МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

ФАКУЛЬТЕТ МЕХАТРОНІКИ ТА КОМП'ЮТЕРНИХ ТЕХНОЛОГІЙ

КАФЕДРА КОМП'ЮТЕРНИХ НАУК

Дипломна магістерська робота

на тему: Розробка програмного забезпечення для визначення показників екосистеми в приміщенні

Виконала: студентка групи МгІТ-21
спеціальності 122 Комп'ютерні науки
освітньої програми Комп'ютерні науки
Діана КОЧУК
Керівник:
к.т.н., доц. Тетяна АСТІСТОВА

Рецензент
_____

Київ 2022

КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ТЕХНОЛОГІЙ ТА ДИЗАЙНУ

Факультет мехатроніки та комп'ютерних технологій

Кафедра комп'ютерні науки

Спеціальність 122 Комп'ютерні науки

Освітня програма Комп'ютерні науки

ЗАТВЕРДЖУЮ

Завідувач кафедри комп'ютерні науки

_____Володимир ЩЕРБАНЬ

«_____»_____2022_____року

З А В Д А Н Н Я

НА ДИПЛОМНУ МАГІСТЕРСЬКУ РОБОТУ СТУДЕНТКІ
Кочук Діані Миколаївні

1.Тема роботи: Розробка програмного забезпечення для визначення показників екосистеми в приміщенні

науковий керівник роботи: Астістова Тетяна Іванівна, к.т.н., доц.,

затверджені наказом закладу вищої освіти від 28 . 09._2022 року № 180-уч.

2. Строк подання студентом роботи 12.11. 2022р.

3. Вихідні дані до роботи:

Розробка кафедри комп'ютерних наук

4. Зміст дипломної роботи (перелік питань, які потрібно розробити)

РОЗДІЛ 1. Огляд систем підтримки діалогових процесів; РОЗДІЛ 2. Алгоритм розробки системи та її моделювання; РОЗДІЛ 3. Розробка та практична реалізація системи. Додатки-програмні коди модулів системи

5.Консультанти розділів дипломної магістерської роботи

Розділ	Ім’я, прізвище та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Вступ	Тетяна АСТІСТОВА, к.т.н., доц.		
Розділ 1	Тетяна АСТІСТОВА, к.т.н., доц.		
Розділ 2	Тетяна АСТІСТОВА, к.т.н., доц.		
Розділ 3	Тетяна АСТІСТОВА, к.т.н., доц.		
Висновки	Тетяна АСТІСТОВА, к.т.н., доц.		

1. Дата видачі завдання: 10. 2022

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломної магістерської роботи	Терміни виконання етапів	Примітка про виконання
1	Вступ		
2	Розділ 1. . Огляд систем підтримки діалогових процесів		
3	Розділ 2. Алгоритм розробки системи та її моделювання		
4	Розділ 3. Розробка та практична реалізація системи		
5	Висновки		
6	Оформлення дипломної магістерської роботи	31.10.2022	
7	Здача дипломної магістерської роботи на кафедру для рецензування	01.11.2022	
8	Перевірка дипломної магістерської роботи на наявність ознак плагіату	04.11.2022	
9	Подання дипломної магістерської роботи на затвердження завідувачу кафедри	07.11.2022	

Студент _____ Діана КОЧУК

Науковий керівник роботи _____ Тетяна АСТІСТОВА

Директор НМЦУПФ _____ Олена ГРИГОРЕВСЬКА

АНОТАЦІЯ

Кочук Д. М. Розробка програмного забезпечення для визначення показників екосистеми в приміщенні.

Дипломна магістерська робота за спеціальністю 122 - «Комп'ютерні науки». – Київський національний університет технологій та дизайну, Київ, 2022 рік.

В основу розробки системи для визначення показників екосистеми в приміщенні було покладено Minimum viable product – продукт з мінімальним функціоналом та мінімальними затратами ресурсів. Концепція розробки системи розглядається на прикладі розумного гуртожитку, де одним із компонентів системи є технологія Інтернет речей, в якій різні пристрої підключені до Інтернету та взаємодіють один з одним. Це дозволить ефективно ідентифікувати проблему у разі відхилень від стандарту при роботі пристроїв та полегшить передбачення можливих аварій, несправностей.

На основі аналізу основних видів систем з підтримки діалогових процесів було обрано основний функціонал створення телеграм чат-боту на платформі Telegram та проведено порівняння наявних аналогів чат-ботів.

В роботі було розроблено архітектуру та модулі системи на основі чат-бота в інформаційній системі Telegram. Програмний продукт був написаний на мові Python; в якості платформи розробки (IDE – інтеграційна платформа розробки) було використано PyCharm.

Ключові слова: Telegram, Python, чат-бот, інформаційні системи

ANNOTATION

Kochuk D. M. Development of software for determining indoor ecosystem indicators.

Diploma master's thesis in specialty 122 - "Computer science", - Kyiv National University of Technology and Design, Kyiv, 2022.

The basis of the development of the system for determining the indicators of the indoor ecosystem was the Minimum Viable Product – a product with minimal functionality and minimal resource costs. The concept of system development is considered on the example of a smart dormitory, where one of the components of the system is the Internet of Things technology, in which various devices are connected to the Internet and interact with each other. This will allow to effectively identify the problem in case of deviations from the standard during the operation of the devices and will facilitate the prediction of possible accidents and malfunctions.

Based on the analysis of the main types of systems supporting dialogue processes, the main functionality of creating telegram chatbots on the Telegram platform was chosen and a comparison of existing chatbot analogues was carried out.

In the work, the architecture and modules of the system based on the chatbot in the Telegram information system were developed. The product software was written in Python; PyCharm was used as a development platform (IDE - integrated development platform).

Keywords: Chatbot, information systems, Telegram, Python, chart

Перелік скорочень умовних позначень

ІКТ (Інформаційно-комунікаційні технології) – термін, який підкреслює роль уніфікованих технологій та інтеграцію телекомунікацій (телефонних ліній та бездротових з'єднань), комп'ютерів, підпрограмного забезпечення, програмного забезпечення, накопичувальних та аудіовізуальних систем, які дозволяють користувачам створювати, одержувати доступ, зберігати, передавати та змінювати інформацію.

COM-порт – це послідовний двонаправлений інтерфейс, призначення якого полягає в обміні бітової інформацією. Цей порт називається послідовним, оскільки він забезпечує передачу інформації біт за бітом.

DDoS-атака – це атака на відмову в обслуговуванні, розподілена атака на відмову в обслуговуванні (англ.DoS attack, DDoS attack, (Distributed) Denial-of-service attack) — напад на комп'ютерну систему з наміром зробити комп'ютерні ресурси недоступними користувачам, для яких комп'ютерна система була призначена.

IoT (англ. Internet of Things, IoT) — Інтернет речей, концепція мережі, що складається із взаємозв'язаних фізичних пристроїв, які мають вбудовані давачі, а також програмне забезпечення, що дозволяє здійснювати передачу і обмін даними між фізичним світом і комп'ютерними системами в автоматичному режимі, за допомогою використання стандартних протоколів зв'язку.

JSON (англ. JavaScript Object Notation, укр. запис об'єктів JavaScript, вимовляється джейсон) – це текстовий формат обміну даними між комп'ютерами. JSON базується на тексті, може бути прочитаним людиною. Формат дає змогу описувати об'єкти та інші структури даних.

ЗМІСТ

ВСТУП

Актуальність теми. XXI століття є початком стрімкого зростання технічного прогресу та інформаційної революції, яка пов'язана з використанням інноваційних рішень. З’являється термін Smart City (розумне місто).

Концепція розумного міста пов'язана з існуванням глобальної мережі, що з'єднує безліч пристроїв та датчиків, які можуть самостійно обмінюватися інформацією для більш ефективного функціонування та відповідності потребам його жителів. Система Smart City функціонує за рахунок збору та безперервної обробки великої кількості даних, які оновлюються щосекунди та надходять з інформаційних каналів (камер відеоспостереження і фотофіксації, засобів відеоаналізу, засобів зв'язку) та комп'ютерних інформаційних технологій.

Концепцію Smart - City можна адаптувати до наших гуртожитків, як один із модулів вирішення глобальної проблеми міста.

Завдання дослідження. Завданням розробки моделі моніторингу екосистем було створити систему, яка дасть змогу в реальному часі отримувати інформацію з об’єктів і використовувати всі ресурси більш продуктивно, отримуючи інформацію з пристроїв, сенсорів. В роботі ця концепція розглядається на прикладі «Smart» гуртожитку, в якому різні пристосування взаємодіють один з одним.

Об’єкт та предмет дослідження. Об’єктом даного дослідження є розробка програмного забезпечення для відтворення температурних показників приміщення та їх контролю.

Мета дослідження. Розробка моделі та програмного забезпечення для визначення показників екосистеми в приміщенні та ідентифікування відхилень від стандарту в реальному часі, на відстані та прийняття рішень для їх усунення.

Методика дослідження. В основу розробки системи було покладено Minimum viable product - продукт з мінімальним функціоналом. та мінімальними затратами ресурсів. Концепція розробки системи розглядається на прикладі розумного гуртожитку, де одним із компонентів системи є технологія інтернет речей, в якій різні пристрої підключені до Інтернету та взаємодіють один з одним. Датчики дозволять пристрою посилати дані в комп'ютерну програму, яка буде збирати та аналізувати їх. Це дозволить ефективно ідентифікувати проблему у разі відхилень від стандарту при роботі пристроїв та полегшить передбачення можливих аварій, несправностей.

Результати дослідження. На основі аналізу характеристик модулів мікроконтролерів з різними типами датчиків, було обрано модуль бездротового зв'язку, який встановлений у пристрій системи моніторингу та типи датчиків для отримання показників екосистеми. Для реалізації задачі обрано відкриту мову програмування Processing та засоби розробки, які б задовольняли всі вимоги та мали необхідний функціонал. У якості інтерфейсу було обрано TelegramBot. Для з'єднання апаратних пристроїв, API та Інтернет-служб було обрано Node-RED, як інструмент програмування. Для написання бота був використаний пакет RedBot для NodeRED. Розроблено структурні елементи підсистеми та програмний код на прикладі плати для датчика.

Наукова новизна. Запропоноване оригінальне рішення для розробки інформаційної моделі показників екосистеми приміщення на відстані, в реальному часі на основі елементної бази обраного модуля моніторингу, протоколів зв’язку та інтерфейсу взаємодії з користувачем через чат - бот.

Практична значимість. Система була протестована на даних, які були отримані в гуртожитку Київського національного університету технологій то дизайну в травні 2021 року. Хостинг для роботи даного середовища було розгорнуто на Amazon Web Services (AWS) EC2. Результати тестування показали працездатність системи, можливість отримувати, аналізувати

інформацію в реальному часі та оперативно реагувати на небезпечні ситуації. Все це дасть можливість покращити життя студентів в гуртожитку та забезпечити своє здоров’я від впливу негативних чинників.

Апробація результатів роботи. Розроблено програмне забезпечення для визначення показників екосистеми в приміщенні дозволяє здійснювати конроль за показниками та оперативно реагувати, використовуючи для спілкування чат-бот.

1. Результати роботи були опубліковані в наступних виступах та статтях: Астістова Т. І., Кочук Д.М, Аналіз та характеристика технології «Internet of things»/ Т.І. Астістова, Д.М. Кочук// Інформаційні технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп’ютерних наук та технологій. – К. : Освіта України, 2021 р. . – С. 224 – 227
2. 2. Astistova T.I, Software development for technology"Internet of things"/ T.I. Astistova, D.M. Kochuk // Тези V Міжнародної науково-практичної конференції «Мехатронні системи: інновації та інжиніринг – «MSIE-2021» К. КНУТД , 4 листопада 2021р. - С.57-58 3.
3. Астістова Т. І., Кочук Д.М. Розробка концепції інформаційної системи «Smart city»/ Т. І. Астістова, Д.М.Кочук //Інформаційні технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп’ютерних наук та технологій. – К. : Освіта України, 2021 р. . – С. 217 – 22

РОЗДІЛ 1. ОГЛЯД СИСТЕМ ПІДТРИМКИ ДІАЛОГОВИХ ПРОЦЕСІВ

1.1. Мета та завдання розробки

XXI століття є початком стрімкого зростання технічного прогресу, який суттєво впливає на функціонування та розвиток управління, ставлячи нові завдання та цілі,. В 1960-х і 1970-х роках з'являється термін Smart City (розумне місто). Розумні міста є результатом поточної інформаційної революції, яка пов'язана з використанням інноваційних ІКТ-рішень.

Концепція розумного міста базується на постійному технічному прогресі та пов'язана з існуванням глобальної мережі, що з'єднує безліч пристроїв та датчиків, які можуть обмінюватися інформацією самостійно задля більш ефективного функціонування та відповідності потребам його жителів Система Smart City функціонує за рахунок збору та безперервної обробки великої кількості даних, які оновлюються щосекунди та надходять з інформаційних каналів (камер відеоспостереження і фотофіксації, засобів відеоаналізу, засобів зв'язку) та комп'ютерних інформаційних технологій.

Концепцію Smart - City можна адаптувати до наших гуртожитків, як один із модулів вирішення глобальної проблеми міста.

Метою дослідження є розробка моделі та програмного забезпечення для визначення показників екосистеми в приміщенні з можливістю ідентифікування відхилень від стандарту в реальному часі, на відстані та прийняття рішень для їх усунення.

Завданням розробки моделі моніторингу екосистем було створити систему, яка дасть змогу в реальному часі отримувати інформацію з об'єктів і використовувати всі ресурси більш продуктивно, отримуючи інформацію з пристроїв, сенсорів.

В роботі ця концепція розглядається на прикладі «Smart» гуртожитку, в якому різні пристосування взаємодіють один з одним.

Весь проект, це Minimum viable product (мінімально життєздатний продукт)—MVP, тобто це продукт з мінімальним функціоналом, який можна дати користувачам для використання в їх задачах. Minimum viable

product використовується для тестування ідей у розробці програм з мінімальними затратами ресурсів. наскільки продукт буде цінним та затребуваним на ринку [2,3]. Для нашої задачі Minimum viable product використовується для тестування ідей у розробці програм з мінімальними затратами ресурсів.

Об’єктом даного дослідження є розробка програмного забезпечення для відтворення температурних показників приміщення та їх контролю.

1.2. Поняття розумного міста

Насьогодні універсального визначення розумного міста не існує. Існує багато концепцій організації міського простору. Розумне місто використовують для більш ефективного використання ключових послуги та елементів міської інфраструктури, таких як: адміністрація, транспорт, освіта, громадська безпека, інформаційно-комунікаційні технології.. Розумне місто, це таке місто , яке пропонує мешканцям максимальну якість життя з мінімальним використанням ресурсів завдяки відповідній комбінації інфраструктурних систем (наприклад, транспорту або передачі енергії.

Можемо сказати, що розумне місто - це кластер, що складається з чотирьох основних елементів:

1. Цифрові простори, широкосмугова інфраструктура, електронні послуги Інтернет-інструменти для управління знаннями.

2. Населення, що здійснює наукомісткі заходи або сукупність таких видів діяльності.

3. Функціонування установи та процедури у галузі створення знань.

4. Доведена здатність до інновацій, вирішення проблем, які виникають вперше, тому що інновації та управління в умовах невизначеності є ключовими для оцінки інтелекту.

