуть генеруватися нові дані, на прикладі він дорівнює одній хвилині;

– “devices” створює список пристроїв (ферм) з ID 5 і 6;

– “measurementConfigs” створює список конфігурацій вимірювань для температури (від 15 °C до 35 °C) та вологості (від 30 % до 70 %).

3.3.2 Підготовка сервісів для датчиків

Використовуючи контейнер залежностей, отримуємо необхідні сервіси для роботи. На рисунку 3.3 наведено приклад продовження коду.

```
25      var serviceProvider = ServiceProviderBuilder.BuildServiceProvider();
26      var mqttService = serviceProvider.GetRequiredService<IMqttService>();
27      var options = serviceProvider.GetRequiredService<IOptions<ProjectOptions>>();
28
29      var mqttClient = await mqttService.CreateClientAsync();
```

Рисунок 3.3 – Приклад підготовки сервісів

Пояснення до частини коду з рисунку 3.3:

– “serviceProvider” створює контейнер залежностей;

– “mqttService” отримує сервіс для роботи з MQTT;

– “options” отримує налаштування проєкту;

– “mqttClient” створює MQTT клієнт для підключення до брокера.

3.3.3 Основний цикл симуляції датчиків

Основний цикл симуляції відповідає за генерацію та відправку даних з датчиків. На рисунку 3.4 наведено приклад продовження коду.

```
30    var currentPosition = start;
31
32    while (true)
33    {
34        foreach (var device in devices)
35        {
36            foreach (var measurementConfig in measurementConfigs)
37            {
38                var mesurement = measurementConfig.CreateValue(currentPosition);
39                string message = JsonSerializer.Serialize(mesurement);
40                await mqttService.PublishAsync(mqttClient, device.CreateTopik(options.Value.Name, c => c.Id), message);
41            }
42        }
43
44        if (currentPosition.Add(step) > DateTime.Now)
45        {
46            var wait = DateTime.Now - currentPosition.Add(step);
47            await Task.Delay(wait);
48        }
49        else
50        {
51            await Task.Delay(TimeSpan.FromSeconds(30));
52        }
53
54        currentPosition = currentPosition.Add(step);
55    }
```

Рисунок 3.4 – Приклад основного циклу симуляції

Пояснення до частини коду з рисунку 3.4:

– “currentPosition” відслідковує поточну позицію часу в симуляції;

– “while (true)” безкінечний цикл, що забезпечує безперервну роботу симуляції.

Збір та обробка даних для кожного пристрою і кожної конфігурації вимірювань виконується наступним чином:

– викликається метод “CreateValue” для генерації значення вимірювання на основі поточного часу (currentPosition);

– вимірювання серіалізується у формат JSON;

– використовується MQTT сервіс для відправки даних до брокера.

Дані передаються через MQTT, а саме: “mqttService.PublishAsync” відправляє серіалізоване повідомлення на відповідну тему, яку створює метод “device.CreateTopik”.

Контроль часу відбувається наступним чином: перевіряється, чи не випереджає симуляція реальний час. Якщо так, програма чекає до досягнення наступного кроку. Якщо симуляція відстає, вона чекає 30 секунд перед наступним циклом.

3.3.4 Узагальнення роботи датчиків програмної системи

Процес збору даних з датчиків та їх передача до центральної системи виглядає наступним чином:

– ініціалізація: встановлюються початкові часові параметри і створюється список пристроїв і конфігурацій вимірювань;

– підключення до сервісів: створюється контейнер залежностей отримуються потрібні сервіси та клієнт для роботи з MQTT;

– генерація даних: для кожного пристрою і для кожного типу вимірювань генерується значення на основі поточного часу;

– серіалізація та передача: вимірювання серіалізуються у форматі JSON та надсилаються через MQTT на відповідні теми;

– синхронізація за часом: перевіряється відповідність часу моделювання і реального часу, і відповідним чином коригується затримка між ітераціями циклу.

3.4 Обробка та зберігання даних

Для забезпечення обробки отриманих даних на серверній частині використовується платформа .NET. Серверна частина відповідає за прийом даних, що надходять з FrontEnd частини та через MQTT-протокол, а також їх подальшу обробку. Виходячи з характеру даних, отриманих від смарт-пристроїв та FrontEnd частини, сервер може виконувати наступні завдання:

– валідація даних: перевірка коректності та цілісності отриманих даних;

– зберігання даних: об’єднання даних з різних джерел для створення більш повної картини;

– шифрування даних: шифрування паролів, щоб покращити безпеку користувачів.

Обробка даних на сервері забезпечує швидку та ефективну реакцію на отримані дані, що дозволяє підтримувати високий рівень контролю параметрів культивування рослин.

Для зберігання оброблених даних у системі використовується реляційна база даних MS SQL Server. Використання MS SQL Server надає наступні переваги:

– надійність і цілісність даних. MS SQL Server забезпечує високу надійність і цілісність збережених даних завдяки підтримці транзакцій та механізмів відновлення даних;

– масштабованість. База даних може легко масштабуватися відповідно до зростання обсягів даних та кількості запитів;

– безпека даних. MS SQL Server надає розширені можливості для забезпечення безпеки даних, включаючи шифрування, контроль доступу та аудит;

– аналітика та звітність. Збережені дані можуть використовуватися для подальшого аналізу та створення звітів, що допомагає користувачам приймати обґрунтовані рішення.

Процес зберігання даних включає кілька етапів:

– інсерція даних. Оброблені дані записуються в таблиці бази даних;

– індексація. Створення індексів для швидкого пошуку та доступу до даних;

– архівація. Переміщення старих даних в архівні таблиці або окремі бази даних для збереження продуктивності системи.

Завдяки обробці даних на сервері та їх зберіганню у MS SQL Server, система забезпечує надійну і ефективну роботу, а також можливість подальшого аналізу та звітності на основі зібраних даних.

3.5 Розробка мобільного застосунку

Для розробки мобільного застосунку була використана технологія .NET MAUI (Multi-platform App UI) [17], яка дозволяє створювати кросплатформені застосунки для Android, iOS, Windows та macOS. Застосунок було написано мовою програмування C# у середовищі розробки Visual Studio 2022. Взаємодія з серверною частиною системи здійснюється через HTTP протокол, а дані передаються у форматі JSON.

Для відображення робочих компонентів клієнтської частини системи та відображення логіки їх взаємодії та інженерних рішень під час проєктування було створено діаграму компонентів мобільного застосунку (рис. 3.5). Ця діаграма ілюструє структуру мобільного застосунку та взаємодію між його компонентами.

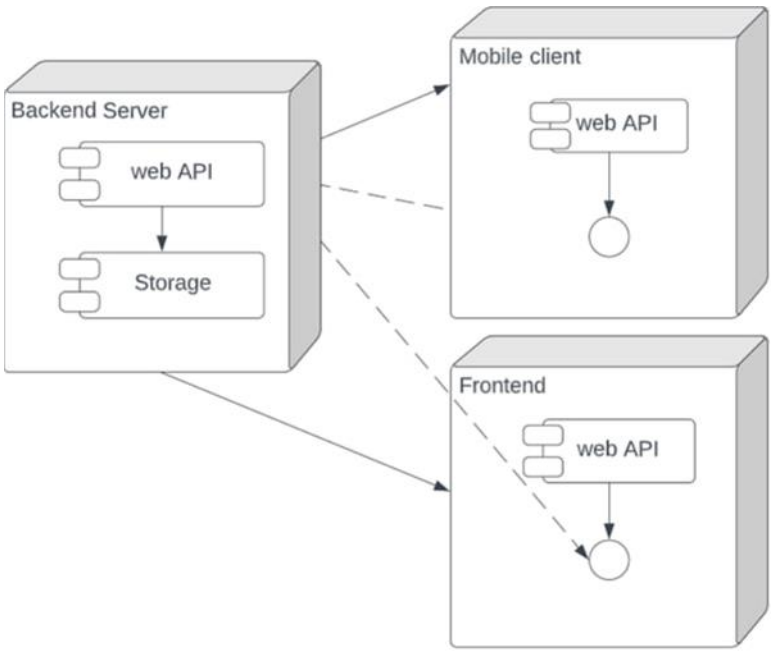

Рисунок 3.5 – Діаграма компонентів програмної системи

Застосунок складається з наступних основних компонентів:

– Backend Server: містить веб API та систему зберігання даних. Він забезпечує всю серверну логіку та обробку запитів від клієнтських застосунків;

– Frontend: клієнтська частина, яка взаємодіє з користувачем через веб API;

– Mobile Client: мобільний клієнт, який також взаємодіє з сервером через веб API.

