Міністерство освіти і науки України

Харківський національний університет радіоелектроніки

Факультет __Комп'ютерної інженерії та управління_____
(повна назва)

Кафедра Автоматизації проектування обчислювальної техніки_____
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти____перший (бакалаврський)_____
(рівень вищої освіти)

Пристрій автоматичного поливу домашніх рослин на базі мікроконтролера
(тема)

Виконав:
здобувач ____4_____ року навчання,
групи КІУКІ-21-8

_____Чуркін Д.Ю._____
(прізвище, ініціали)

Спеціальність *123 _Комп'ютерна інженерія*

Тип програми ____освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма_____

_____*Комп'ютерна інженерія* _____
(повна назва освітньої прграми)

Керівник _проф. Литвинова Є.І._
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ _Чумаченко С.В
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет_____Комп'ютерної інженерії та управління _____

Кафедра_____Автоматизації проектування обчислювальної техніки_____

Рівень вищої освіти____перший (бакалаврський)_____

Спеціальність_____123_Комп'ютерна інженерія _____
(шифр і назва)

Тип програми _____Освітньо-професійна_____
(освітньо-професійна або освітньо-наукова)

Освітня програма _____Комп'ютерна інженерія _____
(повна назва)

ЗАТВЕРДЖУЮ:
Зав. кафедри _____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ

НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачу_____Чуркіну Давіду Юлійовичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи (проекту) Пристрій автоматичного поливу домашніх рослин на базі мікроконтролера_____
затверджена наказом по університету від від " 21 "_____05_____2025 р. № 403 Ст_____

2. Термін подання студентом роботи до екзаменаційної комісії _____20.06.2025_____

3. Вихідні дані до роботи (проекту)_____
Мікроконтролер ATmega328_____
Плата Arduino UNO_____
Середовище розробки Arduino IDE_____
Мова програмування C/C++_____
Wi-Fi модуль ESP8266_____
Середовище розробки Android IDE_____

4. Перелік питань, що потрібно опрацювати у роботі _____
Аналіз предметної галузі та постановка задачі проектування
Розробка програмної частини системи
Розробка апаратної частини системи
Програмування мікроконтролерного пристрою
Тестування роботи пристрою

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (слайдів) _____17 слайдів_____

_____

_____

_____

6. Консультанти розділів роботи (проекту)

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

7. Дата видачі завдання 2.05.2025_____

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи (проекту)	Термін виконання етапів проекту (роботи)	Примітка
1	Видача теми проекту, узгодження і затвердження теми	02.05.2025-02.05.2025	
2	Аналіз проблемної галузі, постановка задачі, вибір інструментальних засобів	03.05.2025 -8.05.2025	
3	Розробка електричної схеми пристрою	9.05.2025 -12.05.2025	
4	Реалізація макету пристрою	13.05.2025 -20.05.2025	
5	Розробка програми для мікроконтролера	25.05.2025 -29.05.2025	
6	Тестування системи	30.05.2025 -04.06.2025	
7	Оформлення пояснювальної записки	05.06.2025 -07.06.2025	
8	Перевірка виконаного проекту керівником, допуск до захисту	08.06.2025 -10.06.2025	
9	Захист проекту	16.06.2025 -25.06.2025	

Студент _____
(підпис)

Керівник роботи (проекту) _____ проф.Литвинова Є.І.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи бакалавра: 51 с., 17 рис., 8 джерел.

ОС, АНДРОЇД, ДОДАТОК, РОЗУМНИЙ ДІМ, ЕКРАН, ІНТЕРФЕЙС КОРИСТУВАЧА, ОПЕРАЦІЇ, КЛАСИ, КНОПКА, РОСЛИНА, GRADLE, ARDUINO, Wi-Fi, ВЕБ-СЕРВЕР, ЗАПИТ

Метою кваліфікаційної роботи є розробка системи автоматичного поливу рослин за допомогою мобільного додатку.

У ході виконання були розглянуті принципи та готові рішення для автоматизації процесу поливу рослин. Встановлені необхідні умови та параметри для розробки своєї системи автоматичного поливу рослин. Згідно із умовами, підібрані компоненти для реалізації задачі та створена схема кінцевого приладу. За схемою був зібраний прототип. Його робота була перевірена у ряді тестів. Запропоновані можливі модифікації для удосконалення системи.

ABSTRACT

Bachelor’s thesis contains 51 pages, 17 figures, 8 sources according to the list of links.

OS, ANDROID, APPLICATION, SMART HOME, SCREEN, USER INTERFACE, OPERATIONS, CLASSES, BUTTON, PLANT, GRADLE, ARDUINO, Wi-Fi, WEB-SERVER

The purpose of the thesis work is to develop a system of automatic watering of plants using a mobile application.

During the implementation, the principles and ready solutions for automation of the plant watering process were considered. Necessary conditions and parameters for development of the system of automatic watering of plants are established. According to the conditions, the components for the task are selected and the scheme of the final device is created. According to the scheme, a prototype was assembled. His work has been tested in a number of tests. Possible modifications for system improvement are offered.

ЗМІСТ

ВСТУП

У наш час людство прагне використовувати технічний прогрес на всі 100 відсотків, тому все більшої популярності набуває технологія “розумний будинок”. Розумний будинок (в перекладі з англійської мови «smarthouse» або «intelligentbuilding») - це сучасне житловий будинок, створений з використанням високотехнічних механізмів і пристосувань, автоматизованих для комфортного проживання в ньому. Одним з компонентів розумного будинку є домашня теплиця і для неї важливим фактором є підтримання мікроклімату, оптимального для вирощування рослин [1].

Власники садів із великою кількістю рослин стикаються із серйозною проблемою, особливо у розпал літа: необхідністю регулярного та адекватного поливу. Деякі вважають за краще брати все у свої руки і поливають самі, використовуючи розпилювачі та спеціальні пістолети. Інші вибирають цікавіші заняття для свого вільного часу. Установка систем поливу та зрошення може покласти край усім подібним незручностям. Початкове планування та встановлення можуть виявитися дорогими, проте вигранийзгодом вільний час безцінний. І знаходячись у відпустці, можна повністю перестати турбуватися про сад.

Впровадження базової системи контролю мікроклімату – фундамент успішного врожаю. Уявіть, що ви більше не прив'язані до щоденних візитів до теплиці для перевірки температури та поливу рослин. Система автоматично підтримує оптимальну вологість ґрунту, запобігаючи як пересиханню, так і перезволоженню кореневої системи. Це особливо важливо для таких вимогливих культур, як томати та перці, де навіть невелике відхилення від норми може призвести до втрати врожаю. Завдяки точному контролю мікроклімату, ви зможете раніше висаджувати розсаду та

продовжувати сезон вирощування на 3-4 тижні. За статистикою, правильний мікроклімат збільшує врожайність на 35-40%.

Таким чином впровадження системи автоматичного контролю за параметрами мікроклімату у розумному будинку, а особливо за підтриманням водного балансу у домашній теплиці або квітнику є актуальною задачею.

1 АНАЛІЗ ЗАВДАННЯ

Вода – головна умова життя будь-яких рослин. І декоративні домашні квіти, і розсада, і сучасна конструкція вертикального городу в домашніх умовах - все потребує якісного і регулярного поливу. Але як забезпечити рослинам належне харчування, якщо господар постійно в справах і часто забуває вчасно полити своїх зелених вихованців? Автоматичний полив кімнатних рослин – ось ідеальне рішення для догляду за домашнім садом!

Автоматична система поливу – це система, що дозволяє здійснювати постійне зрошення грунту необхідним обсягом рідини протягом певного періоду. Це також важливо для кімнатних рослин, які допомагають очищати повітря від шкідливих речовин, таких як формальдегід, бензол та інші. Популярні рослини, які очищають повітря, включають хлорофітум, спатифіллум, сансевієрію, фікус, драцену, алое віра та плющ. Підтримувати для них водний баланс є дуже важливою задачею.

Віддалений моніторинг перетворює керування теплицею або домашнім квітником на зручний та сучасний процес. Замість того, щоб кілька разів на день відвідувати теплицю для перевірки стану рослин, ви можете контролювати всі параметри зі свого смартфона. Система заздалегідь попередить вас про потенційні проблеми: падіння температури, відключення електрики або нестачу вологи в ґрунті. IP-камера дозволяє візуально оцінити стан рослин та роботу систем автоматики. А вбудовані алгоритми аналізу даних допоможуть оптимізувати умови вирощування з урахуванням накопиченої статистики. Це особливо цінно для овочівників-початківців, які можуть вчитися на основі даних, що автоматично збираються, про зростання рослин у різних умовах [2].

Виходячи з цього основним завданням даної роботи є розробка прототипу мікроконтролерної системи автоматичного поливу домашніх рослин, що є рішенням проблеми, яка була викладена вище.

1.1 Апаратна частина

Для реалізації проекту була обрана платформа Arduino.

Arduino – це платформа розробки електронних пристроїв, яка складається з самої плати (Arduino UNO R3, Arduino PRO MINI, Arduino NANO і ін.) та програмного забезпечення (Arduino IDE), яке надає можливість записати розроблений код на плату за допомогою USB-кабелю. Платформа Arduino Uno R3 може використовуватися для створення різних електронних пристроїв – від банальної гірлянди до системи розумноого будинку. Простота і доброзичливість Arduino дозволяє використовувати плату від студента-початківця до досвідченого розробника.

Серед всієї лінійки моделей Arduino, була обрана плата UNO R3. Arduino Uno - це пристрій на основі мікроконтролера ATmega328. На відміну від усіх попередніх плат Arduino, Uno в якості перетворювача інтерфейсів USB-UART використовує мікроконтролер ATmega16U2 (ATmega8U2 до версії R2) замість мікросхеми FTDI (рис. 1.1). Підключіть Arduino Uno R3 до комп'ютера можна за допомогою кабелю USB Type-B.

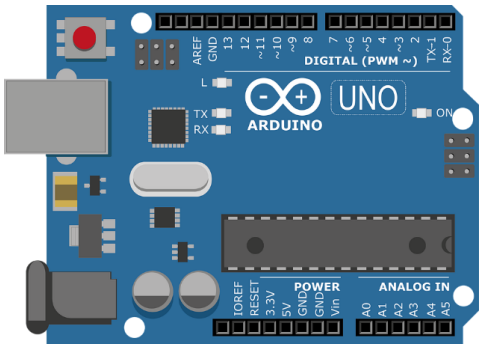

Рисунок 1.1 – Схематичне зображення плати Arduino UNO R3

Платформа Arduino Uno R3 виконана мікроконтролером Microchip ATmega328P сімейства AVR з тактовою частотою 16 МГц.

Процесор має три види пам'яті:

– 32 Кб Flash-пам'яті, з яких 0,5 Кб використовуються завантажувачем, що дозволяє прошивати Arduino Uno зі звичайного комп'ютера через USB. Flash-пам'ять постійна та її призначення - зберігання програм та супутніх статичних ресурсів;

– 2 Кб SRAM-пам'яті, які призначені для зберігання тимчасових даних, наприклад, змінних програми, по суті це оперативна пам'ять платформи. SRAM-пам'ять енергозалежна, при вимкненні живлення всі дані зітруться;

– 1 Кб енергонезалежної EEPROM-пам'яті для довготривалого зберігання даних, які не стираються при вимиканні контролера, за своїм призначенням це аналог жорсткого диска для Arduino.