Інтегроване управління містом, яке і є управлінням в понятті розумного міста, засноване на координації ключових сфер міської політики , таких як просторове планування навколишнього середовища, транспорту, громадського транспорту, економічного розвитку з точки зору простору, об’єкта та часу.

Створення міських стратегій та політик що являють собою ключові аспекти, повинно відбуватися за активної участі всіх осіб та установ, що беруть участь у процесі міського розвитку. Це можуть бути суспільство, університети, підприємців та неурядові організації. (рис. 1.1)

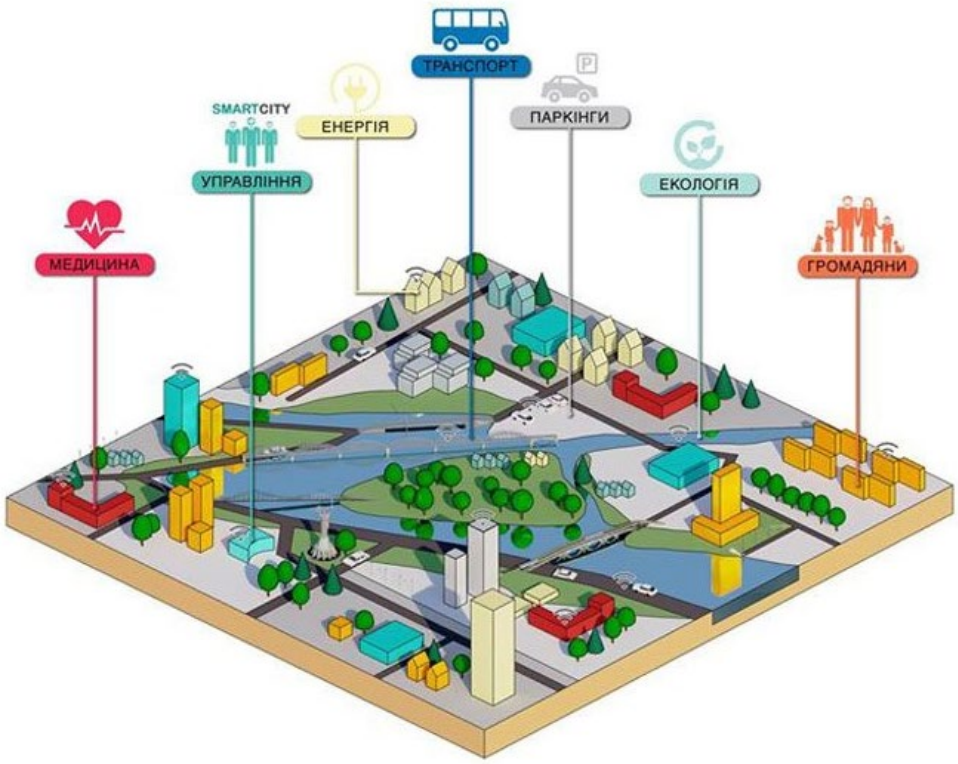


Рисунок 1.1 – Сфери, на які розповсюджуються послуги розумного міста

Існує безліч домашніх пристроїв, здатних виконувати дії і вирішувати певні повсякденні завдання об’єднуються в єдину систему управління, що однозначно раціоналізує життя людини. За допомогою системи розумного будинку, посилюється контроль власного приміщення та більш детальне розуміння кліматичних показників.

Керівники та мешканці міст виконують роль для створення та розвитку розумного міста, що призводять до вигод обом сторонам.

Інформаційно-комунікаційні технології в розумному місті виконують три

важливі завдання:

1. Здійснюють збір та передачу даних службам управління міським господарством
2. Виконують роль засобу зворотного зв'язку між адміністрацією міста та його жителями.
3. Забезпечують швидкі комунікаційні канали передачі інформації;

Розумне місто ставить перед розробниками одним із важливих завдань, це забезпечення громадської безпеки. З цією ціллю в містах застосовуються фотофіксації та камери відеоспостереження, засоби відеоаналізу, зв'язку та комп'ютерних інформаційних технологій .Все це дає можливість забезпечити безпечне міське середовище, комфортне для проживання.

За рахунок безперервної обробки та поновлення даних, що надходять з інформаційних каналів, функціонує система розумного міста

Розумне місто здатне самостійно простежити за належним рухом транспорту і пішоходів, за ситуацією в громадських місцях, за лікарнями і школами.

На рисунку 1.2 показани сфери розповсюдження послуг в розумних містах.

Ми бачимо, що це коло, де задачі кожної з підсистем цієї системи взаємопов’язані і вирішують головну задачу, це комфортне та безпечне життя його мешканців з мінімальними затратами ресурсів для міста.

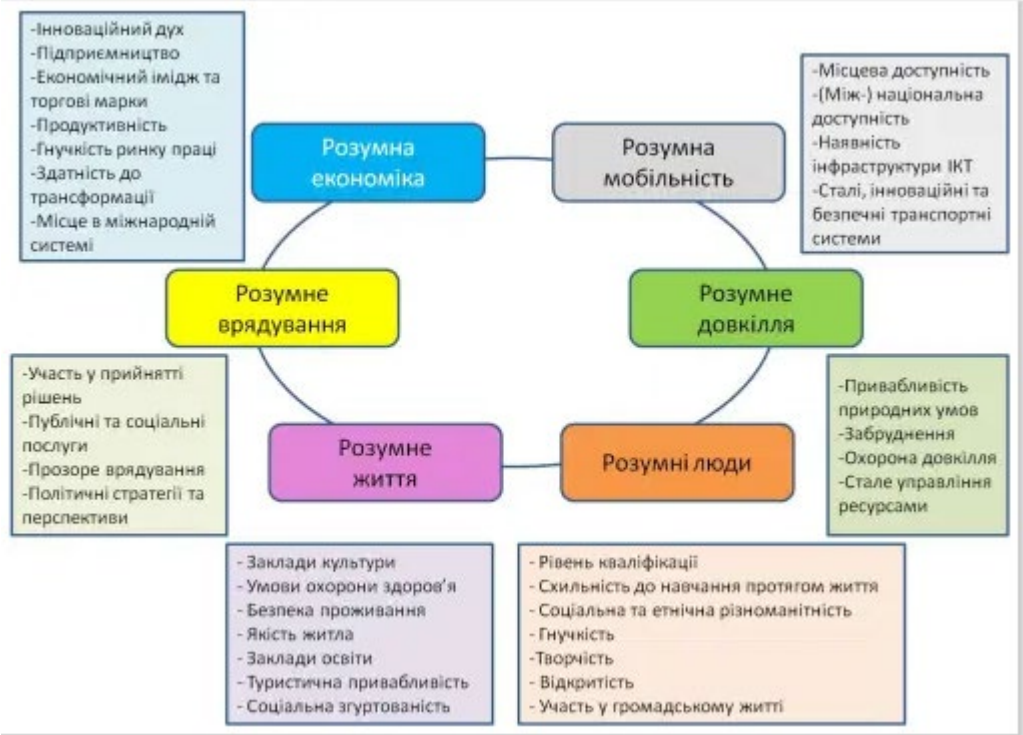


Рисунок 1.2-Сфери розповсюдження послуг в розумних містах

Розгянемо характеристику та задачі, які вирішує кожна з підсистем:

1. Розумна мобільність - вимірюється наявністю інформаційно-комунікаційної інфраструктури за рахунок розвитку інтегрованого, інноваційного та безпечного транспорту. Оцінюється доступність на місцевому рівні.

2. Розумна економіка - вимірюється міським підприємництвом та продуктивністю, адаптацією до змін, гнучкістю ринку праці, інноваціями, співпрацею між наукою та бізнесом та міжнародною співпрацею

3. Розумні люди – ознакою цієї складової є рівень кваліфікації, соціальна та етнічна різноманітність, креативність, навчання протягом усього життя, участь у суспільному житті. Все це відбувається за підтримки інформаційно-комунікаційних технологій. Розумні люди здатні прагнути до постійного поліпшення якості життя у місті.

4. Розумний спосіб життя - місто забезпечує широкий доступ до державних послуг таких, як: онлайн послуги, охорону здоров'я, якісну освіту, сучасну технічну інфраструктуру та соціальну сферу. Воно забезпечує високий рівень безпеки , пропозиції та управління вільним часом, турботу про стан природного середовища та зелених насаджень..

5. Розумне середовище - розумне місто за рахунок використання відновлюваних джерел енергії оптимізує споживання енергії, Розумне місто проводить заходи до зменшення викидів, що забруднюють навколишнє середовище. Управління міськими ресурсами в такому місті базується на принципі сталого розвитку з використанням інфраструктури на основі сучасних технологій. Прикладом цього може бути контроль зі сторони держави електроенергії, води, вуличного освітлення з метою оптимізації екологічних та фінансових витрат на їх експлуатацію. Державні служби проводять постійний моніторинг забруднюючих речовин; будівлі термічно модернізуються з метою зменшення споживання енергії.

6. Розумне врядування - люди приймають участь у приняті рішень, відбувається прозоре сврядування, виробляються перспективи та політичні рішення., вирішується ряд публічних та соціальних послуг.

Покращення якості життя мешканців города є найбільш ефективним та економічно бажаним. Саме інновації лежать в понятті ідеї розумного міста. Їх набагато простіші та швидші у прийнятті у великих центрах, але поширення відбувається лише пізніше.

Сьогодні багато людей звикли використовувати розумні рішення . Наприклад, жителі багатьох міст вже можуть використовувати інформаційні дошки на зупинках, які постійно оновлюють найближчий розклад руху, інтелектуальні пішохідні переходи , що покращують безпеку, або програми, що дозволяють відстежувати дорожній рух у місті та планувати подорож.

Сучасними рішеннями у розумному місті є:

1. Енергонезалежні будівлі,
2. Використання відновлюваних джерел енергії,
3. Місця для паркування,
4. Купівля квитків , оплата місця на парковці за допомогою спеціальних програм,
5. Лічильники, що вимірювання споживання води та енергії,
6. Вивіз сміття,
7. Інтелектуальне міське освітлення,
8. Електронні платежі,
9. Громадянські бюджети,

10. Міські велосипеди та скутери.

1.3 Анадіз переваг та недоліків розумного міста

а) Переваги, що надає розумне місто.

Технічний прогрес, принаймні на загальному рівні, сприяє економічному зростанню . Цієї тенденції дотримуються розумні міста. Сприяння інноваціям дозволяє підприємництву створювати нові робочі місця та конкурентоспроможні економіки. Краще організована інфраструктура призводить до менших витрат. Прикладами може слугувати скорочення часу поїздки на роботу та з роботи, а турбота про кваліфікацію мешканців допомагає їм знайти роботу за гідну зарплату. Такі тенденції, як машинне навчання, автоматизація та IoT, сприяють прийняттю розумних міст.

Взагалі, в управління містом може бути включена будь-яка сфера в ініціативу розумного міста. Гарним прикладом єможе бути розумний паркувальний лічильник. Це програма, яка допомагає водіям знаходити вільні місця для паркування без об'їзду переповнених кварталів міста. Розумний лічильник також дозволяє здійснювати цифрові платежі, тому немає ризику не вистачити монет на лічильнику.

Прикладом розумного міста може бути інтелектуальне управління дорожнім рухом. Воно використовується для моніторингу та аналізу транспортних потоків з метою оптимізації вуличного освітлення та запобігання занадто перевантаженості доріг на основі графіку часу доби або години пік.

Це одним прикладом розумного міста, ще однією гранню , може бути розумний громадський транспорт. Розумні транзитні компанії здатні координувати послуги та задовольняти потреби гонщиків у режимі реального часу, покращуючи ефективність та задоволення вершників.

Основними напрямками діяльності розумних міст є енергозбереження та ефективність. За допомогою розумних датчиків розумні вуличні ліхтарі тьмяніють, коли на дорогах немає автомобілів або пішоходів. На сьогодні, коли

наша країна веде війну з окупантами – рашистами , ми кожного дня бачимо,як важливо проводити енергозбереження. Для поліпшення роботи, технічного обслуговування та планування, а також для постачання електроенергії на вимогу та моніторингу відключень енергії бачимо важливе значення ехнологій .

б) Недоліки та проблеми розумних міст.

Першою проблемою розумних міст є те ,що ІКТ відіграють величезну роль в еволюції розумного міста, але вони порушують конфіденційність. Хоча моніторинг інтенсивності руху дозволяє зменшити затори на найбільш відвідуваних маршрутах, він також дозволяє відстежувати навіть один транспортний засіб.

Другою проблемою розумних міст є вразливість та потенційна нестабільність розумного міста перед кібератаками . При такій високій мірі залежності від технології недостатньої роботи в одному секторі. Висока гнучкість сучасних рішень працює як двосічний меч. Наявність датчиків і камер може сприйматися як втручання в приватне життя або державний нагляд Суперники розумних міст побоюються, що міські менеджери не будуть дотримуватись конфіденційності та безпеки даних. Вони бояться, що дані створені щодня громадянами, піддаються ризику злому чи зловживання. . Для вирішення цієї проблеми зібрані дані розумних міст повинні бути анонімізованими. Вони не повинні бути інформацією, що дозволяє ідентифікувати особу.

Основні переваги та загрози впровадження рішень розумного міста представлені в таблиці 1.1.

Таблиця 1.1. Основні переваги та загрози впровадження рішень розумного міста

Переваги	Загрози
Оптимізація витрат	Можливість виключити певні соціальні групи (наприклад, з низьким рівнем доходу, літні люди, інваліди)
Підвищення безпеки	Ризик для конфіденційності (захист персональних даних
Підвищення якості життя	Потенційна можливість використання даних несанкціонавними особами (наприклад, хакерами, іншими країнами)
Розвиток бізнесу	

Найбільшою проблемою, з якою стикаються розумні міста, є проблема зв'язку. Тисячі або мільйони пристроїв IoT, розкидані по місту, не існували б без надійного зв’язку, а саме розумне місто було б мертвим.

Громадська безпека, постачання електроенергії та природного газу, громадський транспорт, управління дорожнім рухом, управління водою та відходами, можуть бути ненадійними, особливо в міру старіння та зростання системи. Важливість цих операцій буде лише зростати у міру розширення міста та збільшення вимог до його інфраструктури.

Окрім послуг, розумні міста дозволяють передбачати заходи безпеки, такі як моніторинг районів із високим рівнем злочинності або використання датчиків, щоб забезпечити раннє попередження щодо таких інцидентів, як повені, зсуви, урагани або посуха.

1.4. Розумне місто і кібербезпека

Технології Інтернету речей дуже вразливі вразливі до кібератак. Купуючи новомодний гаджет, треба не забувати про кібербезпеку: дозволяючи Інтернету речей приймати за вас рішення, не нехтуйте елементарними запобіжними заходами.

Стійкі загрози в розумному місті є одними з найнебезпечнішими. Загрози цього типу складаються з кілька різних атак, які працюють в унісон.

Призначення атак полягає в порушині роботи міських служб. Для цього часто використовуючи шкідливе програмне забезпечення та вразливості програмного забезпеченні “нульового дня”.

Шкода, від хакерів може завдати тривалого збитку та залишити величезні масиви інфраструктури, які потребують заміни як фізичної , так і цифрової .

Атаки інфраструктури інтелектуального міста :

1. Викрадення пристроїв – один із найстрашніших аспектів кіберзлочинності. Зловмисники можуть взяти вразливі міста, наприклад світлофори та дорожня сигналізація, під контроль і це призведе до зриву всього процесу.