Мобільний застосунок має одного ключового користувача – адміністратора, який може переглядати свій профіль, всі наявні ферми в системі, а також всі наявні шаблони в системі. Оскільки мобільний застосунок реалізований на базі Android-платформи з використанням технології .NET MAUI та мови програмування C#, усі компоненти мобільного застосунку реалізовані у вигляді сторінок Blazor [18]. Це дозволяє ефективно організувати код, полегшує підтримку та розширення функціональності застосунку.

Blazor-сторінки дозволяють створювати інтерактивні веб-інтерфейси з використанням C# замість JavaScript. Це сприяє більшій продуктивності розробки та підтримці коду, оскільки використовується одна мова програмування для клієнтської та серверної частин.

3.6 Безпека системи

Для реалізації рівнів доступу, авторизації була використана бібліотека System.Security.Cryptography, за допомогою цієї бібліотеки було реалізовано:

– механізм аутентифікації, який перевіряє ідентичність користувача перед наданням доступу до системи;

– використання безпечних методів аутентифікації, таких як введення пошти та пароля користувача;

– управління правами доступу до різних функціональних можливостей системи на основі ролей користувачів;

– налаштування обмежень доступу до конкретних даних чи функцій системи залежно від рівня привілеїв користувача.

Крім того, при введенні пароля, були встановлені певні вимоги щодо його складності. Зокрема, паролі повинні містити принаймні один небуквенно-цифровий символ, такий як символи пунктуації або спеціальні символи.

Також, вимагалося, щоб паролі містили принаймні одну літеру нижнього регістру (від 'a' до 'z') і одну літеру верхнього регістру (від 'A' до 'Z'). Це сприяє підвищенню безпеки паролів та ускладнює їхнє вгадування.

Такі перевірки забезпечують, що користувачі створюють міцні паролі, що допомагає уникнути несанкціонованого доступу до їх облікових записів і забезпечує безпеку системи.

Шифрування даних використовується для захисту конфіденційної інформації під час її передачі по мережі. Для цього може використовуватися протокол HTTPS, який забезпечує шифрування даних за допомогою TLS/SSL.

Ці заходи безпеки допомагають забезпечити високий рівень захисту даних у системі контролю параметрів культивування рослин, зменшуючи ризик несанкціонованого доступу, витоку інформації та інших загроз безпеці.

3.7 Аналіз питань охорони праці під час роботи з програмною системою

Під час розробки програмної системи для контролю параметрів культивування рослин у закритому просторі необхідно врахувати можливі небезпеки та ризики, які можуть виникнути під час монтажу, експлуатації та обслуговування системи [19]. Основні небезпеки включають:

– електричні ризики: можливість ураження електричним струмом при роботі з електрообладнанням;

– технічні ризики: можливі несправності обладнання, які можуть призвести до травм;

– фізичні ризики: можливість отримання травм при роботі з обладнанням.

Засоби уникнення небезпек:

– електричні небезпеки: перевірка електрообладнання перед роботою, використання заземленого обладнання та захисних пристроїв, освітлення і позначення небезпечних зон;

– технічні небезпеки: регулярна перевірка та обслуговування обладнання, забезпечення відповідної інструкції з експлуатації та безпеки, проведення навчання персоналу з правильного використання обладнання;

– фізичні небезпеки: навчання персоналу правильним методам роботи з обладнанням, регулярні перевірки на дотримання правил безпеки та вжиття заходів для їх поліпшення [20].

4 РЕЗУЛЬТАТ РОЗРОБКИ ПРОГРАМНОЇ СИСТЕМИ

4.1 Візуалізація даних

4.1.1 Розробка веб-інтерфейсу

Перед початком розробки веб-інтерфейсу було побудовано діаграму UseCase для більш зрозумілого представлення того, який функціонал повинен мати інтерфейс. UseCase – це перелік дій, сценарій, за яким користувач взаємодіє з додатком або програмою для виконання будь-якої дії та досягнення конкретної мети [21]. На рисунку 4.1 наведена UseCase-діаграма програмної системи.

Діаграма UseCase показує дії, доступні для трьох типів користувачів: User, Admin та Not Authorized User.

Для користувача з роллю User доступні наступні дії: User може переглядати та редагувати дані в своєму особистому профілі, переглядати та редагувати шаблони, створені ним самим, створювати нові шаблони для вирощування, а також переглядати і редагувати шаблони, які він вже використовував. Крім того, User може переглядати список всіх доступних шаблонів і створювати нові шаблони на основі існуючих, переглядати та редагувати свої ферми, створювати нові ферми, переглядати дані зібрані датчиками на своїх фермах і виходити з системи.

Для користувача з роллю Admin доступні розширені права та додаткові дії: Admin може переглядати та редагувати всі ферми, що є в системі, переглядати, редагувати та створювати шаблони для вирощування, створювати використання шаблонів для користувачів та редагувати наявні використання, переглядати список всіх зареєстрованих користувачів, переглядати та редагувати свої особисті дані і виходити з системи.

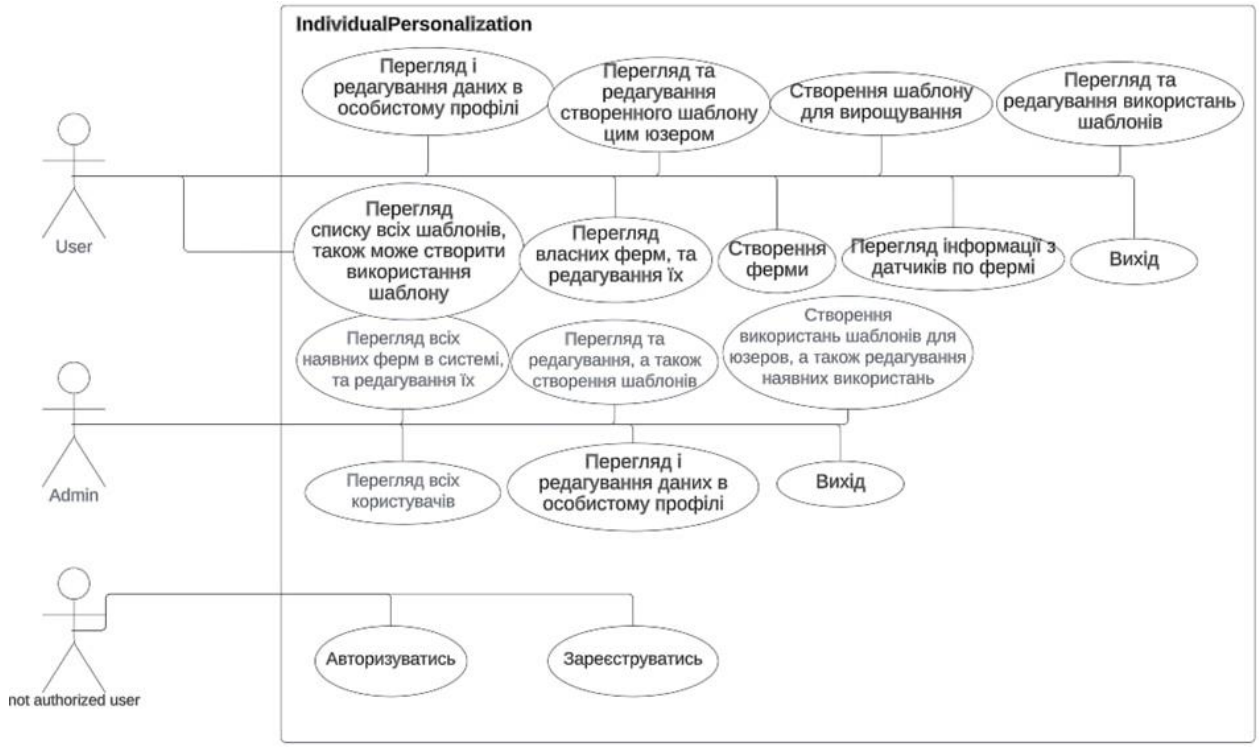

Рисунок 4.1 – UseCase-діаграма програмної системи

Неавторизований користувач (Not Authorized User) має доступ до двох основних дій: він може авторизуватись у системі, якщо у нього є обліковий запис, або зареєструватись, створивши новий обліковий запис.