Для живлення платформи Arduino Uno R3 можна використовувати порт USB, контакт Vin або роз'єм DC Barrel Jack. Джерело визначається автоматично. Під час живлення через USB використовуйте блок живлення на 5 В з USB-виходом та кабель USB. При живленні через пін Vin або роз'єм DC Barrel Jack – використовуйте джерело з вихідною напругою від 7 до 12 В. Наприклад блок живлення на 9 або 12 В.

1.2 Програмна частина

Код проектів для платформи Arduino створюється за допомогою середовища розробки Arduino IDE. Синтаксис – це сукупність рішень з широко поширених мов C та C++. Саме тому немає ніяких проблем з необхідністю глибокого вивчення програмного пристрою даної платформи

Ще однією перевагою цієї платформи є простота редагування або зміни вже написаного коду. Для перепрограмування пристрою достатньо підключити плату до комп'ютеру через USB інтерфейс і запустити перепрошивку плати в середовищі Arduino IDE. Важливо при цьому те, що немає необхідності відключати компоненти від плати, або ж використовувати різні програмні пристрої , так як дана функція передбачена в самій платі.

Мобільний додаток, за допомогою якого буде забезпечено керування системою поливу, буде створюватись в середовищі розробки для мобільних додатків на Android платформі – Android Studio. В якості мови розробки для Android додатку було обрано Java. Основною перевагою обраної мови програмування є те, що вона є об’єктно-орієнтованою, що значно спрощує всі етапи розробки і розуміння роботи системи в цілому.

Сама система автоматичного поливу квітів має працювати за наступним алгоритмом:

- користувач вибирає необхідну рослину за назвою із представлених або додає до стандартного переліку власну схему поливу;
- користувач встановлює час в який полив повинен початись;
- у вибраний час мобільний додаток посилає сигнал до плати за допомогою технології Wi-Fi.

1.3 Огляд існуючих рішень

На сьогоднішній день існують багато програмних додатків для нагадування про полив рослин, але вони не підтримують зв'язок з апаратним засобом для побудови простих систем автоматики.

Як приклад, є додаток з Play Market'y Waterbot: Plants watering + Gardening, розробник Nikola Kosev (рисунок 1.2).

Цей програмний продукт дозволяє:

- відстежувати всі рослини у вашому домі;
- отримувати повідомлення, коли рослині потрібна вода;
- налаштовувати час сповіщень (нагадувань): вранці, опівдні або ввечері;
- підтримувати інтервали поливу від півдня до двадцяти днів;
- створювати квіткові аватари за допомогою камери телефону.

Рисунок 1.2 – Waterbot: Plants watering + Gardening

Інший приклад такого виду додатку – Plants Are Friends - Plant Watering Reminders розробник Eric & Mackenzie (рисунок 1.3).

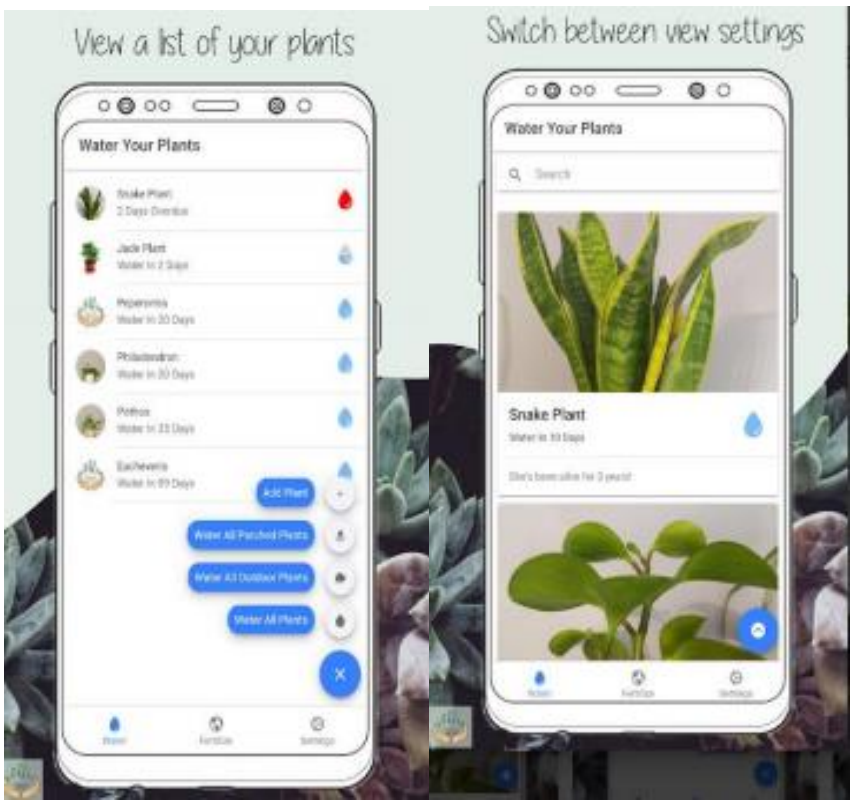

Рисунок 1.3 – Plants Are Friends - Plant Watering Reminders

Цей програмний продукт дозволяє:

- переглядати список своїх рослин і коли наступна дата їх поливання;
- слідкувати за графіком поливу та внесення добрив для кожної рослини;
- налаштувати час, коли ви хочете отримувати сповіщення про полив свого рослини.

Але дані програмні додатки не виконують функцію посиланні сигналу до контролера автоматичного поливу, що необхідно за постановкою задачі.

2 ПОСТАНОВКА ЗАДАЧІ ТА АНАЛІЗ ОБРАНИХ ЗАСОБІВ РОЗРОБКИ

2.1 Постанова задачі

За своєю суттю система повинна являти сукупність мобільного додатку і електронного пристрою, до якого входить плата Arduino та Wi-Fi модуль:

– простота і низька ціна виготовлення кінцевого продукту;

– надійність і коректна робота основних компонентів;

– можливість додаванням нових видів сенсорів та датчиків;

– розробка повноцінного сайту для налаштування системи;

– масштабованість системи, що дозволяє перейти з невеликих прототипів для тестування, на повнорозмірні системи поливу для використання не тільки у квартирах а також і у сільському господарстві.

Внаслідок того, що було вказано раніше, відмінною рисою проекту буде відносно низька ціна виготовлення і компактність, що дозволяють протестувати прототип системи без додаткових компонентів.

Для досягнення поставленої мети мають бути вирішені наступні задачі:

- розгляд існуюсих систем керування мікрокліматом;

- аналіз відомих мобільних операційних систем та вибір програмної платформи реалізації;

- вибір апаратної платформи, на якій буде будуватись весь проект;

- розробка програмної частини проекту, що вміщує керування тестовою платформою та роботу самої системи автоматичного поливу;

- реалізація пристрою дистанційного керування тестовим прототипом системи на монтажній платі з використанням світлодіодів.

2.2 Аналіз відомих мобільних операційних систем

Дуже важливо знати про різні мобільні ОС, які використовують виробники, і що ховається за барвистим сенсором вашого смартфона. Варто почати з того, що обидві операційні системи досить добрі. Принаймні для того, щоб бути монополістами на ринку мобільних пристроїв: Android - близько 70%, iOS - трохи менше 30%.

Виходячи з цієї логіки, можна відразу зробити висновок: Android краще, адже на ньому працює майже в 2,5 рази більше аксесуарів, ніж на iOS. І це буде неправильна точка зору, оскільки на системі від Google випускаються моделі різних цінових сегментів, у тому числі ультрабюджетні та бюджетні, які займають переважну частку ринку. Apple ж виробляють виключно флагманські моделі, навіть якщо розглядати їхню продукцію в контексті початкових версій iPhone. І ціни на них відповідні.

Тому далі ми розглядатимемо ці системи у відриві від вартості пристроїв. Хоча до фінансового питання ми все одно повернемось.

Концепція та універсальний користувальницький досвід.

Просунуті користувачі Android як один з основних аргументів на користь системи наводять політику відкритого вихідного коду. Це означає, що кожна людина може змінювати систему. Насамперед відкритий вихідний код актуальний для виробників смартфонів – вони створюють фірмові версії системи (оболонки) з власними характеристиками юзабіліті (зручності використання) та унікальними «фішками». Завдяки цьому Android пристрої різних виробників у контексті одного цінового сегмента можуть забезпечувати абсолютно різний досвід користувача.

Також Android смартфони мають відкриту файлову систему і мають повноцінний файловий менеджер, що робить роботу з файлами максимальною простою, інтуїтивно зрозумілою і навіть різноманітною. Простіше кажучи, користувач смартфона на цій ОС може завантажити на девайс все, що завгодно. Якщо розширення (формат) файлу підтримується

системою, його можна ще й відкрити чи запустити. Це спрощує і роботу з файлами-все за аналогією з ПК на звичній для багатьох ОС Windows.

Операційна система Apple не може похвалитися подібними можливостями. Є фірмовий магазин App Store, що жорстко регулюється компанією, є окремі попередньо встановлені або завантажувані програми для різних типів файлів, є обмеження за форматами і в принципі діям користувача на пристрої.

Але саме завдяки кардинально різним політикам обидві операційні системи мають плюси та мінуси. Так, свобода вибору та гнучкість Android – не лише перевага, а «тоталітаризм» iOS – не лише недолік.

Зручність використання. При розгляді Android не з точки зору зручності роботи з файлами, а в цілому він може здаватися менш зрозумілим, якщо не виходити за рамки базового функціоналу - дзвінки, месенджери, соцмережі, ігри і т.д. Це пов'язано саме з гнучкістю системи та масою різноманітних налаштувань. Так, якщо мова про не найвпевненішого користувача комп'ютера на Windows, то користуватися всім доступним функціоналом девайса на Android буде складно, оскільки про багато «фіч» він просто не знатиме. Або не зможе їх знайти та налаштувати. У той же час просунутому користувачеві доступна детальна кастомізація системи під себе та свої цілі – з нею можна робити майже все.

iOS набагато простіше: є типи файлів – для кожного з них окрема програма, є функція – для кожної з них, умовно, теж окрема програма. Усі на своїх місцях. Крім того, недосвідчений користувач, навіть якщо кудись випадково натисне, навряд чи зможе нашкодити системі.

Апаратна частина та швидкість.Якщо загалом, швидкість смартфона залежить від двох складових: продуктивність апаратного забезпечення (центральний процесор, оперативна пам'ять, графічний прискорювач і т.д.) і оптимізованості (налагодженості) системи. Система Apple, об'єктивно, оптимізована краще - саме тому користувачі iPhone практично ніколи не стикаються з підгальмуванням, некоректною роботою додатків і самої ОС.

Більше того, девайси Apple зазвичай не кладуть навіть через 3-4 роки використання.

Бюджетні смартфони на Android можуть перманентно гальмувати «з коробки», а нові флагмани підгальмовувати періодично. Але все залежить від моделі пристрою та версії операційної системи.