2. Викрадення любих типів даних – це найвідоміший кіберзлочин. Хакери проникаючи в банки даних можуть викрасти особисту інформацію. Інфраструктура розумних міст особливо вразлива до цього, і хакери, як відомо, витягують персональні дані з публічної платіжної інфраструктури з серйозними наслідками.

3. Атаки " Людина в середині" – хакер може перервати зв'язок між двома пристроями і виступити в якості відправника, надсилаючи неправдиву інформацію. Приклад такого вида атаки: хакер може отримати доступ до платформи мобільності та повідомити про затримки громадського транспорту , що може призвести до того, що більше людей візьмуть машину на роботу, спричиняючи приплив дорожнього руху, який призводить до зупинки міста.

4. Розподілена відмова в обслуговуванні – DDoS-атаки. Хакер завалює

систему, бомбардуючи її запитами, блокуючи послугу для тих, хто її потребує. Оскільки реальні користувачі не можуть отримати доступ до послуги, міські системи не зможуть підтримати своїх громадян.

5.Вимагач – використано комплекс видів атак для для викупу. Хакери використовують їх для компрометації або оприлюднення конфіденційних даних, якщо не виконуються певні вимоги. Виплата викупу створила б небезпечний прецедент.

1.4.1 Вирішення ризиків кібербезпеки в smart city

Існує два способи щоб мінімізувати ризики кібербезпеки, вживаючи декілька заходів обережності та залучаючи відповідну допомогу.

Перший захід. Найм незалежної охоронної компанії, яка намагається проникнути та знайти вади в мережі.

Сторонні фірми будуть імітувати атаки та намагатись використати будь-які слабкі місця. Після нападу охоронна компанія пояснить будь-які вразливі місця та запропонує реалістичні заходи захисту. Таке тестування на проникнення є чудовим, хоча краще розвивати інфраструктуру, яка неприступна з першого дня.

Другий захід. Забезпечити безпеку підключеної інфраструктури від хакерів, навіть якщо їм вдається отримати доступ.

Функції, які мають бути регулярною частиною міської програми кібербезпеки для захисту smart city :

1. Зашифровані дані.

Всі види даних завжди треба шифрувати. Шифрування використовують, щоб дані були марними та нечитабельними, за винятком тих, що мають ключ шифрування, який може їх розшифрувати. Такий метод називають скремблювання даних.

Скремблювання даних слід використовувати з ключем шифрування двофакторну автентифікацію. Оскільки інфраструктура smart city має справу з дуже конфіденційними даними, шифрування слід використовувати

як стандарт. Таким чином, якщо хакери отримують доступ до конфіденційних даних, що ідентифікують особисті дані, вони не мають можливості їх використовувати.

2. Постійний моніторинг безпеки .

Для цього вида безпеки потрібна спеціальна команда, яка може стежити за дорожнім рухом та шукати будь-які аномалії. Цей вид захисту можна автоматизувати за допомогою програмного забезпечення, яке може аналізувати об'ємні дані та шукати індикатори компромісу. Після виявлення потенційні зони ризику можна ізолювати, запобігаючи порушенням даних.

РОЗДІЛ 2. Алгоритм розробки системи та її моделювання

2.1 Поняття та концепція Інтернету речей

Четверта промислова революція почалась з поточних змін у сфері технологій та передачі даних. Все частіше стали обговорюються теми штучний інтелекту, нейронні мережи та Інтернет речей.

Інтернет речей (IoT) - це концепція, що дозволяє фізичним об’єктам здійснювати взаємодію між собою або з зовнішнім світом, частково або повністю без участі людини [5].

Коли кількість речей та предметів, підключених до Інтернету, перевищила кількість людей було введено термін Інтернет речей. Інтернет речей можна визначити як сукупність інтелектуальних об’єктів, які можуть реагувати на навколишнє середовище та обробляти інформацію, усуваючи розрив між фізичним та віртуальними світами і надсилати її іншим об’єктам за допомогою Інтернет-протоколів [4]. У 2000 році у світі було підключено до мережі 500 мільйонів пристроїв, на початок 2009 року ця кількість вже перевищила кількість мешканців Землі. У 2011 році завдяки популяризації смартфонів, планшетів та інших мобільних пристроїв кількість пристроїв, підключених до Інтернету, становила понад 13 мільярдів (населення досягло 7 мільярдів) (Raymond, 2014).

Вбудовані «давачі» і програмне забезпечення за допомогою використання стандартних протоколів зв'язку дозволяють здійснювати передачу і обмін даними між фізичним світом і комп'ютерними системами в автоматичному режимі. Саме цьому Інтернет більше не буде мережею підключених комп’ютерів, а стане мережею об’єднаних об’єктів»[5]. В даний час за допомогою таких систем як GSM, WiFi, Bluetooth, ZigBee, Z-Wave, але також маяки, фотоелементи або бездротових сенсорних мереж (Wireless Sensor) створюється нова концепія мережи.

За оцінками Гартнера, в 2020 році IoT охопить понад 26 мільярдів пристрої (Middleton et al., 2013). Очікується, що IoT знайде багато додатків у різних сферах, зокрема у розумних містах, енергетика, транспорт, промисловість,

будівництво, логістика, охорона здоров’я, IT-сектор.

В багатьох країнах інтернет речей не є інновацією. Це поняття перейшло до розряду завдань, які вирішуються повсякденно за допомогою пристроїв та алгоритмів. В галузях, які є великі за розмірами, інтернет речей та алгоритмізація можуть принести ефект миттєвої економії та швидкої монетизації . Саме в цих галузях ми і бачимо великі прориви в використанні інтернет речей.

Прикладами таких галузей може бути сільське господарство чи транспорт, їх логістика «кишить» новаціями, що вражають розум простої людині. При цьому сьогодні вже стоїть питання не створення новації, а її імплементації у великих масштабах. Багато з того, що необхідно для повсюдної зміни усталених операцій та дій, вже є, але потрібно інтегрувати та почати застосовувати ці технології

Очікування щодо швидкого розвитку Інтернету речей також пов'язані із застосуванням цієї технології в розумних містах, інтелектуальному будівництві та автомобілях, а також у промисловій автоматизації, відомій як промисловість 4.0. В таблиці 2.1 показані сфери застосування інтернет речей

Таблиця 2.1 – Сфери застосування інтернет речей

№	Назва	Опис і приклади застосування
1	Місто	Міське середовище з громадською інфраструктурою, наприклад, розумними лічильниками паркування, контролем якості води та вуличним освітленням.
2	Людина	Предмети, які слід ковтати та носити, пов’язані з моніторингом та покращенням здоров’я, добробуту та продуктивності праці, наприклад, розумні планшети, пульсометри, фітнес-ремінці.
3	Робоче	Організовані робочі місця, такі як будівельний

	середовище	майданчик, видобуток корисних копалин, наприклад, системи контролю робочого стану.
4	Будинок	Будинки та резиденції, наприклад розумний будинок, системи безпеки.
5	Торгівля та послуги	Місця для продажу та пропонування таких послуг, як готелі, ресторани, банки, магазини тощо. Прикладом можуть слугувати рекламні акції на основі місцезнаходження.
6	Виробниче середовище	Виробниче середовище, таке як заводи, ферми, наприклад самохідні навантажувачі.
7	Транспорт	Особисті транспортні засоби, такі як автомобілі, мотоцикли, велосипеди, наприклад, навігація, уникнення зіткнень, самохідні машини тощо.
8	Офіс	Комерційні та офісні будівлі, наприклад, розумні термостати та кондиціонери.
9	Навколишнє середовище	Усі інші зовнішні середовища, крім згаданих вище, визначаються як повітря і космос, логістика тощо, наприклад управління розташуванням флоту.

У серпні 2020 року стало відомо, що Google планує інвестувати $450 млн в компанію з надання електронних послуг для підприємств ADT. Ціль вкладень – розвиток сфери Інтернету речей. Інакше кажучи, підтримка IoT, "розумних" девайсів, смарт-будинків, великих даних. Інтернет речей, разом із штучним інтелектом і нейронними мережами, стає однією з найактуальніших тем 2020 року. У яких напрямках розвивається IoT сьогодні, поговоримо у матеріалі.

Приклади Інтернету речей включають безліч гаджетів, підключених до Інтернету. Особливість Інтернету речей (IoT) – у цьому, що звичний домашній прилад збирає інформацію, передає їх у “хмару” (чи інший сервер), там дані аналізуються, і подальші команди повертаються в девайс.

Наприклад, господиня будинку купує "розумний" холодильник. Завантажує продукти, на сенсорному екрані ставить біля страви термін зберігання. Вирушаючи в магазин, щаслива власниця холодильника заходить у додаток, і в ньому відображається інформація, які продукти закінчуються, а що ще залишилося на вечерю. Або екстремальніша версія (зрозуміло, в цьому випадку смарт-холодильник буде коштувати дорожче). Рефрижератор самостійно аналізує, які продукти необхідні господареві, як часто у будинку з'являється та чи інша їжа. Холодильник “віддає команду” додатку, який, своєю чергою, запитує у продуктовий Інтернет-магазин. Сума також автоматично знімається з картки, а господареві залишається лише відкрити двері кур'єру та прийняти замовлення.

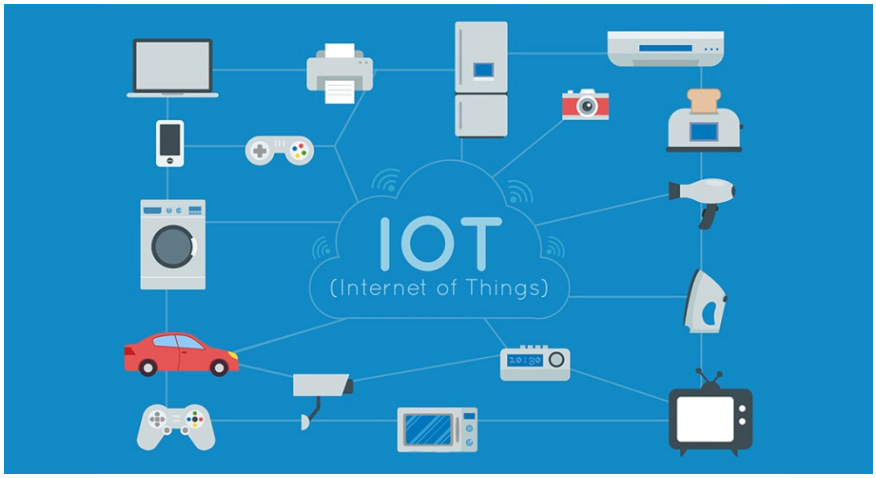
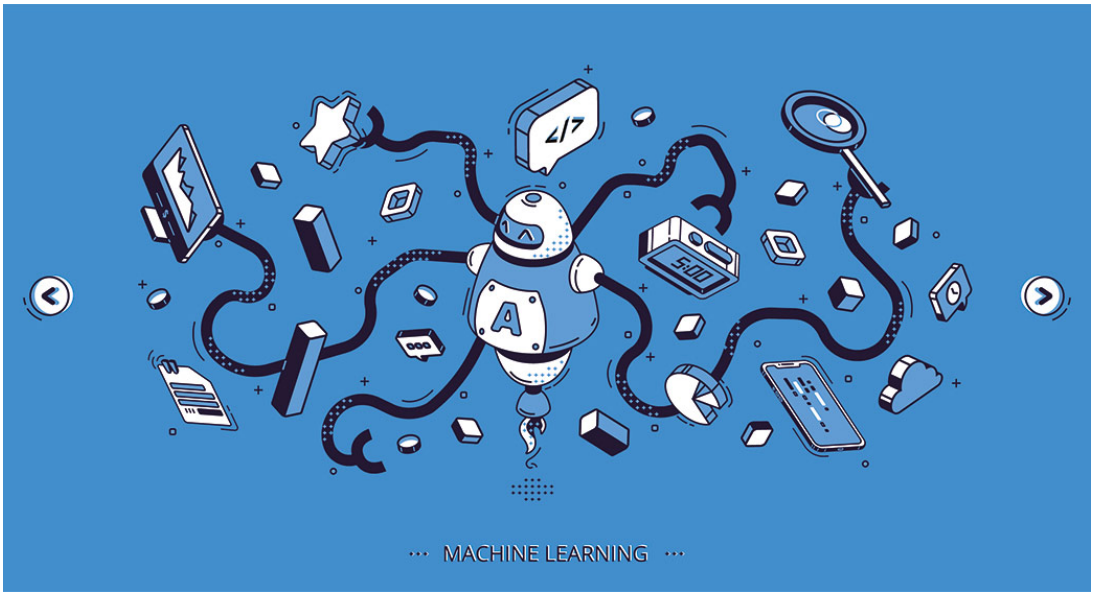


Рисунок 2.1. Приклади інтернет -речей

Якщо розбирати цю ситуацію з погляду технологій, виходить така картина. Холодильник збирає дані (продукти, терміни, частота закупівель, традиційні страви), а комп'ютер видає вердикт – що і як потрібно зробити. "Наказ" надходить у додаток, і смартфон завершує послідовність дій.

Точки доступу Інтернету речей обробляють колосальну кількість даних, які випливають із різних ресурсів – девайсів, підключених до Інтернету. Великі дані вимірюються у петабайтах, терабайтах та екзабайтах. У той же час далеко не кожна комп'ютерна система зможе проаналізувати таку кількість інформації.

Хоча і Big Data, і IoT збирають дані, лише Інтернет речей спроможний проаналізувати отримані відомості та миттєво передати команду девайсу.



Таким же чином працює більшість девайсів, які також являють собою Інтернет речей. Роботи-пилососи сканують будинок на наявність сміття та прибирають у квартирі, коли це необхідно. Інтуїтивні лампочки "реагують" на голос господаря та змінюють колір, залежно від переваг. Камери спостереження відбивають на смартфоні, що відбувається у будинку. Тепер не страшно залишати дитину з нянею, адже в режимі реального часу можна відстежити те, що відбувається.

Візьмемо приклад комерційної компанії, яка займається продажем косметики. Раніше маркетологи мали влаштовувати опитування серед покупців, розробляти стратегію, тестувати нову продукцію. Тепер достатньо встановити в магазині нейронні мережі, які працюють з великими даними. У основі лежить система машинного навчання: тобто комп'ютер на різних прикладах вчить, як людині треба приймати рішення у конкретних ситуаціях. Нейронні мережі спостерігають за споживачами, фіксуючи інформацію. Потім ці дані обробляються, причому завдяки можливостям Big Data це займає кілька хвилин. І комп'ютер видає маркетологам готову стратегію

подальшого розвитку.

Кожного року у систему IoT інвестують досить щедро. Вірять, що майбутнє – за Інтернетом речей, віртуальною реальністю та нейронними мережами. Хоча науково-технічний прогрес розвивається зі швидкістю світла, не слід забувати у тому, що сучасні технології залишаються вразливими. Купуючи новомодний гаджет, треба не забувати про кібербезпеку: дозволяючи Інтернету речей приймати за вас рішення, не нехтуйте елементарними запобіжними заходами.

2.2. Аналіз переваг використання інтернет речей.

Інтернет речей в процесі своєї роботи надє інформацію про датчики, забезпечує зв'язок від пристрою до пристрою, викликає широкий спектр програм.

Найпопулярніші програми в роботі інтернет речей:

1. Нова ефективність у виробництві. Вона створюється завдяки моніторингу машин та контролю якості продукції.

Інтернет речей дає можливість постійно контролювати та аналізувати машини, щоб переконатися, що вони працюють у межах необхідних допусків. Важливим є те , ще продукцію можна контролювати в режимі реального часу для виявлення та усунення недоліків якості.