Для зручності розробки та розуміння як повинна виглядати Front End частина була побудована діаграма діяльності яка демонструє, які кроки повинен пройти користувач для того щоб скористатися програмною системою (рис. 4.2)

Діаграма діяльності описує процес взаємодії користувача з системою, починаючи з етапу авторизації або реєстрації. Спочатку користувач вводить свої дані, які сервер отримує та перевіряє на коректність. Якщо введені дані неправильні, процес завершується. Якщо дані правильні, проводиться перевірка ролі користувача. Якщо користувач не є адміністратором, йому надається можливість переглядати інформацію про свою ферму або переглядати свою систему.

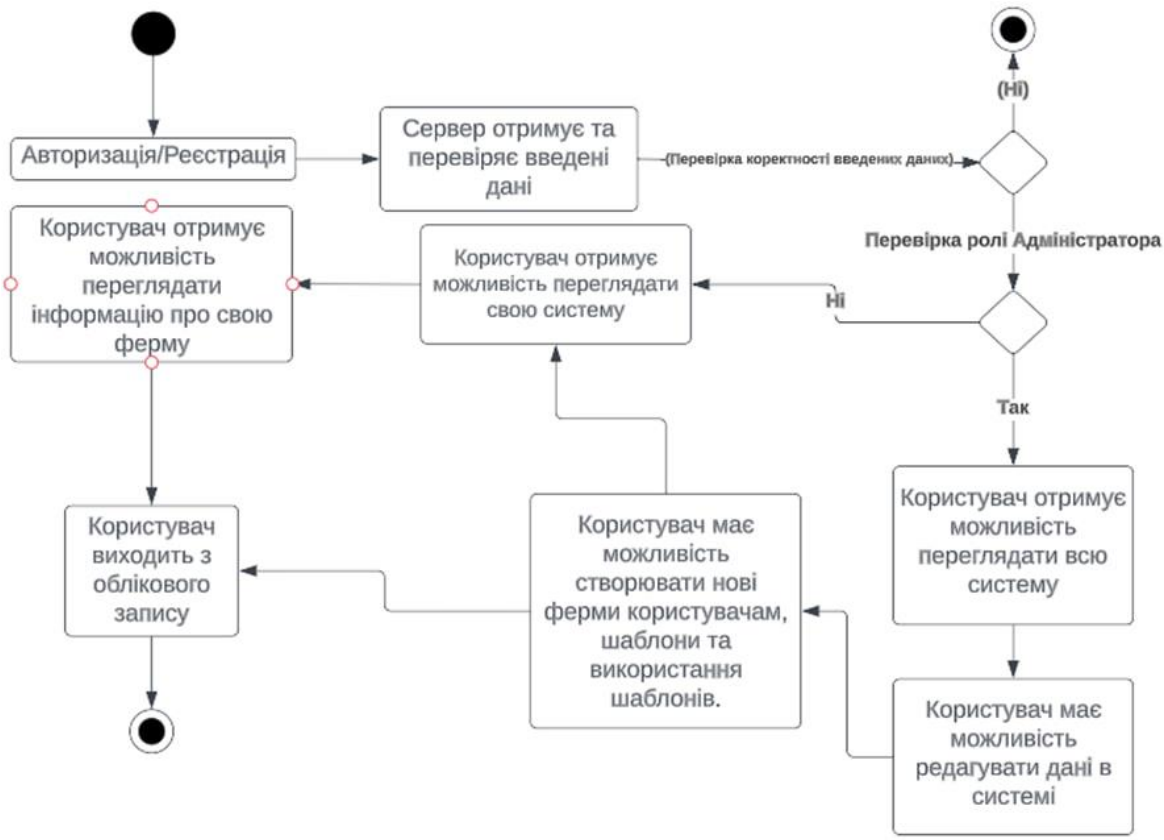

Рисунок 4.2 – Діаграма Діяльності

Далі користувач може створювати нові ферми, шаблони та використання шаблонів. Він також може переглядати інформацію про свою ферму. Після завершення дій користувач може вийти з облікового запису, що завершує процес.

Якщо перевірка ролі показує, що користувач є адміністратором, йому надається можливість переглядати всю систему.

Адміністратор може редагувати дані в системі, а також створювати нові ферми для користувачів, шаблони та використання шаблонів. Після завершення всіх необхідних дій адміністратор може також вийти з облікового запису, завершуючи процес.

Таким чином, діаграма показує різні сценарії використання системи залежно від ролі користувача та дій, які він може виконувати.

Для створення веб-інтерфейсу використовується фреймворк Angular, який дозволяє створювати динамічні односторінкові додатки (SPA). Angular забезпечує модульну структуру, двосторонню прив’язку даних, компоненти та служби, що значно спрощує розробку і підтримку інтерфейсів.

Приклад розробленого інтерфейсу сторінки “Ферми” наведено на рисунку 4.3.

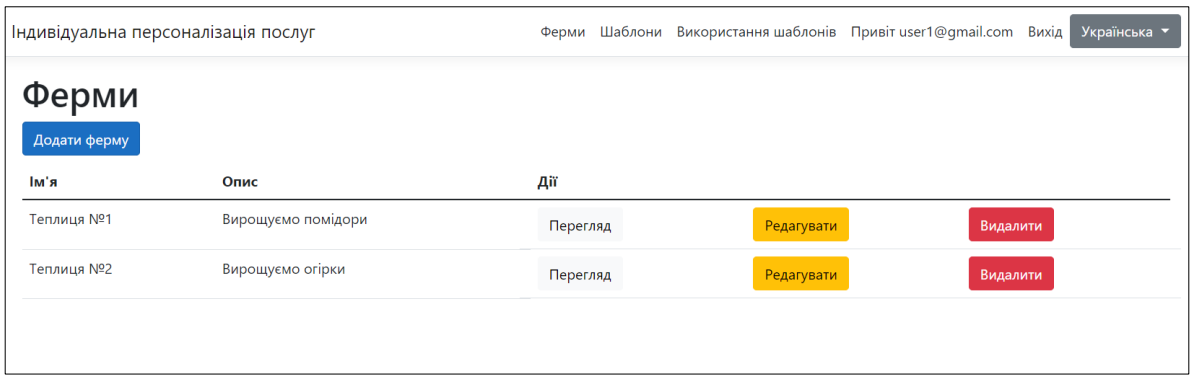

Рисунок 4.3 – Приклад розробленого інтерфейсу сторінки “Ферми”

На сторінці “Ферми” користувач може переглядати свої ферми та інформацію про кожну з них окремо, створювати нові, редагувати та видаляти існуючі.

Також інтерфейс підтримує англійську мову, та є змога зручного переключення на неї. Нижче на рисунку 4.4 наведено приклад ідентичного інтерфейсу, але англійською мовою.

