МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ
(підпис)

09 червня 2023 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерних наук,

освітньо- професійної програми «Інформатика»

на тему: «Веб-орієнтована інформаційна система «Розумний будинок»»

здобувача групи ІН-91 Щербань Дмитро Олегович

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Дмитро ЩЕРБАНЬ
(підпис)

Керівник
доцент, кандидат фізико-математичних наук Сергій ШАПОВАЛОВ _____
(підпис)

Суми – 2023

Сумський державний університет

Факультет електроніки та інформаційних технологій

Кафедра комп'ютерних наук

«Затверджую»

В.о. завідувача кафедри

_____ Ігор ШЕЛЕХОВ

(підпис)

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

на здобуття освітнього ступеня бакалавра

зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»

здобувача групи ІН-91 Щербаня Дмитра Олеговича

1. Тема роботи: «Веб-орієнтована інформаційна система «Розумний будинок»»
затверджую наказом по СумДУ від *«01» червня 2023 р. № 0475-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 09 червня 2023 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити)
1) Аналіз основних аспектів проблеми, аналіз конкурентів та потреб аудиторії, постановка й формування завдань дослідження. 2) Огляд існуючих інструментів, що використовуються для розробки систем та вибір серед них оптимальних. 3) Розроблення веб-орієнтованої системи. 4) Аналіз та виклад результатів
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень)_____
6. Консультанти до проєкту (роботи), із значенням розділів проєкту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «____» _____ 20 ___ р.

Завдання прийняв до виконання _____ Керівник _____

(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз основних аспектів проблеми, аналіз конкурентів та потреб аудиторії, постановка й формування завдань дослідження*		
2	*Огляд існуючих інструментів, що використовуються для розробки систем та вибір серед них оптимальних*		
3	*Розроблення веб-орієнтованої системи*		
4	*Аналіз та виклад результатів*		
5	*Оформлення пояснювальної записки до кваліфікаційної роботи*		

Здобувач вищої освіти _____ Керівник _____

(підпис) (підпис)

АНОТАЦІЯ

Записка: 77 стр., 13 рис., 1 додаток, 32 використаних джерел.

Обґрунтування актуальності теми роботи – Тема кваліфікаційної роботи є актуальною, так як присвячена автоматизації функцій розумного будинку шляхом застосування інформаційних технологій.

Об'єкт дослідження – процес автоматизації функцій розумного будинку.

Мета роботи – розробка веб-системи для автоматизації функцій розумного будинку з використанням сучасних мов програмувань та технологій.

Методи дослідження – моделі та методи керування функціями розумного будинку та методи й алгоритми побудови веб-систем.

Результати – проаналізовано особливості роботи систем розумного будику, виконано моделюванння і проєктування основних функцій розумного будинку, розроблено інформаційну систему для розумного будинку з використанням сучасних веб-технологій.

ВЕБ-СИСТЕМА, РОЗУМНИЙ БУДИНОК, UML-ДІАГРАМА, ARDUINO UNO

ЗМІСТ

ВСТУП 6
1 АНАЛІЗ ПРОБЛЕМИ. ПОСТАНОВКА ЗАВДАННЯ ДОСЛІДЖЕННЯ 9
1.1. Поняття та особливості роботи системи розумний дім 9
1.2. Аналітичній огляд систем розумний будинок 12
1.3. Постановка завдання 13
2 ОБ'ЄКТИВНА ТИПОЛОГІЯ СИСТЕМИ РОЗУМНИЙ БУДИНОК 16
2.1. Короткий огляд відомих типологій 16
2.2. Середовище розробки та інструменти створення додатка 19
2.3. Вибір мови програмування 21
3 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ 25
3.1. Розробка UML –діаграми 25
3.2. Процедури та функції 31
3.3. Архітектура даних розроблювальної системи 34
4 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОГО Й ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ РОЗУМНИЙ БУДИНОК 39
4.1. Обґрунтування вибору технічних засобів реалізації інформаційної системи розумний будинок 39
4.2. Алгоритм оптимізації параметрів функціонування системи розумний будинок 42
4.3. Короткий опис програмної реалізації алгоритму 44
4.4. Розробка веб-сервісу для системи «Розумний дім» 47
4.5. Тестування системи «Розумний дім» 54
ВИСНОВКИ 61
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 65
ДОДАТОК 68

ВСТУП

Актуальність дослідження. Сучасні технології інформаційних систем впливають на всі сфери нашого життя, у тому числі й на житловий комфорт. Розумний будинок є одним із перспективних напрямків в розвитку інформаційних систем, що поєднує в собі концепції автоматизації та забезпечення комфорту у побуті. Він надає можливість забезпечити ефективне управління будинком за допомогою сучасних веб-технологій, що відкривають безліч нових можливостей для користувачів.

Одним з ключових аспектів цієї роботи є застосування сучасних веб-технологій. Веб-технології дозволяють створювати інтуїтивно зрозумілі та зручні інтерфейси для користувачів, що забезпечують доступ до функціональності розумного будинку через будь-який пристрій з підключенням до Інтернету. Це дозволяє користувачам зручно керувати будинком навіть на віддаленій відстані, використовуючи смартфони, планшети або комп'ютери.

Інформаційна система розумного будинку, заснована на сучасних веб-технологіях, відкриває широкі можливості для покращення енергоефективності та сталості будинків. Шляхом інтеграції датчиків та управління електричними пристроями, система може автоматично регулювати споживання електроенергії, опалення та освітлення з метою зниження витрат та екологічного впливу.

Також варто відзначити, що розумний будинок, заснований на веб-технологіях, сприяє покращенню життєвої безпеки. Завдяки вбудованим системам моніторингу, датчикам вогню, витоку газу та системам безпеки, користувачі можуть отримувати сповіщення та контролювати стан безпеки свого будинку з будь-якого місця, де є доступ до Інтернету. Це дозволяє запобігти небажаним подіям та негайно реагувати на них.

Необхідно також враховувати соціальний аспект розумних будинків. Інформаційна система розумного будинку може сприяти покращенню якості

життя людей з обмеженими фізичними можливостями або похилого віку, надаючи їм більшу незалежність та комфорт. Віддалене управління пристроями та автоматичне виконання рутинних завдань зменшують потребу в фізичних зусиллях та сприяють більш активному та здоровому способу життя.

Загалом, інформаційна система розумного будинку, побудована з використанням сучасних веб-технологій, відкриває безліч переваг і можливостей для поліпшення нашого життя. Вона сприяє ефективному управлінню будинком, підвищенню енергоефективності, покращенню безпеки та сприяє більш комфортному та доступному життю для людей. Дослідження та розробка в цій галузі мають великий потенціал для майбутнього розвитку інформаційних систем розумних будинків та зростання їх прийняття у суспільстві.

Тема інформаційної системи розумного будинку з використанням сучасних веб-технологій є надзвичайно актуальною у сучасному світі. За останні роки спостерігається стрімкий розвиток інтернету речей, штучного інтелекту, хмарних технологій та зростаючої доступності до високошвидкісного Інтернету. Ці технології створюють унікальні можливості для реалізації інформаційних систем розумних будинків з використанням веб-технологій.

Перш за все, зростаюча цікавість до енергоефективності та сталого розвитку ставить перед нами виклик ефективного управління ресурсами. Інформаційна система розумного будинку, побудована з використанням веб-технологій, дозволяє ефективно контролювати та керувати споживанням енергії, води та інших ресурсів. Це дозволяє не лише знизити витрати на комунальні послуги, але й зменшити негативний вплив на навколишнє середовище.

Загалом, інформаційна система розумного будинку з використанням сучасних веб-технологій відповідає потребам сучасного суспільства у комфортному, безпечному та енергоефективному житті. Розвиток таких систем має великий потенціал у покращенні якості життя людей, зменшенні впливу на

навколишнє середовище та стимулюванні інноваційного розвитку в сфері будівництва та технологій.

Враховуючи все вищесказане, нами і була обрана тема дипломної роботи:

"Інформаційна система розумний будинок із застосуванням сучасних веб-технологій".

Об’єкт дослідження – Особливості функціонування, роль та значення системи розумний будинок із застосуванням сучасних веб-технологій.

Предмет – способи та методи проектування системи розумний будинок із застосуванням сучасних веб-технологій.

Мета роботи: дослідити основні аспекти проектування системи розумний будинок із застосуванням сучасних веб-технологій.

Відповідно до мети були визначені наступні **завдання**:

1) Дати загальну характеристику системи розумний будинок у науковій літературі;

2) проаналізувати особливості розробки об’єктної моделі системи розумний будинок із застосуванням сучасних веб-технологій

3) зробити опис роботи з програмним додатком та його тестування.

Для розв’язання поставлених завдань нами були використані такі **методи дослідження**: теоретико-критичний аналіз літератури з теми дослідження; зіставлення, узагальнення і синтезування здобутої інформації тощо.

Робота може бути використана студентами ВНЗ для підготовки до семінарських занять, також може бути використана викладачами для проведення лекції, практик тощо.

Структура роботи. Робота складається зі вступу, чотирьох розділів, висновків, списку використаних джерел, що містить 32 найменування. Повний обсяг роботи: 60 сторінок.

1 АНАЛІЗ ПРОБЛЕМИ. ПОСТАНОВКА ЗАВДАННЯ ДОСЛІДЖЕННЯ

1.1. Поняття та особливості роботи системи розумний дім

Система розумного дому визначається як інтегрована мережа електронних пристроїв, сенсорів та розумних алгоритмів, що спрямована на автоматизацію та оптимізацію різних аспектів життя в будинку. Основна ідея полягає в тому, щоб забезпечити зручність, ефективність, безпеку та енергоефективність в управлінні різними системами будинку, такими як освітлення, опалення, кондиціонування повітря, безпека, розваги та багато інших.

Особливості роботи системи розумного дому включають:

Інтеграція різних пристроїв: Система розумного дому об'єднує різні пристрої та системи в єдину інтегровану мережу. Це можуть бути освітлювальні пристрої, термостати, системи безпеки, аудіо- та відеосистеми, електроприлади та інші пристрої. Інтеграція дозволяє керувати цими пристроями централізовано та забезпечує взаємодію між ними [23, .с 65].

Автоматизація та програмування: Система розумного дому дає можливість налаштовувати різні сценарії та розклади роботи для пристроїв у будинку. Наприклад, можна програмувати автоматичне включення світла або регулювання температури в певний час або за певних умов. Це дозволяє автоматизувати багато процесів та забезпечує зручність та ефективність в управлінні будинком.

Віддалений доступ та керування: Система розумного дому може бути керована з будь-якого пристрою з Інтернет-підключенням, наприклад, смартфона або планшета. Це дає можливість користувачу контролювати та керувати системою навіть здалеку. Наприклад, можна включити опалення або перевірити стан системи безпеки, коли ви не знаходитесь вдома.

Система аналітики та навчання: Деякі системи розумного дому можуть навчатися та аналізувати ваши звички, пристосовуючись до вашого способу

життя та потреб. Вони можуть автоматично регулювати налаштування пристроїв, прогнозувати ваші звички та пропонувати оптимальні рішення для енергоефективного використання ресурсів.

Забезпечення безпеки: Система розумного дому також може включати функції безпеки, які допомагають захистити будинок та жителів. Це можуть бути системи відеоспостереження, датчики протипожежної та протиопікової сигналізації, системи контролю доступу та інші [19, с. 85].

Усі ці особливості роботи системи розумного дому спрямовані на забезпечення зручності, комфорту, ефективності та безпеки в будинку. Вона дозволяє користувачам налаштовувати та контролювати різні аспекти домашнього середовища, забезпечуючи більш усвідомлене та енергоефективне життя.

Система розумного дому стала одним з найбільш актуальних та інноваційних розвитків в сфері домашньої автоматизації. Її популярність постійно зростає завдяки великій кількості переваг, які вона надає користувачам.

Однією з головних переваг системи розумного дому є зручність та комфорт. Завдяки можливості вдалого керування різними пристроями з одного інтерфейсу або навіть за допомогою голосових команд, користувач може контролювати освітлення, температуру, безпеку та інші системи будинку без зусиль. Це забезпечує максимальну зручність і забезпечує адаптацію системи до індивідуальних потреб користувача.

Ще однією вагомою перевагою є енергоефективність. Система розумного дому дозволяє оптимізувати споживання енергії, наприклад, автоматично вимикаючи світло або регулюючи температуру відповідно до присутності або відсутності людей у приміщенні. Це не лише знижує рахунки за комунальні послуги, але й сприяє збереженню енергоресурсів та екологічно чистому способу життя.

Розумний дім має великий потенціал для поліпшення зручності, комфорту та енергоефективності життя в будинку. Однак, важливо розуміти, що впровадження такої системи потребує уважного планування, вибору підходящих пристроїв та розгляду можливих проблем та викликів. Завдання полягає в тому, щоб забезпечити належну безпеку, зручність та ефективність роботи системи розумного дому, забезпечуючи задоволення та комфорт користувачів.

Крім того, важливо розглянути й інші аспекти системи розумного дому. Одна з особливостей полягає в тому, що вона може надавати користувачеві повний контроль над різними аспектами будинку, такими як освітлення, опалення, безпека, енергозбереження та розваги. Користувач може віддалено керувати пристроями за допомогою мобільних додатків або голосових помічників, що додає зручності і флексібельності.

Одна з ключових переваг систем розумного дому полягає у можливості автоматизації рутинних завдань. Наприклад, система може автоматично вмикати й вимикати освітлення, регулювати температуру в будинку в залежності від погодних умов або розпочинати процес приготування їжі перед приходом додому. Це забезпечує зручність та ефективність, а також може допомогти зменшити споживання енергії та знизити витрати [26, с 87].

У підсумку, система розумного дому, що використовує сучасні веб-технології, має великий потенціал у поліпшенні комфорту, безпеки та енергоефективності житлових приміщень. Проте, вона також зіткнувається з певними викликами, такими як безпека даних, технічні проблеми та сумісність з іншими пристроями. Для розв'язання цих проблем варто вживати заходів, таких як захист даних, регулярне обслуговування, вибір надійних пристроїв, резервне джерело живлення та системи резервного копіювання. Користувацький досвід, взаємодія зі стандартами та протоколами, розширюваність та масштабованість є також важливими аспектами. Загалом, система розумного дому може стати

важливою складовою сучасного життя, принесе багато переваг та сприятиме створенню зручного та ефективного середовища проживання.

1.2. Аналітичній огляд систем розумний будинок

Аналітичний огляд систем розумного будинку розкриває широкий спектр можливостей і переваг, які ці системи пропонують для покращення якості життя. Огляд розглядає ключові характеристики, функціональні можливості та вплив цих систем на енергоефективність, безпеку та комфорт життя в будинку.

Системи розумного будинку дозволяють мешканцям забезпечувати контроль та керування різними аспектами будинку, такими як освітлення, опалення, кондиціонування повітря, безпека, розселення та багато іншого. Вони можуть бути управляні через веб-інтерфейс, мобільний додаток або голосові помічники, надаючи мешканцям зручність та гнучкість управління будинком.

Одна з головних переваг систем розумного будинку полягає в їх здатності до автоматизації. Завдяки розумним датчикам, програмованим сценаріям та штучному інтелекту, системи можуть автоматично виконувати задані дії відповідно до умов та налаштувань користувача. Наприклад, система може розпізнавати, коли немає мешканців в будинку і автоматично вимикати зайве освітлення та прилади, забезпечуючи енергозбереження [28, с. 64].

Огляд також розглядає важливість безпеки в системах розумного будинку. За допомогою відеокамер, датчиків руху та систем моніторингу, системи розумного будинку забезпечують постійний контроль та можливість отримання сповіщень про потенційні небезпеки, такі як пожежа, витік води або незвичайні активності. Це дозволяє мешканцям вчасно реагувати на небезпеку та запобігати можливим збиткам.

Вплив систем розумного будинку на енергоефективність є значним. Вони дозволяють оптимізувати використання енергії, працюючи в режимі

енергозбереження, автоматично вимикаючи електричні прилади, коли вони не використовуються, або регулюючи освітлення та температуру відповідно до потреб мешканців та умов зовнішнього середовища.

У підсумку, системи розумного будинку представляють значний потенціал для покращення якості життя, забезпечення енергоефективності та безпеки в будинку. Вони перетворюють будинки в розумні екосистеми, які можуть пристосовуватися до потреб та пріоритетів мешканців. З ростом інтересу до смарт-технологій та зростання доступності цих рішень на ринку, системи розумного будинку стають все більш популярними та важливими компонентами сучасного життя [37, с. 40].

З впровадженням систем розумного будинку відкриваються безліч перспективних можливостей для розширення функціональності та зручності життя. Сучасні системи розумного будинку базуються на використанні сучасних веб-технологій, таких як Інтернет речей (IoT), хмарні обчислення, штучний інтелект (AI) та блокчейн.

Загалом, системи розумного будинку із застосуванням сучасних веб-технологій відкривають нові можливості для покращення якості життя, ефективного використання ресурсів та створення розумних, інтелектуальних середовищ для мешканців. Вони представляють справжню революцію в сфері будівництва і мають потенціал перетворити наші будинки у справжні розумні домівки, де кожен аспект життя можна контролювати та налаштовувати під свої потреби.

1.3. Постановка завдання

Постановка завдання для розробки Інформаційної системи розумного будинку із застосуванням сучасних веб-технологій передбачає визначення

конкретних цілей та завдань проекту, що дозволять досягти успішної реалізації системи. Основні етапи постановки завдання включають наступні аспекти:

1. Аналіз вимог: Перший етап передбачає детальний аналіз потреб та вимог користувачів до системи розумного будинку. Це можуть бути вимоги щодо енергоефективності, безпеки, комфорту, зручності управління та інших функціональних аспектів. Важливо визначити основні функції, які має виконувати система, а також необхідність інтеграції з іншими пристроями та сервісами [16 ,с. 54].