2. Покращення відстеження та "обмеження" фізичних активів. Відстеження дозволяє компаніям швидко визначити місце розташування активів. Кільцеве огородження дозволяє їм переконатися, що коштовні активи захищені від крадіжок та вивезення.

3. Використання портативних пристроїв .

Дуже часто використовують інтернт- речей для задач моніторингу. Наприклад портативні пристрої для моніторингу аналітики здоров’я людини та умов навколишнього середовища.Портативні пристрої дозволяють людям

краще зрозуміти власне здоров'я та дозволяють лікарям віддалено контролювати пацієнтів. Ця технологія також дозволяє компаніям відстежувати стан здоров'я та безпеку своїх працівників, що особливо корисно для працівників, зайнятих у шкідливих умовах.

Рис. 2.2 – Основні переваги використання IOT

4. Підвищення ефективності та безпеки існуючих процесів.

Використання інтеннрнет речей для підвищення ефективності та безпеки у підключеній логістиці для управління флотом. Наприклад, компанії можуть використовувати моніторинг парку своїх , щоб направляти вантажівки в режимі реального часу для підвищення ефективності. Основні переваги використання IOT (рис. 2.2).

5. Внесення змін у бізнес-процеси.

Можливість віддаленого моніторингу машин дає змогу створювати нові бізнес-моделі продукту як послуги, коли клієнтам більше не потрібно купувати продукт, а натомість платити за його використання. Організації, які найкраще підходять для IoT - це ті організації, яким було б корисно використовувати

сенсорні пристрої у своїх бізнес-процесах. Прикладом цього є використання пристроїв інтернет речей для підключених активів для моніторингу працездатності віддалених машин та ініціювання викликів служби для профілактичного обслуговування.

2. 3. Огляд інтерфейсів користувача

Для розробки інтерфейсу системи визначення показників екосистеми в приміщені обрано платформу Telegram, а в якості інтерфейсу бази даних екосистеми було обрано TelegramBot.

Telegram є хмарним сервісом для обміну повідомленнями, що об'єднує новітні технології, розвинену екосистему. Він надзвичайно надійний, потужний, приватний і захищений, простий у використанні. така кількість переваг показує,чому користувачі обирають для комунікації саме цей сервіс. Telegram відіграє функцію платформи для розробки боту, а в якості інтерфейсу бази даних екосистеми було обрано TelegramBot.

Обираючи бот необхідно було визначити , до якої категорії він буде відноситись. У схожих програм є різні функції та можливості, тому така сфера у використанні віртуальних помічників постійно зростає. За своїм призначенням та ціллю роботи можна розділити на чотири основних типи роботів. Вони показані в табл. 1.1.

Таблиця 2.2 – Види чат - ботів

Види чат -	Категорії

ботів	Новини	Квитки	Їжа	Пошуковий
Розважальні	-	-	-	+
Консультанти	-	-	-	-
Помічники	-	-	-	+
CRM-система	-	-	-	-

Задачею бота є відтворювати автоматичу відповідь після вводу до нього команди користувача. Однією із головних функцій бота TelegramBot є можливість виконувати команди в чаті Telegram. Ці команди потім безпосередньо запускають дії або запитують інформацію. Після обробки запиту сервіси надсилають результат в на пристрій, а сам результат буде знаходитись у самому боті. Пошук проводиться у вигляді спілкування в інтерфейсі, де є певна прошивка, в якій прописано, які дані збиратимуться

На пристрої користувача завантажені API Google та месенджер Телеграм.

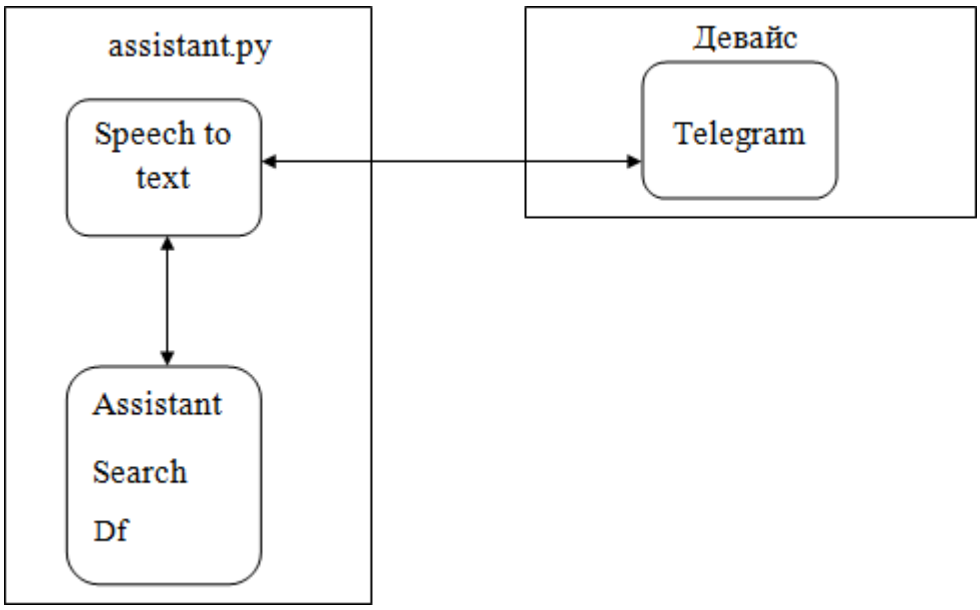


Рисунок 2.4. – Структура чат-боту

Розглянемо структуру чат – боту:

1. Додаток Telegram відіграє призначення платформи для розробки боту, а також його розміщення,

2. Assistant Search Df - assistant. py - файл з логічною роботою сервісів, котрі є у самому боті.

3. Speech-to-text recognition відіграє роль поєднання мов, яка після обробки аудіо чи тексту робить запит на сервіси Assistant, Search, Dialogflow. Запит обробляється і надсилається результат вже на пристрій, а в підсумку буде знаходитись у самому боті, де і був проведений запит. Весь необхідний пошук проводиться в постійному спілкуванні.

2.4. **Алгоритм розробки системи для визначення показників екосистеми в приміщенні**

Розглянемо на прикладі роботу алгоритму.

1. Спочатку треба обов’язково провести функціональне тестування.

Функціональне тестування, це перевірка функціоналу інтерфейсу системи:

- Перевірка роботи обов’язкових функцій боту;
- Тестування працездатності користувацьких форм інтерфейсу;
- Налаштування та перевірка роботи сповіщень.

2. Далі. Наш модуль моніторингу, який складається з двох датчиків підключають до WI-FI, який має вихід до інтернету. При цьому на модулі створюється точка доступу і далі користувачу потрібно зайти за певною ip-аддресою,192.168.4.1.

2. Після цього користувач потрапляє в налаштування. Далі потрібно вказати назву WI-FI, ввести пароль до WI-FI, до якого ми підключились та сервер на який скидати дані, вказуємо нашу кімнату,тобто ми налаштовуємо його під нашу кімнату. В інфтерфейсі вже є певна прошивка в якій прописано, які дані збиратимуться, далі скидається на серверт і власне опрацювання.

В нашому випадку це середовище NODered, де будуть зберигатися дані з датчиків. Вони будуть в цьому середовищі опрацьовуватися і далі розкидаються в телеграм.

2.5 . Аналіз мов програмування

2.5.1.Мова програмування Java.

Java є об’єктно-орієнтованою, строго типізована мовою для загального призначення. Її розробником була корпорація Sun Microsystems. Зараз підтримкою мови займається Oracle. Це одна з найбільш важливих методологій розробки. Вона ґрунтується на тому, що програма уявляється як сукупність об’єктів, і кожен з них є екземпляром певного класу. Класи у свою чергу утворюють ієрархію наслідування. Основною перевагою об’єктно-орієнтованого програмування є краща модульність програмного забезпечення. Тобто велику кількість функцій процедурної мови можна замінити на набагато меншу кількість класів, які будуть мати свої методи. Основні концепції об’єктно-орієнтованого програмування – це інкапсуляція,

поліморфізм, наслідування та абстракція

Програми, на писані на Java можна запускати на будь-яких платформах, тому що використовується віртуальна машина (JVM – Java Virtual Machine). Принцип її роботи полягає в тому, що написаний програмістом код спочатку транслюється в байт-код, а вже після цього виконується віртуальною машиною.

Синтаксис Java є C-подібним синтаксисом. Зокрема, об’єктна модель взята з мови C++, але її модифіковано.

Полегшено процес розробки додатків та усунуто деякі конфліктні ситуації, які могли з’явитися при розробці через помилки програміста.

У мові Java інкапсуляція реалізована за допомогою системи класів, які дозволяють зібрати інформацію про об'єкт в одному місці; пакетів, які групують класи по певному критерію, і модифікаторів доступу, якими можна позначити весь клас або його поле чи метод.

Існує чотири модифікатори доступу: private, default, protected і public. Private дозволяє отримувати доступ до доданих чи методів лише в межах одного класу, в якому їх оголошено. Default – це неявний модифікатор за замовчуванням, який дозволяє отримувати доступ до сутностей з будь-якого класу в межах одного пакету. Protected забезпечує доступ до даних для класів в межах свого пакету та для класів нащадків. Public надає повний доступ до сутностей з будь-якого пакету.

Головних принципів ООП є наслідування. Цей принцип дозволяє створювати ієрархічні структури об’єктів. Використовуючи цей принцип створюється клас предок. Він задає характеристики, поведінку та властивості певному набору пов’язаних об’єктів. Цей клас може наслідувати іншими класами. Їх називають нащадками. Вони успадковують всі характеристики класу предка і можуть доповнювати їх і модифікувати або доповнювати поведінку базового класу. На відміну від C++ в Java відсутнє множинне наслідування. Тобто класи нащадки можуть мати всього один батьківський клас. А ті класи, для яких немає явно вказаного суперкласу, наслідують клас

Object.

Поліморфізм – це можливість використання однойменних методів з однаковими або різними наборами параметрів в одному класі або в групі класів, пов’язаних відношенням наслідування. Завдяки цьому код стає більш гнучким. Використовується перегрузка методів або перевизначення. Перегрузка – це випадок, коли декілька методів мають однакове ім’я, але різний набір параметрів. А перевизначення – це можливість дочірного класу забезпечувати специфічну реалізацію метода, який уже реалізовано у суперкласі.

Абстракція – це теоретичний прийом, який дозволяє відокремитися від деяких неважливих властивостей та сконцентруватися на важливих. Фокусування розробника на конкретних властивостях об'єкта залежить від тих задач, які повинен вирішувати об'єкт. Наслідком такого підходу є те, що, якщо в імперативних мовах програмісту необхідно думати в термінах комп'ютерної логіки, то в об'єктно-орієнтованих мовах розробник думає в термінах проблемної сфери, в якій він розробляє програму.

Також мова Java має високу надійність. Завдяки введенню істинних масивів та забороні арифметики покажчиків програмісти в принципі не можуть стерти дані з пам’яті через неправильне використання покажчиків. Також виключена можливість явного виділення і звільнення пам’яті. Цим займається спеціальний механізм. І виключена можливість переплутати оператор порівняння на рівність з оператором присвоєння.

2.5.2. Мова програмування Processing

Мова програмування - це набір спеціальних інструкцій та правил їх запису у програмах для обчислювальної машини, які є записом алгоритмів у формі, зрозумілій для виконання комп’ютером.

Програмування - це є як написання коду, так і спосіб думати, як зробити щось нове та унікальне, а також обійти деякі обмеження існуючих програмних засобів.

Мова програмування Processing була створена ще в 2002 році.

Мова програмування Processing – це проект ініційований Бенжаміном Фраєм і Кейсі Різом. Він народився з ідей, вивчених в The Aesthetics and Computation Group в MIT Media Lab.

Processing - це повністю функціональна мова програмування, відкрита мова програмування, заснована на Java. Разом з тим, Processing - це і професійний інструмент, який використовується для сотень проектів високого класу в широкому діапазоні областей, від мультимедійних установок до візуалізації інформації, це не іграшка чи «навчальна» мова, як Скретч, незважаючи на своє коріння як навчальний засіб.

Processing побудовано на мові програмування Java, самне тому не доведеться робити надто великих зусиль при переході на інші мови програмування, таких як C/C++.

Processing постачається з невеликим, але досить ефективним середовищем розробки (IDE), чудовою документацією, великою бібліотекою розширень та значним набором прикладів та демонстраційних програм, він є безкоштовним, з відкритим кодом і добре задокументовим - все це робить його дуже доступним.

Як мова програмування, Processing виступає лише як "шар" поверх Java. Весь код Processing спочатку перекладається на код Java. Це означає, що ви можете писати код Java та імпортувати бібліотеки Java у свій код Processing в (або за межами) Processing IDE. Методично це допомагає Processing служити мовою програмування яка має «безшовний» перехід до Java та інши повнофункціональни мов, таких як C/C++. Ви можете розпочати програмування за допомогою Processing, фактично опанувавши Java у звичному середовищі, а потім перейти до потужних інструментів.

Можна сказати що Processing – Java.

Це насправді просто Java, з дещо простішим синтаксисом, зі спрощеним маніпулюванням графікою у поєднанні з простим у використанні IDE. Коли ви натискаєте «Запустити», Processing запускає ваші файли .pde через препроцесор, який перетворює їх в один гігантський клас Java. Ваші

класи - це всі внутрішні класи вашого основного класу, який розширює PApplet. Всі ваші користувацькі об’єкти можуть отримати доступ до «глобальних» змінних та функцій, визначених у вашому основному класі, а також до всіх методів Processing API. Це означає, що ви можете використовувати багато класів зі стандартної бібліотеки Java. Разом з тим буде гарною ідеєю спробувати відкинути свої уявлення про Java під час першого навчання Processing.

Існує хибне сприйняття Processing як надто простої мови програмування, однак Processing - це також інструмент, який полегшує життя навіть досвідченим розробникам, він має є набагато більше, адже спільнота Processing надає бібліотеки, які розширюють функціональні можливості середовища на комп’ютерне бачення, роботу з аудіопристроями та різними типами інтерфейсу, таких як Kinect або Leap Motion. Існують бібліотеки для експорту PDF-файлів, роботи з вебкамерами, створення 3D-зображень, анімації, надсилання текстових повідомлень SMS, включаючи дані про погоду, створення типографіки та багато іншого.

Окрім цих можливостей, протягом багатьох років люди розробляли інструменти, які можуть зробити ваші навички Processing корисними у багатьох контекстах.

Кілька прикладів:

1.веб/браузер: Processing.js - це бібліотека JavaScript, яка дозволяє вам запускати код Processing в браузері.

2. p5.js - це бібліотека за допомогою якої ви можете конвертувати свої ескізи для інтеграції з вашими вебсайтами.

3. мобільні пристрої: Ви можете розробляти програми для Android за допомогою Processing, використовуючи IDE у режимі «Android".

Мова програмування мікроконтролерів Arduino теж має корені у Processing.

Саме тому була обрана мова Processing для своєї розробки.

РОЗДІЛ 3. РОЗРОБКА ТА ПРАКТИЧНА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1. Середовище розробки програмного забезпечення.

Апаратною платформою середовища системи буде система на кристалі з мікроконтролером ESP8266 ESP-12E; в якості середовища для програмування даних плат, використовується середовище розробки Arduino IDE - Arduino .