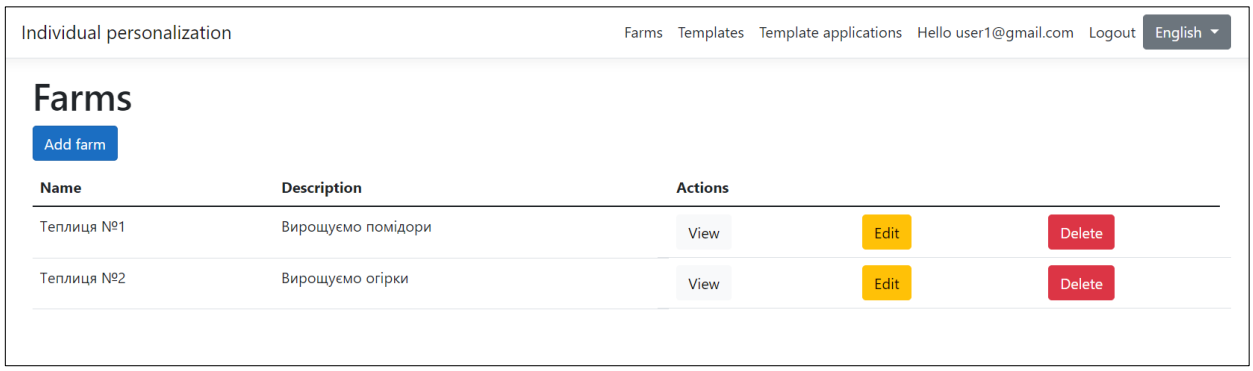

Рисунок 4.4 – Приклад розробленого інтерфейсу англійською мовою у вікні “Ферми”

На сторінці ферм користувача при натисканні на кнопку “Перегляд” відкривається повна інформація про ферму, на якій можемо бачити її назву, опис та дані, які були виміряні датчиками, а саме: температура і вологість та відображені на графіках, детально цю сторінку відображено на рисунку 4.5.

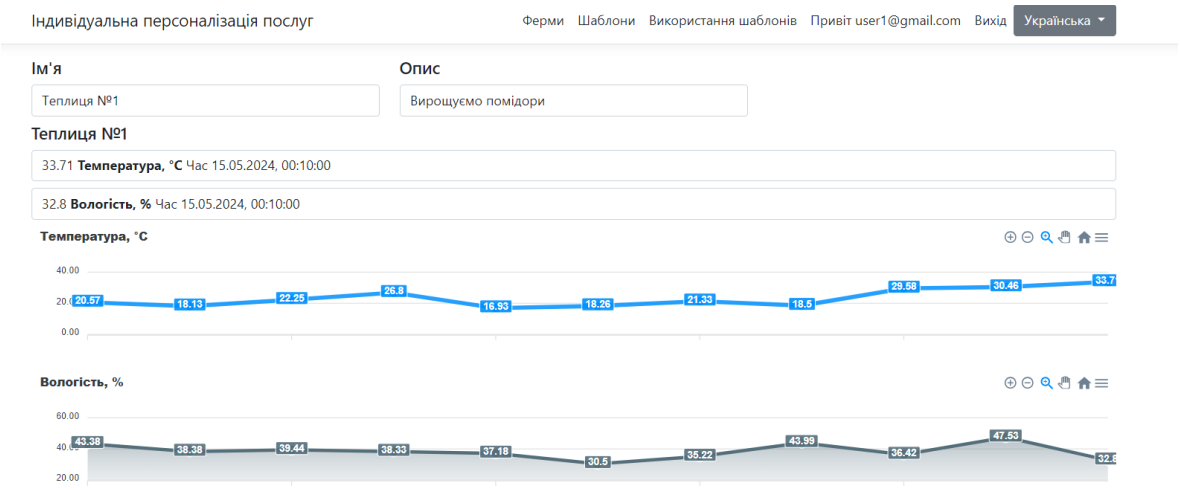

Рисунок 4.5 – Приклад розробленого інтерфейсу

Сторінка “Шаблони” розроблена для того, щоб користувачі могли самостійно створити або використати вже існуючі шаблони налаштувань системи для вирощування рослин. Користувач має змогу створювати нові шаблони, переглядати існуючі, редагувати свої шаблони, а також видаляти їх. Приклад сторінки “Шаблони” представлений на рисунку 4.6.

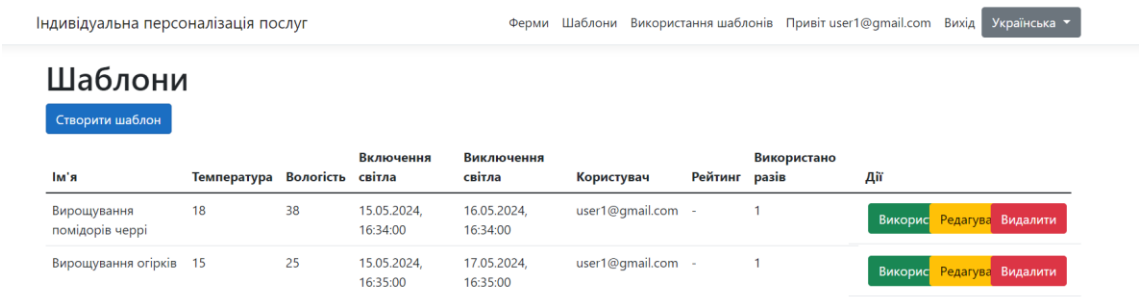

Рисунок 4.6 – Приклад розробленого інтерфейсу сторінки “Шаблони”

Сторінка “Використання шаблонів” розроблена для того, щоб користувач міг застосувати створені шаблони до своїх ферм. Користувач має змогу переглядати всі свою використання шаблонів, створювати використання шаблонів, редагувати вже існуючі та видаляти їх. Приклад сторінки “Використання шаблонів” представлений на рисунку 4.7.

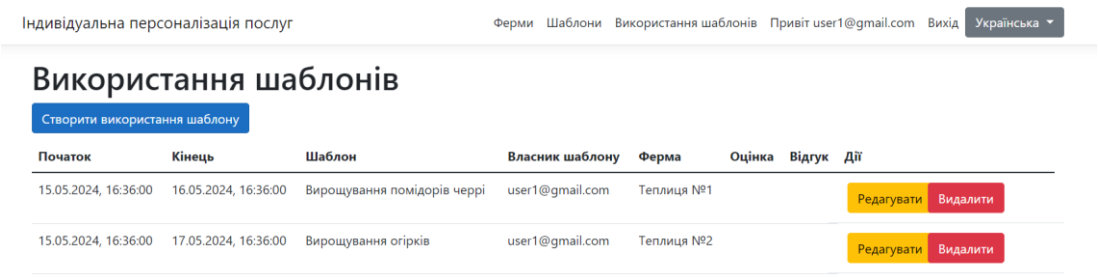

Рисунок 4.7 – Приклад розробленого інтерфейсу сторінки “Використання шаблонів”

На сторінці “Профілю користувача” користувач може переглядати інформацію про свій особистий профіль, додавати свій номер телефону та переглянути свою пошту (рис. 4.8).

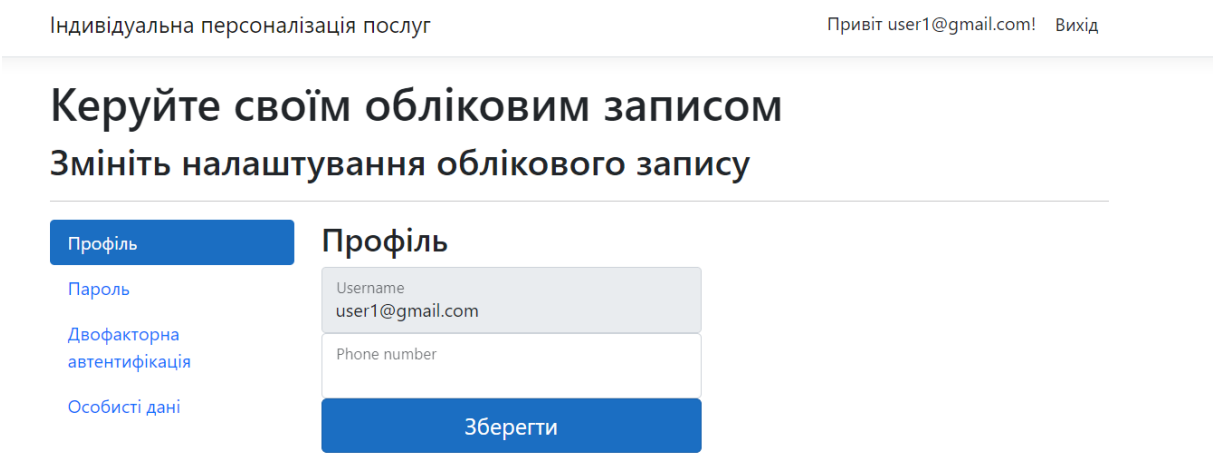

Рисунок 4.8 – Приклад розробленого інтерфейсу сторінки “Профіль користувача”

На сторінці “Профілю користувача” у вкладці пароль користувач може змінити свій пароль на інший за допомогою введення свого поточного паролю та написання нового (рис. 4.9).