2. Технічне проектування: На цьому етапі визначається технічна архітектура системи розумного будинку. Вона включає в себе вибір веб-технологій для розробки інтерфейсу та зв'язку, а також визначення апаратних компонентів і сенсорів, що необхідні для збору даних та керування різними системами будинку.

3. Розробка та тестування: На даному етапі розробляються інформаційні та програмні компоненти системи розумного будинку. Це включає розробку веб-інтерфейсу, бази даних, модулів керування та алгоритмів, які забезпечують функціональність системи. Після розробки проводиться тестування, щоб переконатися в коректності роботи всіх компонентів та їх взаємодії.

4. Впровадження та підтримка: Цей етап передбачає впровадження системи розумного будинку в реальне середовище. Важливо забезпечити налагодження та інтеграцію всіх компонентів, а також забезпечити навчання користувачів щодо використання системи. Крім того, необхідна підтримка та обслуговування системи після впровадження, щоб забезпечити її надійну та безперебійну роботу.

5. Оцінка результатів: На завершальному етапі проводиться оцінка результатів розробки Інформаційної системи розумного будинку. Важливо визначити, наскільки вдало вдалося виконати вимоги користувачів, оцінити

функціональність та продуктивність системи, а також врахувати отримані відгуки користувачів.

Розробка такої системи вимагає аналізу потреб та вимог користувачів, вибору технологічної платформи, розробки інтегрованої системи, забезпечення безпеки та конфіденційності даних, а також тестування та впровадження.

Загалом, інформаційна система розумного будинку з використанням сучасних веб-технологій є потужним інструментом для створення зручного та ефективного домашнього середовища. Вона сприяє покращенню якості життя користувачів, забезпечує енергоефективність та допомагає зберігати енергію, а також сприяє розвитку екологічно чистого способу життя.

2 ОБ’ЄКТИВНА ТИПОЛОГІЯ СИСТЕМИ РОЗУМНИЙ БУДИНОК

2.1. Короткий огляд відомих типологій

В наші дні існує безліч відомих та нових брендів, але кожен з них має свої переваги і недоліки. Саме про це розповідатиметься в цьому розділі, де ми розглянемо і визначимо сильні та слабкі сторони деяких світових брендів, зокрема: Amazon Echo; Nest Learning Thermostat; Apple HomeKit Netatmo; Meizu LifeKit; Google Home [14, с. 57].

Основними параметрами, на які ми звернемо увагу, будуть: Масштабованість; Ціна; Підтримка сучасних операційних систем смартфонів; Мова розпізнавання команд, якщо вона передбачена; Підтримка сторонніх засобів управління .

Amazon Echo (скорочено Echo) - це лінійка розумних пристроїв, розроблених компанією Amazon.com. Ці пристрої підключаються до голосового інтелектуального помічника під назвою Alexa, який реагує на ім'я "Alexa". Це ключове слово може бути змінене користувачем на "Amazon", "Echo" або "Computer". Пристрій забезпечує голосову взаємодію, відтворення музики, створення списків справ, налаштування нагадувань, трансляцію подкастів, прослуховування аудіокниг, надання інформації про погоду, дорожній рух та багато іншого в режимі реального часу.

Nest Learning Thermostat - це інтелектуальний термостат, розроблений компанією Nest Labs (див. рис. 1.4). Цей електронний програмований пристрій є "навчаємим" і має вбудовану підтримку Wi-Fi. Він оптимізує системи опалення та охолодження в будинках та підприємствах для забезпечення енергоефективності.

Рис. 1.1 – Вигляд системи Nest Learning Thermostat

Заснований на алгоритмі машинного навчання, Nest Learning Thermostat вимагає від користувачів регулювання термостата протягом перших тижнів, щоб створити набір еталонних даних. Після цього пристрій "навчається" графіку використання температури користувачами і включає режим енергозбереження, коли він розпізнає, що ніхто не перебуває вдома.

HomeKit - це програмне забезпечення від Apple, яке дозволяє користувачам настроювати свої iPhone або інші пристрої Apple для керування розумною побутовою технікою, як показано на рисунку 1.5. Користувачі можуть управляти побутовими пристроями в своєму будинку простим голосовим диктуванням через Siri або за допомогою додатків. HomeKit був випущений разом з iOS 8 у вересні 2014 року.

Цей продукт має такі переваги: Вбудований голосовий помічник. Підтримка всіх сучасних смартфонів. Можливість підключення сторонніх засобів домашньої автоматизації. Багатий набір готових функцій.

Проте він має деякі недоліки: Висока ціна. Проблеми з масштабованістю. Не підтримує OS Android.

Рис. 1.2. Вигляд системи управління Apple HomeKit Netatmo

Компанія Meizu з Китаю також приєдналася до виробників домашньої автоматизації, випустивши нову систему Lifekit, яка складається з кількох пристроїв, таких як розумна лампочка (X-Plus SMART Bulb), підлогові ваги (RyFit), очищувач повітря (Air Box) і розумний кондиціонер (Smart A/C), що контролює якість повітря в приміщенні.

Система від Google дозволяє користувачам взаємодіяти з сервісами за допомогою інтелектуального персонального помічника Google, відомого як Google Assistant. Вона зображена на рисунку 1.10. В системі інтегровано велику кількість внутрішніх та сторонніх функцій. За допомогою голосових команд користувачі можуть слухати музику, керувати відтворенням відео і фотографій, отримувати новини та багато іншого. Продукти Google також мають вбудовану підтримку домашньої автоматизації, що дозволяє користувачам керувати розумними пристроями голосом. Google Home можна розміщувати в різних кімнатах будинку для синхронного відтворення музики. Оновлення в квітні 2017 року додало підтримку декількох користувачів, дозволяючи пристрою розпізнавати до шести осіб за голосом.

У травні 2017 року Google опублікувала кілька оновлень для Google Home, зокрема: можливість безкоштовних телефонних дзвінків в режимі "вільні руки" в США та Канаді, оновлення з можливістю попередження про заплановані події.

Рис. 1.3. Вигляд системи Google Home

Підсумовуючи проведений огляд, можна зазначити декілька проблем, які відзначаються в сучасних системах управління для розумних будинків. Це, зазвичай, висока ціна, обмежений перелік готових функцій та відсутність українського та російського розпізнавання голосових команд. Варто зазначити, що сучасні бренди поступово намагаються вирішити ці проблеми, але це складне завдання.

2.2. Середовище розробки та інструменти створення додатка

Веб-сервіс - це програмна система, яка призначена для сприяння взаємодії між комп'ютером і машиною через мережу. Він має інтерфейс, який описаний у форматі, що працює з серверами, зокрема WSDL. Інші системи взаємодіють з веб-службою за допомогою SOAP-повідомлень, які зазвичай передаються через HTTP, використовуючи XML для серіалізації, разом з іншими Інтернет-стандартами. Отже, в основному веб-сервіси базуються на базовому наборі стандартів, які описують синтаксис і семантику комунікації. XML використовується для загального синтаксису представлення даних, SOAP забезпечує семантику обміну даними, а WSDL надає механізм для опису можливостей веб-служби. PHP є чудовим вибором для веб-програмування. PHP-скрипти виконуються на сервері і підтримують безліч баз даних, таких як

MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL, Generic ODBC та інші. Одна з основних переваг PHP полягає в тому, що це програмне забезпечення з відкритим кодом, яке можна безкоштовно завантажити та використовувати. PHP в поєднанні з MySQL є крос-платформовим і працює як в середовищі Windows, так і на платформі UNIX. PHP сумісний з різними веб-серверами, що використовуються сьогодні. [46, с. 55].

Веб-сервіс складається з сервера, який обслуговує запити до веб-сервісу, та клієнта, який взаємодіє з веб-сервісом за допомогою його методів. В PHP доступні розширення SOAP для розробки серверів і клієнтів SOAP, а також розширення XMLRPC для створення серверів і клієнтів XML-RPC.

Однією з ключових переваг веб-сервісів є їхній крос-платформений і крос-мовний характер. Наприклад, PHP-скрипт, що працює на Linux, може легко взаємодіяти з сервером IIS на Windows, використовуючи ASP [67].

Під час проектування системи був проведений аналіз предметної області і вимог замовника. Було вирішено розробити програмний продукт, який базується на веб-технологіях та включає мобільний додаток. Технології, використані на сервері, відповідають принципам відкритого вихідного коду, актуальності в наш часі та можливості використання на будь-якій операційній системі. Тому було обрано Express.js, який дозволяє швидко та якісно розробляти як великі, так і малі серверні системи[23, с. 75].

На стороні клієнта були вибрані технології, що відповідають тим же вимогам, що й на сервері, з урахуванням їх виконання у браузері. Фреймворк React.js був вибраний для розробки складних графічних інтерфейсів, які легко тестувати та розширювати під час подальшої розробки програмного забезпечення. Мова програмування C# була обрана через її можливість несуворої типізації та підтримки системи модулів. Використання даної мови програмування надає різноманітний "синтаксичний цукор", що дозволяє скоротити обсяг шаблонного коду. Для реалізації архітектури Flux була

використана бібліотека Redux, яка дозволяє повністю інтегрувати цю архітектуру у фреймворк React.js.

Цей набір обраних технологій є ефективним і гнучким рішенням для будь-якого проекту, дозволяючи забезпечити високу якість, надійність і безпеку програмного продукту. Використання відкритого вихідного коду дозволяє залучати спільноту розробників, отримувати підтримку та спільно вдосконалювати інструменти.

В результаті, обраний стек технологій допомагає забезпечити успішну розробку і експлуатацію програмного продукту, забезпечуючи його конкурентоспроможність, стабільність і високу якість на ринку.

2.3. Вибір мови програмування

C++ - компіліруєма, статично типізована мова програмування загального призначення.

Підтримує такі парадигми програмування, як процедурне програмування, об'єктно-орієнтоване програмування, узагальнене програмування. Мова має багату стандартну бібліотеку, яка містить у собі розповсюджені контейнери й алгоритми, введення- висновок, регулярні вираження, підтримку многопоточности й інші можливості. C++ поєднує властивості як высокоуровневых, так і низкоуровневых мов. У порівнянні з його попередником - мовою C - найбільша увага приділена підтримці об'єктно-орієнтованого й узагальненого програмування [5, с. 81].

C++ широко використовується для розробки програмного забезпечення, будучи одним із самих популярних мов програмування. Область його застосування включає створення операційних систем, різноманітних прикладних програм, драйверів обладнань, додатків для, що вбудовуються систем, високопродуктивних серверів, а також комп'ютерних ігор. Існує безліч реалізацій

мови С++, як безкоштовних, так і комерційних і для різних платформ. Наприклад, на платформі x86 це GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder і інші. С++ вплинув на інші мови програмування, у першу чергу на Java і С++.

Синтаксис С++ успадкований від мови С. Споконвічно одним із принципів розробки було збереження сумісності з С. Проте С++ не є в точному значенні надбезліччю С; безліч програм, які можуть однаково успішно транслюватися як компіляторами С, так і компіляторами С++, досить велике, але не включає всі можливі програми на С.

Стандарт С++ складається із двох основних частин: опис ядра мови й опис стандартної бібліотеки.

Перший час мова розбудовувалася поза формальними рамками, спонтанно, у міру завдань, що вставали перед ним. Розвитку мови супроводжував розвиток крос- компілятора cfront. Нововведення в мові відбивалися в зміні номера версії кросу-компілятора. Ці номери версій крос- компілятора поширювалися й на саму мову, але стосовно до теперішнього часу мова про версії мови С++ не ведуть. Лише в 1998 році мова стала стандартизованою.

С++ підтримує як коментарі в стилі С (/* коментар */), так і однорядкові (// уся частина, що залишився, рядка є коментарем), де // позначає початок коментаря, а найближчий наступний символ нового рядка, який не випереджений символом \ (або еквівалентним йому позначенням ??/), уважається закінченням коментаря. Плюс цього коментаря в тому, що його не обов'язково закінчувати, тобто позначати закінчення коментаря.

Специфікатор inline для функцій. Функція, певна усередині тіла класу, є inline за замовчуванням. Даний специфікатор є підказкою компіляторові й може вмонтувати тіло функції в код замість її безпосереднього виклику [9, с. 65].

Квалификаторы const і volatile. На відміну від З, де const позначає тільки доступ на читання, в С++ змінна із квалификатором const повинна бути

инициализирована. volatile використовується в описі змінних і інформує компілятор, що значення даної змінної може бути змінене способом, який компілятор не в змозі відстежити. Для змінних, оголошених volatile, компілятор не повинен застосовувати засоби оптимізації, що змінюють положення змінної в пам'яті (наприклад, що поміщають її в регістр) або належні на незмінність значення змінної в проміжку між двома присвоюваннями їй значення. У многоядерной системі volatile допомагає уникати бар'єрів пам'яті 2- го типу.

Спеціальним випадком є безіменний простір імен. Усе імена, описані в ньому, доступні тільки в поточній одиниці трансляції й мають локальне зв'язування. Простір імен std містить у собі стандартні бібліотеки C++.

- Для роботи з пам'яттю введені оператори new, new[], delete і delete[]. На відміну від бібліотечних malloc і free, пришедших з C, дані оператори роблять ініціалізацію об'єкта. Для класів це виклик конструктора, для POD типів ініціалізацію можна або не проводити(new Pod;), або провести ініціалізацію нульовими значеннями (new Pod(); new Pod {};).

В C++ доступні наступні вбудовані типи. Типи C++ практично повністю повторюють типи даних в C:

- символьні: char, wchar_t (char16_t і char32_t, у стандарті C++11);
- числові знакові: signed char, short int, int, long int (і long long, у стандарті C++11);
- беззнакові: unsigned char, unsigned short int, unsigned int, unsigned long int (і unsigned long long, у стандарті C++11);
- із плаваючою точкою: float, double, long double;
- логічний: bool, що має значення або true, або false.

Операції порівняння повертають тип bool. Вираження в дужках після if, while приводяться до типу bool[14].

Мова ввела поняття посилань, а зі стандарту C++11 rvalue- посилання й передане посилання (англ. forwarding reference) [13 ,с. 80].

C++ додає до C об'єктно-орієнтовані можливості. Він уводить класи, які забезпечують три найважливіші властивості ООП: інкапсуляцію, спадкування й поліморфізм.

У стандарті C++ під класом (class) мається на увазі користувацький тип, оголошений з використанням одного із ключових слів class, struct або union, під структурою (structure) мається на увазі клас, певний через ключове слово struct, і під об'єднанням (union) мається на увазі клас, певний через ключове слово union. Залежно від використаного ключового слова, міняються також і деякі властивості самого класу. Наприклад, у класі, оголошеним через struct, члени без вручну прописаного модифікатора доступу будуть за замовчуванням мати публічний рівень доступу, а не приватний.

У якості використовуваних користувачем операційних систем були обрані системи сімейства Windows, оскільки саме вони найчастіше використовуються для роботи.

3 МОДЕЛЮВАННЯ ТА ПРОЕКТУВАННЯ

3.1. Розробка UML –діаграми

UML-діаграма прецедентів відображає зв'язки між акторами та прецедентами, тоді як UML-діаграма діяльності описує складові частини конкретної діяльності. У даному технологічному процесі вплив на систему мають температура та вологість у кімнатах.

UML-діаграма діяльності веб-сервісу для інформаційної системи "Розумний дім" на основі веб-технологій може включати наступні складові частини:

1. Актори:

Користувач: основний користувач системи "Розумний дім", який взаємодіє з веб-сервісом.

Система: представляє інформаційну систему "Розумний дім" як цілісний компонент.

2. Діяльності:

Автентифікація: користувач вводить свої облікові дані для аутентифікації в системі.

Налаштування: користувач має можливість змінювати налаштування системи, такі як температура, освітлення, безпека тощо.

Моніторинг: користувач може переглядати поточний стан різних пристроїв та датчиків в розумному домі.

Розклад: користувач може налаштовувати графік роботи пристроїв, наприклад, автоматичне вмикання світла або регулювання температури в певний час.

Сповіщення: система може надсилати сповіщення користувачеві про події, такі як спрацювання сигналізації або використання енергії [37, с. 62].