Відомо, що мікроконтролер – це специальна мікросхема, призначена для керування різними електроними пристроями. На сьогодні існує більше 200 модифікаций мікроконтролерів, які випускаються двумя десятками компаніями. Для вирішення задачі потрібно було проаналізувати та обрати тип мікроконтролер, який би відповідав мети завдання.

Нами було проаналізовано мікроконтролери Arduino UNO, Arduino Nano, MEGA, Espressi. Аналізуя технічіх показники мікроконтролерів, такі як: об’єм пам’яті, тактову частоту, тип мікроконтролера, потрібно було обрати такий, що відповідав би ідеї розробки. Було обрано мікроконтролер Espressif («рішення на кристалі»), який є модулем бездротового зв'язку і може бути встановлений у пристрої системи розумного будинку, інтелектуальному пристрої або модулі безпеки. [7]. Під інтелектуальним пристроєм мається на увазі пристрій, яким можна керувати через Wi-Fi мережу.

Саме Espressif має мікросхему, на якій розміщений Wi-Fi та ядро системи. Плюс цього модуля в тому, шо його можна запрограмувати на мале споживання енергії. Це відбувається завдяки тому, що він відправляє дані, потім «падає в сон», надалі «прокидається», «скидає» дані, і знову "засинає". Espressi дасть змогу втілити ідеї Minimum viable product в життя, а Arduino не має такої можливості. На рисуноку 3.1 показан код та зовнішній вигляд середовища розробки Arduino IDE.

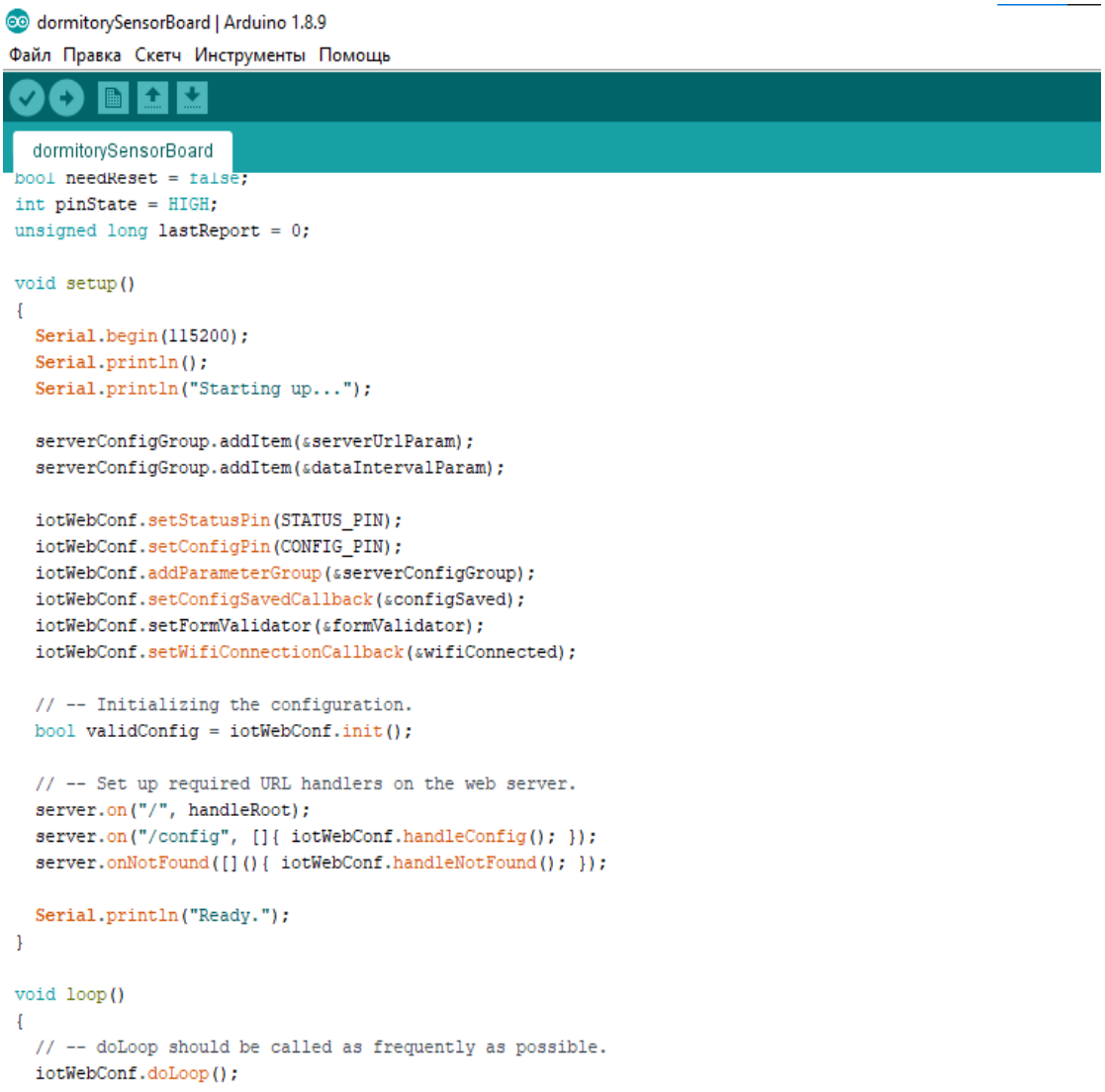


Рисунок 3.1 – Зовнішній вигляд середовища розробки Arduino ID

Мікроконтролери для Arduino відрізняються наявністю заздалегідь прошитого в них завантажувача (англ. bootloader). За допомогою цього завантажувача користувач завантажує свою програму в мікроконтролер без використання традиційних окремих апаратних програматорів. Завантажувач з'єднується з комп'ютером через інтерфейс USB (якщо він є на платі) або окремим перехідником UART-USB. Підтримка завантажувача вбудована Arduino IDE і виконується в один клацання миші. Монітор COM порту використовують для відлагодження роботи прошивки платформи в систему ESP8266 (див. рис. 3.2), до якого підключений мікроконтролер та через який можна побачити сервісні повідомлення та

усунути проблеми при необхідності. Повний лістинг коду модуля моніторингу наведено в Додатку А.

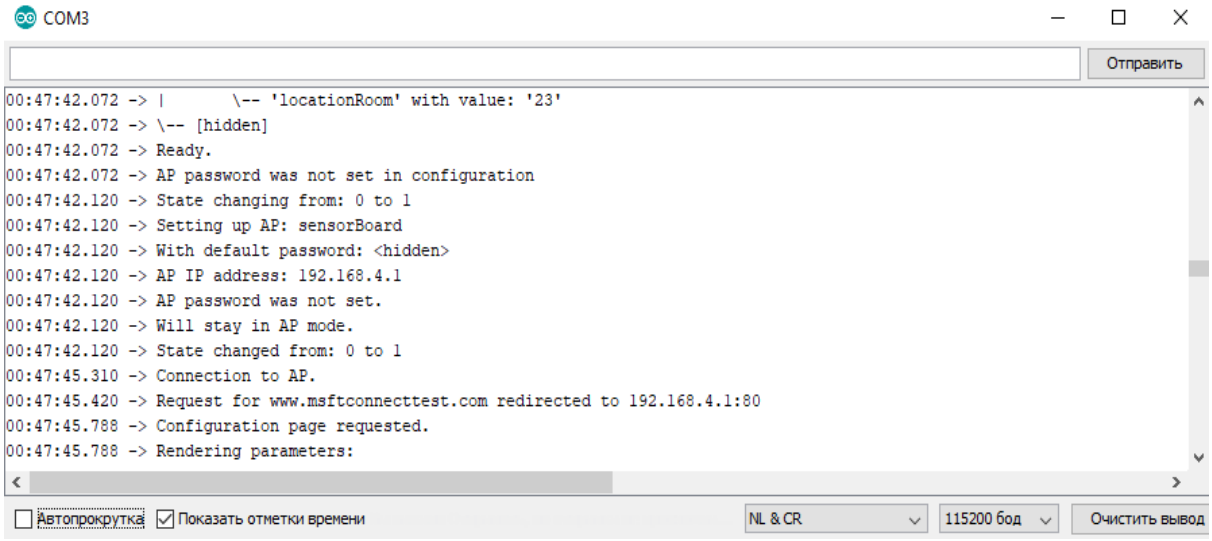


Рисунок 3.2 – Монітор COM порта

На випадок затирання завантажувача або придбання мікроконтролера без завантажувача розробники надають можливість прошити завантажувач у мікроконтролер самостійно. Для цього в Arduino IDE вбудована підтримка кількох популярних дешевих програматорів, а більшість плат Arduino має штирьовий роз'єм для внутрішньосхемного програмування (ICSP для AVR, JTAG або SWD для ARM).

В Arduino IDE вбудована можливість створення власних програмно-апаратних платформ. Цією можливістю користуються сторонні компанії, які додають Arduino IDE свої набори плат і компіляторів-завантажувачів до них

У деяких платах склад доступних портів та частота тактування можуть відрізнятися. Розглянемо приклади деяких.

ARM

Поступово у лінійці плат почали з'являтися процесори ARM. Спочатку це був AT91SAM3X8E на платі класичного конструктиву (Due). Пізніше з'явилася лінійка плат Arduino MKR в конструктиві DIP, оснащена контролером SAMD21 (Cortex-M0, 48 МГц, 256к Flash, 32к RAM).

З 2020 року у такому конструктиві MKR з'явилися модулі Portenta з ARM Cortex-M7 (STM32H747 @ 480 МГц).[28]

Напруга живлення процесорів ARM на платах Arduino – 3,3 вольта. На таку ж напругу мають бути розраховані датчики цих плат.

ESP8266

Сторонні розробники портували Arduino підтримку популярного Wi-Fi мікроконтролера ESP8266 і його клону ESP12. Тепер компілювати та завантажувати прошивку для ESP8266 зі своїми скетчами та підтримкою Wi-Fi можна прямо з Arduino IDE, отримуючи одноплатну схему з підтримкою мережі Wi-Fi. Плати з обв'язаним ESP8266 продаються під маркою Wemos, мають 2 форм-фактори (один як у Uno, другий - зменшений свій) і два покоління в кожному форм-факторі (R1 та R2).

Intel x86

У рамках співпраці зі сторонніми виробниками Arduino IDE була включена підтримка деяких апаратних засобів Intel x86. Intel Galileo (англ.) рос. (процесор Intel Quark X1000 400 МГц), Intel Edison (англ.) рос. та Arduino 101 [29] - Arduino-сумісні плати на Intel x86 архітектурі. Плати механічно та електрично сумісні з периферійними платами Ардуїно. Плати функціонують під власною ОС Linux, поверх якої працює додаток, що дозволяє завантажувати та виконувати скетчі Arduino.

Периферія мікроконтролерів.

Порти введення-виведення мікроконтролерів оформлені у вигляді штирьових лінійок. Жодної буферизації, захисту, конвертації рівнів, як правило, немає. Мікроконтролери живляться від 5 або 3,3. В залежно від моделі плати. Відповідно, порти мають такий же розмах допустимих вхідних та вихідних напруг. Програмістові доступні деякі спеціальні можливості портів введення-виведення мікроконтролерів, наприклад широтно-імпульсна модуляція (ШІМ), аналогово-цифровий перетворювач (АЦП), інтерфейси UART, SPI, I2C. Кількість та можливості портів введення-виведення визначаються конкретним варіантом мікропроцесорної плати.

Крім портів, на платах мікроконтролерів іноді встановлюється периферія як інтерфейсів USB чи Ethernet. Опціональний набір зовнішньої периферії на модулях розширення включає:

• USB Device (найчастіше як віртуальний COM порт через FTDI FT232, є також версії з емуляцією USB HID Class клавіатур та мишок).

• Дротовий і бездротовий Ethernet як у основній платі, і на платах расширения.

• Модуль GSM та інші бездротові інтерфейси[34]. USB Host.

• SD-картка.

• Модуль керування низьковольтним двигуном на базі L298. Підтримуються кроковий та колекторний двигуни з напругою до 12 В та струмом до 2 А на канал. Можуть підключатися також реле, електромагніти тощо. Модуль немає гальванічної розв'язки.

• Графічний РКІ-індикатор.

• Модуль із макетним полем.

Сторонні виробники випускають широку гаму датчиків та виконавчих пристроїв, що підключаються до Arduino.

При підключенні Uno до комп'ютерів, що працюють на Mac OS X або Linux, його мікроконтролер скидатиметься при кожному з'єднанні програмного забезпечення з платою. Після скидання на Arduino Uno активізується завантажувач на час близько півсекунди. Незважаючи на те, що завантажувач запрограмований ігнорувати сторонні дані (тобто всі дані, що не стосуються процесу прошивки нової програми), він може перехопити кілька перших байт даних з посилки, що надсилається платі відразу після встановлення з'єднання.

Відповідно, якщо програма, що працює на Ардуїно, передбачає отримання від комп'ютера будь-яких налаштувань або інших даних при першому запуску, переконайтеся, що програмне забезпечення, з яким взаємодіє Ардуїно, здійснює відправлення через секунду після встановлення з'єднання.

На платі Uno існує доріжка (позначена як "RESET-EN"), розімкнувши яку, можна вимкнути автоматичне скидання мікроконтролера. Для відновлення функції автоматичного скидання необхідно спаяти між собою висновки, розташовані по краях цієї доріжки. Автоматичне скидання також можна вимкнути, підключивши резистор номіналом 110 Ом між виведенням RESET та 5В; для отримання більш детальної інформації див. відповідну гілку форуму. В таблиці 1. наведено приклад характеристик Arduino Uno с микроконтроллером.

Характеристики Arduino Uno з мікроконтролером ESP-12E

таблиця 3.1.

Мікроконтроллер	ESP8266 ESP-12E
Робоче напряжение	5В
Напруга живлення (рекомендуєма)	7-12В
Напруга живлення (граничне)	6-20В
Цифрові входи/виходи	14 (з них 6 можуть використовуватися в якості ШИМ-виходів)
Аналогові входи	6
Максимальний струм одного вивода	40 мА
Максимальний вихідной струм вихода 3.3V	50 мА
Flash-пам’ять	32 КБ із яких 0.5 КБ використовуються завантажником
SRAM	2 КБ
EEPROM	1 КБ
Тактовая частота	16 МГц

3.2 Елементна база системи.

Після проведеного аналізу мікроконтролерів для Arduino було обрано мікроконтролер ESP8266. На сьогодні він є одним із високоінтегрованих

рішень для роботи з WiFi. Найбільше популярна на даний момент версія ESP-12E[8]. Він має високу швидкість роботи, більше доступних виводів та великий об’єм Flash пам’яті, що дозволяє підключити велику кількість давачів даних та дає можливість завантажити більше програмного коду і тим самим розширити функціонал системи.. На рисунку 3.3.представлена ESP-12E версія

Системи, засновані на цій версії мають можливість завантажити більше програмного коду і розширити функціонал системи.

Для реалізації системи моніторингу екосистеми ми обрали саме цю версію.

На момент тестування задачі, комунікація з мікроконтролером (МК) відбувалася через USB to Serial конвертер. Serial конвертер виконував завантаження коду в пам’ять МК та проводити відлагодження його роботи.

Рисунок 3.3 – Версія ESP-12E

Виконання програми на мікроконтролері ведеться шляхом динамічного підвантаження необхідних ділянок програми в кеш інструкцій з зовнішньої SPI ПЗУ. Вантаження йде прозоро для програміста. Підтримується до 16 МБ зовнішньої пам'яті програм. Найчастіше ESP8266 використовують як апаратну основу Інтернету речей, тобто установку в будинках або офісах. Підключення до мережі здійснюється до домашньої /

офісної локальної мережі з виходом в інтернет через роутер. Власник пристрою може контролювати його за допомогою планшета або комп'ютера через свою локальну мережу або віддалено, через Інтернет.