Чи означає це, що Android – погана ОС? У жодному разі. Справа в тому, що компанії Apple потрібно оптимізувати свою систему для 15-20 щодо актуальних моделей свого виробництва, частина з яких оснащується ідентичним або схожим за архітектурою і технологіями «залізом». Моделей на Android – сотні, якщо не тисячі. Природно, за таких умов ніхто не оптимізуватиме свою систему під кожну. Почасти проблема оптимізації вирішується фірмовими оболонками.

По суті, сьогодні смартфони на Android змагаються не в тому, у кого ексклюзивна однокристальна система (помилково: процесор), великі об'єм і швидкість оперативної пам'яті, а в технологіях штучного інтелекту, зручності інтерфейсів користувача, якості зйомки, дизайну та ергономіці.

"На папері" топові Android мають явну перевагу, проте фактична продуктивність девайсів практично ідентична. Причина, в першу чергу, полягає саме в оптимізованості систем: iOS та додатки для неї менш ресурсомісткі. Відповідно, немає потреби в надзвичайно потужному апаратному забезпеченні. Варто згадати і розмір техпроцесу виробництва актуальної СНК Apple: 3 нанометри проти 4 чіпсетів флагманів Android. Говорячи простими словами, що менший розмір техпроцесу, то більш продуктивною і енергоефективною є платформа. Це правило працює майже завжди.

Існують ще інші важливі відмінності.

1. Функціональність. Актуальні версії iOS і Android аналогічні за функціоналом і за умовчанням навіть мають схожі інтерфейси користувача. Якісь унікальні «фішки», щойно діставшись до iOS, через короткий час у модифікованому вигляді потрапляють на Android і навпаки. У чому Android

дійсно кращий за iOS — гнучка кастомізація, завдяки якій користувач може тонко налаштувати систему під себе. Значки, іконки, шрифти, віджети і т.д.

2. Безпека. У плані безпеки iPhone перевершують девайси на Android. І головний аргумент — навіть не просунуті механізми захисту та шифрування, а саме «закритість» системи. Не можна встановлювати програми зі сторонніх джерел, є обмеження за форматами файлів, що завантажуються і запускаються. Все це знижує можливість проникнення шкідливого програмного забезпечення на пристрої. Крім того, найшкідливішого ПЗ для iOS кратно менше, ніж для Android.

Таким чином однозначно відповісти на це питання не можна, оскільки ці системи – повноцінні та повноправні конкуренти, кожна з яких має переваги та недоліки. Але можна відповісти інакше: краще та ОС, яка більшою мірою відповідає вашим вимогам та побажанням. Більш стабільні, безпечні, швидкодіючі пристрої iPhone. Якщо важливі можливості персоналізації та комплексного налаштування («підгону» девайсу під себе), а також доступ до прозорої та зрозумілої файлової системи – Android.

Отже, зважаючи на все вищенаведене, було вирішено, що в якості мобільної операційної системи для розробки програмного застосунку буде обрано Android.

2.2 Аналіз інструментальних засобів для розробки застосунків для операційної системи Android

Компанія Google випустила офіційне інтегроване середовище розробки для роботи з платформою Android - Android Studio. Android Studio прийшло на зміну плаґіну ADT для платформи Eclipse. Середовище побудоване на базі вихідного коду продукту IntelliJ IDEA Community Edition, що розвивається компанією JetBrains. Android Studio розвивається в рамках відкритої моделі розробки та поширюється під ліцензією Apache 2.0 [3].

Середовище надає засоби для розробки застосунків не тільки для смартфонів і планшетів, але і для носимих пристроїв на базі Android Wear, телевізорів (Android TV), окулярів Google Glass і автомобільних інформаційно-розважальних систем (Android Auto).

Крім можливостей, присутніх в IntelliJ IDEA, в Android Studio реалізовано кілька додаткових функцій, таких як нова уніфікована підсистема складання, тестування і розгортання застосунків, заснована на складальному інструментарії Gradle і підтримуюча використання засобів безперервної інтеграції.

Основними особливостями даного середовища розробки є :

- живі макети: редагувальник WYSIWYG — живе кодування “подання” програми в реальному часі;
- базування на Gradle;
- android - орієнтований рефакторинг та швидкі виправлення;
- lint утиліти для охоплення продуктивності, юзабіліті, сумісності версій та інших проблем;
- шаблони для створення поширених Android дизайнів та компонентів;
- багатий редактор макетів (layouts) що дозволяє користувачам перетягнути і покласти (drag-and-drop) компоненти користувацького інтерфейсу, як варіант, переглянути одночасно макети (layouts) на різних конфігураціях екранів.

2.3 Загальні поняття Android-розробки

Компоненти програми є свого роду «будівельними блоками» для додатка Android. Ці компоненти пов'язані файлом-маніфестом додатку AndroidManifest.xml, який описує кожен компонент програми та взаємодію цих компонентів між собою [4].

Є чотири базових типи компонентів, які можуть бути використані в додатку Android:

- операції (Activities) представляють собою елементи призначеного для користувача інтерфейсу (одна операція – один екран) і відповідають за взаємодію користувача з екраном мобільного пристрою;

- служби (Services) представляють собою тривалі операції, що працюють у фоновому режимі і не мають призначеного для користувача інтерфейсу (наприклад, передача даних), замість цього вони, як правило, запускаються іншими елементами, що вже мають користувальницький інтерфейс, і взаємодіють з ними;

- приймачі широкомовних повідомлень (Broadcast receivers) представляють собою компоненти, що реагують на оголошення самій ОС, що передаються всій системі (як хороший приклад – оголошення про низький рівень заряду батареї пристрою). Вони також не мають призначеного для користувача інтерфейсу, проте можуть передавати дані до інших компонентів, де вони демонструються користувачеві у вигляді повідомлень;

- постачальники контенту (Content providers) представляють собою компоненти, що управляють взаємодією додатки з його базою даних – за допомогою постачальника контенту інші компоненти програми можуть запитувати або змінювати дані.

Крім чотирьох базових типів, існують додаткові типи компонентів, що використовуються для вибудовування взаємодій і зв'язків як між базовими типами компонентів, так і між компонентами і зовнішніми елементами. До них відносяться:

- фрагменти (Fragments) – частини призначеного для користувача інтерфейсу в операціях;

- види (Views) – елементи призначеного для користувача інтерфейсу, які відображаються на екрані, наприклад, кнопки, списки;

- макети (Layouts) – визначають елементи призначеного для користувача інтерфейсу, їх властивості та розташування;

- наміри (Intents) – з'єднують разом різні компоненти програми або пов'язують один з одним роботу різних додатків;

- ресурси (Resources) – зовнішні елементи, такі, як рядки, константи або зображення;
- маніфест (Manifest) – конфігураційний файл програми.

2.4 Середовище розробки Arduino

Arduino IDE – середовище розробки Arduino, яке дозволяє швидко і доволі просто розробляти скрипт для прошивки плат. За допомогою вбудованих інструментів можна без труднощів слідкувати за станом апаратури.

Розглянемо основні переваги використання Arduino IDE.

Багатоплатформна програма. Arduino IDE працює на трьох найпопулярніших операційних системах: Windows, Mac OS та Linux. Крім цього, додаток також доступний із хмари. Ці параметри надають можливість створювати та зберігати свої скрипти в хмарі або будувати свої програми локально та завантажувати їх безпосередньо на плату.

Управління правлінням. Arduino IDE постачається з модулем управління платою, де користувачі можуть вибрати плату, з якою хочуть працювати в даний момент. Якщо вони хочуть це змінити, вони можуть це зробити легко зі спадного меню. Змінюючи їх вибір, також автоматично оновлюється інформація PORT з даними, необхідними їм щодо нової плати.

Прямі скрипти. За допомогою Arduino IDE користувачі можуть створювати програми, звані скриптами, які будуються за допомогою текстового редактора.

Документація проекту. Arduino IDE пропонує можливість документувати свої проекти. Ця функція дозволяє відстежувати досягнення та будь-які зміни, які вносяться щоразу. Крім цього, документація дозволяє іншим людям легко застосовувати скрипти на своїх платах.

Простий обмін ескізами. Окрім збереження та архівування скриптів та завантаження їх на плату, Arduino IDE також може обмінюватися ними

(доступно лише у хмарній версії). Кожному скрипту надається своя унікальна URL-адреса, яку користувачі можуть поділитися зі своїми колегами та колегами-любителями Arduino. Потім одержувач має доступ до коду, вони можуть зберегти його в хмарі або завантажити для власного використання.

Обширна бібліотека. Arduino IDE має понад 700 інтегрованих бібліотек. Вони були написані та поширені членами спільноти Arduino, які інші користувачі можуть використовувати для власних проектів, не встановлюючи нічого. Це дозволяє програмістам додати інший функціонал своїм скриптам.

Підтримка стороннього обладнання. Незважаючи на те, що Arduino IDE розроблений спеціально для плат Arduino, він також підтримує з'єднання зі стороннім обладнанням. Це робить використання програми більш широким, а не обмеженим власними платами.

3 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ДЛЯ КЕРУВАННЯ СИСТЕМОЮ ПОЛИВУ

3.1 Розробка інтерфейсу користувача

Кожен мобільний застосунок для успішного функціонування потрібен мати UI (інтерфейс користувача). Інтерфейс користувача (англ. user interface, UI, дружній інтерфейс) – засіб зручної взаємодії користувача з інформаційною системою.

Створення якісного інтерфейсу передбачає реалізацію принципу «інтереси користувача понад усе» відповідну методологію розробки всього програмного продукту. В англомовній літературі для опису такого підходу використовується термін User-centered Design («Розробка, орієнтована на користувача»). Ця технологія, крім усього іншого, передбачає якомога більш раннє проектування інтерфейсу з подальшим його розвитком в процесі розробки самого програмного продукту.

Основна перевага хорошого інтерфейсу полягає в тому, що користувач завжди відчуває, що він управляє програмним забезпеченням, а не програмне забезпечення управляє ним. Для створення у користувача такого відчуття «внутрішньої свободи» інтерфейс повинен мати цілу низку властивостей.

Також важливо зазначити, що навіть при наявності добре спроектованого інтерфейсу користувачі можуть робити ті чи інші помилки. Ці помилки можуть бути як «фізичного» типу (випадковий вибір команди або даних) так і «логічного» (прийняття неправильного рішення на вибір команди або даних). Ефективний інтерфейс повинен дозволяти запобігати ситуації, які, ймовірно закінчаться помилками. Він також повинен вміти адаптуватися до потенційних помилок користувача і полегшувати йому процес усунення наслідків таких помилок.

Зважаючи на всі вище сказані вимоги було розроблено даний інтерфейс користувача (рисунок. 3.1).