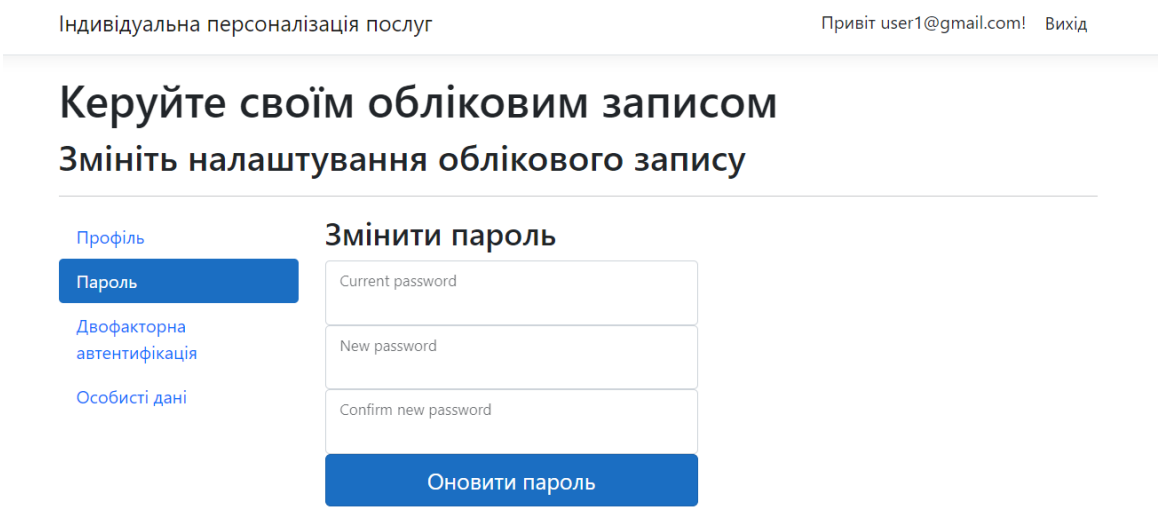

Рисунок 4.9 – Приклад розробленого інтерфейсу сторінки “Пароль”

На сторінці “Профіль користувача” користувач має змогу видалити свій обліковий запис, для чого йому потрібно буде написати пароль від облікового запису (рис. 4.10).

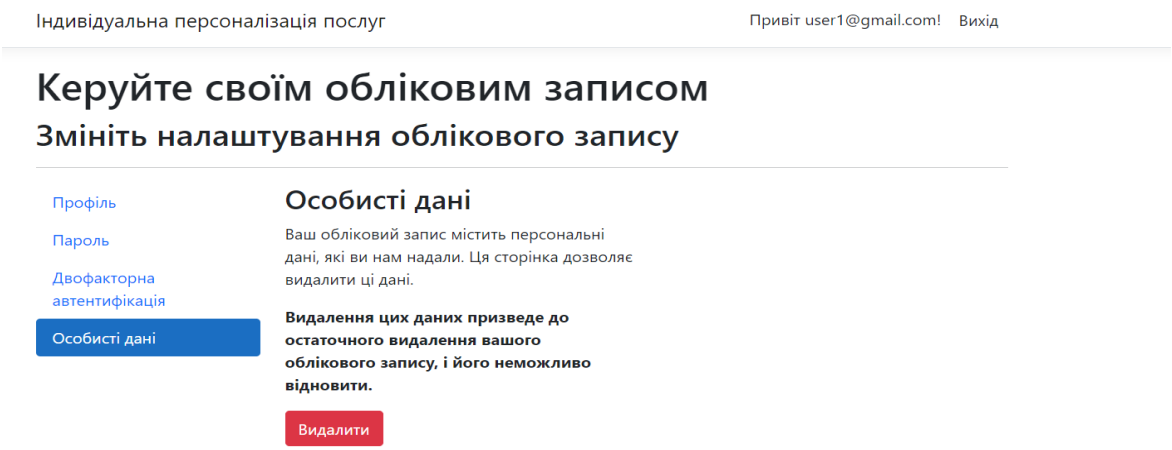

Рисунок 4.10 – Приклад розробленого інтерфейсу сторінки

На сторінці “Профілю користувача” у вкладці двофакторна автентифікація, на якій користувач може прив’язати свій акаунт до певної програми, наприклад “Google Authenticator” (рис.4.11)[22].

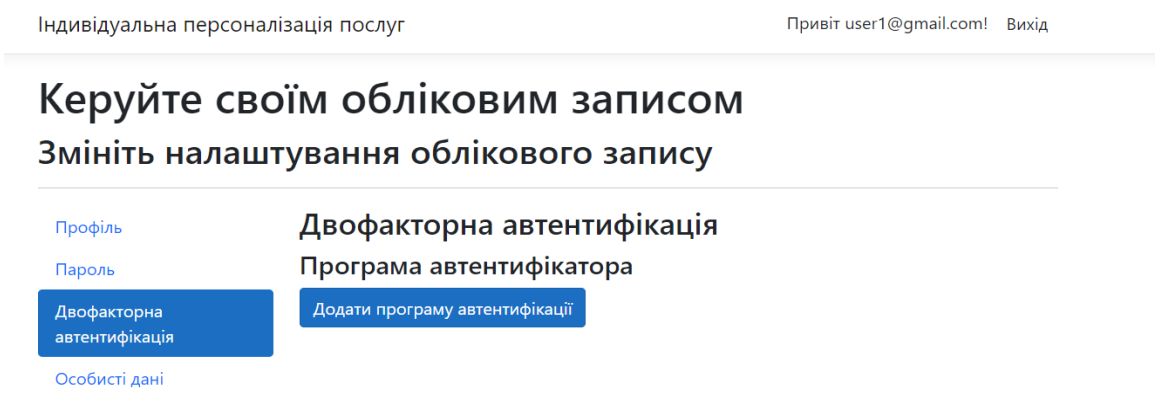

Рисунок 4.11 – Приклад розробленого інтерфейсу сторінки

Нижче на рисунку 4.12 наведено процес додавання двофакторної автентифікації до акаунту шляхом перенесення згенерованого ключа у застосунок двофакторної автентифікації та підтвердження цього згенерованим шестизначним кодом, який згенерує застосунок для автентифікації.