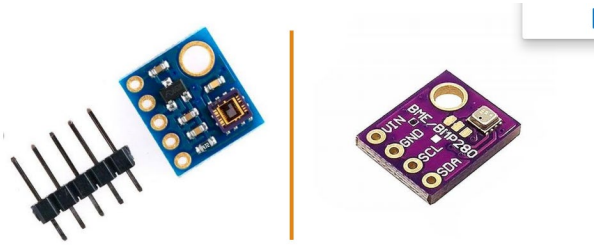
3.2.1. Модуль моніторінгу

Модуль моніторингу- це завжди сукупність датчиків.

В модулі моніторингу використовується два типа датчиків:

1. Цифровий (BME280)- для вимірювання температури, атмосферного тиску, вологості та CO_2 (видає кінцеве значення);

2. Аналоговий (GYL8511)- для вимірювання ультрафіолетового випромінювання.(оптичний сенсор, який чутливий до певного спектру).



Датчики BME280 та GYML8511

Рисунок 3.4. –Види датчиків модуля моніторингу

Вони компактні, мало споживають енергії характеризуються високою точністю вимірювання, високою швидкодією інтерфейсу та мають широке застосовування у смарт пристроях,телефонах.

Показники датчиків знімаються по різним протоколам,тобто різні датчики працють по різним протоколам.

Умовно: якщо датчик цифровиий він обмінюється по якимось протоколам, якщо ж аналоговий, то в нього є пін (вивід\ніжка), який підключається до мікроконтролера , до певного виводу в ньому,і знімається звідти аналоговий сигнал.

Цифрові датчики видають кінцеве значення.

Тобто всі датчики працюють за власним протоколам та мають свій даташит (документ, в якому прописується як працює датчик, які алгоритми використовувались). В нашій розробці це протокол i2c, що є послідовною шиною даних для зв'язку інтегральних схем. Датчики обмінюються даними з основним мікроконтролером, який опрацьовує отриману інформацію. Кожен модуль запрограмований з певним набором датчиків, для збору певних даних.

3.2.2. Інтерфейс системи

Для розробки інтерфейсу інформаційної системи було обрано платформу Telegram. Telegram є хмарним сервісом для обміну повідомленнями, що об'єднує новітні технології, розвинену екосистему. Він надзвичайно надійний, потужний, приватний і захищений, простий у використанні. така кількість переваг показує,чому користувачі обирають для комунікації саме цей сервіс. Telegram відіграє функцію платформи для розробки боту, а в якості інтерфейсу бази даних екосистеми було обрано TelegramBot.

Задача бота - це автоматична відповідь після вводу до нього команди користувача. Важливою функцією бота Telegram є можливість виконувати команди в чаті Telegram. Ці команди потім безпосередньо запускають дії або запитують інформацію. Після обробки запиту сервіси надсилають результат в на пристрій, а сам результат буде знаходитись у самому боті. Пошук проводиться у вигляді спілкування в інтерфейсі, де є певна прошивка, в якій прописано, які дані збиратимуться.

Для з'єднання апаратних пристроїв, API та Інтернет-служб було обрано NodeRED, як інструмент програмування. Для бота був використаний пакет RedBot для NodeRED. NodeRED забезпечує редактор потоків на основі браузера, що спрощує з'єднання потоків за допомогою широкого діапазону вузлів у палітрі. Потоки можна розгорнути в середовищі виконання одним клацанням миші. NodeRED має спеціальне сховище даних від всіх датчиків і

надалі, ця інформація посилається в Telegram.

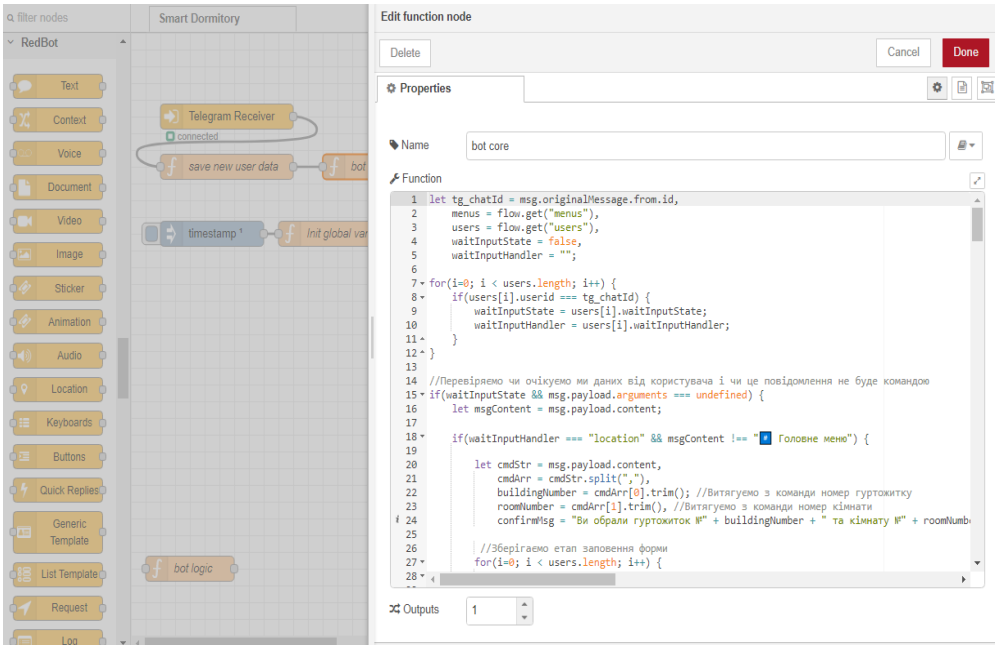


Рисунок 3.5 – Середовище розробки NodeRed

Більшість функціоналу написана через елементи Function Node, які надають можливість вставити користувацький програмний код і тим самим розширити можливості вбудованих в пакет інструментів.

Хостинг для тестування роботи даного середовища було розгорнуто на Amazon Web Services (AWS)Ecx [1,10]. Amazon Web Services (AWS) є дочірньою компанією Amazon. Компанія надає платформу для хмарних обчислень на вимогу приватних осіб та урядам на основі платної підписки. Але є і безкоштовна підписка якою ми скористались. Безкоштовна підписка доступна впродовж перших 12 місяців. Технологія дозволяє абонентам мати у своєму розпорядженні повноцінний віртуальний кластер комп'ютерів, який завжди доступний через Інтернет.

Тестування інформаційної моделі моніторингу екосистеми було проведено у гуртожитку №7 Київського національного університету технологій та дизайну в травні 2021року .

3.3. Алгоритм тестування основних функцій системи.

1. Створюються бездротова точки доступу. Модуль підключається до Wi-Fi з виходом до Інтернет. Створюється точка доступ.

2. За певною IP-адресою переходимо в режим налаштування**.** Перевірка системи на створення нею точки доступу. Створена точка доступу захищена паролем для забезпечення безпеки налаштувань підсистеми та уникнення зміни налаштувань користувачем, якому не було надано доступу.

3. Вказуємо назву та пароль Wi-Fi - мережі, до якої має підключитись модуль моніторингу

4. Обираємо сервер для відправки даних. Вказуємо номер гуртожитку і кімнату до якої буде закріплено модуль з датчиками.

5. Встановлюємо інтервал обміну даними (Таймінг видачі інформації). В налаштуваннях вказуємо період часу, через який відбуватиметься оновлення, наприклад 20 хвилин. Кожні 20 хв.він буде скидати дані на сервер

6. Для початку комунікації з ботом потрібно натиснути «start». Користувач отримує повідомлення, в якому власне можна побачити, чим може бути корисний даний бот

7. Натиснувши на «Перегляд показників», користувач отримає всі дані стану екосистеми кімнати.

Розглянемо на прикладі роботу алгоритму.

Спочатку треба обов’язково провести функціональне тестування.

Функціональне тестування, це перевірка функціоналу інтерфейсу системи:

1. Перевірка роботи обов’язкових функцій боту;
2. Тестування працездатності користувацьких форм інтерфейсу;
3. Налаштування та перевірка роботи сповіщень.

3.4. Режим тестування системи

Далі проводимо тестування основних функцій роботи системи.

Починаємо з перевірки системи на створення нею точки бездротового доступу (рис. 3.6). Ця функція є першочерговою та надає користувачу доступ до системи. На цьому кроці модуль підключається до Wi-Fi з виходом до Інтернет та створюється точка доступ. Точка доступу захищена паролем, для забезпечення безпеки налаштувань підсистеми та уникнення зміни налаштувань системи користувачем, якому не було надано доступу

При успішному підключенні, модуль моніторингу перестане створювати точку доступу для налаштування та стане невидимий в списку доступних точок для підключення і почне відправку даних на вказаний сервер з заданим інтервалом.

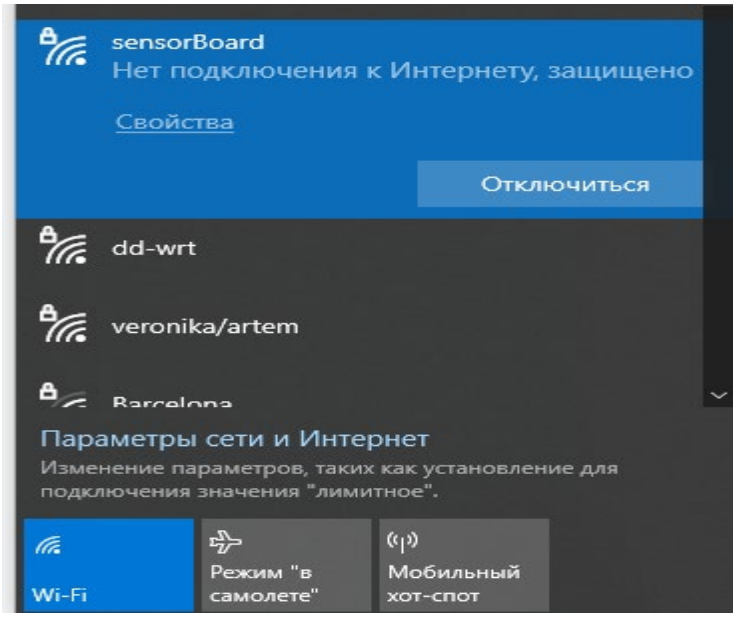


Рис.3.6- Створена бездротова точка доступу

За певною IP-адресою переходимо в режим налаштування. Йде перевірка системи на створення нею точки доступ .

Далі будемо виконувати вхід у веб-інтерфейс . Заповнюємо форму (рис. 3.7.), вказуючи потрібні дані :

1. Назва та пароль wifi мережі, до якої має підключитись модуль моніторингу
2. Сервер для відправки даних з заданим інтервалом
3. Номер гуртожитку і кімнату до якої буде закріплено модуль з датчиками

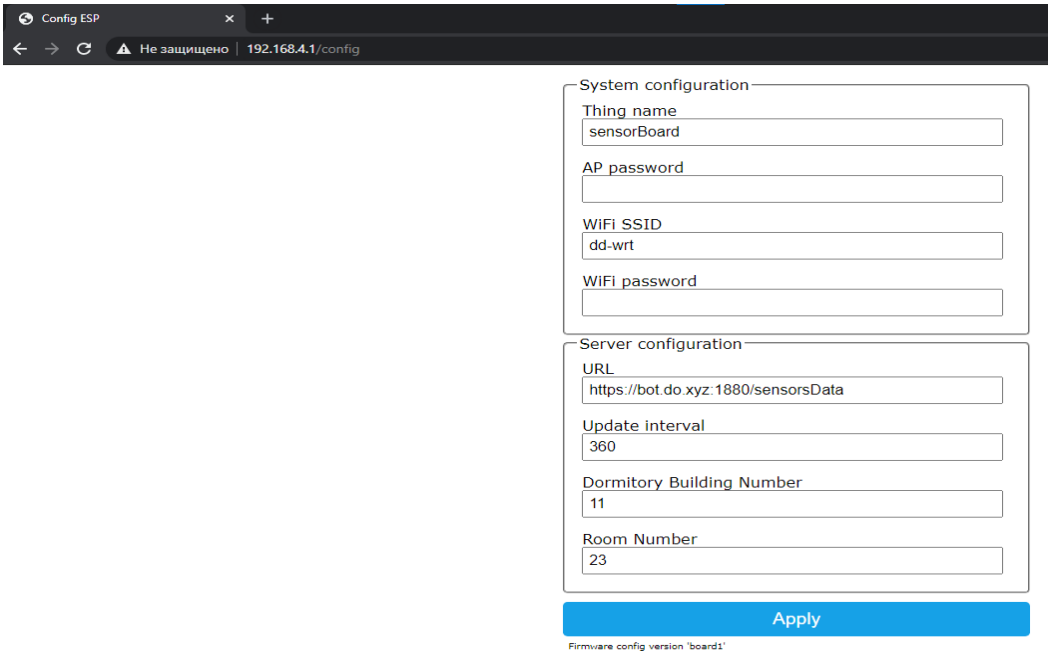


Рисунок 3.7 - Вхід у веб-інтерфейс

Лістинг 3.1 - Фрагмент коду підключених датчиків та ініціалізації модулю моніторингу:

```
void setup()
{
  Serial.begin(115200);
  Serial.println();
  Serial.println("Starting up...");
  serverConfigGroup.addItem(&serverUrlParam);
  serverConfigGroup.addItem(&dataIntervalParam);
  serverConfigGroup.addItem(&locationBuildingParam);
  serverConfigGroup.addItem(&locationRoomParam);

  iotWebConf.setStatusPin(STATUS_PIN);
  iotWebConf.setConfigPin(CONFIG_PIN);
```

```
  iotWebConf.addParameterGroup(&serverConfigGroup);
  iotWebConf.setConfigSavedCallback(&configSaved);
  iotWebConf.setFormValidator(&formValidator);
  iotWebConf.setWifiConnectionCallback(&wifiConnected);

  // -- Initializing the configuration.
  bool validConfig = iotWebConf.init();

  // -- Set up required URL handlers on the web server.
  server.on("/", handleRoot);
  server.on("/config", []{ iotWebConf.handleConfig(); });
  server.onNotFound([](){ iotWebConf.handleNotFound(); });

  Serial.println("Ready.");

  //  Sensors init
  Wire.begin(D4, D3);

  pinMode(A0, INPUT);

  while(!bme.begin())
  {
    Serial.println("Could not find BME280 sensor!");
    delay(450);
  }

  switch(bme.chipModel())
  {
    case BME280::ChipModel_BME280:
      Serial.println("Found BME280 sensor! Success.");
      break;
    case BME280::ChipModel_BMP280:
      Serial.println("Found BMP280 sensor! No Humidity
available.");
      break;
    default:
      Serial.println("Found UNKNOWN sensor! Error!");
  }
}
```

Інформація з модулю моніторингу при успішному підключенні (дані місцерозташування та покази з датчиків), формуються в пакет даних та відправляються на вказану в налаштуванні адресу сервера з заданим інтервалом.

На стороні NodeRED за допомогою HTTP-кодів та блоку користувацького коду виконується обробка цих даних та зберігання/відправка сповіщень користувачу. (рис. 3.8).