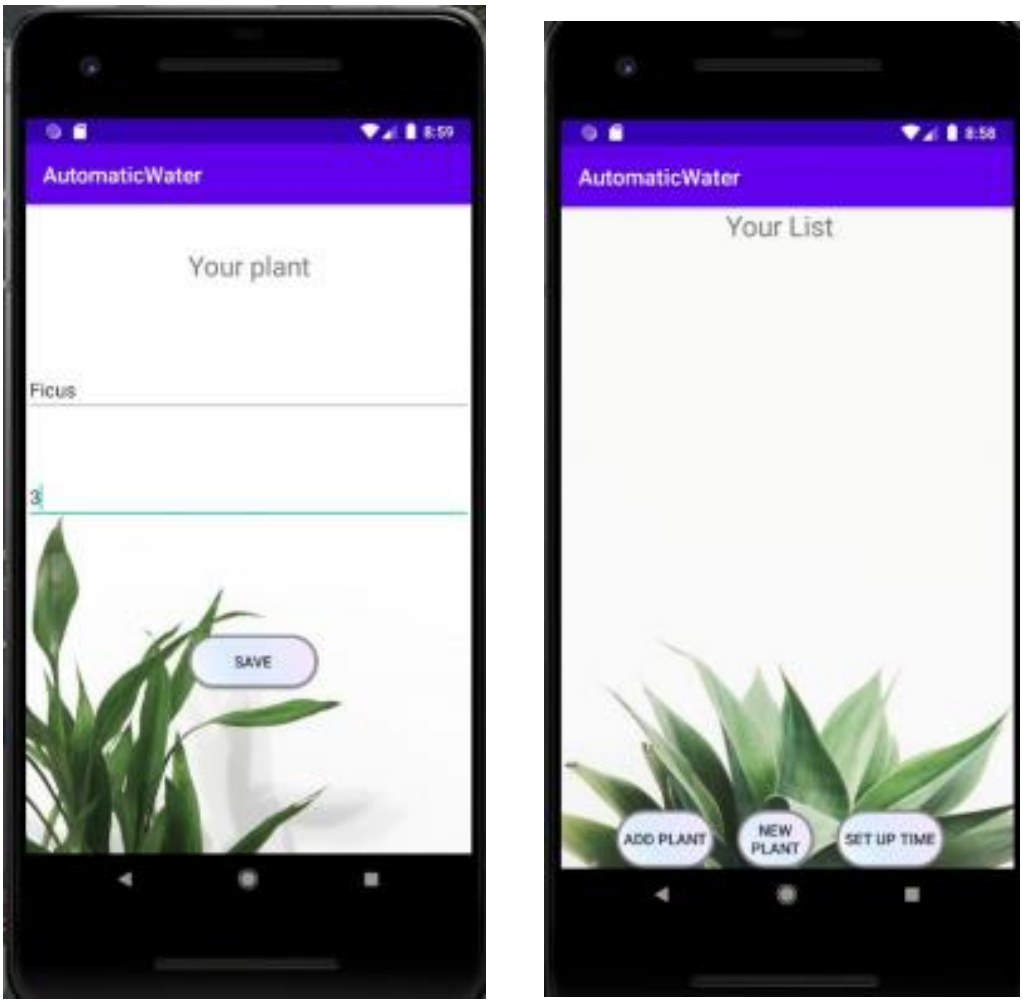

Рисунок 3.1 – Інтерфейс розробляємого програмного забезпечення AutomaticWater

3.2 Розробка основного функціоналу програмного забезпечення

3.2.1 Реалізація функції переходу між операціями(Activities)

Дана функція реалізована за допомогою події Button–setOnClickListene, який обробляє натискання на кнопку. У лістингу 3.1 представлена функція переходу з головної операції (MainActivity) до операції вибору схеми поливу рослин із заданого списку (ChoiceOfPlantActivity).

Лістинг 3.1 – Функція переходу з головної операції до операції вибору схеми поливу із заданого списку

```
addPlants.setOnClickListener(v ->
{
      Intent intent = new  Intent(MainActivity.this,
      ChoiceOfPlantActivity.class);  startActivityForResult(intent,
      OPEN_NEW_ACTIVITY);
      onActivityResult(OPEN_NEW_ACTIVITY, 0,  intent);
 }
);
```

3.2.2 Реалізація функції обрання схем поливу рослин із заданого списку

Дана функція реалізована за допомогою події ListView – setOnItemClickListener, який обробляє натискання на пункт списку. В даній ситуації ListView зберігає список рослин, для яких вже заздалегідь встановлена схема поливу. Реалізація даної функції представлена в лістингу 3.2.

Лістинг 3.2 – Функція обрання схеми поливу рослин із заданого списку

```
listOfPlants.setOnItemClickListener((adapterView,  view, i, l) -> {
      PlantsNameEnum plant =
      PlantsNameEnum.valueOf(listOfPlants.getItemAtPosition(i)
      .toString());
      listOfChosenPlant.add(plant.name());
});
```

3.2.3 Реалізація функції зберігання та перенесення внесених змін з однієї операції до іншої

Для забезпечення правильного функціонування даної функції було використано класу намір (Intent) та його метод putStringArrayListExtra, який додає розширені дані. .. Реалізація функції представлено в лістингу 3.3.

Отримання даних з іншої операції також реалізується за допомогою класу намір (Intent) та його методу getSerializableExtra, який отримує розширені дані. Реалізація даної функції представлено у лістингу 3.4.

Лістинг 3.3 – Додавання розширених даних до наміру

```
Intent intent = getIntent();
intent.putStringArrayListExtra("listOfChosenPlant",
                      (ArrayList<String>) listOfChosenPlant);
setResult(Activity.RESULT_OK, intent);
```

Лістинг 3.4 – Отримання розширених даних з наміру

```
ArrayList<String>names=(ArrayList<String>)
data.getSerializableExtra("listOfChosenPlant");
```

3.5.4 Функція додавання нових схем поливу користувачем

Дана функція реалізується за допомогою події класу кнопка (Button) setOnClickListener. Вона забезпечує передачу даних до головної операції і збереження їх у списку обраних рослин. Зчитування даних з форми введення реалізується за допомогою функції getText(). Реалізація представлена у листингах 3.5 і 3.6 відповідно.

Лістінг 3.5 – Зчитування даних із форми введення і передача до головної операції

```
back.setOnClickListener((View.OnClickListener) v ->
{
      String name=nameCustomPlant.getText().toString();
      Integer  interval=Integer.parseInt(numberOfWatering.getText()
                                                    .to String());
      customPlant.put(name,interval);
      Intent intent = getIntent();  intent.putExtra("customPlant",
      (Serializable) customPlant);
      setResult(4, intent);
      finish();
});
```

Лістинг 3.6 – Отримання і збереження введених даних

```
Map<String,Integer> customPlant= (Map<String,  Integer>)
data.getSerializableExtra("customPlant");
String  nameOfCustomPlant=Objects.requireNonNull(customPlant.ke ySet()
      .toArray())[0]
                  .toString();
```

```
myList.add(nameOfCustomPlant);
 adapter.notifyDataSetChanged();
int interval=Objects.requireNonNull(customPlant
                                    .get(nameOfCustomPlant));
```

3.5.5. Встановлення часу та інтервалу посилання сигналу до контролера

Дана функція була реалізована за допомогою класу AlarmManager. Цей клас забезпечує доступ до систем сигналізації. Вони дозволяють запланувати запуск програми в певний момент у майбутньому. Коли спрацьовує будильник, намір, який був для нього зареєстрований, транслюється системою, автоматично запускаючи цільову програму, якщо вона ще не запущена. Зареєстровані будильники зберігаються, поки пристрій не працює, але буде видалено, якщо його вимкнути та перезавантажити.

Alarm Manager утримує блокування процесора, доки виконується метод onReceive() приймача сигналу. Це гарантує, що телефон не буде спати, поки ви не закінчите обробку трансляції. Після того, як onReceive() повернеться, Alarm Manager звільняє цю функцію блокування. Це означає, що телефон у деяких випадках буде спати, як тільки завершиться метод onReceive(). Якщо ваш приймач сигналу викликав Context.startService(), можливо, телефон заснує до запуску запитуваної послуги. Щоб цього не сталося, вашому BroadcastReceiver та службі потрібно буде застосувати окрему політику блокування будильника, щоб гарантувати, що телефон продовжить працювати, поки послуга не стане доступною.

3.5.6. Функція посилання сигналу до контролера

Реалізована функція відправлення сигналу від мобільного додатка до контролера системи.

У адрес отримувача необхідно вказати IP – адресу контролера, що буде отримувати сигнали. Було ініційовано з’єднання за допомогою класу HttpURLConnection. Реалізація даної функції представлено у Лістингу 3.8.

Лістинг 3.8 – Встановлення зв'язку між мобільним додатком з контролером системи (фрагмент)

```
public static final String REQUEST_METHOD = "GET";
public static final int READ_TIMEOUT = 15000;
public static final int CONNECTION_TIMEOUT = 15000;

@Override
    protected String doInBackground(String... params){
        String stringUrl = params[0];
        String result;
        String inputLine;
        try {
            //Create a URL object holding our url
            URL myUrl = new URL(stringUrl);
            //Create a connection
            HttpURLConnection connection =(HttpURLConnection)
                    myUrl.openConnection();
            //Set methods and timeouts

            connection.setRequestMethod(REQUEST_METHOD);
            connection.setReadTimeout(READ_TIMEOUT);
            connection.setConnectTimeout(CONNECTION_TIMEOUT);

            //Connect to our url
            connection.connect();
            //Create a new InputStreamReader
            InputStreamReader streamReader = new
                    InputStreamReader(connection.getInputStream());
            //Create a new buffered reader and String Builder
            BufferedReader reader = new BufferedReader(streamReader);
            StringBuilder stringBuilder = new StringBuilder();
            //Check if the line we are reading is not null
            while((inputLine = reader.readLine()) != null){
                stringBuilder.append(inputLine);
            }
```

Так як клас Background спадкоємець класу AsyncTask <String, Void, String> з превизначеним методом doInBackground(String... params), сигнал можна послати до контролера шляхом створення нового екземпляру класу Background і викликати метод execute із потрібною командою для контролера. Реалізація даної функції представлено у лістингу 3.9.

Лістинг 3.9 – Відправлення контролеру сигнал з командою

```
@Override
    public void onReceive(Context context, Intent intent) {
        if(Objects.equals(intent.getAction(),
PlantsNameEnum.DRACAENA.name())){
            Toast.makeText(context, "Time to water
flower!"+intent.getAction(), Toast.LENGTH_LONG).show();
        }
        new
Background().execute("http://192.168.1.4/waterPlants_ON"+intent.getAct
ion());
    }
}
```

4 РОЗРОБКА АПАРАТНОЇ СКЛАДОВОЇ СИСТЕМИ АВТОМАТИЧНОГО ПОЛИВУ

4.1 Технологія зв'язку апаратної складової з системою керування

Перед тим як почати розробляти скрипт для обраної плати, необхідно обрати спосіб встановлення зв'язку апаратної складової з системою керування. Було розглянуто і порівняно декілька варіантів, а саме: Bluetooth та Wi – Fi зв'язок.

Технологія Bluetooth корисна при передачі інформації між двома або більше пристроями, які знаходяться поруч один з одним, коли швидкість не є проблемою, наприклад телефонами, принтерами, модемами і гарнітурами. Він найкраще підходить для додатків з низькою пропускною здатністю, таких як передача звукових даних за допомогою телефонів (наприклад, за допомогою гарнітури Bluetooth) або байтових даних за допомогою портативних комп'ютерів (передача файлів) або клавіатури і мишей.