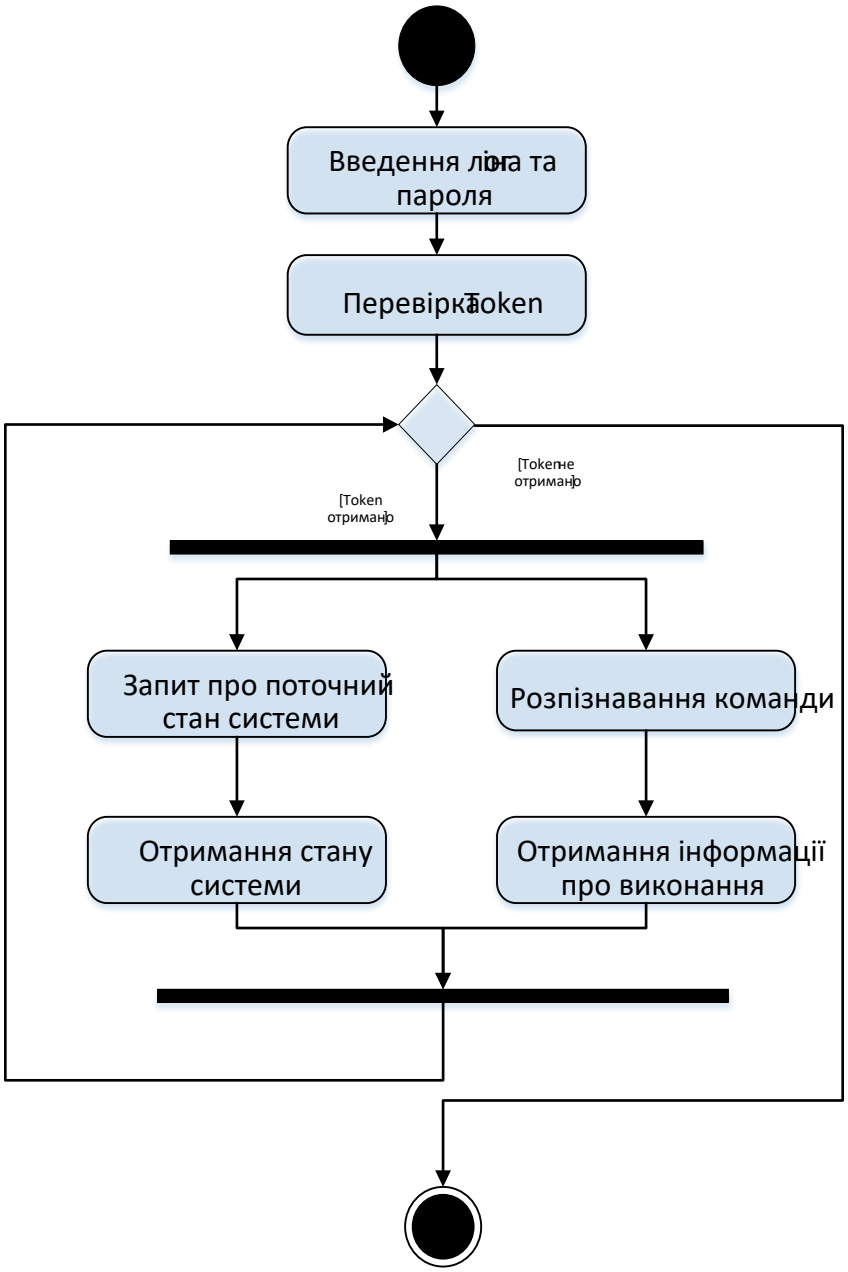


Рис. 3.1 UML-діаграма діяльності клієнта

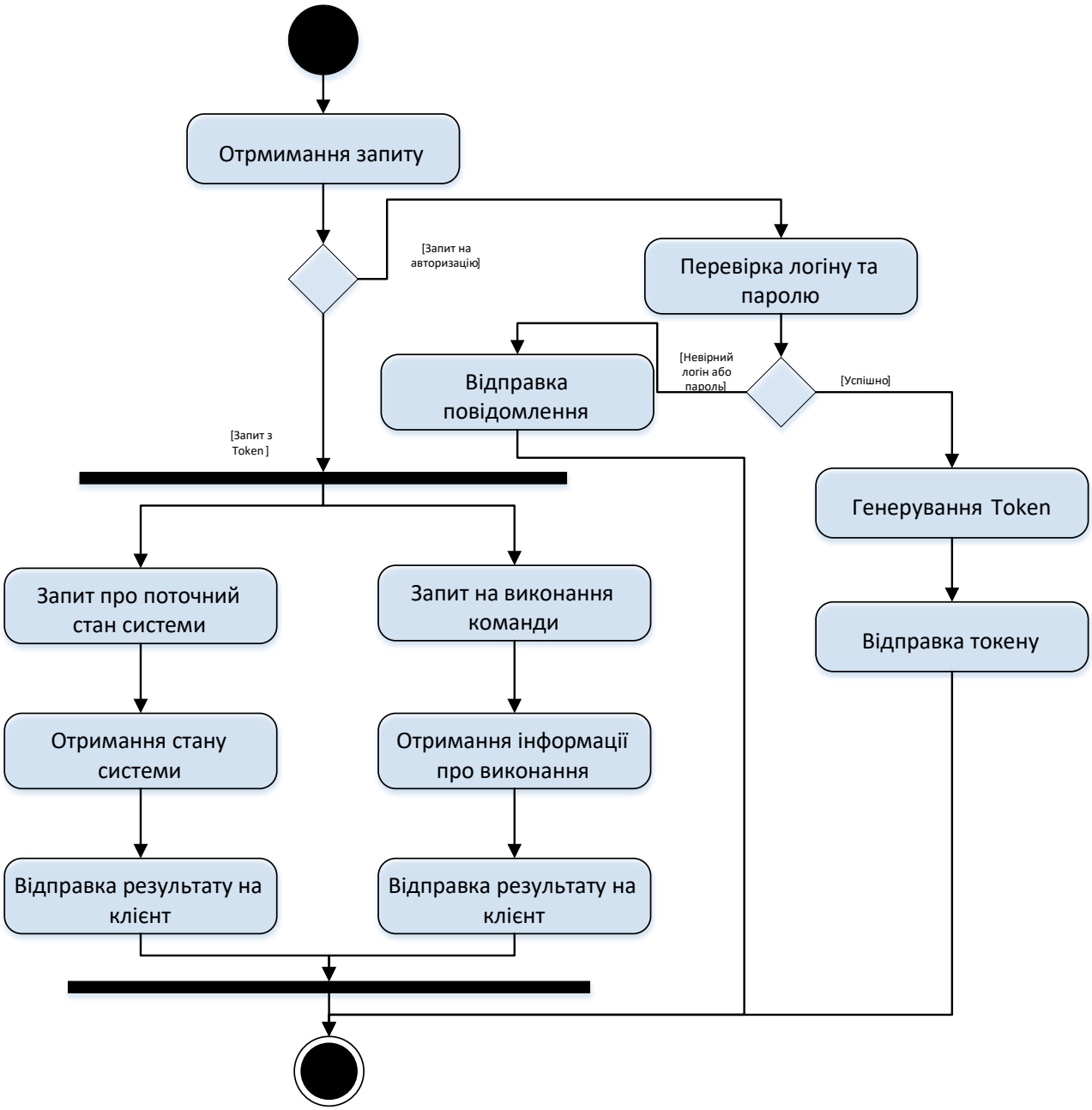


Рис. 3.2. UML-діаграма діяльності веб-сервісу

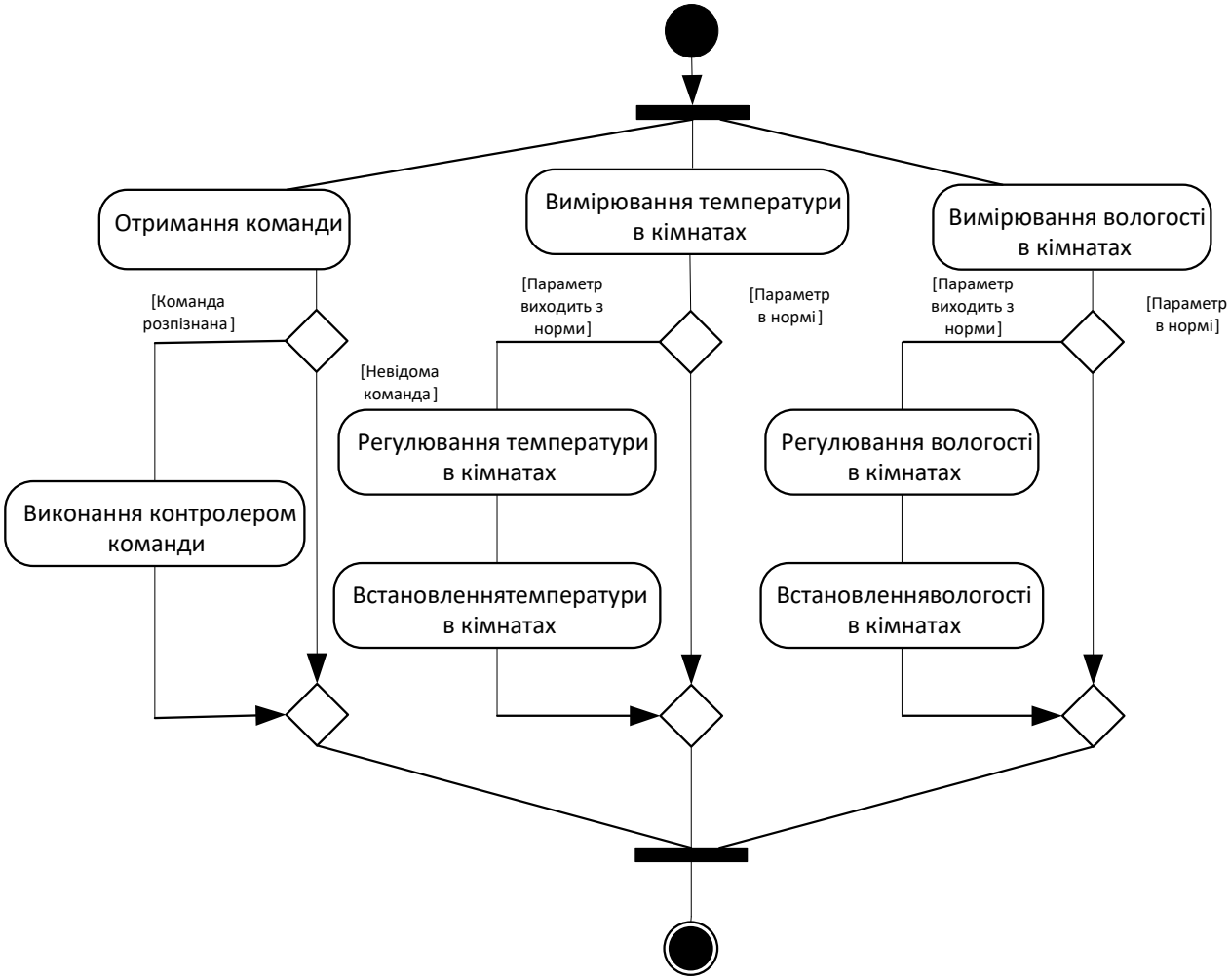


Рис. 3.3. UML-діаграма діяльності мікроконтролера

UML-діаграма діяльності мікроконтролера може відображати послідовність виконання дій та взаємодію з пристроями та датчиками у розумному домі. Вона дозволяє уявити процес обробки та керування даними в мікроконтролері, а також комунікацію з веб-сервісом. На діаграмі можуть бути зображені актор (мікроконтролер), пристрої, датчики та стрілки, що показують порядок виконання дій [51 ,с. 87].

Ця діаграма допомагає розуміти, як мікроконтролер взаємодіє з пристроями та датчиками, як він зчитує дані, виконує логіку та керує пристроями у розумному домі. Вона є важливим інструментом для розробників та інженерів, щоб визначити та уточнити логіку роботи мікроконтролера в системі "Розумний дім" на основі веб-технологій.

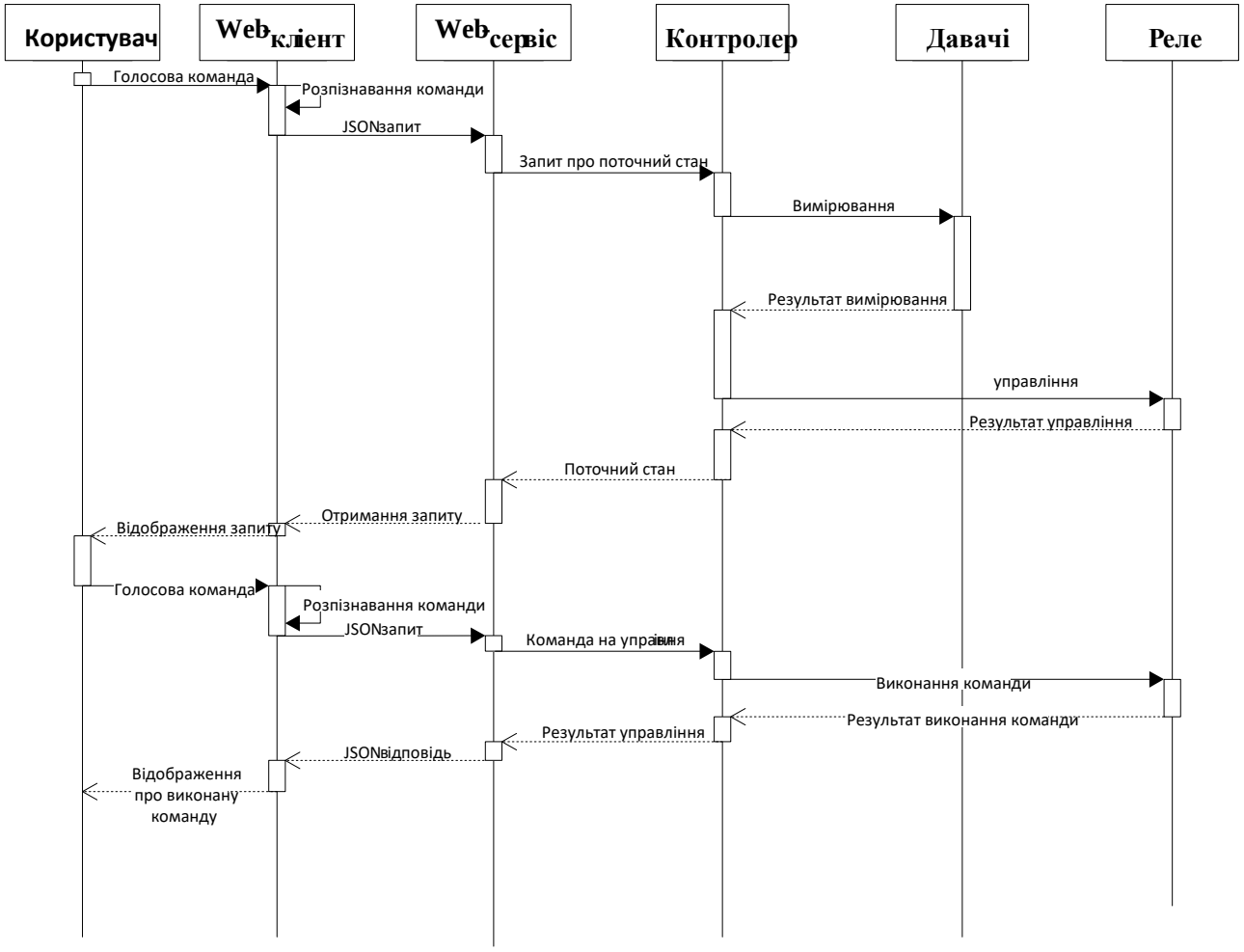


Рис. 3.4. UML-діаграма послідовності

Розроблена система має здійснювати вимірювання температури. У випадку, коли температура перевищує оптимальне значення, вмикається реле. Це реле, в свою чергу, активує систему охолодження. Якщо температура опускається нижче допустимої межі, активується система обігріву. Крім того, система повинна виконувати голосові команди. Це можливо реалізувати за допомогою клієнта, веб-сервісу та мікроконтролера. Останній з перерахованих виконує функцію вмикання та вимикання спеціалізованих реле [44 ,с. 74].

Розроблена структурна схема даної системи зображена на рисунку 3.5.

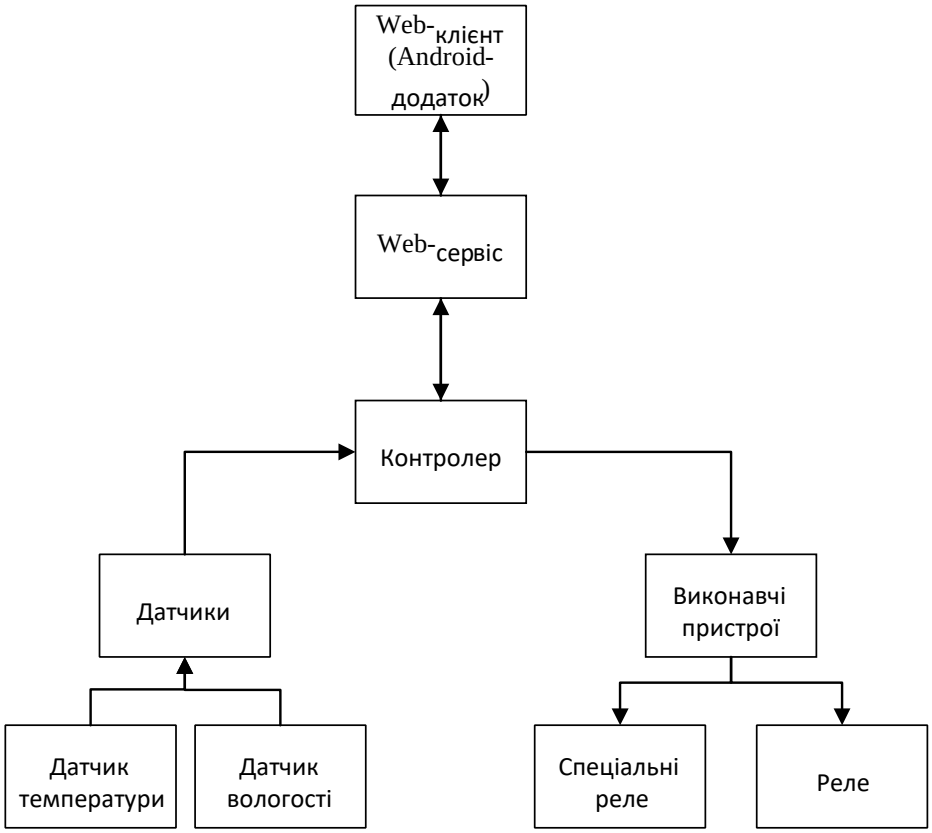


Рис. 3.5. Структурна схема системи

Основними впливовими факторами під час роботи апарату для розумного будинку є температура в кімнатах та вологість в кімнатах. Для вимірювання цих параметрів використовуються датчики, які передають результати на мікроконтролер. Отримані значення параметрів передаються до регулюючих пристроїв, які обчислюють регульований вплив і передають його до виконавчих механізмів, у даному випадку - спеціалізовані реле [25, с. 87].