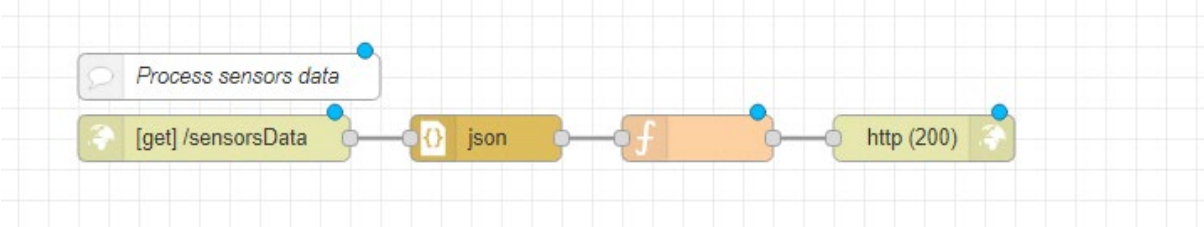


Рисунок 3.8 – Процес обробки даних

Лістинг 3.2 – Структура JSON пакету з даними

```
[
  {
    "location":{
      "building":"",
      "room":""
    },
    "sensors":[
      {
        "id":0,
        "value":""
      },
      {
        "id":3,
        "value":""
      }
    ]
  }
```

Лістинг 3.3 – Фрагмент коду з відправкою даних на сервер (фрагмент коду) :

```
unsigned long now = millis();
  if (dataIntervalParam < now - lastReport)
  {
    lastReport = now;

    Serial.println("Data send.");

    bme.read(pres, temp, hum, tempUnit, presUnit);
    pressure = pres * 0.007500637554192;

    outputVoltage = 3.3 / 1024 * averageAnalogRead(A0);
    uvIIntensity = mapfloat(outputVoltage, 0.99, 2.8, 0, 15);
//Convert the voltage to a UV intensity level

    //Check WiFi connection status
```

```
    if(WiFi.status()== WL_CONNECTED){
      HTTPClient http;

      String serverPath = serverUrlParam;

      // Your Domain name with URL path or IP address with path
      http.begin(serverPath.c_str());

      http.addHeader("Content-Type", "application/json");

      int httpResponseCode =
http.POST("{\"location\":{\"building\":\"\" +
locationBuildingParam + "\"\",\"room\":\"\" + locationRoomParam
+ "\"\"},\"sensors\":[{\"id\":0,\"value\":\"\" + temp +
"\"\}",{\"id\":4,\"value\":\"\" + uvIIntensity + "\"\]}");

      if (httpResponseCode>0) {
        Serial.print("HTTP Response code: ");
        Serial.println(httpResponseCode);
        String payload = http.getString();
        Serial.println(payload);
      }
      else {
        Serial.print("Error code: ");
        Serial.println(httpResponseCode);
      }
      // Free resources
      http.end();
  }
```

Для початку комунікації з ботом після створення бездротової точки доступу та вдалого проходження ініціалізації модулю моніторингу, потрібно натиснути «start». Користувач отримує повідомлення, в якому власне можна побачити, чим може бути корисний даний бот (рис.3.9).

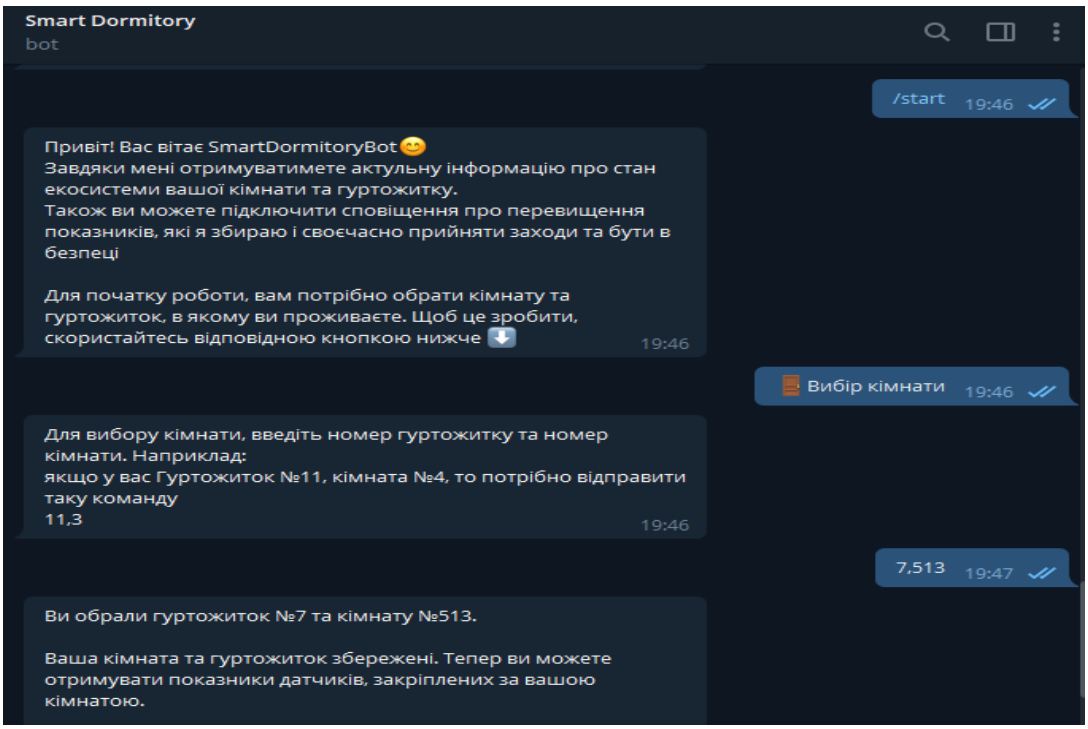


Рис.3.9. Комунікація з ботом. Вибір кімнати

Система має користувацьку клавіатуру з різними командами, в залежності від обраної дії (рис. 3.10). Незмінними лишаються кнопки «Головне меню» та «Назад».

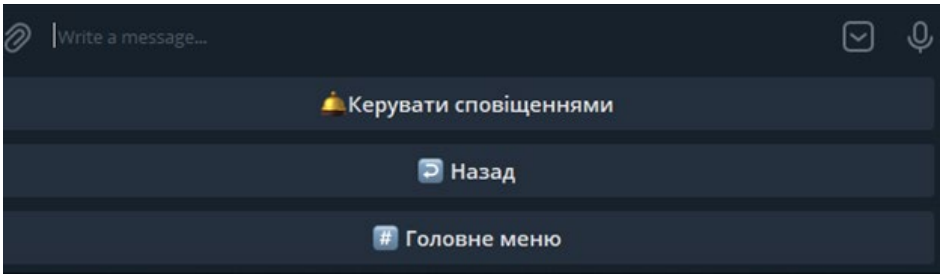


Рис.3.10. Користувацька клавіатура з командами

Фрагмент коду, для виконання команд з меню(при натисканні певних кнопок).

Лістинг 3.4 – фрагмент коду для роботи меню бота

```
let inboundMsg = msg.payload.content,
        hier,
        findMenuFlag = false;
```

```
    menus.forEach((element) => {
        if(inboundMsg == element.name) {
            findMenuFlag = true;
            element.payloadObj.chatId = tg_chatId;
                //Зберігаємо дані для ієрархії меню
            hier = element.hierarchy;
            //Встановлюємо вказівник для кнопки Назад
            menus.forEach((elem) => {
              if(elem.id == hier) {
                for(i=0; i < users.length; i++) {
                    if(users[i].userid === tg_chatId) {
                        users[i].lastMenu = elem.name;
                        //Вимикаємо очікування вводу
користувача
                        users[i].waitInputState = false;
                  }
                }
                flow.set("users", users);
              }
            });
```

Натиснувши на «Перегляд показників», користувач отримає всі дані стану екосистеми кімнати, які досліджуються: температура повітря, рівень вологості в кімнаті, рівень CO_2, якість повітря (рис.3.11)

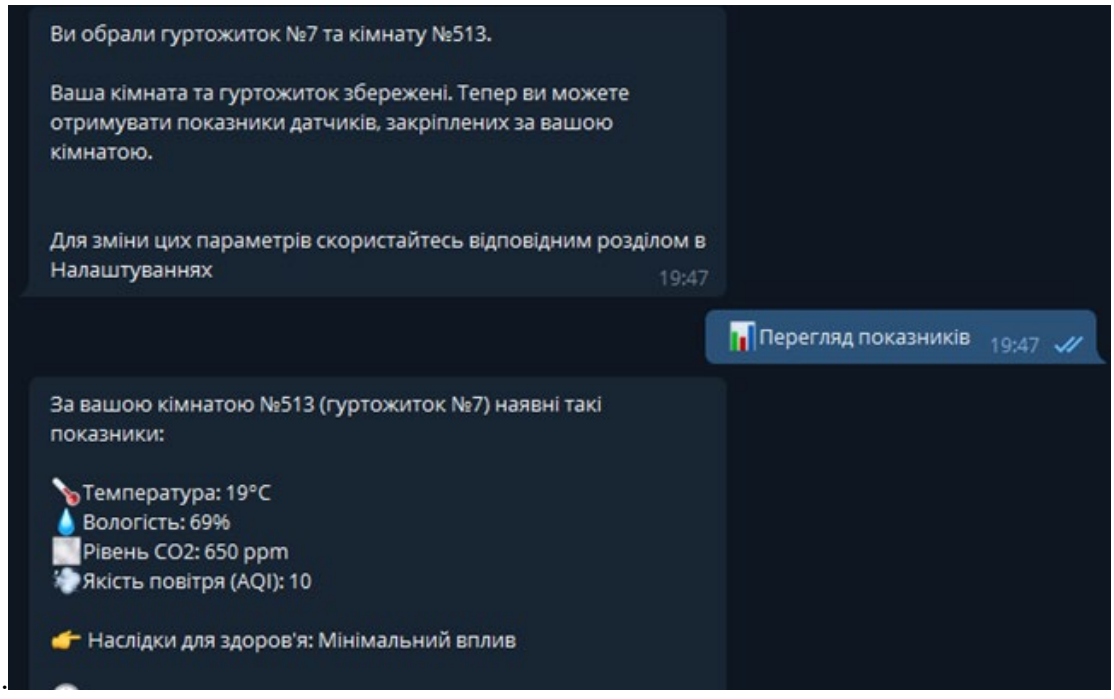


Рис.3.11. Перегляд показників екосистеми кімнати

Наприклад,

1. Показник концентрації рівеня CO_2 , характеризує накопичення вірусів у повітрі, що особливо важливо насьогодні в період пандемії коронавірусної інфекції

2. Зміна температури в кімнатах гуртожитку, покаже, де є не санкціоноване використання обігрівачів та інших пристроїв, які можуть привести до пожежа небезпечної ситуації.

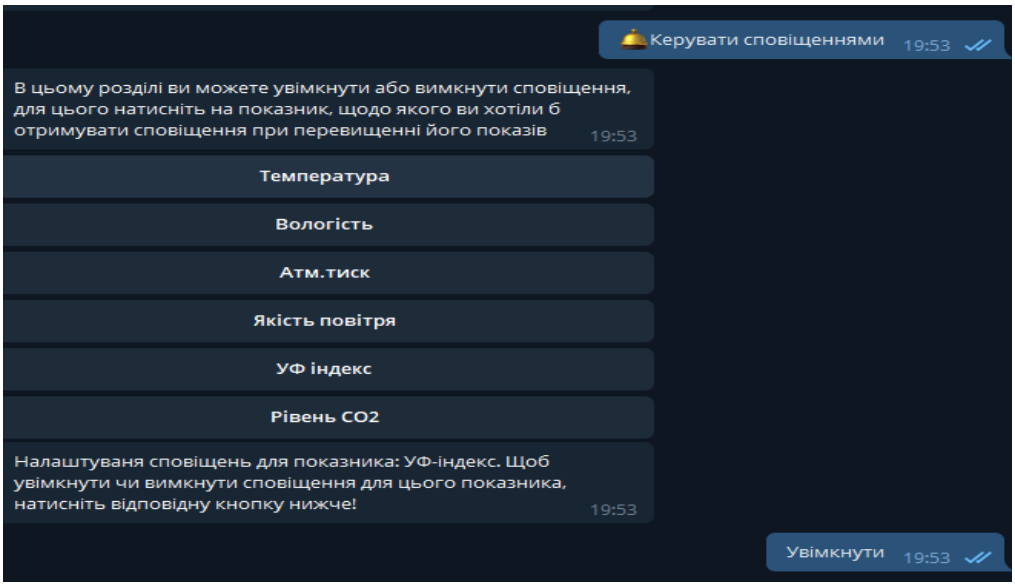


Рис.3.12 Увімкнення сповіщень про зміну показників

Щоб забезпечити своє здоров’я від впливу негативних чинників, можна увімкнути сповіщення про зміну або перевищення тих чи інших показників. (рис.3.12).

ВИСНОВКИ

Під час виконання даної дипломної роботи було зроблено наступне:

1. Концепцію розробки інформаційної моделі моніторингу екосистеми розглянуто на прикладі розумного гуртожитку, де одним із компонентів системи є технологія інтернет речей.

2. На основі аналізу характеристик мікроконтролерів та датчиків була реалізована логіка роботи модулів моніторингу з різними типами датчиків, протоколами зв’язку та побудувана система моніторингу з інтерфейсом взаємодії з користувачем через чат-бот. Було обрано модуль бездротового зв'язку, який встановлений у пристрої системи моніторингу та обрано два типи датчиків для отримання показників екосистеми. Це дало можливість отримувати та аналізувати інформацію екосистеми кімнати в реальному часі та на відстані.

3. Розроблені структурні елементи системи та програмний код на прикладі плати для датчика, дозволили ефективно ідентифікувати проблему у разі відхилень від стандарту та полегшити передбачення можливих аварій та їх усунень.

4. Платформою для розробки інтерфейсу. взаємодії з користувачем була обрано TelegramBot. Система моніторингу з інтерфейсом взаємодії з користувачем через бот, буде відслідковувати різні показники екосистеми в кімнатах гуртожитку і в цілому та надсилатиме сповіщення про їх перевищення, або вихід із норми.