Проаналізувавши сучасний стан технології Bluetooth, можна визначити плюси і мінуси. До переваг стандарту відносяться:

- високий рівень стандартизації і сумісність з іншими пристроями Bluetooth різних виробників;
- захист переданих даних;
- низька вартість;
- висока дальність дії (до 1000 м);
- універсальність і велика різноманітність модулів під різні завдання.

Серед недоліків відзначимо:

- відносне енергоспоживання (робота від автономних джерел живлення не завжди можлива). Передбачається, що цього недоліку буде позбавлена нова версія специфікації Bluetooth 4.0.
- відносно невисока швидкість обміну даними (до 1 Мбіт / с). Як

правило, реальна швидкість обміну даними обмежується пропускною спроможністю зовнішніх апаратних інтерфейсів модуля.

Одне з основних переваг стандарту Bluetooth полягає в його високому рівні стандартизації та найширшому поширенні в складі користувальницьких електронних пристроїв. Це дозволяє в ряді випадків практично в два рази заощадити час і витрати на розробку при проектуванні деякої системи збору даних, телеметрії або управління на основі Bluetooth, оскільки в якості однієї зі сторін бездротового обміну даними може виступати, наприклад, звичайний серійно випускається ноутбук або комунікатор з підтримкою даної технології.

- Щодо технології Wi-Fi, даний стандарт бездротової передачі даних був створений спеціально для об'єднання декількох комп'ютерів в єдину локальну мережу. Wi-Fi краще підходить для роботи в повномасштабних мережах, оскільки він забезпечує більш швидке з'єднання, кращий діапазон від базової станції і кращу безпеку бездротового зв'язку (при правильному налаштуванні), ніж Bluetooth.

Зважаючи на все вищесказане, було прийнято рішення, що в якості основного способу зв'язку обрати Wi-Fi.

4.2 Засіб зв'язку апаратної складової та системи керування за допомогою технології Wi-Fi

Для взаємодії моделі системи автоматичного поливу з системою керування вирішено застосувати модуль ESP8266 ESP-01 (рисунок. 4.1). Цей модуль Wi-Fi, який дозволяє мікроконтролерам отримати доступ до мережі Wi-Fi.

ESP-01 дуже маленький і вміщується в будь-якому корпусі, тому ідеально підходить для готових проектів. Оновлена версія має флеш-пам’ять розміром 1 МБ (попередня версія мала 512 кБ). Він пропонує чотири GPIO для управління та підключення периферійних пристроїв (два з яких - TX та

RX для послідовного зв'язку). Плата не має вбудованого регулятора напруги, тому вам потрібно використовувати джерело живлення 3V3 або додати регулятор напруги, щоб вхідна напруга знизилася до 3V3.

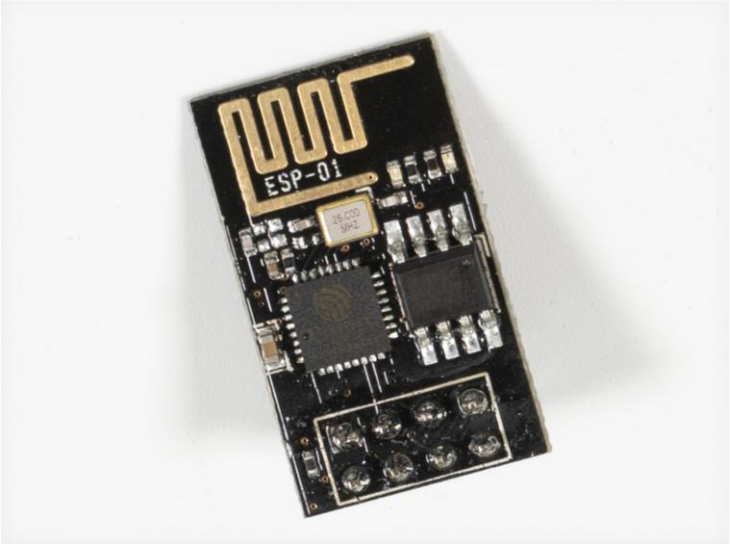

Рисунок 4.1 – Зображення модуля ESP8266 ESP-01

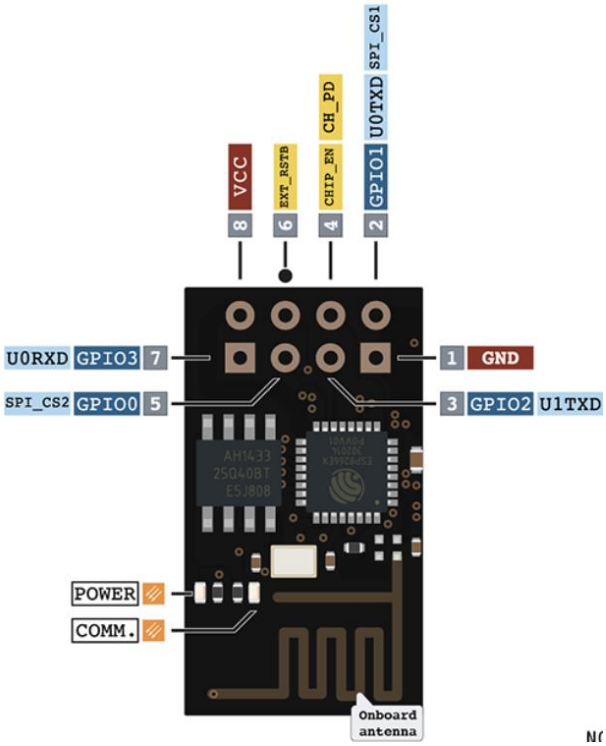

Рисунок 4.2 – Схематичне зображення модуля ESP8266 ESP-01

Модуль ESP8266 дозволяє записувати власні додатки. Можливо скомпілювати код з мови C і завантажити в модуль. Така процедура називається "прошивкою" (flashing). Для того, щоб розроблений додаток працював коректно, у нього повинна бути можливість відправляти і приймати дані по мережі і / або працювати з зовнішніми датчиками, входами і виходами. ESP8266 оснащений базовими функціями для цього, набір яких представляє собою примітивну "операційну систему". Служби цієї ОС можуть бути викликані додатком.

Так як пристрій має працювати вдома тому було прийнято рішення, що краще використовувати живлення за допомогою USB. Дане рішення також дає змогу зменшити витрати на розробку даного пристрою.

4.3 Підсумкова схема пристрою

В результаті об'єднання всіх компонентів, пристрій набуває наступний схематичний вигляд (рисунок 4.1).

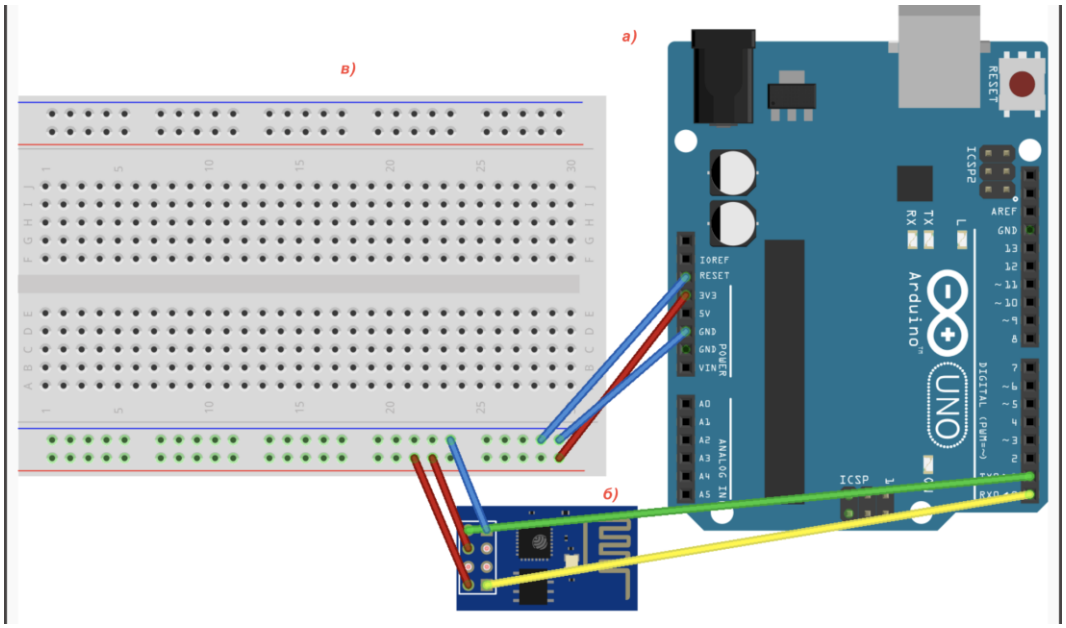

Рисунок 4.3 – Схематичне зображення пристрою та його компонентів: а) плата Arduino UNO R3; б)Wi-Fi модуль ESP8266; в) макетна плата

4.4 Розгортання веб-сервера за допомогою ESP8266-01 для зв'язку з системою керування

Одна з найважливіших функцій, яку забезпечує ESP8266, полягає в тому, що він може не тільки підключатися до існуючої Wi-Fi мережі і працювати в якості веб-сервера, але він також може встановлювати власну мережу, дозволяючи іншим пристроям підключатися безпосередньо до нього і отримувати доступ до веб-сторінок. Це можливо, тому що ESP8266 може працювати в трьох різних режимах: режим станції, режим точки доступу і обидва перших режиму одночасно. Це забезпечує можливість побудови пористих мереж.

По-перше, необхідно визначити що таке веб-сервер. Веб-сервер – це місце, де зберігаються, обробляються і доставляються веб-сторінки веб-клієнтам. Веб-клієнт – це не що інше, як веб-браузер на наших ноутбуках та смартфонах. Зв'язок між клієнтом і сервером здійснюється за допомогою спеціального протоколу, званого протоколом передачі гіпертексту (HTTP).

У цьому протоколі клієнт ініціює зв'язок, роблячи запит для певної веб-сторінки за допомогою HTTP, і сервер відповідає вмістом цієї веб-сторінки або повідомленням про помилку, якщо це не вдається зробити (наприклад, знаменита Помилка 404).

Веб-сторінки - це переважно статичний вміст, що включає документи HTML, зображення, таблиці стилів, тести тощо. Окрім HTTP, веб-сервер також підтримує протоколи SMTP (Простий протокол передачі пошти) та FTP (Протокол передачі файлів) для надсилання електронної пошти та передачі файлів та зберігання.

Основною роботою веб-сервера є відображення вмісту веб-сайту. Якщо веб-сервер не є загальнодоступним і використовується всередині, тоді він називається Intranet Server. Коли хтось запитує веб-сайт, додавши URL-адресу або веб-адресу в адресному рядку веб-браузера (наприклад, Chrome

або Firefox) (наприклад, www.dl.nure.ua), браузер надсилає запит в Інтернет на перегляд відповідної веб-сторінки для цього адресу. Сервер доменних імен (DNS) перетворює цю URL-адресу на IP-адресу (наприклад, 192.168.216.345), яка в свою чергу вказує на веб-сервер.