Також поточні дані передаються на веб-сервіс, який розташований далеко від будинку споживача. Зв'язок між веб-сервісом та мікроконтролером здійснюється через мережу. Веб-сервіс, в свою чергу, взаємодіє з клієнтом також за допомогою мережі, передаючи розпізнані голосові команди. На рисунку 3.6 зображена функціональна схема даної системи.

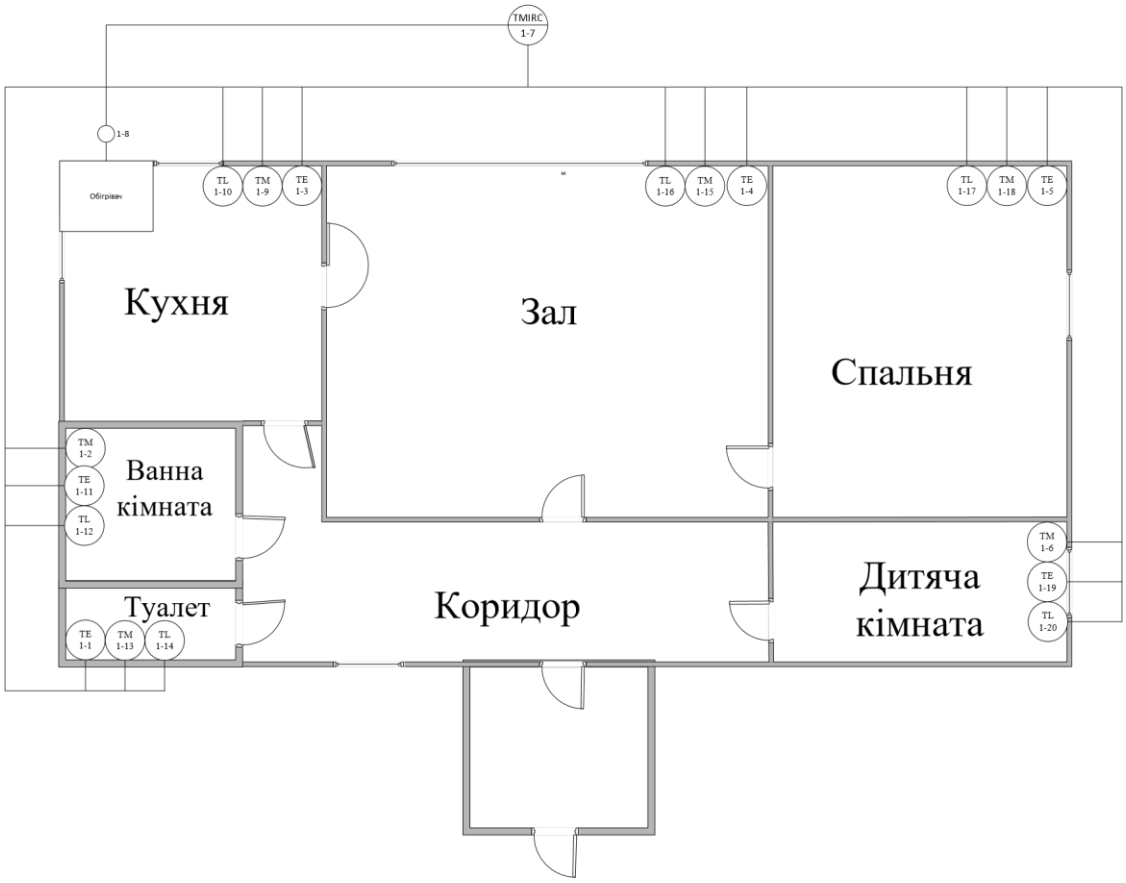


Рис. 3.6. Функціональна схема КСУ розумним домом

Ця функціональна схема системи розумного будинку на основі веб-технологій ілюструє важливі елементи та взаємозв'язки між ними. Значення температури та вологості в кімнатах вимірюються датчиками, які передають дані на мікроконтролер.

3.2. Процедури та функції

В даній роботі передбачається розробка системи управління "розумний дім", яка буде складатися з трьох рівнів, що є типовим для подібних систем. Перший рівень виступатиме в ролі клієнта і буде включати модуль голосового розпізнавання, який передаватиме розпізнані команди на другий рівень. Другий рівень буде виконувати функції веб-сервісу і буде взаємодіяти з першим та третім

рівнями. Цей рівень буде розроблений для віддаленого користування та базуватиметься на концепції "інтернету речей".

Переваги використання веб-сервісів на другому рівні включають можливість зміни обладнання на третьому рівні (наприклад, підвищення продуктивності мікроконтролера) або зміну операційної системи на першому рівні (наприклад, заміна смартфона з Android OS на Mac OS). Крім того, весь код системи залишається прихованим від користувачів, що забезпечує більшу безпеку для системи. Третій рівень виступатиме у ролі контролера, який виконуватиме вказівки, надіслані з веб-сервісу [26 ,с. 75].

Основні функції системи включають:

1. На рівні клієнта:

- Розпізнавання голосових команд.

- Відображення стану системи.

- Підключення до веб-сервісу.

2. На рівні веб-сервісу:

- Авторизація клієнтів.

- Обмін даними між клієнтом та мікроконтролером.

3. На рівні мікроконтролера:

- Виконання команд, отриманих від веб-сервісу.

- Відправлення поточного стану на веб-сервіс у відповідь на запит.

- Вимірювання та регулювання температури та вологості.

- У даній системі ролі виконуються такими елементами, як користувач, клієнт, веб-сервіс та мікроконтролер.

UML-діаграма прецедентів подана на рисунку 3.7.

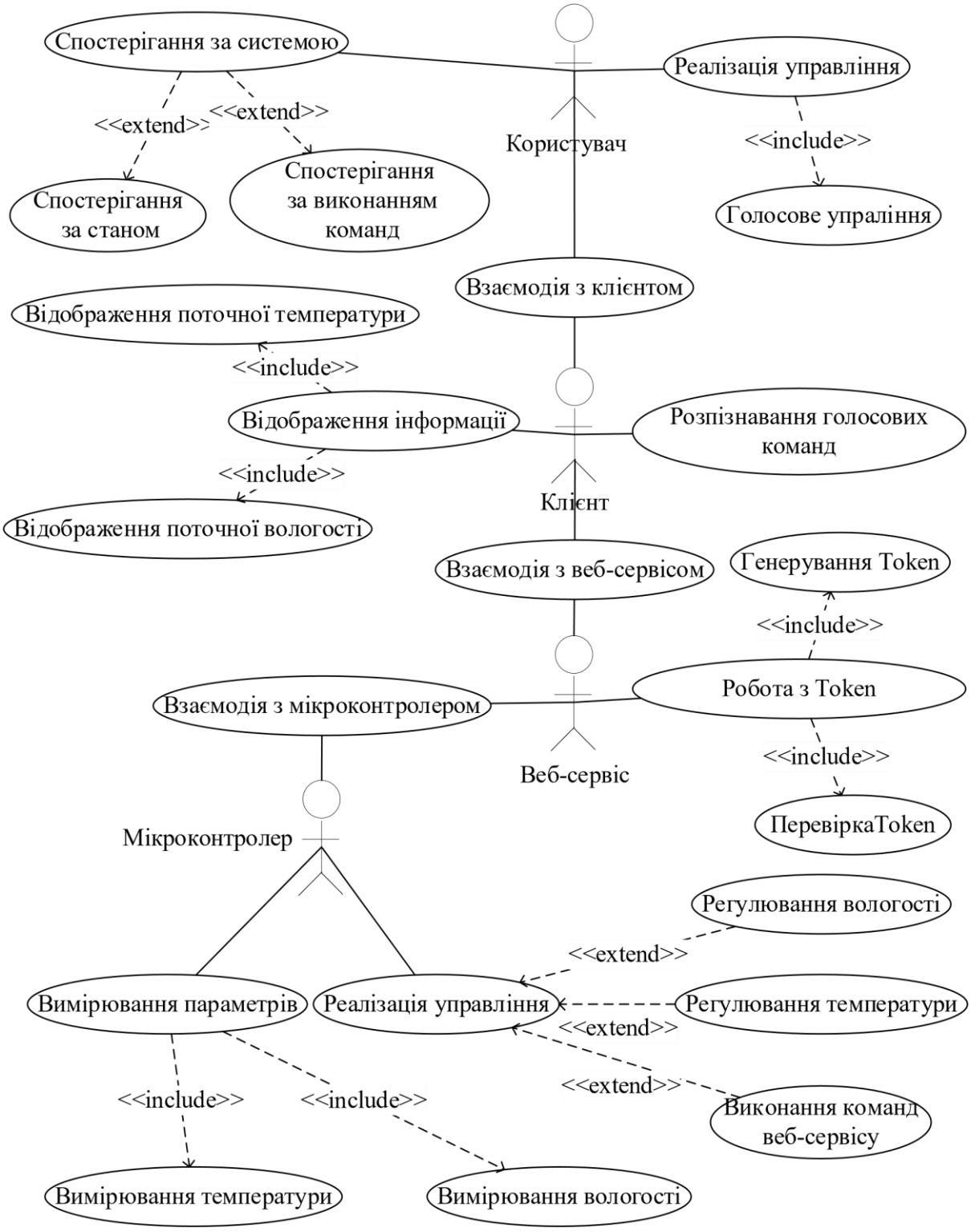


Рис. 3.7. UML-діаграма прецедентів

3.3. Архітектура даних розроблювальної системи

Для досягнення поставленої мети було вирішено використати таку архітектуру системи, яка включає веб-застосунок, мобільний застосунок, сервер та базу даних.

Веб-застосунок є центральним елементом системи, який забезпечує доступ користувачів до функціональності системи через веб-інтерфейс. Він надає зручний та інтуїтивно зрозумілий інтерфейс для управління різними аспектами "розумного дому", такими як освітлення, температура, безпека та інші. Користувачі можуть здійснювати дії та контролювати систему з будь-якого пристрою з підключенням до Інтернету [38, с. 54].

Мобільний застосунок розширює можливості системи, надаючи користувачам зручний спосіб керування "розумним домом" зі своїх мобільних пристроїв, таких як смартфони або планшети. Це дозволяє користувачам контролювати систему навіть віддалено, незалежно від їхнього місцезнаходження.

Сервер є центральним узлом системи, де знаходиться логіка обробки даних, обміну інформацією та керування взаємодією між веб-застосунком, мобільним застосунком та іншими компонентами системи. Він забезпечує безперебійну роботу системи, зберігає та обробляє дані, відповідає на запити користувачів та забезпечує координацію всіх компонентів системи.

Схема обраної архітектури зображена на рисунку 3.8.

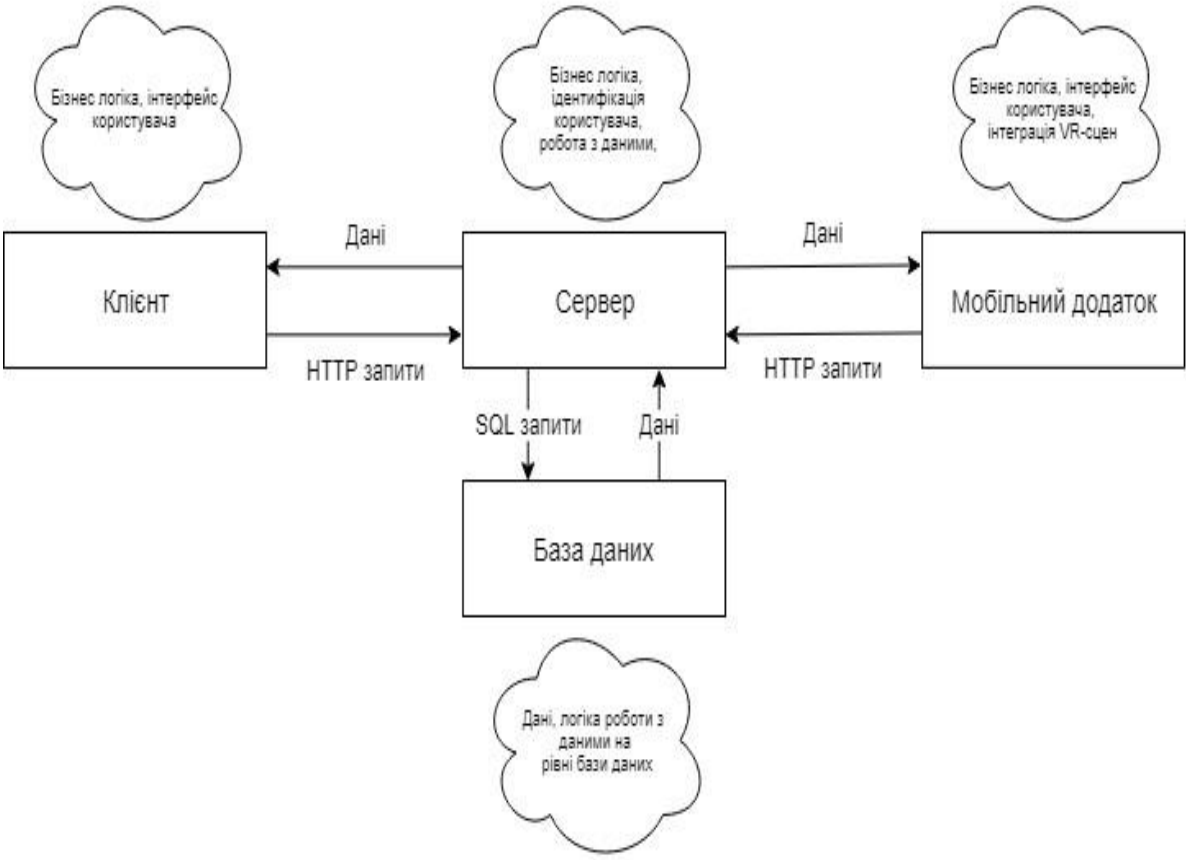


Рис. 3.8. Архітектура програмного комплексу

У вибраній архітектурі, головним елементом є сервер, який забезпечує централізоване управління бізнес-логікою та контроль доступу до бази даних. Сервер відповідає за ідентифікацію користувача для надання персоналізованого доступу до додатку. Це робиться для запобігання несанкціонованому використанню та пошкодженню даних. Авторизація користувача в системі реалізується на рівні сервера, оскільки це необхідна логіка для користування програмою.

Таким чином, шаблон CoR надає гнучкість та можливість легко розширювати систему обробки запитів, забезпечуючи послідовну передачу запитів по ланцюжку обробників [19 ,с. 73].

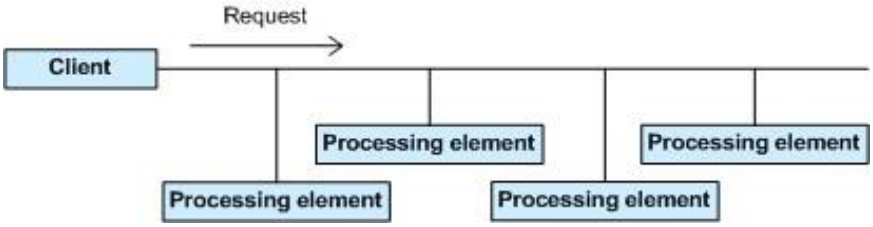


Рис. 3.9. Схема роботи CoR шаблону

Основна мета використання шаблону Chain of Responsibility полягає в полегшенні розширення обробників запитів, уникненні складних умов та розгалужень. Замість цього, сервер може бути розділений на умовні частини, де кожна частина відповідає за конкретний вид запитів.

Шаблон Chain of Responsibility дозволяє уникнути залежності між відправником запиту та його одержувачем, дозволяючи кільком об'єктам обробляти запит. Об'єкти-одержувачі зв'язуються в ланцюжок, і запит передається вздовж цього ланцюжка, поки не буде оброблений.

Цей шаблон дозволяє гнучко налаштовувати обробку запитів та уникати прямої залежності між клієнтом та обробниками, розділяючи обов'язки між різними об'єктами у ланцюжку.

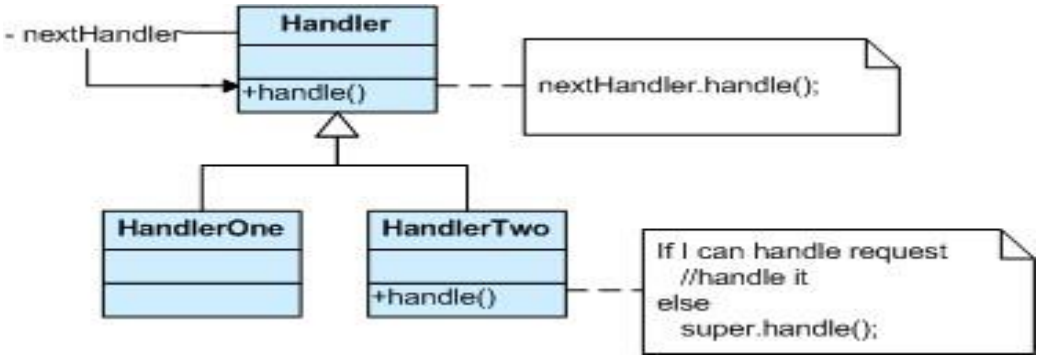


Рис. 3.10 UML-діаграма класів CoR шаблону

Обробники можуть вносити свій вклад в обробку кожного запиту. Запит може бути переданий по всій довжині ланцюжка до самого останнього ланки.