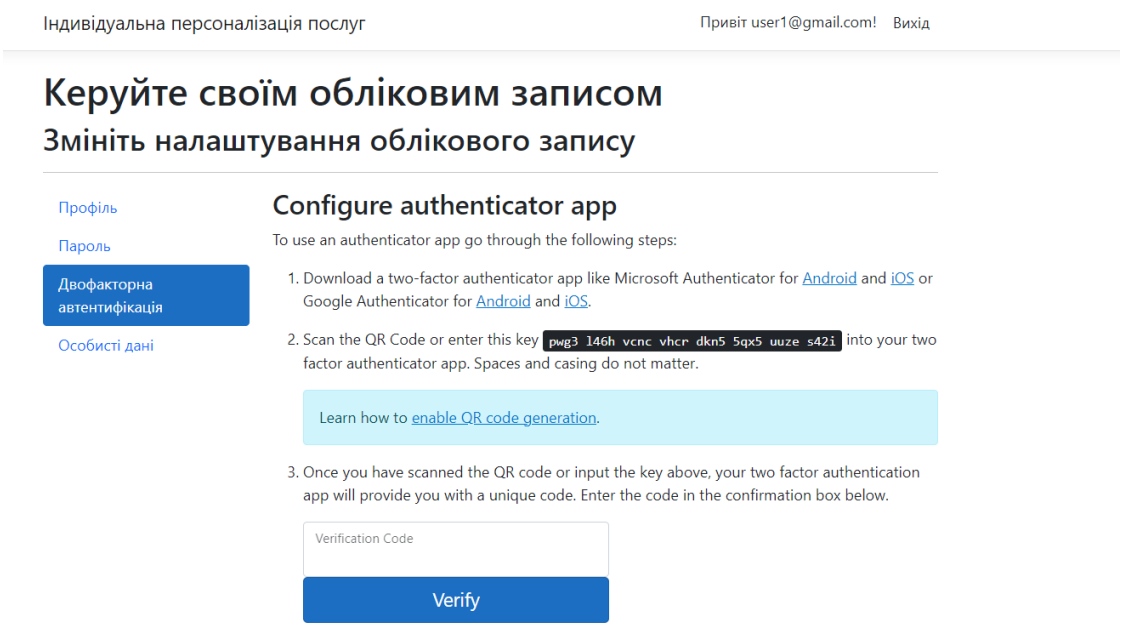

Рисунок 4.12 – Приклад розробленого інтерфейсу сторінки “Профіль користувача вкладка двофакторна автентифікація”

Приклад як виглядають згенеровані шестизначні коди у застосунку “Google Authenticator” (рис.4.13).

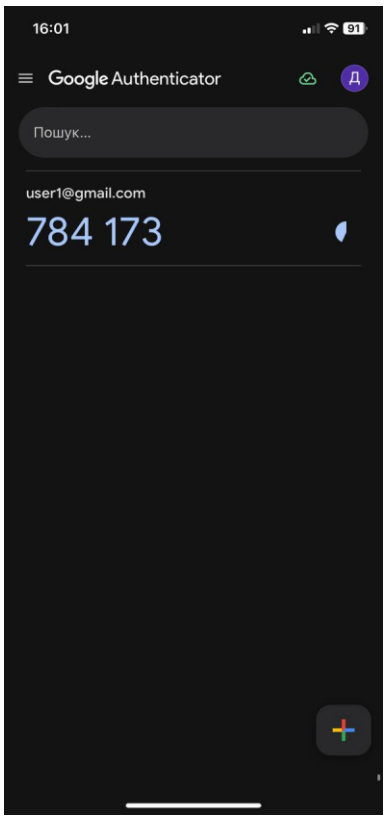

Рисунок 4.13 – Згенерований шестизначний код

Після вірного введення шестизначного коду відображається сторінка з успішним підтвердженням того, що двофакторна автентифікація була додана та представлені згенеровані ключи відновлення доступу, якщо користувач загубить доступ до свого застосунку “Google Authenticator” (рис. 4.14).

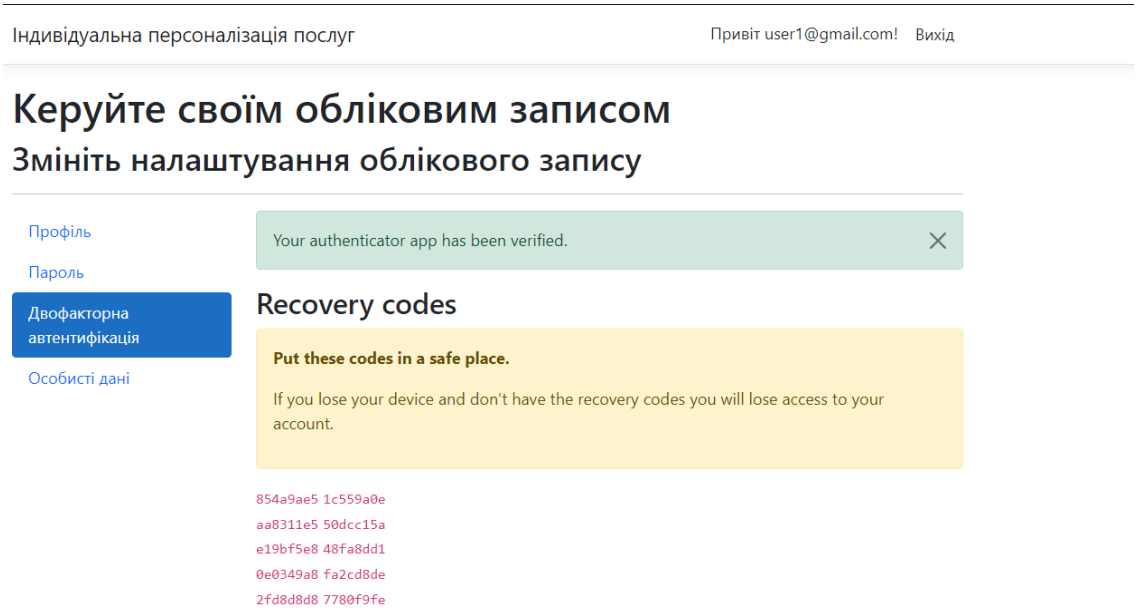

Рисунок 4.14 – Приклад розробленого інтерфейсу сторінки “Двофакторна автентифікація”

Сторінка “Реєстрація” розроблена для того, щоб нові користувачі могли створити собі акаунт у програмній системі, на сторінці додана перевірка на правильність введених даних, а саме, щоб у полі пошти користувач обов’язково вказав валідну адресу та ввів складний пароль, який включає в себе мінімум 6 символів, принаймні: один небуквенно-цифровий символ, один нижній регістр ('a'-'z'), один верхній регістр ('A'-'Z') (рис. 4.15).

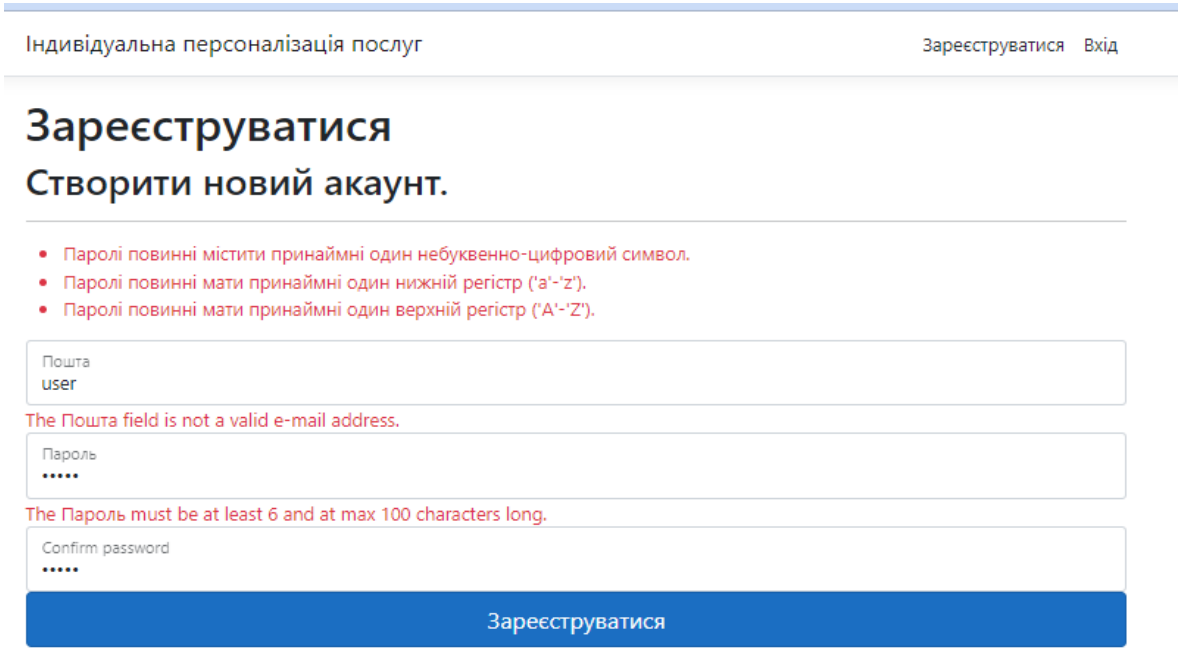

Рисунок 4.15 – Приклад розробленого інтерфейсу сторінки “Реєстрація”

Сторінка авторизації у застосунок створена для того, щоб користувач міг авторизуватися у свій профіль (рис. 4.16).