Веб-серверу пропонується представити веб-сайт із вмістом у браузері користувача. Усі веб-сайти в Інтернеті мають унікальний ідентифікатор з точки зору IP-адреси. Ця адреса Інтернет-протоколу використовується для зв'язку між різними серверами в Інтернеті.

У наші дні сервер Apache - найпоширеніший веб-сервер, доступний на ринку. Apache - це програмне забезпечення з відкритим кодом, яке обробляє майже 70 відсотків усіх веб-сайтів, доступних сьогодні. Більшість веб-програм використовують Apache як середовище веб-сервера за замовчуванням. Іншим загальнодоступним веб-сервером є Інформаційна служба Інтернету (IIS). IIS належить Microsoft.

Модуль ESP8266 може працювати в трьох різних режимах.

1. Станція (STA) – модуль ESP8266, який підключається до існуючої Wi-Fi мережі (створеної бездротового маршрутизатора), називається Станція (STA). У режимі станції ESP8266 отримує IP-адресу від бездротового маршрутизатора, до якого він підключений. З цим IP-адресою він може налаштувати веб-сервер і доставляти веб-сторінки на всі підключені пристрої в існуючій мережі Wi-Fi;

2. Точка доступу (AP) – модуль ESP8266, який створює свою власну WiFi мережі та виступає концентратор (точно так само як маршрутизатор WiFi) для однієї або декількох станцій, називається Точкою доступу (AP). На відміну від WiFi-роутера, він не має підключення до дротової мережі. Максимальна кількість станцій, які можуть підключитися до нього, обмежена п'ятьма. У режимі точка доступу ESP8266 створює нову WiFi мережу і встановлює SSID (ім'я мережі) і IP-адреса для неї. За допомогою цього IP-адреси він може доставляти веб-сторінки на всі підключені пристрої в своїй власній мережі;

3. Комбінований - поєднання двох попередніх режимів.

Будемо використовувати ESP8266 у першому режимі.

4.5.1 Оголошення змінних у скриптах для Arduino

Ескіз починається з включення бібліотеки ESP8266WiFi.h. Ця бібліотека забезпечує специфічні для ESP8266 методи WiFi, які забезпечують підключення до мережі. Після цього ми також включаємо бібліотеку ESP8266WebServer.h, яка має деякі доступні методи, які допоможуть налаштувати сервер та обробляти вхідні HTTP-запити, не турбуючись про деталі впровадження на низькому рівні.

Оскільки ми встановлюємо ESP8266 в режимі станція, нам потрібно встановити змінні, які характеризують його SSID та пароль.

Змінна - це місце для зберігання фрагмента даних. Вона має ім’я, тип і значення.

Наприклад, у (Лістинг 4.1) оголошує змінну з іменем ssid, типом char* та початковим значенням ”531”. Він використовується для позначення SSID Wi-Fi мережі. SSID - це назва бездротової мережі. Це те, що необхідно шукати при підключенні бездротових комп’ютерів та пристроїв.
Кожного разу, коли ім'я ssid з'являється в коді, його значення буде отримано.

Перевага використання змінної полягає в тому, що простіше змінити значення ssid тільки в одному місті, а ніж у кожному рядку.

Однак часто значення змінної змінюватиметься під час роботи ескізу. Наприклад, можна зберегти значення, прочитане з вводу, у змінну.

Лістинг 4.1 – Оголошення змінних у коді, які позначають SSID та пароль Wi-Fi мережі

```
const char* ssid = "531"; // Wi-Fi SSID
const char* password =  "maximumride"; //Wi-Fi Password
```

Далі оголошується об’єкт бібліотеки ESP8266WebServer, за допомогою функції ESP8266WebServer server(80);, щоб ми могли отримати доступ до його функцій. Конструктор цього об'єкта приймає за параметр порт (де сервер буде слухати). Оскільки 80 є портом за замовчуванням для HTTP, будемо використовувати це значення. Тепер можна отримати доступ до сервера без необхідності вказувати порт в URL-адресі.

4.5.2 Функція void setup() у скрипті для Arduino

Функція setup() викликається, коли стартує програма. Використовується для ініціалізації змінних, визначення режимів роботи виводів, запуску використовуваних бібліотек та ін. Функція налаштування запускається лише один раз, після кожної подачі живлення або брошури плати Arduino.

Лістинг 4.2 представляє собою функцію налаштування для цього проекту. По-перше, функція Serial.begin(9600); – ініціює послідовне з'єднання і задає швидкість передачі даних в біт / секунду (бод). Для обміну даними з комп'ютером використовують наступні значення: 300, 1200, 2400, 4800, 9600, 14400, 19200, 28800, 38400, 57600 або 115200. Для нашого випадку було встановлено 9600 тому, що модуль ESP8266 стабільніше працює на даній швидкості.

По-друге функція WiFi.begin(ssid, password); – ініціалізує налаштування мережі бібліотеки WiFi і надає поточний стан. Після цього перевіряється поточний стан:

- WL_CONNECTED при підключенні до мережі;
- WL_IDLE_STATUS, коли не підключений до мережі, але ввімкнений;

Після цього необхідно роздрукувати дані на послідовний порт як зручний для читання текст ASCII, за яким слідують символ повернення каретки (ASCII 13, або '\ r') та символ нового рядка (ASCII 10, або '\ n'). Це

можна зробити за допомогою функції Serial.print(). В нашому випадку було виведено IP адресу де буде розвернено веб-сервер.

Важливо зазначити, що в даній функції також необхідно вказати , який код виконувати для обробки вхідних HTTP-запитів за вказаною URL-адресою. Для цього ми використовуємо метод server.on(). Цей метод приймає два параметри. Перший - це шлях до URL-адреси, а другий - це назва функції, яку ми хочемо виконати, коли ця URL-адреса потрапляє.

Наприклад у Лістингу 4.2 перший рядок вказує, що коли сервер отримує HTTP-запит на кореневий (/) шлях, він ініціює функцію handle_index(). Зверніть увагу, що вказана URL-адреса є відносним шляхом.

Лістинг 4.2 – Пов’язання функцій обробника із шляхом

```
server.on("/", handle_index);
server.on("/waterPlants_WATER_ON_DRACAENA",
handle_WATER_ON_DRACAENA);
server.on("/waterPlants_WATER_ON_SANSEVIERIA",
handle_WATER_ON_SANSEVIERIA);
server.on("/waterPlants_OFF", handle_WATER_OFF);
server.begin();
```

Функції handle_WATER_ON_DRACAENA() та handle_WATER_ON_SANSEVIERIA() функції, які виконуються коли ініціюється GET запит на розгорнутий сервер.

GET запит – це один із HTTP запитів, який запрошує уявлення ресурсу. Запити з використанням цього методу можуть тільки отримувати дані.

Тепер, щоб запустити веб-сервер, необхідно викликати метод begin на об'єкті сервера.

У нашому випадку функції обробки (Лістинг 5.1) керують світлодіодом, який виконує функцію тестування розробленої системи. Вони будуть розглянуті у наступному розділі.

4.5.3 Функція void loop() у скрипті для Arduino

Після виконання циклу setup, програма переходить в цикл loop, який буде повторюватися до тих пір, поки на плату подано харчування. Якщо цикл містить одну команду, то вона буде виконуватися тисячі разів в секунду. Якщо ви вирішите написати скетч для миготіння світлодіодом на Arduino, то необхідно додавати в код затримку для виконання програми, інакше миготіння світлодіода не буде помітно.

Функція loop це те місце, куди ми повинні помістити команди, які будуть виконуватися весь час, поки включена плата Arduino. Почавши виконання з першої команди, мікроконтролер дійде до кінця і відразу ж перестрибне в початок, щоб повторити ту ж послідовність. І так безліч разів (до тих пір, поки на плату буде поданий електрику). Найбільш доречний переклад англійського слова loop в даном випадку - це цикл (петля) .

Функція loop в Aduino IDE має наступну конструкцію:

void loop () {

// основний код програми розташовується тут

}

У нашому випадку функція loop має тільки одну команду, яка забезпечує початок обробки вхідних HTTP-запитів (Лістинг 4.3) . Для цього нам потрібно викликати метод handleCLient() обьєкту server.

Лістинг 4.3 – Пов’язання функцій обробника із шляхом

```
void loop() {
  server.handleClient(); //Handling of incoming client requests
}
```

Так виглядає основна сторінка розгорнутого веб-сервера, якщо звернутися у браузері до його IP-адреси (Рисунок 4.5).

IP-адресу можна дізнатися за допомогою інструменту Serial Monitor у

Arduino IDE (рисунок 4.4).

```
Waiting to connect...
Waiting to connect...
Waiting to connect...
Waiting to connect...
Waiting to connect...
Waiting to connect...
IP address: 192.168.1.4
Server listening
```

Рисунок 4.4 – IP-адреса розгорнутого веб-сервера, отриманого за допомогою інструмента Serial Monitor у Arduino IDE

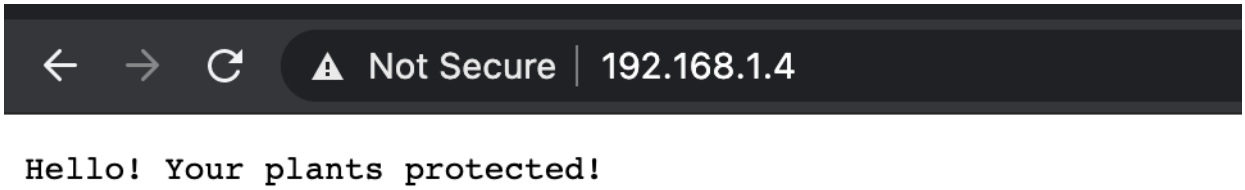

Рисунок 4.5 – Головна сторінка розгорнутого веб-сервера

5 ТЕСТУВАННЯ РОЗРОБЛЕНОГО ПРОТОТИПУ ПРИЛАДУ

5.1 Постанова задач на тестування

Будь-яка розроблена система або продукт повинні бути якісно протестовані перед подальшим розвитком та випуском на ринок у вигляді кінцевого продукту.

До таких приладів відносяться і системи, які входять до складу системи “розумний будинок”. Як виробник, нова технологія надає вам можливість забезпечити споживачам більший контроль над своїми будинками. Однак проблема полягає в тому, щоб переконатися, що ці нові технології працюють належним чином та належним чином працюють з іншими продуктами.

Клієнти не хочуть возитися з налаштуваннями на пральній машині - вони хочуть просто підключити її до мережі та забезпечити її роботу. То як ви переконаєтесь, що все просто працює для вашого клієнта? Ви перевіряли сумісність з іншими продуктами? Чи може ваша пральна машина спілкуватися з іншими бездротовими маршрутизаторами, наприклад? Як щодо продуктивності та асортименту вашого продукту? Якщо моя шайба знаходиться занадто далеко від мого бездротового маршрутизатора, що трапиться?