Для реалізації клієнтського застосунку було обрано фреймворк, який базується на шаблоні проектування MVC (Model-View-Controller) [24 ,с. 54].

Шаблон MVC передбачає поділ системи на три взаємопов'язані частини: Model (модель даних), View (інтерфейс користувача) та Controller (модуль керування) (рис. 3.2). Основна ідея шаблону полягає у відокремленні логіки застосунку від його представлення. Він застосовується для розділення даних (моделі) від інтерфейсу користувача (вигляду) таким чином, щоб зміни в інтерфейсі користувача мінімально впливали на роботу з даними, а зміни в моделі даних могли здійснюватися без необхідності змінювати інтерфейс користувача.

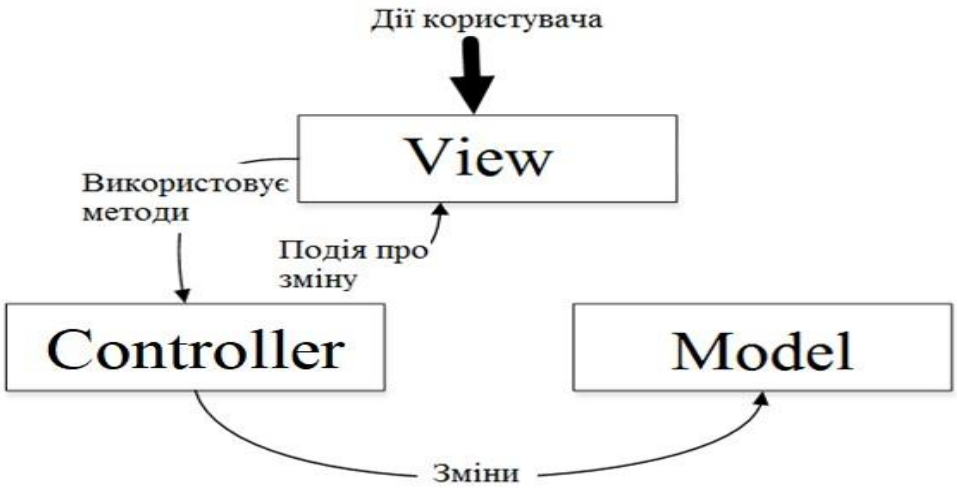


Рис. 3.11. Схема роботи MVC шаблону

Модель включає знання про предметну область, дані та правила доступу до даних, проте не володіє інформацією про контролери та представлення. У моделі реалізується бізнес-логіка застосунку, а контролер отримує дані від моделі, запитані користувачем або іншою системою.

Представлення забезпечує можливість візуалізувати дані, отримані від моделі, різними способами і може містити в собі логіку. Представлення є кінцевим інтерфейсом, з яким взаємодіє користувач, і користувач може передавати дані через нього [45, с. 76].

Загалом, ці характеристики та архітектурні підходи допомагають реалізувати можливість переміщення вперед і назад у часі в програмному забезпеченні.

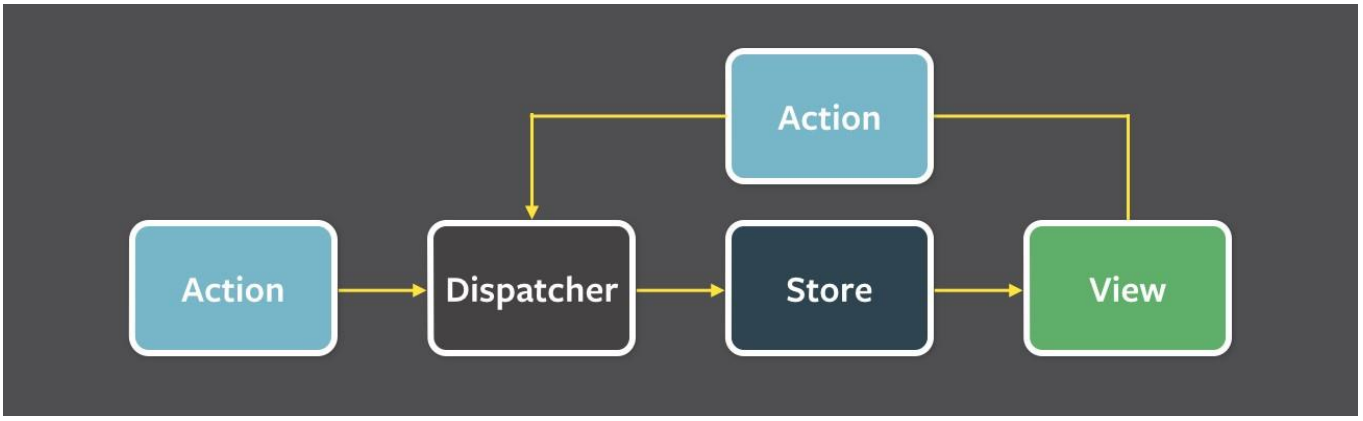


Рис. 3.12. Схема роботи архітектури Flux

Сховище включає в себе стан програми та логіку управління. Його роль схожа на модель у традиційній архітектурі MVC, але воно керує станом багатьох об'єктів, а не просто єдиним записом даних, як у моделях ORM, або колекціями, як у Backbone. Сховище не просто керує набором об'єктів, але відповідає за управління станом програми для конкретної області в межах програми.

Представлення відповідає за відображення інформації користувачу. У Flux-архітектурі, представлення є кінцевою точкою потоку даних і технічно не пов'язане зо внутрішнім уявленням системи. Інформаційна архітектура вимагає, щоб дані потрапляли в систему (тобто, назад до сховищ) лише за допомогою дій. Представлення також відповідає за введення даних користувачем та надає можливість викликати певні дії [54 ,с. 97].

Основна мета використання цієї концепції полягає в розділенні бізнес-логіки (моделі) від її представлення. Цей розподіл підвищує гнучкість системи та можливість повторного використання. Для великих проектів застосування даного шаблону дозволяє тестувати різноманітні компоненти програмної системи.

4 РОЗРОБЛЕННЯ ІНФОРМАЦІЙНОГО Й ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ РОЗУМНИЙ БУДИНОК

4.1. Обґрунтування вибору технічних засобів реалізації інформаційної системи розумний будинок

При виборі приладів контролю і регулювання слід враховувати наступні вимоги:

Прилади повинні забезпечувати необхідну точність вимірювання, бути достатньо чутливими та надійними у роботі.

Показники приладів повинні мати зрозумілу шкалу і покажчик. Самописні прилади мають регулювати свої покази у вигляді чіткої та видної кривої.

Місцеві прилади повинні бути розташовані так, щоб їхні показання було легко спостерігати.

Похибка не повинна перевищувати припустимі межі при зміні зовнішніх умов, температури та тиску навколишнього середовища.

Вимірювальні та регулюючі прилади повинні задовольняти вимоги щодо безпеки щодо вибуху та пожежі [43, с. 76].

При виборі приладів контролю і регулювання необхідно враховувати властивості об'єктів регулювання та регуляторів, щоб забезпечити стійкість систем регулювання і якісний процес регулювання без великих відхилень регульованої величини від заданого значення.

У даному технологічному процесі для контролю потрібні такі датчики:

- Датчики температури для вимірювання температури.

- Датчики вологості для вимірювання вологості у кімнатах.

- Спеціальні реле.

Завдяки цим характеристикам і було обрано цей давач який зображено на рисунку 4.1

Рис. 4.1 – Давач температури та вологості DHT22

Arduino Ethernet Shield 2 забезпечує підключення Arduino до мережі. Цей модуль легко підключається до Arduino-плати. Для з'єднання з мережею використовується RJ45-кабель. Як і у всіх компонентах Arduino, цей модуль має апаратну та програмну документацію, яка доступна з відкритим вихідним кодом. Це означає, що ви можете докладно дізнатися, як використовувати його і використовувати його дизайн як основу для своїх проектів [24 ,с. 86].

Рис. 4.2 - Контролер Ethernet: W5500

Для спеціальних реле ми обираємо реле від фірми SONGLE моделі SRD-05VDC, оскільки ці пристрої виявили високу надійність в подібних системах. Реле працює з напругою 5 В і може керувати пристроями, які вимагають високої напруги до 250 В та струму до 10 А.

На рисунку 4.3 показана схема реле, а на рисунку 4.4 - його зовнішній вигляд. Між контактами A1 і A2 розміщений металевий сердечник, який притягується рухомим якорем (2), коли через нього проходить струм. Контакти 1 і 3 є нерухомими. Важливо відзначити, що якор має пружини, і до тих пір, поки струм не пройде через сердечник, якор залишатиметься прижатим до контакту 3. При подачі струму, як вже було зазначено, сердечник перетворюється на електромагніт і притягується до контакту 1. При відключенні струму пружина знову повертає якір до контакту 3 [8, с. 61].

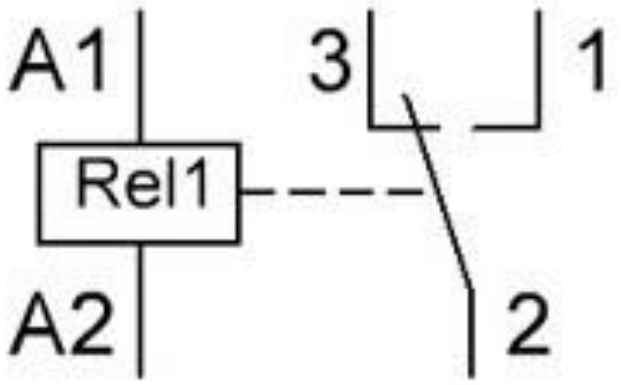


Рис. 4.3 – Схема реле

Існує велика кількість мікроконтролерів і платформ для "physical computing", таких як Parallax Basic Stamp, Netmedia's BX-24, Phidgets, MIT's Handyboard і багато інших, які надають подібні можливості. Всі ці пристрої збирають розрізнену програмну інформацію та упорядковують її у просту у використанні бібліотеку. Але Arduino має деякі переваги над іншими пристроями, особливо для викладачів, студентів та ентузіастів [41, с. 76].

Низька вартість: плати Arduino відносно дешеві порівняно з іншими платформами. Можна навіть зібрати модуль Arduino вручну, а деякі готові модулі коштують менше 5 доларів.

Завдяки цим перевагам і було обрано Arduino UNO, який зображено на рисунку 4.5

Рис. 4.5 – Плата Arduino UNO

4.2. Алгоритм оптимізації параметрів функціонування системи розумний будинок

Більшість методів розпізнавання зазвичай працюють зліва направо, обробляючи по одній фонемі за раз і розташовуючи їх в кінці префіксу слова. Ці префікси, відомі як гіпотези, представлені у вигляді об'єктних пар (o1 ... oj, [A1, ..., Aj]). Додавання дозволеної фонеми та нового закінчення до гіпотези називається її продовженням. У нашій роботі ми використовуємо два основних алгоритми пошуку: алгоритм багатостадійного декодування та метод пошуку в пучку Вітербі. Ці алгоритми можуть бути реалізовані шляхом використання стеків для зберігання даних в будь-якому зв'язковому місці між фонемами. Ці стеки, які не слід плутати зі стеками LIFO або FIFO, просто зберігають гіпотези на основі їх ймовірності [28, с. 75].

Припустимо, що функція векторної послідовності еталонних шаблонів виглядає наступним чином $X = \{x_1, x_2, ..., x_i\}$, та векторна послідовність – $Y = \{y_1, y_2, ..., y_j\}, i \neq j$.

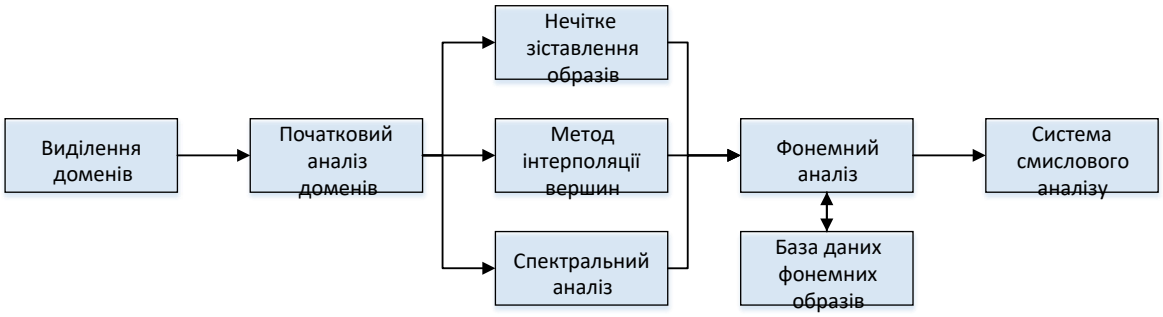


Рис. 4.6. Використання динамічних доменів в системі

Алгоритм DTW - знайти оптимальну функцію деформації часу, яка мінімізує загальну суму кумулятивних спотворень та нелінійно відображає часову шкалу невідомого еталонного шаблону до часової шкали і еталонного шаблону [7, с. 64].

Припустимо, що функція деформації часу є:

$$C = \{c(1), c(2), ..., c(N)\},$$

де N є довжина шляху, $c(N) = (i(N), j(N))$вказує на перші N точки збіру, які складаються з першого вектору $i(N)$ або вектору еталонного шаблону та першого $j(N)$вектору ознак випробуваного шаблону. Відстань між двома шаблонами (або значенням спотворення) $d(x_{i(N)}, y_{j(N)})$, називається частковою відповідністю відстані.

У формулі вибору вагової функції W_n слід розглянути дві передумови.

Спочатку, при розгляді кроку перед частковим шляхом перших N точок карати, локальний шлях розташовується під кутом 45 градусів, а потім адаптується відповідно до ситуації [20 ,с. 85].

Другий аспект полягає у розгляді різних частин мови, щоб надати різні ваги для посилення певних відмінних рис. У випадку вибору функції зважування, отриманої з третього стану, значення SMR нелінійно визначаються згідно з

результатами психоакустичної моделі, які відповідають функції перцептуальної ваги Wn, враховуючи критичні коефіцієнти вагової частоти Мела.

Отже зведемо данні в таблицю 4.1.

Таблиця 4.1.

Зведені дані експеременту

Метод	Отпимальні параметри	Правильність	Швидкість
Basic method		98.72%	100%
SMR для (MFCC)	i = 10, j = 17	99.45%	97.87%
SMR для (WFCC)	i = 8, j = 28	95.29%	78.29%
DTW для (MFCC)	i = 15, j = 34	93.22%	137.02%
DTW для (WFCC)	i = 11, j = 24	96.37%	155.57%

Як видно з таблиці, під час експерименту DTW з використанням функції WFCC мав кращу швидкість, близьку до половини швидкості одного з відомих виробників. Однак, у нього є недолік - менша точність розпізнавання.

4.3. Короткий опис програмної реалізації алгоритму

Цей підхід застосовується не лише у розробці програмного забезпечення, але й при створенні драйверів для різних пристроїв. Він добре підходить для використання в програмному середовищі Arduino IDE, оскільки початково мова програмування для пристроїв Arduino була заснована на C++. На сьогоднішній день цей підхід є найпоширенішим і зручним способом програмування мікроконтролерів.

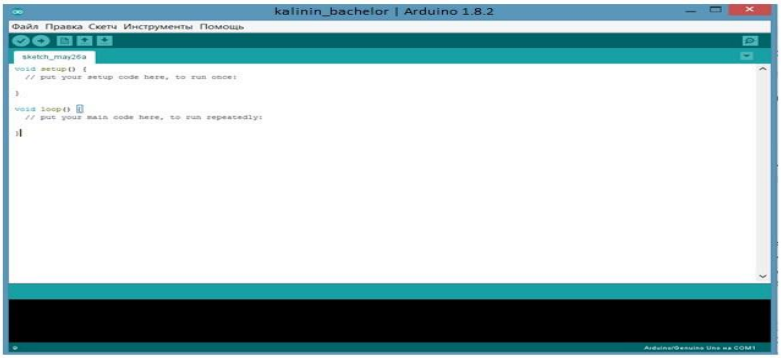


Рис. 4.7. Інтерфейс середовища програмування Arduino IDE

Центральний контролер збирає дані з локальних датчиків та, в залежності від отриманих результатів, автоматично виконує запрограмовані дії без потреби втручання користувача. Користувач може змінити режим роботи, прочитавши дані з RFID ключа за допомогою RFID зчитувача і змінюючи параметри роботи центрального процесора. Стан об'єкта періодично моніториться датчиками і передається до центрального контролера через визначені проміжки часу. Розділ коду, який описує введення вхідних даних, показаний на рисунку 4.8.