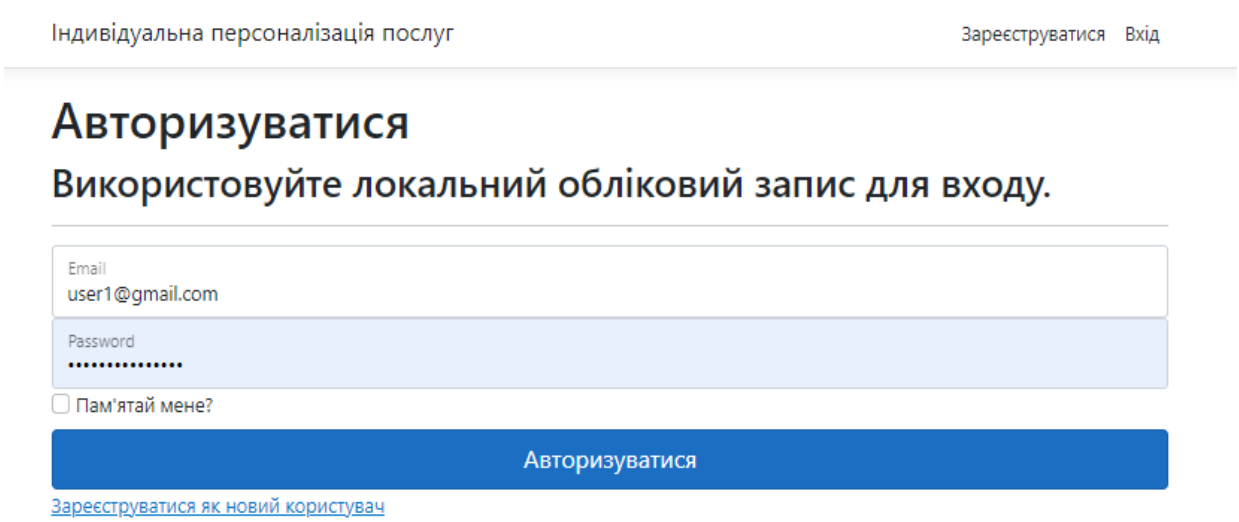

Рисунок 4.16 – Приклад розробленого інтерфейсу сторінки “Авторизація”

Коли користувач вводить дані від свого аккаунту та натискає на кнопку “Авторизуватися”, дані відправляються на сервер, і якщо такий користувач вже зареєстрований, то його перенаправляє на головну сторінку програмної системи, або якщо до його аккаунту прив’язана двофакторна автентифікація, то перенаправляє на сторінку введення двофакторного коду із застосунку “Google Authenticator”, приклад цієї сторінки наведено на рисунку 4.17.

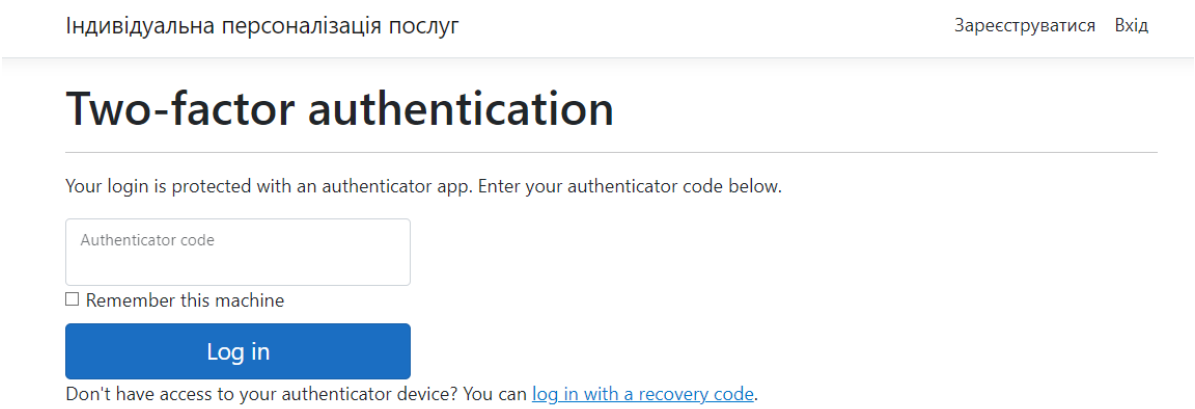

Рисунок 4.17 – Приклад розробленого інтерфейсу сторінки “Двофакторна автентифікація

Після успішної автентифікації користувач отримує повний доступ до програмної системи та може керувати нею.

4.1.2 Архітектура веб-інтерфейсу

Angular додаток складається з компонентів, кожен з яких відповідає за певну частину інтерфейсу. Наприклад, компонент для відображення списку ферм, шаблонів, використання шаблонів, профілю користувача та перегляд інформації ферми, на якій відображається температура, вологість і графіки по зміні температури та вологості за проміжок часу.

Сервіси використовуються для отримання даних з сервера, обробки цих даних і надання їх компонентам. Наприклад, сервіс для підключення до MQTT брокера, сервіс для отримання даних з бази даних. На рисунку 4.18 відображені сервіси, які використовуються в програмній системі.

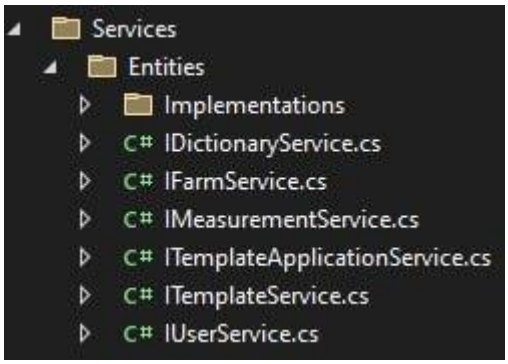

Рисунок 4.18 – Складові Services Angular

Angular-додаток складається з різних модулів, що дозволяє розділяти функціональність на логічні блоки. Це покращує організацію коду і полегшує його масштабування.

4.1.3 Реалізація реального часу

Для відображення даних у режимі реального часу використовуються вебсокети або інші технології, що підтримують постійний зв’язок між сервером і клієнтом. Angular має бібліотеки та інструменти для роботи з вебсокетами, що дозволяє безперервно отримувати і відображати оновлення даних без перезавантаження сторінки.

Процес візуалізації даних для користувачів в системі контролю параметрів культивування рослин у закритому просторі включає створення інтуїтивного і зручного веб-інтерфейсу за допомогою Angular. Веб-інтерфейс забезпечує відображення даних у режимі реального часу, використовуючи інтерактивні графічні елементи, наприклад графіки, що допомагають користувачам швидко і ефективно аналізувати стан системи та приймати обґрунтовані рішення.

4.2 Розрахунки параметрів програмної системи

Розрахунки включають оцінку продуктивності системи, вимоги до апаратного забезпечення, навантаження на мережу, використання пам’яті та

процесорних ресурсів.

Основні показники продуктивності програмної системи включають:

– час відгуку системи;

– пропускна здатність;

– стійкість до навантажень.

Час відгуку системи повинен бути не більше 1 с для забезпечення реального часу контролю параметрів. Програмна система повинна обробляти до 1000 запитів за хвилину, що включає дані від сенсорів та команди управління. Система повинна витримувати пікові навантаження без значного зниження продуктивності.

Для ефективної роботи програмної системи необхідно використовувати апаратне забезпечення, яке наведено у таблиці 4.1.

Таблиця 4.1 – Апаратне забезпечення та вимоги до нього

Компонент	Вимоги
Сервер	2 процесори, 16 ГБ ОЗП, 500 ГБ SSD
Контролери	ARM Cortex-M4, 256 КБ ОЗП
Сенсори	Цифрові сенсори температури та вологості

Обсяг даних, що передається від сенсорів до серверу, становить приблизно 10 Мб на годину за частоти оновлення 1 раз на хвилину для кожного сенсора.