Відповідно до цього, розроблена у рамках проекту система також повинна бути протестована. Це зумовлено тим, що клієнт має бути впевнений, що їх рослини знаходиться у повній безпеці і будуть политі у вказаний час.

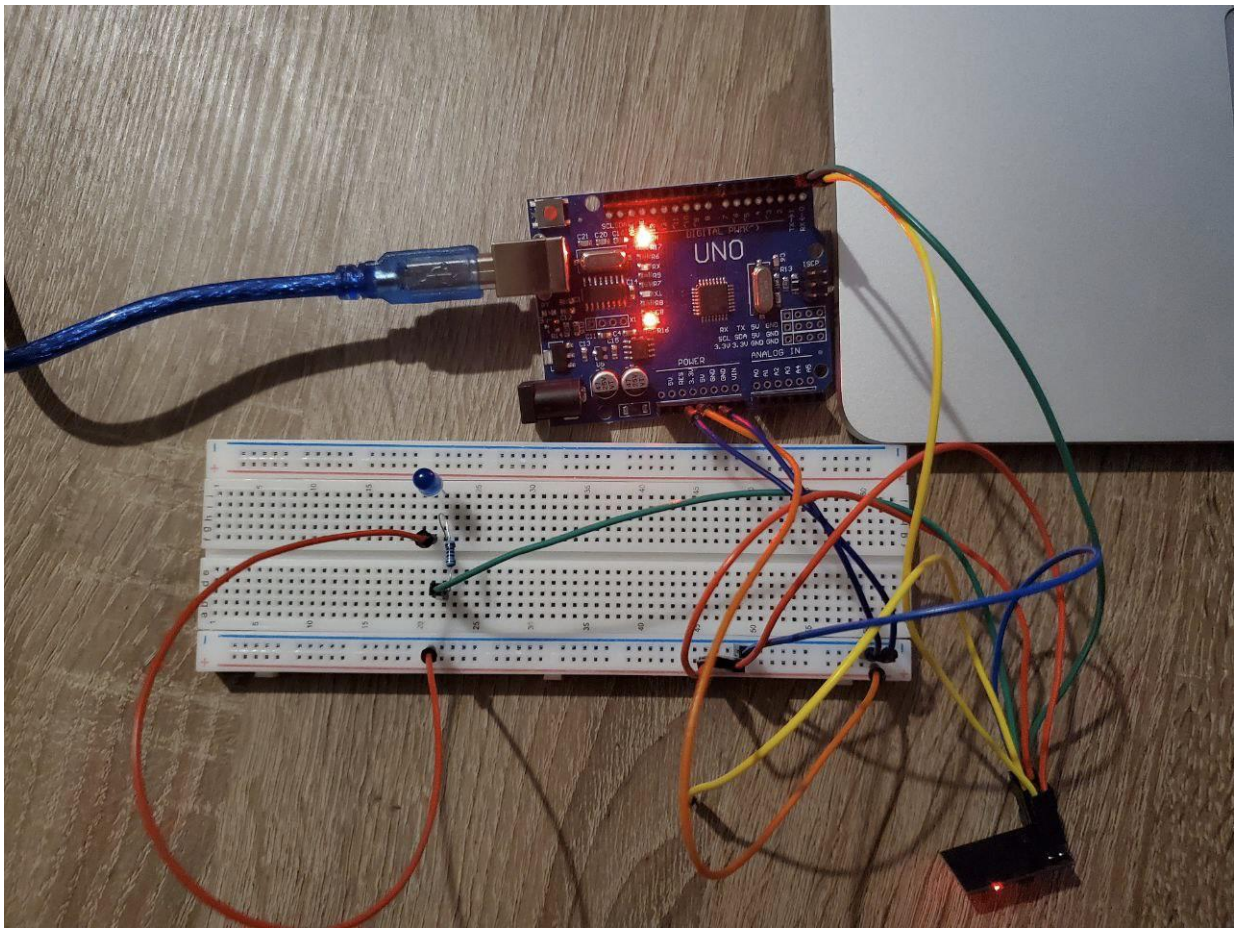

Рисунок 5.1 – Зовнішній вигляд прототипу систем

5.2 Тестування мобільного застосунку

Основним аспектом, що необхідно протестувати у мобільному додатку, є можливість посилання запиту на веб-сервер у конкретний момент часу, який встановлює користувач. Але оскільки веб-сервер ще не розгорнуто, було прийнято рішення, що для тестування будемо використовувати Toast Notification.

Розробник Android програми може «програмно» створювати та відкривати в інтерфейсі вікна спливаючі повідомлення Toast. Об'єкт Toast не можна описати в будь-якому файлі в форматі xml, Toast можна тільки сформувати в коді java. Додаток зберігає свою працездатність для користувача при появи повідомлення Toast. Як правило, спливаюче

повідомлення Toast використовується для представлення коротких текстових повідомлень. Наприклад, при натисканні на кнопку формування "Замовлення" спливає повідомлення з текстом «Товар буде обраний». Через пару секунд повідомлення зникає. Це і є Toast повідомлення.

Для створення спливаючого повідомлення необхідно ініціалізувати об'єкт Toast за допомогою методу Toast.makeText (), а потім викликати метод show () для підтвердження.

За допомогою функції: Toast.makeText(context, "Time to water flower!"+intent.getAction(), Toast.LENGTH_LONG).show(); у заданий час на мобільний телефон приходить повідомлення, яке вказує на те, що необхідно полити конкретну рослину. Виглядає це так:

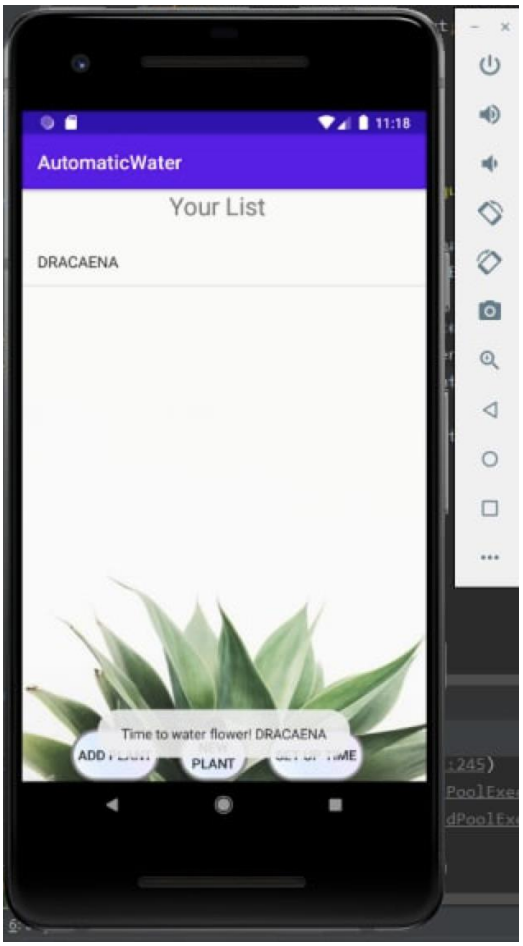

Рисунок 5.2 – Toast Notification у зазначений час для конкретної рослини

5.2 Тестування апаратної складової

Основним аспектом, що необхідно протестувати в апаратній складовій системи - є поведінка при запиті на розгорнутий сервер за конкретну адресу. Для цього у браузері можна прописати IP-адресу/"та адресу необхідної сторінки".

Для того, щоб протестувати реакцію на запит до сервера, було вирішено додати до готової системи світлодіод, який має блимати із різною частотою, ґрунтуючись на виді рослини.

Світлодіоди відмінно служать в пристроях для різного роду індикації. Вони споживають мало електроенергії і при цьому довговічні.

В даному прикладі ми використовуємо найпоширеніші світлодіоди діаметром 5 мм. Також поширені світлодіоди діаметром 3 міліметри, ну і великі світлодіоди діаметром 10 мм.

Підключати світлодіод безпосередньо до батарейці або джерела напруги не рекомендується. По-перше, треба спочатку розібратися, де саме у світлодіода негативна і позитивна ноги. Ну а по-друге, необхідно використовувати струмообмежуючі резистори, інакше світлодіод дуже швидко перегорить.

Якщо ви не будете використовувати резистор з світлодіодом, останній дуже швидко вийде з ладу, так як через нього буде проходити дуже велика кількість струму. В результаті світлодіод нагрівається і контакт, що генерує світло, зруйнується.

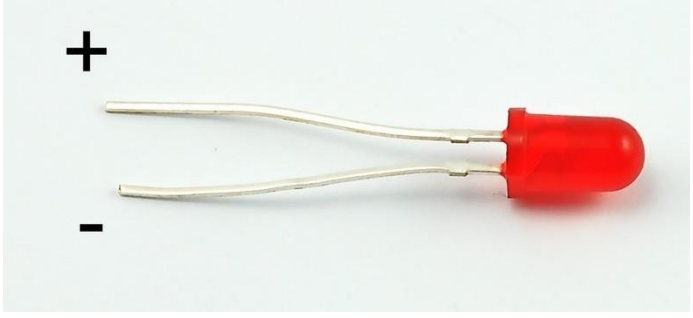

Рисунок 5.3 – Приклад типу використаного світлодіоду

Необхідно зрозуміти що таке резистор взагалі. З назви можна здогадатися, що резистори пручаються потоку електрики. Чим більше номінал (Ом) резистора, тим більше опір і тим менше струму пройде по ланцюгу, в якій він встановлений. Ми будемо використовувати цю властивість резисторів для регулювання струму, який проходить через світлодіод.

Щоб додати світлодіод до схеми, необхідно підключити пін GPIO2 на модулі ESP-01 до одного із контактів обраного світлодіоду. Інший контакт необхідно підключити до “землі” у макетній платі.