Рис. 4.8. Функціональна схема розумного будинку

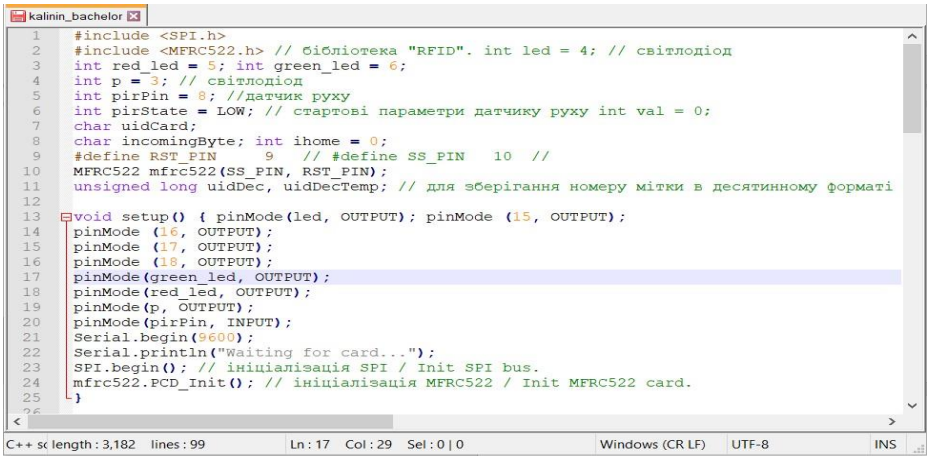


Рис. 4.9. Введення вхідних даних

На рис. 4.10. показано процес реалізації сигналізації для «розумного будинку».

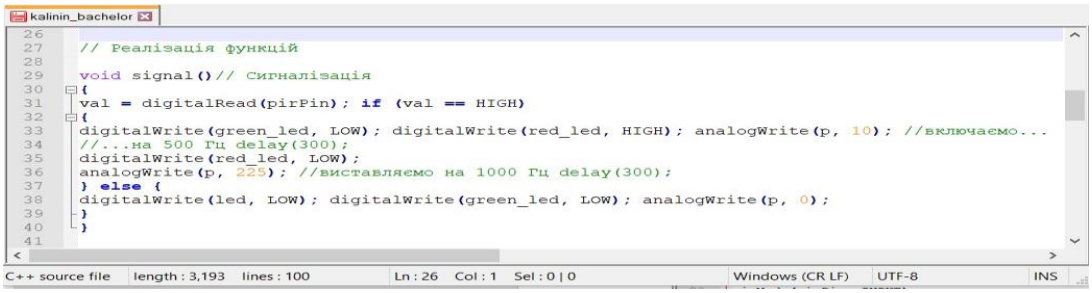


Рис. 4.10. Реалізація сигналізації

На рис. 4.11. показано керування освітленням «розумного будинку». Частина коду на рис. 4.12. забезпечує зрівняння міток (зареєстрованих ключів) за допомогою радіозв’язку.

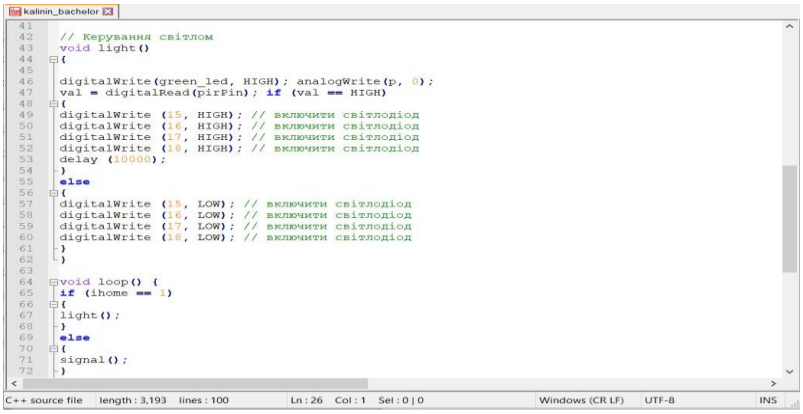


Рис. 4.11. Реалізація керування освітленням

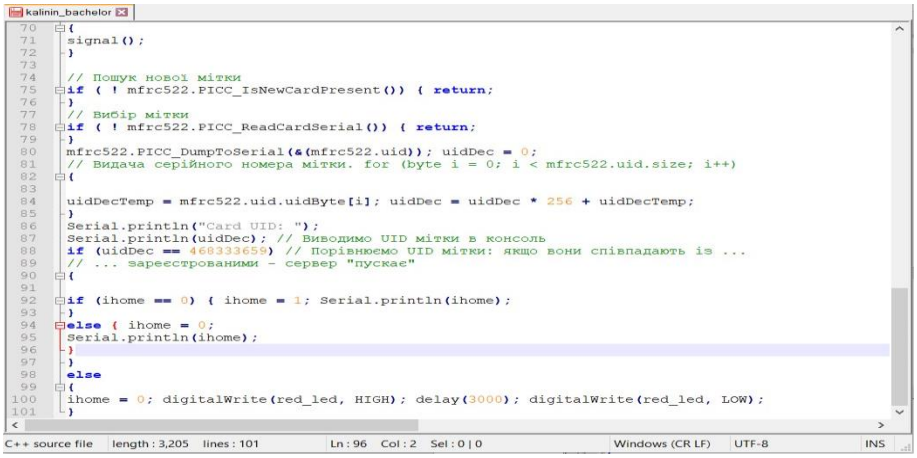


Рис. 4.12. Реалізація RFID-зчитувача

Далі у програмному коді було реалізовано інші необхідні функціональності для "розумного будинку". Наприклад, були додані блоки коду для обробки отриманих даних з датчиків, прийняття рішень та керування виконавчими пристроями. Програма також містила логіку для взаємодії з RFID зчитувачем, щоб користувач міг змінювати параметри роботи системи за допомогою RFID ключа [28 ,с. 65].

Використання мови програмування C# у середовищі Arduino IDE є досить поширеним та зручним підходом для розробки програмного забезпечення для мікроконтролерів, зокрема для пристроїв Arduino. Це дозволяє розробникам ефективно використовувати функціональні можливості мікроконтролера та легко взаємодіяти з його периферійними пристроями.

4.4. Розробка веб-сервісу для системи «Розумний дім»

Під час дослідження сучасних мов програмування у першому розділі, було вирішено обрати мову програмування C++ та фреймворк ASP.NET Core для розробки веб-сервісу. Вибір цих технологій зумовлений їхньою популярністю та ефективністю у створенні потужних та безпечних веб-додатків.

Одним із ключових аспектів безпеки веб-сервісу є система генерації токенів. Токен використовується для ідентифікації та автентифікації користувачів. Використання системи генерації токенів дозволяє забезпечити захист від несанкціонованого доступу та зловживань у веб-сервісі.

На рисунку 4.13. можна побачити ілюстрацію використання системи генерації токенів та протоколу HTTPS у веб-сервісі. Ці заходи безпеки гарантують надійну та захищену роботу сервісу, допомагають уникнути порушень безпеки та зловживань [24, с. 86].

Вибір мови програмування C++ та фреймворка ASP.NET Core для розробки веб-сервісу забезпечує не тільки потужність та широкі можливості, але й високий рівень безпеки. Використання системи генерації токенів та протоколу HTTPS дозволяє побудувати надійну та захищену інфраструктуру, що відповідає сучасним стандартам безпеки веб-додатків.

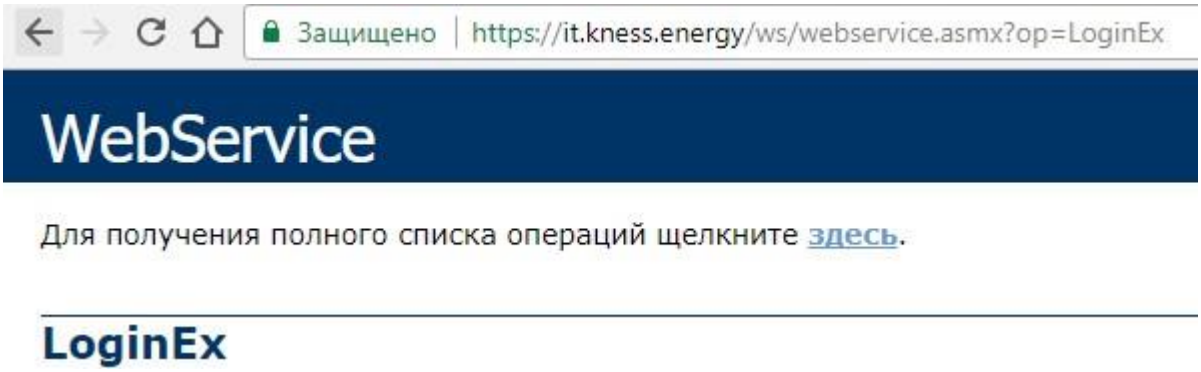
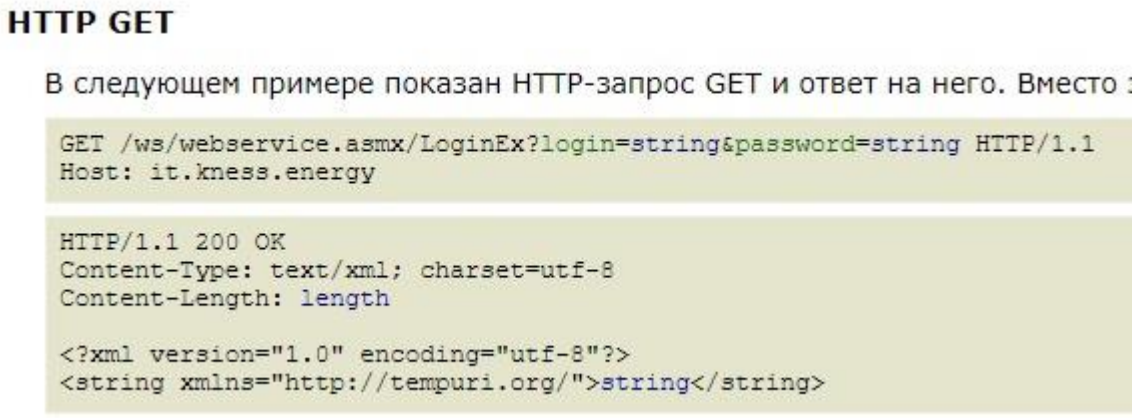


Рис. 4.13. HTTPS з’єднання з веб-сервісом

Такий підхід до безпеки веб-сервісу є особливо важливим в сучасному цифровому середовищі, де кіберзагрози стають все складнішими і розповсюдженішими. Застосування системи генерації токенів дозволяє забезпечити ідентифікацію користувачів та контроль доступу до ресурсів веб-сервісу. Кожен користувач отримує унікальний токен, який використовується для перевірки його прав доступу під час кожного запиту.

Для прикладу на рисунках 4.14.-4.15. відображено http-get та http-post запити та відповіді до веб-сервісу.



HTTP GET

В следующем примере показан HTTP-запрос GET и ответ на него. Вместо

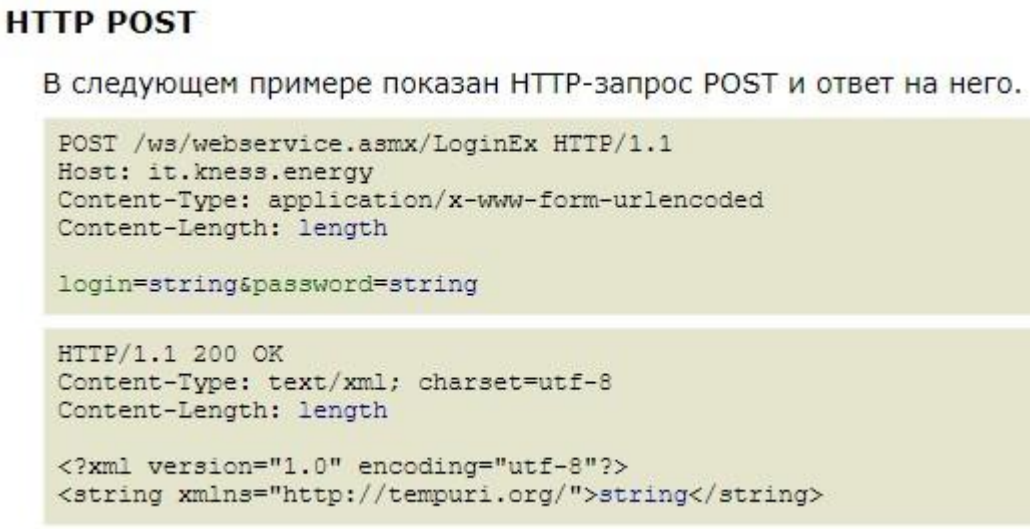
```
GET /ws/webservice.asmx/LoginEx?login=string&password=string HTTP/1.1
Host: it.kness.energy
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">string</string>
```

Рис. 4.14. HTTP-GET запит та відповідь

Застосування системи генерації токенів та протоколу HTTPS у веб-сервісі не тільки забезпечує безпеку, але й сприяє покращенню користувацького досвіду. Користувачі можуть бути впевнені, що їхні дані захищені, і вони можуть безпечно взаємодіяти з веб-сервісом, надаючи свої дані чи виконуючи різноманітні дії онлайн.



HTTP POST

В следующем примере показан HTTP-запрос POST и ответ на него.

```
POST /ws/webservice.asmx/LoginEx HTTP/1.1
Host: it.kness.energy
Content-Type: application/x-www-form-urlencoded
Content-Length: length

login=string&password=string
```

```
HTTP/1.1 200 OK
Content-Type: text/xml; charset=utf-8
Content-Length: length

<?xml version="1.0" encoding="utf-8"?>
<string xmlns="http://tempuri.org/">string</string>
```

Рис. 4.15. HTTP-POST запит та відповідь

Давайте розглянемо кілька прикладних сценаріїв взаємодії пристроїв у розумному будинку. Завдяки зручному інтерфейсу, користувач може легко

переглянути список всіх підключених пристроїв Інтернету речей. Цю можливість надає вкладка "IoT Monitor -> Home".

На даній вкладці користувач може отримати повний перелік пристроїв, які входять до системи розумного будинку. Це можуть бути різноманітні пристрої, такі як освітлення, системи безпеки, опалення, кондиціонування повітря, розетки та багато інших. Кожен пристрій має свою унікальну ідентифікацію та статус підключення [47, с. 55].

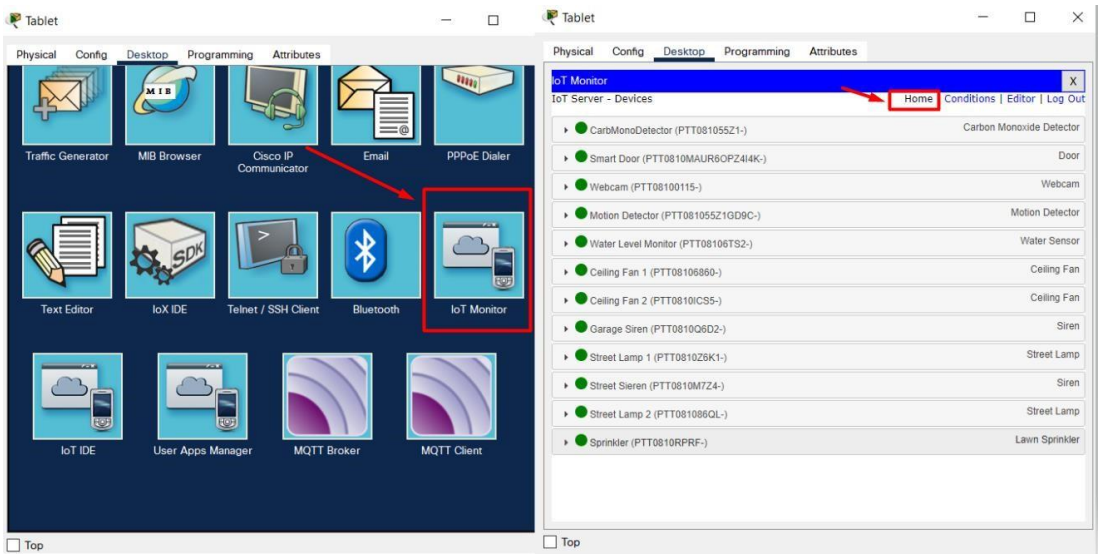
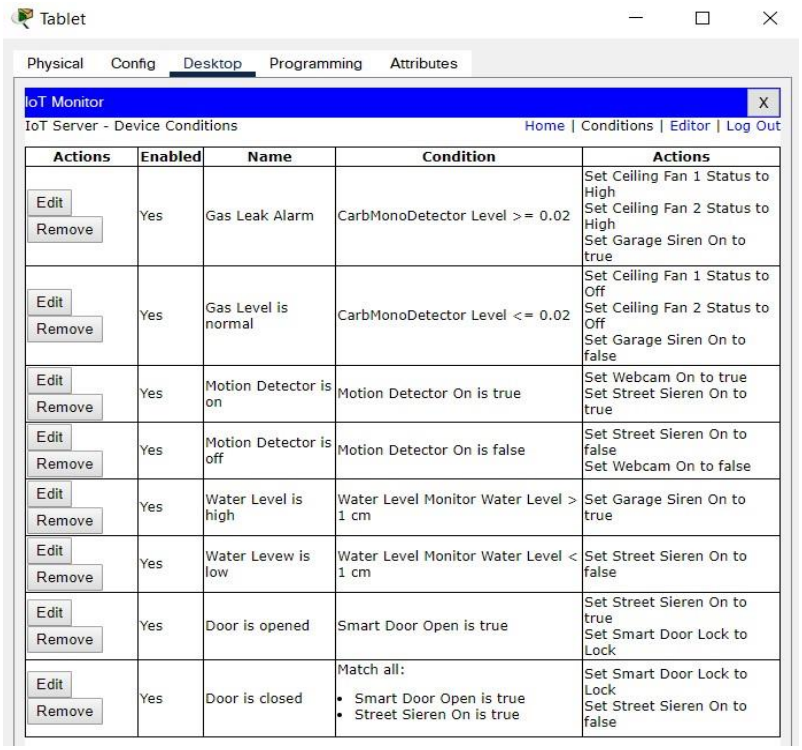


Рис. 4.16. Стартове вікно демонстрації всіх підключених до мережі розумних пристроїв

На вкладці "Conditions" користувач може легко створювати, змінювати та видаляти сценарії взаємодії пристроїв у розумному будинку. На рисунку 4.17. показані деякі приклади умов, за яких активуються датчики в розумному будинку.