5. Модель моніторингу екосистеми була апробована на даних, які були отримані в гуртожитку № 7 Київського національного університетй технологій та дизайну у травні 2021 році. Хостинг для роботи даного середовища було розгорнуто на Amazon Web Services (AWS) EC

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Види тестування ПЗ [Електронний ресурс] – Режим доступу: http://qalearning.com.ua/theory/lectures/material/testing-types-functional/ – Дата доступу: 02.05.2021
2. Астістова Т.І. Розробка інформаційної моделі моніторингу екосистеми/ Т.І Астістова // Технології та інжиніринг («Вісник КНУТД. Серія Технічні науки») Київський національний університет технологій та дизайну, Україна №4, 2021р. С. 9-17
3. Astistova T. I. Dormitory as a cluster of a smart house/ T. I. Astistova T. I., D.M. Kochuk// Збірник наукових праць II Всеукраїнської конференці «Інноватика в освт, науц та бзнес: виклики та можливост» 18 листопада 2021р., КНУТД, Том 2, С. 5-10 https://drive.google.com/drive/folders/1JzJtQ4_dEWkSG46jANcROYuwkn3mtbrd
4. Розумне місто та його компоненти [Електронний ресурс] – Режим доступу URL: https://www.sea.com.ua/ua/smart-city/news/rozumni-mista-abo-smart-cities-happy-citizens/ – Дата доступу: 02.05.2021
5. Астістова Т. І., Розробка концепції інформаційної системи «Smart city»/ Т. І., Астістова Т. І., Д.М Кочук Д.М //Інформаційні технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук– К. : Освіта України, 2021 р. . – С. 217 – 220
6. Тестування програмного забезпечення, яке використовується для моніторингу екосистеми. [Електронний ресурс] – Режим доступу : URL : https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення – Дата доступу : 28.05.2021.
7. Астістова Т. І., Кочук Д.М, Аналіз та характеристика технології «Internet of things»/ Т.І. Астістова, Д.М. Кочук// Інформаційні

технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук та технологій. – К. : Освіта України, 2021 р. . – С. 224 – 227

8. Інтернет речей [Електронний ресурс] – Режим доступу URL: https://www.sas.com/ru_ru/insights/big-data/internet-of-things.html – Дата доступу: 04.05.2021
9. Amazon Web Services [Електронний ресурс] – Режим доступу URL: https://uk.wikipedia.org/wiki/Amazon_Web_Services – Дата доступу: 01.06.2021
10. Astistova T.I Chat-bot development for telegram social network/ T.I. Astistova,O.V.Tура // Тези V Міжнародної науково-практичної конференції «Мехатронні системи: інновації та інжиніринг – «MSIE-2021» К. КНУТД , 4 листопада 2021р. - С. 155
11. Астістова Т. І.Тюпа О. В., Розробка пошукової системи в месенджері Telegram з використанням Google Assistant та Google Search API/ Т. І. Астістова, О.В. Тюпа// Інформаційні технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук та технологій. – К. : Освіта України, 2021 р. . – С. 220 – 223
12. Astistova T.I, Smart house management system, user interface/ T.I. Astistova, M.A. Kolva //Тези V Міжнародної науково-практичної конференції «Мехатронні системи: інновації та інжиніринг – «MSIE-2021» К. КНУТД , 4 листопада 2021р. - С.156-157
13. Астістова Т. І., Кольва М. А., Розробка інтерфейсу системи управління розумним будинком / Т.І. Астістова , М.А.Кольва // Інформаційні технології в науці, виробництві та підприємництві: зб. наук. праць молодих вчених, аспірантів, магістрів кафедри комп'ютерних наук та технологій. – К. : Освіта України, 2021 р. – С. 212 – 214
14. JSON, що це таке [Електронний ресурс] – Режим доступу : URL: https://uk.wikipedia.org/wiki/JSON – Дата доступу: 01.06.2021

15. Internet rzeczy [Електронний ресурс] – Режим доступу URL: https://www.copadata.com/pl/produkty/platform-editorial-content/co-to-jest-iot-oraz-iiot/ – Дата доступу: 28.05.2021
16. What is smart city? [Електронний ресурс] – Режим доступу URL: https://placesjournal.org/article/a-city-is-not-a-computer/?gclid=Cj0KCQjw5PGFBhC2ARIsAIFIMNf0iWo9q2IcrsSWm1NhvmpfQSyCjHPhXlRABoJhN0TgPvwInwopZHUaAr2HEALw_wc
17. Бойко В. І.,. Гуржій А. М,. Жуйков В. Я. Мікрпроцесори Сокол Є. І,. Домнін І. Ф.,.. Спеціалізовані мікроконтролерні системи. Теорія і практика Харків: НТУ “ХПІ”, 2007. – 252 с.
18. What is smart city? https://placesjournal.org/article/a-city-is-not-a-computer/?gclid=Cj0KCQjw5PGFBhC2ARIsAIFIMNf0iWo9q2IcrsSWm1NhvmpfQSyCjHPhXlRABoJhN0TgPvwInwopZHUaAr2HEALw_wcB
19. Middleton P,. Kjeldsen P., Tully J., "Forecast: The Internet of Things, Worldwide,"- Gartner, 2013
20. Keo C, S. Kumar S., Tschofenig H. , "Securing the internet of things: A standardization perspective," IEEE Internet of Things Journal, Vol. 1, No. 3, pp. 265-275 2014. https://doi.org/10.1109/JIOT.2014.2323395
21. Astistova T.I, Kochuk D.M..Software development for technology"Internet of things" , V Міжнародної науково-практичної конференції «Мехатронні системи: інновації та інжиніринг – «MSIE-2021» К. КНУТД , 4 листопада 2021р. - С.57-58 анг.
22. MVP Explained: A Systematic Mapping Study on the Definitions of Minimal Viable Product*" (2016). 42th Euromicro Conference on Software Engineering and Advanced Applications (SEAA).p.: 112–119.* DOI:10.1109/SEAA.2016.56
23. Amazon Web Services URL: https://uk.wikipedia.org/wiki/Amazon_Web_Services - Access date: 01.06.2021 testuvannia prohramnoho zabezpechennia. Iake bykorystovuetsia dlia monitorinqy yekosystemy

24. Тестування програмного забезпечення, яке використовується для моніторингу екосистеми: UR:https://uk.wikipedia.org/wiki/Тестування_програмного_забезпечення – Дата доступу : 08.09.2021.
25. Radoff, Jon Minimum Viable Product rant. http://web.archive.org/web/20140323181121/http://radoff.com/blog/2010/05/04/minimum-viable-product-rant/. Jon Radoff's Internet Wonderland (Дата обращения: 19 августа 2014)
26. P. Middleton, P. Kjeldsen, Tully J., "Forecast: The Internet of Things, Worldwide," Gartner, 2013
27. Emerging Trends and Applications of the Internet of Things Advances in Wireless Technologies and Telecommunication (2327-3305) Kocovic, Petar, Behringer, Reinhold, Ramachandran, Muthu, Mihajlovic, Radomir/ IGI Global, 2017
28. NodeJS Tutorial [Електронний ресурс] – Режим доступу до ресурсу: https://www.w3schools.com/nodejs/.
29. Введення в NodeJS [Електроний ресурс] – Режим доступу до ресурсу: https://metanit.com/web/nodejs/1.1.php
30. NodeJS [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Node.js.
31. Про NodeJS [Електронний ресурс] – Режим доступу до ресурсу: https://nodejs.org/uk/about/.
32. Websockets [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/WebSocket
33. Інтернет речей [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%86%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82_%D1%80%D0%B5%D1%87%D0%B5%D0%B9.
34. Що таке інтернет речей? [Електронний ресурс] – Режим доступу до ресурсу:http://iot.lviv.ua/%D1%89%D0%BE%D1%82%D0%B0%D0%BA

%D0%B5%D1%96%D0%BD%D1%82%D0%B5%D1%80%D0%BD%D0%B5%D1%82-%D1%80%D0%B5%D1%87%D0%B5%D0%B9/.

35. JSON, що це таке [Електронний ресурс] – Режим доступу : URL: https://uk.wikipedia.org/wiki/JSON – Дата доступу: 01.06.2021
36. Internet rzeczy [Електронний ресурс] – Режим доступу URL: https://www.copadata.com/pl/produkty/platform-editorial-content/co-to-jest-iot-oraz-iiot/ – Дата доступу: 28.05.2021
37. What is smart city? [Електронний ресурс] – Режим доступу URL: https://placesjournal.org/article/a-city-is-not-a-computer/?gclid=Cj0KCQjw5PGFBhC2ARIsAIFIMNf0iWo9q2IcrsSWm1NhvmpfQSyCjHPhXlRABoJhN0TgPvwInwopZHUaAr2HEALw_wcB

ДОДАТОК А.

Програмний код для плати датчика

```
#include <IotWebConf.h>
#include <IotWebConfUsing.h> // This loads aliases for easier
class names.
#include <BME280I2C.h>

// -- Initial name of the Thing. Used e.g. as SSID of the own
Access Point.
const char thingName[] = "sensorBoard";

// -- Initial password to connect to the Thing, when it creates
an own Access Point.
const char wifiInitialApPassword[] = "smrtTHNG8266";

#define STRING_LEN 128

// -- Configuration specific key. The value should be modified
if config structure was changed.
#define CONFIG_VERSION "board1"

// -- When CONFIG_PIN is pulled to ground on startup, the Thing
will use the initial
//      password to buld an AP. (E.g. in case of lost password)
#define CONFIG_PIN D2

// -- Status indicator pin.
//      First it will light up (kept LOW), on Wifi connection it
will blink,
//      when connected to the Wifi it will turn off (kept HIGH).
#define STATUS_PIN LED_BUILTIN

// -- Method declarations.
void handleRoot();
// -- Callback methods.
void wifiConnected();
void configSaved();
bool formValidator(iotwebconf::WebRequestWrapper*
webRequestWrapper);

DNSServer dnsServer;
WebServer server(80);
WiFiClient net;

BME280I2C bme;    // Default : forced mode, standby time = 1000
ms
                  // Oversampling = pressure ×1, temperature ×1,
humidity ×1, filter off,
```

```
float temp(NAN), hum(NAN), pres(NAN), pressure, outputVoltage,
uvIIntensity, uvAIntensity;

BME280::TempUnit tempUnit(BME280::TempUnit_Celsius);
BME280::PresUnit presUnit(BME280::PresUnit_Pa);

//Takes an average of readings on a given pin
//Returns the average
int averageAnalogRead(int pinToRead)
{
  byte numberOfReadings = 8;
  unsigned int runningValue = 0;

  for(int x = 0 ; x < numberOfReadings ; x++)
    runningValue += analogRead(pinToRead);
  runningValue /= numberOfReadings;

  return(runningValue);
}

//The Arduino Map function but for floats
//From: http://forum.arduino.cc/index.php?topic=3922.0
float mapfloat(float x, float in_min, float in_max, float
out_min, float out_max)
{
  return (x - in_min) * (out_max - out_min) / (in_max - in_min)
+ out_min;
}

char serverUrlValue[STRING_LEN];
char dataIntervalValue[STRING_LEN];
char locationBuildingValue[STRING_LEN];
char locationRoomValue[STRING_LEN];

IotWebConf iotWebConf(thingName, &dnsServer, &server,
wifiInitialApPassword, CONFIG_VERSION);
// -- You can also use namespace formats e.g.:
iotwebconf::ParameterGroup
IotWebConfParameterGroup serverConfigGroup =
IotWebConfParameterGroup("serverConfig", "Server
configuration");
IotWebConfTextParameter serverUrlParam =
IotWebConfTextParameter("URL", "serverUrl", serverUrlValue,
STRING_LEN);
IotWebConfTextParameter dataIntervalParam =
IotWebConfTextParameter("Update interval", "dataInterval",
dataIntervalValue, STRING_LEN);
IotWebConfTextParameter locationBuildingParam =
IotWebConfTextParameter("Dormitory Building Number",
"locationBuilding", locationBuildingValue, STRING_LEN);
IotWebConfTextParameter locationRoomParam =
IotWebConfTextParameter("Room Number", "locationRoom",
locationRoomValue, STRING_LEN);
```

```
bool needReset = false;
int pinState = HIGH;
unsigned long lastReport = 0;

void setup()
{
  Serial.begin(115200);
  Serial.println();
  Serial.println("Starting up...");

  serverConfigGroup.addItem(&serverUrlParam);
  serverConfigGroup.addItem(&dataIntervalParam);
  serverConfigGroup.addItem(&locationBuildingParam);
  serverConfigGroup.addItem(&locationRoomParam);

  iotWebConf.setStatusPin(STATUS_PIN);
  iotWebConf.setConfigPin(CONFIG_PIN);
  iotWebConf.addParameterGroup(&serverConfigGroup);
  iotWebConf.setConfigSavedCallback(&configSaved);
  iotWebConf.setFormValidator(&formValidator);
  iotWebConf.setWifiConnectionCallback(&wifiConnected);

  // -- Initializing the configuration.
  bool validConfig = iotWebConf.init();

  // -- Set up required URL handlers on the web server.
  server.on("/", handleRoot);
  server.on("/config", []{ iotWebConf.handleConfig(); });
  server.onNotFound([](){ iotWebConf.handleNotFound(); });

  Serial.println("Ready.");

  //  Sensors init
  Wire.begin(D4, D3);

  pinMode(A0, INPUT);

  while(!bme.begin())
  {
    Serial.println("Could not find BME280 sensor!");
    delay(450);
  }

  switch(bme.chipModel())
  {
    case BME280::ChipModel_BME280:
      Serial.println("Found BME280 sensor! Success.");
      break;
    case BME280::ChipModel_BMP280:
      Serial.println("Found BMP280 sensor! No Humidity
available.");
      break;
```

```
    default:
      Serial.println("Found UNKNOWN sensor! Error!");
  }
}

void loop()
{
  // -- doLoop should be called as frequently as possible.
  iotWebConf.doLoop();

  if (needReset)
  {
    Serial.println("Rebooting after 1 second.");
    iotWebConf.delay(1000);
    ESP.restart();
  }

  unsigned long now = millis();
  if (dataIntervalParam < now - lastReport)
  {
    lastReport = now;

    Serial.println("Data send.");

    bme.read(pres, temp, hum, tempUnit, presUnit);
    pressure = pres * 0.007500637554192;

    outputVoltage = 3.3 / 1024 * averageAnalogRead(A0);
    uvIIntensity = mapfloat(outputVoltage, 0.99, 2.8, 0, 15);
//Convert the voltage to a UV intensity level

    //Check WiFi connection status
    if(WiFi.status()== WL_CONNECTED){
      HTTPClient http;

      String serverPath = serverUrlParam;

      // Your Domain name with URL path or IP address with path
      http.begin(serverPath.c_str());

      http.addHeader("Content-Type", "application/json");

      int httpResponseCode =
http.POST("{\"location\":{\"building\":\"\" +
locationBuildingParam + "\"\",\"room\":\"\" + locationRoomParam
+ "\"\"},\"sensors\":[{\"id\":0,\"value\":\"\" + temp +
"\"\}",{\"id\":4,\"value\":\"\" + uvIIntensity + "\"\]}");

      if (httpResponseCode>0) {
        Serial.print("HTTP Response code: ");
        Serial.println(httpResponseCode);
        String payload = http.getString();
        Serial.println(payload);
```

```
      }
      else {
        Serial.print("Error code: ");
        Serial.println(httpResponseCode);
      }
      // Free resources
      http.end();
  }
}

/**
 * Handle web requests to "/" path.
 */
void handleRoot()
{
  // -- Let IotWebConf test and handle captive portal requests.
  if (iotWebConf.handleCaptivePortal())
  {
    // -- Captive portal request were already served.
    return;
  }
  String s = "<!DOCTYPE html><html lang=\"en\"><head><meta
name=\"viewport\" content=\"width=device-width, initial-scale=1,
user-scalable=no\"/>";
  s += "<title>Dormitory Sensor
Board</title></head><body>Dormitory Sensor Board Config";
  s += "<ul>";
  s += "<li>Server URL: ";
  s += serverUrlValue;
  s += "</ul>";
  s += "Go to <a href='config'>configure page</a> to change
values.";
  s += "</body></html>\n";

  server.send(200, "text/html", s);
}

void wifiConnected()
{
//  needMqttConnect = true;
}

void configSaved()
{
  Serial.println("Configuration was updated.");
  needReset = true;
}

bool formValidator(iotwebconf::WebRequestWrapper*
webRequestWrapper)
{
  Serial.println("Validating form.");
  bool valid = true;
```

```
  int l = webRequestWrapper-
>arg(serverUrlParam.getId()).length();
  if (l < 3)
  {
    serverUrlParam.errorMessage = "Please provide at least 3
characters!";
    valid = false;
  }

  return valid;
}
```