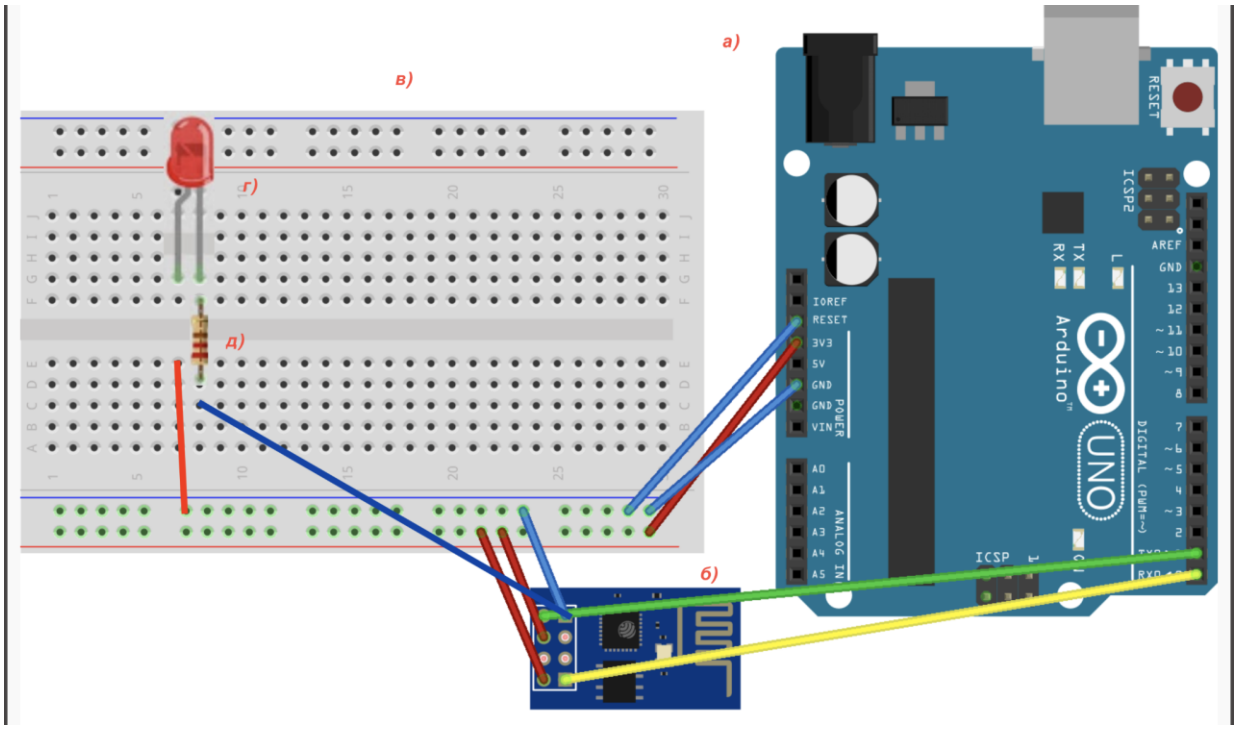

Рисунок 5.4 – Схематичне зображення пристрою та його компонентів: а) плата Arduino UNO R3; б)Wi-Fi модуль ESP8266; в) макетна плата; г) світлодіод (синій); д) резистор

Отже, якщо у браузері ввести “http://192.168.1.4/waterPlants_ON_DRACAENA” то ми можемо побачити реакцію на наш запит.

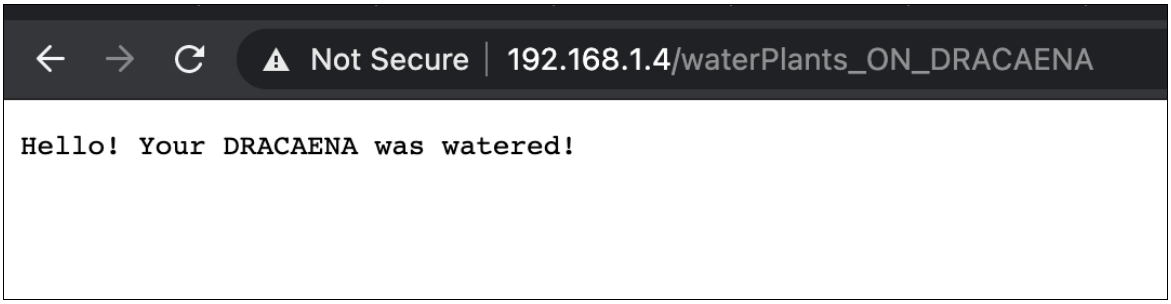

Рисунок 5.5 – Сторінка розгорнутого веб-сервера, як реакція на GET запит

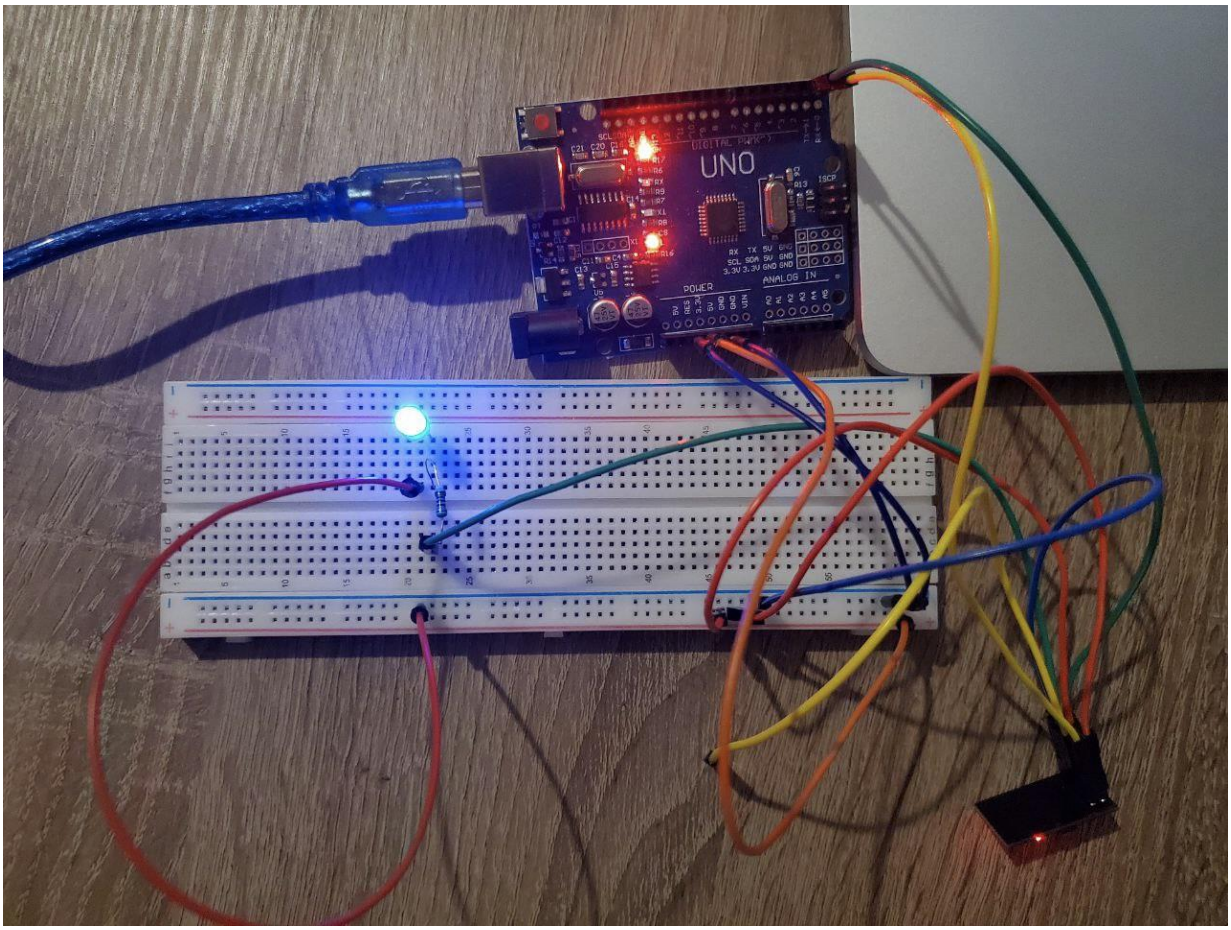

Рисунок 5.6 – Реакція апаратури на GET запит до веб-сервера

5.3 Тестування розробленої системи в цілому

Оскільки основні аспекти були протестовані та описані вище, основним аспектом тестування всієї системи полягає в тому, щоб перевірити, що запит з мобільного додатку посилається корректно до веб-серверу.

За допомогою функції Background().execute ("http://192.168.1.4/waterPlants_ON_DRACAENA"); був відправлений запит до веб-серверу і реакція апаратної складової була такою же як було описано вище.

ВИСНОВКИ

В даній роботі було представлено сучасний підхід до створення систем автоматичного поливу рослин.

Виходячи з поставленої задачі були розглянуті наявні рішення цієї проблеми – різноманітні мобільні додатки для нагадування про полив рослин та системи розумного будинку. Серед них, як широко розповсюдженні на ринку, так і різні специфічні, або новітні рішення, що не мають великого використання через свою недосконаліть, або велику ціну.

На основі розглянутого, запропонована більш автоматизована система, що дозволяє максимально полегшити догляд за рослинами, коли користувач виїжджає на відпочинок, або просто не знаходить час на турботу про рослини.

Для підключення системи до інтернету за допомогою Wi-Fi технології було розглянуто модуль ESP-01 . У якості апаратної та програмної платформи були обрані плата Arduino UNO на базі мікроконтролеру ATmega328P та середовище розробки Arduino IDE.

Для розробки мобільного додатку було обрано операційну систему Android. У якості програмної платформи було обрано середовище розробки Android IDE та мову програмування Java.

Також були розглянуті інші компоненти, необхідні для створення тестового прототипу такої системи та їх визначальні особливості перед іншими рішеннями. На підставі цього, запропонована схема кінцевого приладу.

Згідно до розробленої схеми, був зібраний прототип, що відповідає встановленим вимогам. Був сформований програмний код у середовищі Arduino IDE для здійснення встановлених операцій та виконання симуляції поливу рослин.

Крім того, був створений мобільний додаток для пристроїв на базі

системи Android з метою керування та налаштування розробленої системи.

Для підтвердження працездатності та безпечності системи, було проведене повноцінне тестування всіх аспектів роботи приладу. Пристрій пройшов всі тести без виникнення непередбачених подій та збоїв у роботі системи.

У разі подальшого розвитку проекту є ряд можливих модифікацій та вдосконалень:

1) збільшення кількості заздалегідь встановлених графіків поливу рослин;

2) заміна апаратної платформи на більш продуктивну, наприклад Raspberry PI;

3) масштабованість та тестування системи на повноцінних системах поливу із залученням клапанів для подачі води;

4) розробка і вдосконалення веб-сервісу для автоматизованої системи поливу рослин, щоб користувач мав змогу використовувати для налаштування не тільки мобільний телефон, а й свій комп’ютер.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Петин В.В. Створення розумного будинку на базі Arduino. – ДМК ПРЕСС, 2018. – 449 с.

2. ТЕХНОЛОГІЯ «РОЗУМНИЙ ДІМ»: МАЙБУТНЄ ВЖЕ ПОРУЧ [Текст]. – Т. А. Волосова – Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

3. Программы и приложения для Android [Електронне джерело]. – Режим доступу до ресурсу: http://android-phones.ru/.

4. Методичні вказівки до виконання лабораторних робіт з дисципліни “Програмування малих та мобільних платформ” для магістрів спеціальності 121 "Інженерія програмного забезпечення" / Укл.: О.О. Степаненко, Ю.В. Твердохліб.– Запоріжжя: ЗНТУ, 2016. – 59 с.

5. Рік Роджерс, Джон Ломбардо Android. Разработка приложений – М.: ЕКОМ Паблішерс, 2010 – 400 с.

6. Архитектура Android-приложений [Електронне джерело]. – Режим доступу: http://habrahabr.ru/post/141201/.

7. Сетевые и облачные контроллеры: различия и преимущества – [Электронный ресурс] – 2021. – Режим доступу до ресурсу: https://worldvision.com.ua/setevye-i-oblachnye-kontrollery-razlichiya-i-preimushchestva/

8. Плата Arduino Uno R3: схема, описание, подключение устройств. [Электронный ресурс] – 2019. – Режим доступу до ресурсу: https://arduinomaster.ru/platy-arduino/plata-arduino-uno/

}