Пропускна здатність мережі повинна бути не менше 1 Мбіт/с для забезпечення стабільного з’єднання та обміну даними у реальному часі.

Середнє використання пам’яті на сервері становить близько 8 Гб з піковими навантаженнями до 12 Гб. Процесорне навантаження при цьому становить близько 60%.

Контролери використовують приблизно 50% наявної пам’яті та 70% процесорних ресурсів за максимального навантаження.

Програмна система для контролю параметрів культивування рослин у закритому просторі розроблена з урахуванням високих вимог до продуктивності, надійності та стабільності роботи. Система здатна обробляти велику кількість запитів, витримувати пікові навантаження та забезпечувати своєчасну передачу даних у реальному часі. Виконані розрахунки підтверджують, що обрана архітектура та апаратне забезпечення повністю відповідають поставленим завданням і забезпечують необхідний рівень продуктивності та надійності.

ВИСНОВКИ

У рамках кваліфікаційної роботи була розроблена програмна система для контролю параметрів культивування рослин у закритому просторі, що сприяє підвищенню ефективності та якості вирощування сільськогосподарської продукції. Основні результати роботи включають наступне:

– досліджено сучасні тенденції автоматизації у сільському господарстві, зокрема в галузі культивування рослин у закритих приміщеннях;

– розглянуто приклади автоматизованих теплиць і гроубоксів, що дозволило визначити ключові параметри та функціональні можливості, які повинна мати розроблена система;

– розроблена концепція системи, яка включає чотири основні компоненти: веб-застосунок, емулятор IOT-пристроїв, серверну частину та мобільний додаток;

– створено програмну систему, що дозволяє користувачам створювати власні шаблони для оптимізації процесів вирощування рослин, які можуть бути оцінені та випробувані іншими фермерами;

– проєктування архітектури програмної системи забезпечило надійну роботу всіх компонентів і інтеграцію з датчиками IoT;

– забезпечено збір даних з датчиків, їх обробку та зберігання, а також візуалізацію результатів через веб-інтерфейс;

– система передбачає моніторинг та управління основними параметрами мікроклімату у теплицях, такими як температура, вологість та освітлення;

– проведено аналіз потенційних небезпек і ризиків під час роботи з програмною системою, що дозволяє забезпечити її надійну та безперебійну роботу;

– впроваджено механізми безпеки для захисту даних та забезпечення стабільного функціонування системи;

– запропонована система дозволяє значно підвищити ефективність

вирощування рослин у закритих приміщеннях, забезпечуючи оптимальні умови для росту;

– використання розробленої програмної системи сприяє зниженню витрат на підтримку теплиць і підвищенню врожайності завдяки автоматизації процесів контролю;

– розроблено мобільний застосунок, який дозволяє адміністратору системи зручно переглядати аспекти вирощування рослин з будь-якого місця. Мобільний застосунок створений на базі технології .NET MAUI та мовою програмування C#.

Розроблена програмна система має широкі перспективи для подальшого розвитку та впровадження у сільськогосподарських підприємствах, що прагнуть оптимізувати свої виробничі процеси за допомогою сучасних технологій. У майбутньому планується розширення функціоналу системи та її адаптація до різних умов вирощування рослин, що дозволить забезпечити ще більшу гнучкість та ефективність у сільськогосподарському виробництві.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

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

2. Методичні вказівки з підготовки та захисту кваліфікаційної роботи здобувачами першого (бакалаврського) рівня вищої освіти спеціальності 172 Телекомунікації та радіотехніка, освітньої програми «Інтелектуальні технології засобів радіоелектроніки» / Упоряд. Н. П. Демська, В. В. Євсєєв, О. М. Замірець, В. В. Невлюдова, Ю. М. Олександров. Харків : ХНУРЕ, 2022. – 38 с.

3. Харківський національний університет радіоелектроніки [Електронний ресурс]. – Режим доступу: www/URL: https://nure.ua/department/kafedra-komp-yuterno-integrovanih-tehnologiy avtomatizatsiyi-ta-mehatroniki-kitam.

4. Кафедра комп'ютерно-інтегрованих технологій, автоматизації та мехатроніки (КІТАМ) [Електронний ресурс]. – Режим доступу: www/URL: https://tapr.nure.ua.

5. FarmBot [Електронний ресурс]. – Режим доступу: URL: https://en.wikipedia.org/wiki/FarmBot#:~:text=It%20can%20perform%20almost%20all,seeds%20for%20the%20first%20time.

6. Автоматична промислова теплиця. [Електронний ресурс]. – Режим доступу: URL: https://teplitca.kiev.ua/ua/p631717548-avtomaticheskaya-teplitsa.html.

7. Що таке гроубокс? Ідеї для вирощування та особливості конструкції [Електронний ресурс]. – Режим доступу: URL: https://www.cbdukr.com/2022/08/growbox.html.

8. Гроубокс medium.Grow box стелс [Електронний ресурс]. – Режим доступу: URL: https://growboxua.com.ua/ua/p441525380-grouboks-mediumgrow-box.html

9. DESIGN AND CONSTRUCTION OF PHYTOTRON-GREENHOUSE

COMPLEXES [Електронний ресурс]. – Режим доступу: URL: https://moderntechno.de/index.php/meit/article/view/meit26-02-017/5612

10. ER-діаграма [Електронний ресурс]. – Режим доступу: URL: https://www.guru99.com/uk/er-modeling.html

11. Ms SQL Server [Електронний ресурс]. – Режим доступу: URL:https://learn.microsoft.com/en-us/sql/sql-server/?view=sql-server-ver16.

12. Framework .Net [Електронний ресурс]. – Режим доступу: URL:https://learn.microsoft.com/en-us/dotnet/.

13. Learning Angular – Fourth Edition: A no-nonsense guide to building web applications with Angular by Aristeidis Bampakos (Author), Pablo Deeleman (Author)

14. Learning TypeScript: Enhance Your Web Development Skills Using Type-Safe JavaScript by Josh Goldberg (Author).

15. Mathematics for the Life Sciences Kindle Edition by Erin N. Bodine (Author), Suzanne Lenhart (Author), Louis J. Gross (Author)

16. Невлюдов І. Ш. Теорія автоматичного управління (збірник задач): навчальний посібник / І. Ш. Невлюдов, О. В. Токарєва. Харків: ХНУРЕ, 2020. 240 с.

17. Ye, R. (2023). . NET MAUI Cross-Platform Application Development: Leverage a first-class cross-platform UI framework to build native apps on multiple platforms. Packt Publishing Ltd.

18. Sandberg, E. (2021). Evaluating Blazor : A comparative examination of a web framework (Dissertation). Retrieved from https://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-173239

19. Комплекс навчально-методичного забезпечення навчальної дисципліни "Безпека життєдіяльності. Охорона праці та цивільний захіст" підготовки бакалаврів спеціальності 172 - Телекомунікації та радіотехніка [Електронний ресурс] : спеціалізації "Телекомунікації", "Інфокомунікаційна інженерія"; спеціальності 125 - Кібербезпека, спеціалізація "Управління інформаційною безпекою" / ХНУРЕ ; розроб. І. В. Терещенко. – Харків, 2017. –

125 с.

20. Методичні вказівки до виконання розділу "Охорона праці" у випускних роботах ОКР "бакалавр" усіх форм навчання / упоряд.: В. А. Айвазов. Т. Є. Стиценко., Н. Л. Березуцька ; М-во освіти і науки України, ХНУРЕ. – Харків : ХНУРЕ, 2018. – 28 с. – 1,81

21. Ozkaya, M. (2019). Are the UML modelling tools powerful enough for practitioners? A literature review. IEt software, 13(5), 338-354.

22. Google Authenticator [Електронний ресурс]. – Режим доступу: URL:https://apps.apple.com/ua/app/google-authenticator/id388497605