Наприклад, якщо датчик виявляє рівень газу, що перевищує значення 0.02, то спрацьовують два вентилятори, а також активується звукове сигнальне сповіщення сиреною. У такий спосіб, система розумного будинку реагує на потенційну небезпеку та забезпечує вчасну реакцію.

Це дозволяє впевнено керувати пристроями та створювати персоналізовані сценарії відповідно до потреб та вимог користувача [19 ,с. 52].



Actions	Enabled	Name	Condition	Actions
Edit Remove	Yes	Gas Leak Alarm	CarbMonoDetector Level >= 0.02	Set Ceiling Fan 1 Status to High Set Ceiling Fan 2 Status to High Set Garage Siren On to true
Edit Remove	Yes	Gas Level is normal	CarbMonoDetector Level <= 0.02	Set Ceiling Fan 1 Status to Off Set Ceiling Fan 2 Status to Off Set Garage Siren On to false
Edit Remove	Yes	Motion Detector is on	Motion Detector On is true	Set Webcam On to true Set Street Sieren On to true
Edit Remove	Yes	Motion Detector is off	Motion Detector On is false	Set Street Sieren On to false Set Webcam On to false
Edit Remove	Yes	Water Level is high	Water Level Monitor Water Level > 1 cm	Set Garage Siren On to true
Edit Remove	Yes	Water Levew is low	Water Level Monitor Water Level < 1 cm	Set Street Sieren On to false
Edit Remove	Yes	Door is opened	Smart Door Open is true	Set Street Sieren On to true Set Smart Door Lock to Lock
Edit Remove	Yes	Door is closed	Match all: • Smart Door Open is true • Street Sieren On is true	Set Smart Door Lock to Lock Set Street Sieren On to false

Рис. 4.17. Налаштування правил автовмикання вентиляторів та звукового сигналу при спрацьовуванні датчика загазованості

Перевіримо, чи правильно працює інтелектуальна мережа відповідно до встановлених вище правил [24, с. 85].

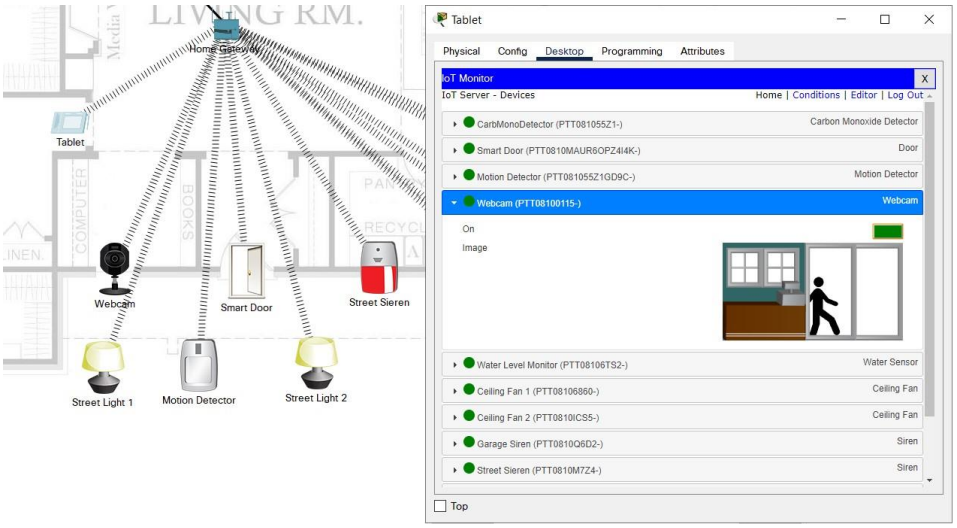


Рис. 4.18. Спрацювання датчика відкриття дверей

У наступному сценарії, який зображений на рисунку 4.19., ми розглянемо спрацювання датчика протікання води. При виявленні вологості датчиком, автоматично активується звуковий сигнал. Крім того, користувач може

контролювати рівень витоку води за допомогою екрану свого смартфона або планшету [26, с. 43].

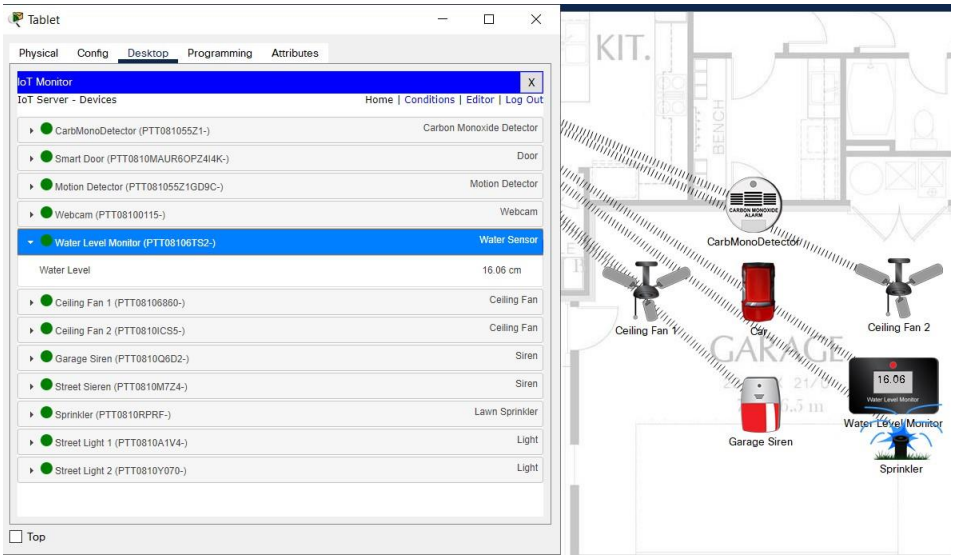


Рис. 4.19. Спрацювання датчика затопленності

У третьому сценарії, який ілюструє спрацювання датчика загазованості, ми спостерігаємо наступні дії. При досягненні рівня газу, виміряного датчиком, значення більше або рівне 0.02, активуються два вентилятори для видалення газу, а також звукова сигналізація через сирену. При зниженні рівня загазованості до значення менше або рівне 0.02, вентилятори та сирена автоматично вимикаються.

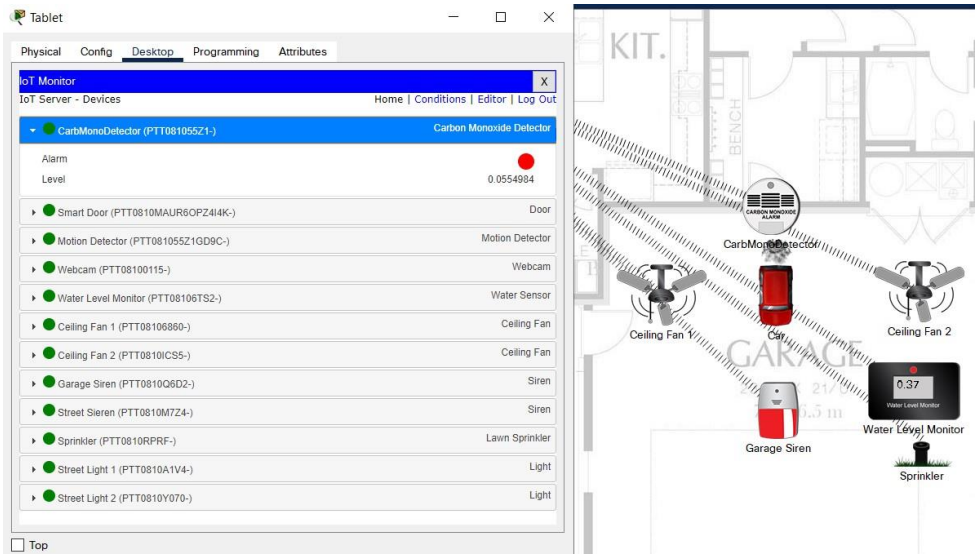


Рис. 4.20. Спрацювання датчика загазованості

Сценарій спрацювання датчика руху зображено на рисунку 4.21. Спрацювання супроводжується звуковим сигналом, а також ввімкненням веб-камери.

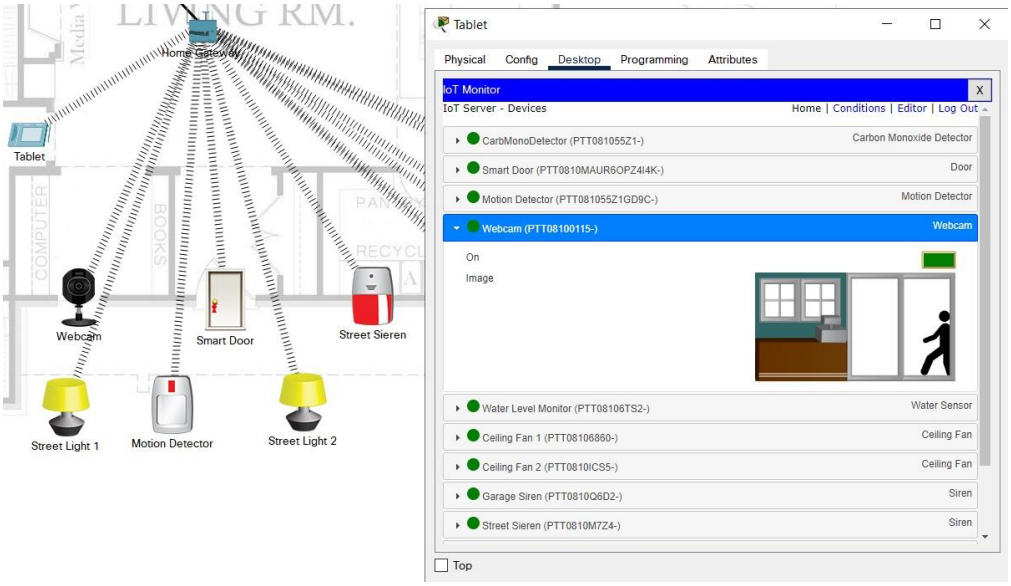


Рис. 4.21. Спрацювання датчика руху

У випадку спрацювання датчика відкриття дверей, вуличне освітлення не вмикається, відмінно від попереднього сценарію. Це означає, що у розумному будинку можуть бути різні конфігурації та умови взаємодії пристроїв, залежно від вимог та вподобань користувача [37, с. 85].

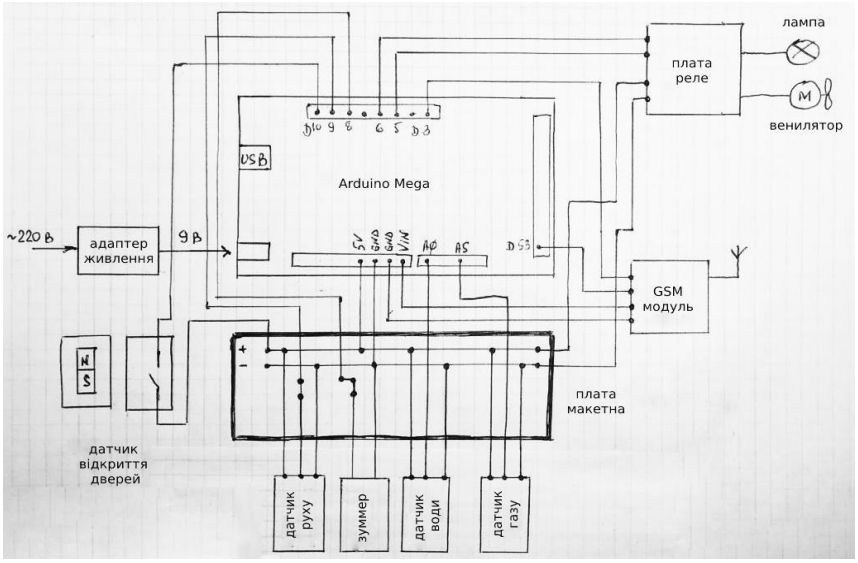


Рис. 4.22. Схема підключення датчиків до плати Arduino Mega 2560

– D4 – вихід на реле освітлення 1;

− D5 – вихід на реле освітлення 2;

− D6 – вихід на реле вентилятора;

− D7 – вихід на реле освітлення 3;

− D8 – вихід на зумер;

− D9 – цифровий вхід від датчика руху;

− D10 – цифровий вхід від датчика відкриття дверей;

− A0 – аналоговий вхід від датчика протікання води;

− A5 – аналоговий вхід від датчика загазованості;

− D53 – шина RX, дані GSM модуля; − D3 – шина TX, дані GSM модуля.

Робота системи базується на простій логіці: якщо будь-який датчик спрацьовує, то звучить звуковий сигнал тривоги через зумер. Крім того, система відправляє текстове повідомлення на запрограмований номер мобільного телефону власника за допомогою модуля GSM, що вказує на спрацьований датчик. Крім цього, при спрацюванні датчика загазованості вмикається вентиляція, а при спрацюванні датчика руху – освітлення.

4.5. Тестування системи «Розумний дім»

Почнемо з тестування клієнта разом з покращеною бібліотекою для розпізнавання голосових команд. Для зручності тестування ми будемо використовувати локальний сервер (localhost) і настроєний порт 56247. На рисунку 4.23. видно, що фраза була повністю розпізнана. Крім того, в консолі відображалося, як алгоритм наближався до правильного рішення.

Під час тестування ми звернули увагу на точність розпізнавання фрази, а також на прогрес алгоритму в процесі аналізу голосових команд. Завдяки покращеній бібліотеці, система була здатна ефективно розпізнавати вимовлені фрази і повертати точні результати.

Це демонструє, що наша система готова до реалізації голосового керування, і може успішно сприймати та інтерпретувати команди користувача.

Тестування на локальному сервері дозволило нам перевірити функціональність системи перед її розгортанням на більш широкому масштабі [27 ,с. 32].

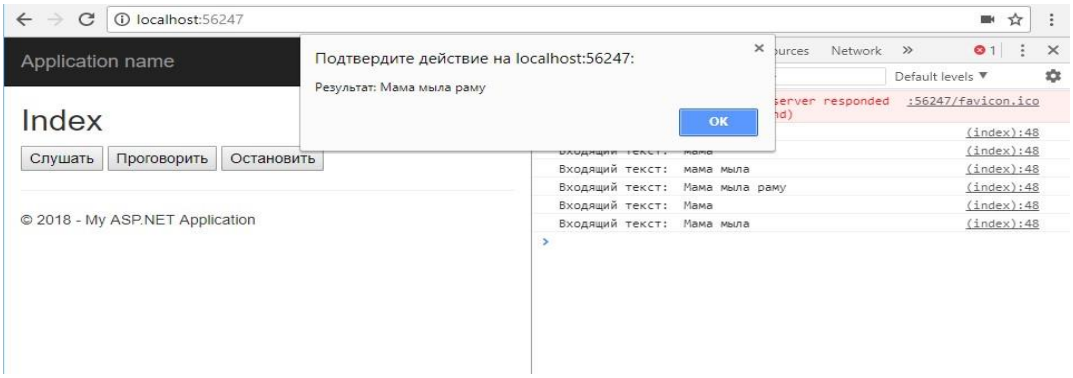


Рис. 4.23. Результати виводу клієнта

Наступним етапом буде перевірка веб-сервісу. Для цього ми відкриємо початкову сторінку веб-сервісу і з випадаючого меню функцій оберемо опцію авторизації (рис. 4.24.). Після цього введемо свій логін та пароль в відповідні поля і натиснемо кнопку "Запуск" для продовження.

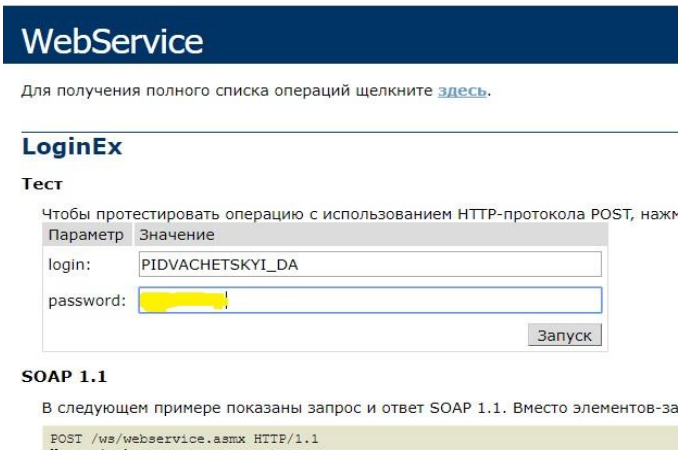


Рис. 4.24. Введення логіна та пароля для авторизації

У результаті, якщо логін та пароль є правильними, отримуємо назву користувача, та Token, саме це зображено на рисунку 4.25.

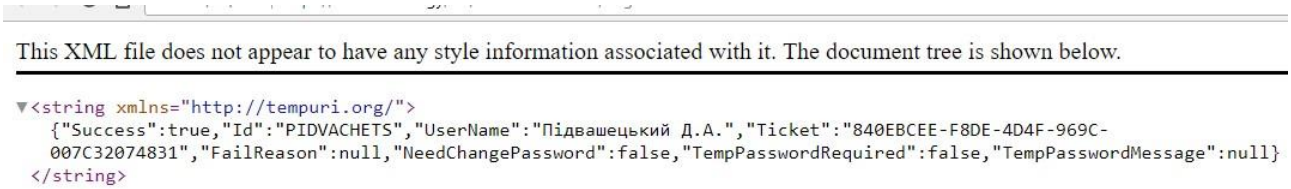


Рис. 4.25. Вдала авторизація

Останнім етапом буде перевірка роботи мікроконтролера Arduino. Ми проведемо тестування датчика температури та вологості, контроль за відхиленням значень та спрацювання спеціального реле. У разі коли температура у приміщенні перевищує 27 градусів за Цельсієм, система охолодження буде ввімкнена. У випадку коли температура знаходиться нижче або дорівнює 27 градусам, система охолодження залишиться вимкненою [30, с. 62].

Таблиця 4.2.

Тестовий сценарій «Автентифікація у системі»

Назва	Тест для перевірки RFID автентифікації у системі	
Use case	Автентифікація користувача у системі.	
Дія	Очікуваний результат.	Результат тесту
Передумова		
Увімкнути живлення Arduino Mega 2560	Діод, автентифікації RFID увімкнутий	Пройдений
Кроки тесту		
Підносим мітку до рідера з ключем 16909060	Сигналізація не спрацювала,	Пройдений
Відчиняємо двері	Виведено надпис ("DOOR IS OPEN!");	Пройдений

Таблиця 4.3.

Тестовий сценарій «Спроба зчитування хибної мітки користувача у системі»

Назва	Тест для перевірки автентифікації у системі	
Use case	Спроба автентифікації користувача з хибною міткою.	
Дія	Очікуваний результат	Результат тесту
Передумова		
Живлення увімкнуте	Діод, автентифікації RFID увімкнутий	Пройдений
Підносим мітку до рідера з ключем 18009060	Виводиться напис «ALARM! UNKNOWN CARD».	Пройдений
«ALARM! UNKNOWN CARD».	Спрацювала сигналізація	Пройдений
Спробуєм відкрити замок	Замок зачинено	Пройдений

Таблиця 4.4.

Назва	Тест для перевірки автентифікації у системі.	
Use case	Відімкненя дверей без мітки RFID.	
Дія	Очікуваний результат	Результат тесту
Передумова		
Живлення увімкнуте	Діод, автентифікації RFID увімкнутий	Пройдений
Відчиняємо двері ключем	("DOOR IS OPEN!")	Пройдений
Заходим у будинок	Спрацювала сигналізація	Пройдений
Потрапляння води на датчик	«ALARM! WATER».	Пройдений
«ALARM! WATER».	«WATER VALVE IS WORKING».	Пройдений

Тестовий сценарій «Відімкненя дверей без мітки RFID »

Таблиця 4.5.

Тестовий сценарій «Витоку газу»

Назва	Тест для перевірки витоку газу	
Use case	Моделювання ситуації витоку газу	
Дія	Очікуваний результат	Результат тесту
Передумова		
Перевищена концентрація газу у повітрі	«ALARM! GAS».	Пройдений
«ALARM! GAS».	«FAN IS WORKING».	Пройдений
«FAN IS WORKING».	Спрацювала сигналізація	Пройдений
Повторна перевіка витоку газу	Сигналізація вимкнулась, клапан закритий	Пройдений

Таблиця 4.6.

Тестовий сценарій «Взяття квартири під охорону»

Назва	Тест взяття квартири під охорону	
Use case	Моделювання ситуаці взяття квартири під охорону	
Дія	Очікуваний результат	Результат тесту
Передумова		
За допомогою телефону підєднуємось до bluetooth модуля вводим пароль (ErhfYf6tM5)	Аутентифікацію пройдено	Пройдений
Заходим у додаток bluetooth terminal hc-05 І вводимо змінну 6	("SECURITY MODE")	Пройдений
За допомогою телефону підєднуємось до bluetooth модуля вводим пароль (ErhfYf6tM5)	Аутентифікацію пройдено	Пройдений
Заходим у додаток bluetooth terminal hc-05 І вводимо змінну 7	Квартиру знято з охорони	Пройдений

Таблиця 4.7.

Тестовий сценарій «Займання»

Назва	Тест для перевірки датчика вогню	
Use case	Моделювання ситуації займання	
Дія	Очікуваний результат	Результат тесту
Передумова		
Підносим вогонь до датчика	"ALARM! FIRE"	Пройдений
"ALARM! FIRE"	Спрацювала сигналізвція	Пройдений

Ми вибрали фреймворк NUnit для написання модульних тестів. Для позначення класу, який використовується для юніт-тестів, ми використовуємо атрибут TestFixtureAttribute. Кожен метод, що представляє собою окремий тест, позначається атрибутом TestAttribute. Також є можливість використовувати метод, який встановлює базові налаштування середовища перед виконанням кожного тесту. Цей метод позначається атрибутом SetupAttribute і зазвичай має назву "Setup". На рисунку 4.26. показаний результат виконання тестів для класу RequirmentController [29, с. 65].

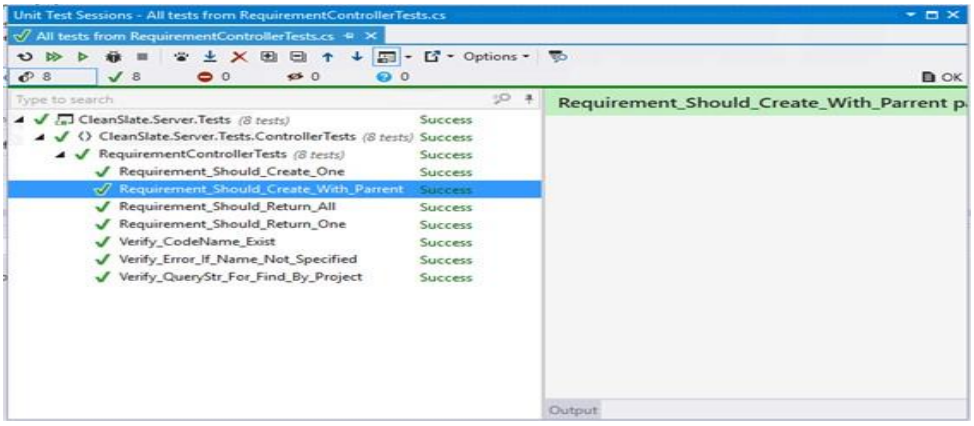


Рис. 4.26. Результат виконання модульних тестів

Результати виконання тестів для класу RequirmentController, як показано на рисунку 4.26., свідчать про успішне протестування основних функціональностей контролера. Використання фреймворку NUnit дозволяє нам зручно та ефективно виконувати модульні тести, що допомагає перевірити правильність роботи окремих компонентів системи.

Атрибути TestFixtureAttribute, TestAttribute і SetupAttribute забезпечують ясну позначку та структуру тестів, допомагаючи легко розуміти їх при огляді коду. Метод Setup використовується для попередньої підготовки середовища перед виконанням кожного тесту, що сприяє стабільності та надійності тестової суіти.

ВИСНОВКИ

У представленій бакалаврській роботі було проведено дослідження та розробку інформаційної системи для розумного будинку з використанням сучасних веб-технологій. Загальна структура роботи включала декілька основних етапів, які дозволили досягти поставлених цілей.

На початку роботи було проведено аналіз сучасного стану проблеми, що включало огляд наявних розумних будинків і їх функціональності. Цей аналіз дозволив з'ясувати переваги та недоліки існуючих рішень, а також визначити потенційні можливості для покращення.

Далі було проведено проектування інформаційної системи розумного будинку. Цей етап включав визначення вимог до системи, вибір необхідних компонентів та технологій, а також створення архітектури системи. Проектування враховувало не лише функціональні аспекти, але й аспекти безпеки та ефективності.

Після проектування розпочалась розробка імплементації інформаційної системи. Було розроблено програмне забезпечення, що забезпечує функціонування розумного будинку. Це включало розробку інтерфейсу для взаємодії з системою, збір і обробку даних з датчиків, управління різними пристроями та автоматизацію різних процесів в будинку.

Останнім етапом було тестування та оцінка розробленої інформаційної системи. Це включало виконання різних тестів для перевірки функціональності, безпеки та стабільності системи. Отримані результати тестування допомогли виявити можливі проблеми та вдосконалити систему.

Висновком до бакалаврської роботи є те, що було успішно розроблено інформаційно-програмне забезпечення для функціонування розумного будинку з використанням сучасних веб-технологій. Результати дослідження показали, що така система має значний потенціал для покращення комфорту, безпеки та

енергоефективності в розумних будинках. Проте, існують питання, які потребують додаткового дослідження, такі як інтеграція з іншими системами та адаптація до різних сценаріїв використання.

Розроблена інформаційна система розумного будинку з використанням сучасних веб-технологій демонструє потужний потенціал у сфері автоматизації та контролю різних аспектів побутового середовища. Вона забезпечує зручну та ефективну взаємодію між мешканцями та різними пристроями в будинку, що призводить до підвищення рівня комфорту та забезпечення більшого контролю над енергоспоживанням.

Застосування сучасних веб-технологій, таких як хмарні обчислення, Інтернет речей (IoT) та веб-додатки, дозволяє забезпечити доступ до системи з будь-якого пристрою, що має підключення до Інтернету. Це дає змогу мешканцям контролювати різні аспекти будинку, такі як освітлення, опалення, кондиціонування повітря, системи безпеки та інші, навіть коли вони знаходяться далеко від будинку. Крім того, система здатна збирати та аналізувати дані з різних датчиків і пристроїв, що дозволяє забезпечити інформаційну зворотну зв'язок та приймати розумні рішення для оптимізації роботи будинку.

Проте, додаткові дослідження та розвиток системи можуть бути спрямовані на інтеграцію з іншими домашніми системами та платформами, такими як голосові помічники або інші розумні пристрої. Це дозволить забезпечити ще більшу автоматизацію та зручність для мешканців, а також забезпечити більш широкий функціонал системи. Крім того, важливим аспектом є адаптація системи до різних сценаріїв використання та потреб користувачів, забезпечуючи гнучкість та налаштування системи під конкретні умови та вимоги.

У підсумку, бакалаврська робота показує, що інформаційна система розумного будинку з використанням сучасних веб-технологій має значний потенціал для покращення життя мешканців, забезпечення комфорту, безпеки та енергоефективності. Проте, ще багато можливостей залишаються для

подальшого дослідження, розвитку та вдосконалення системи з метою надання ще більшого спектру функцій та покращення її інтеграції з іншими технологіями та платформами.

Завершуючи думку, можна зазначити, що бакалаврська робота довела ефективність та потенціал інформаційної системи розумного будинку з використанням сучасних веб-технологій. Ця система сприяє автоматизації та оптимізації різних аспектів будинку, забезпечуючи комфорт, безпеку та енергоефективність. Проте, для подальшого розвитку і вдосконалення системи рекомендується провести додаткові дослідження та інтеграцію з іншими технологіями, щоб розширити її функціональні можливості та забезпечити ще більшу зручність та гнучкість для користувачів.

Результати дослідження та розробки вказують на потенціал такої системи для покращення комфорту, безпеки та енергоефективності житла. Інтеграція хмарних обчислень, Інтернету речей та веб-додатків дозволяє забезпечити зручний доступ до системи з будь-якого пристрою та забезпечити контроль над різними аспектами будинку. Проте, для подальшого розвитку системи рекомендується провести додаткові дослідження, зосередившись на інтеграції з іншими домашніми системами та розумними пристроями, а також на адаптації до різних сценаріїв використання. Подальші зусилля слід спрямувати на розширення функціональності, підвищення зручності та гнучкості системи, а також на покращення її інтеграції з іншими технологіями та платформами.

Загалом, бакалаврська робота підтверджує, що інформаційна система розумного будинку з використанням сучасних веб-технологій має великий потенціал для покращення якості життя мешканців, забезпечення ефективності та зручності. Дальший розвиток цієї області досліджень відкриває шлях до створення інтелектуальних розумних будинків, що сприятимуть сталому розвитку та забезпеченню збалансованого та комфортного життя.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Абдуліна І. Роблячи ставки на розробку програмного забезпечення [Текст] /І. Абдуліна, В. Березанська// Інтелетуальна власність.–2012.–№9.–С8-12.
2. Алгоритми і структура даних: Навчальний посібник / В.М.Ткачук. - ІваноФранківськ: Видавництво Прикарпатського національного університету імені Василя Стефаника, 2016. 286 с.
3. Алгоритми та структури даних. Навчальний посібник / Т. О. Коротєєва. Львів: Видавництво Львівської політехніки, 2014. - 280 с.
4. Алексенко О. В.. Технології програмування та створення програмних продуктів: конспект лекцій. – Суми : Сумський державний університет, 2013. – 133 с.
5. Бадд Т. Об'єктно-орієнтоване програмування в дії / Перекл. з англ. - Спб .: Питер, 1997. - 464 с.
6. Баркмейер Едвард Дж. (2003). Концепції автоматизації системної інтеграції NIST 2003. – 435 с.
7. Бичков О.С. Основи сучасного програмування [Текст] : підручник / О. С. Бичков ; Київ. нац. ун-т ім. Т. Шевченка. – К. : Київ. ун-т, 2008. – 272 с.
8. Васюков І.В., Міночкін А.І., Шаціло П.В. Розробка алгоритмів задач управління зв’язком. ВІТІ. К.2004р. – 430 с.
9. Глоба Л. С. Розробка інформаційних ресурсів та систем [Електронний ресурс]: конспект лекцій / Л. С. Глоба, Т. М. Кот. - Київ : НТУУ "КПІ", 2014. - 318 с.
10. Грязнова В. О., Єфіменко С. В. Основи методології програмування. - К.: ВПЦ "Київський університет", 2010.
11. Жерновий Ю. В. Імітаційне моделювання систем масового обслуговування: Практикум. / Ю. В. Жерновий. – Львів: Видавничий центр ЛНУ імені Івана Франка, 2007. – 307 с.

12. Забуранна Л.В. Опимізаційні методи і моделі / Л.В. Забуранна, Н.В. Попрозман, Н.А. Клименко, О.І. Попрозман, С.В. Забуранний. – Київ, 2014. – 372 с.
13. Зовнішній датчик руху «Розумного» будинку: [Електронний ресурс]: // sutem – Режим доступу до ресурсу: http://sutem.com.ua/7135smartbus.php
14. Імітаційне моделювання систем масового обслуговування: навч. посіб. [для студентів техн. спец. вищ. навч. закл.] / В. Б. Толубко, А.Д. Кожухівський, В.В. Вишнівський, Г.І. Гайдур, О.А. Кожухівська. – Київ: 2018 р. - 175 с.
15. Інженерія якості програмного забезпечення: навч. посібник / Г.В Табунщик, Р.К. Кудерметов, Т.І. Брагіна. - Запоріжжя: ЗНТУ, 2013. - 180 с.
16. Інтелектуальні системи освітлення: [Електронний ресурс]: // dss-bi – Режим доступу до ресурсу: http://dss-bi.com.ua/sitelab1/
17. Історія розумний дім: [Електронний ресурс]: // wiki – Режим доступу до ресурсу:http://wiki.tntu.edu.ua/%D0%A0%D0%BE%D0%B7%D1%83%D0%BC%D0%BD %D0%B8%D0%B9_%D0%B4%D1%96%D0%BC
18. Караванова Т.П. Інформатика: методи побудови алгоритмів та їх аналіз. Обчислювальні алгоритми: Навч. посіб. для 9-10 кл. із поглибл. вивч. інформатики – К.: Генеза. – 2008.- 333 с.
19. Керований пристрої «Розумного» будинку: [Електронний ресурс]: // sutem – Режим доступу до ресурсу: http://sutem.com.ua/7113smartbus.php
20. Керуючі пристрої «Розумного» будинку: [Електронний ресурс]: // sutem – Режим доступу до ресурсу: http://sutem.com.ua/7121smartbus.php
21. Кислий, В.М. Методологія та організація наукових досліджень: конспект лекцій для студ. спец. 8.050201 "Менеджмент організацій" усіх форм навчання / В.М. Кислий. – Суми: СумДУ. 2009. – 111 с.
22. Ковалюк Т.В. Алгоритмізація та програмування: Підручник. – Львів: «Магнолія 2006», 2013. – 400 с.

23. Котунова., Д. Г. Огляд DIY елементів для систем «Smart Home» / Д. Г. Котунова, О. М. Павловський // XIII Науково-практична конференція студентів, аспірантів та молодих вчених «Погляд у майбутнє приладобудування», 13-14 травня 2020 р., м. Київ, Україна : збірник праць конференції. – Київ : КПІ ім. Ігоря Сікорського, 2020. – С. 35–38.
24. Кравець П. Об'єктно-орієнтоване програмування: навч. посібник / П.О. Кравець. – Львів: Видавництво Львівської політехніки, 2012. – 624 с.
25. Кудряченко. А.А. Верифікація / Політична енциклопедія. Редкол.: Ю. А. Левенець (голова), Ю. Б. Шаповал (заст. голови) та ін. – Київ: Парламентське видавництво, 2011. – 91 с.
26. Лавріщева К. М. Програмна інженерія / К. М. Лавріщева. – К. : Академперіодика, 2008. – 319 с
27. Леонов И.В. Введення в методологію розробки програмного забезпечення за допомогою Rational Rose // Ескейп, 2004. – 301 с.
28. Литвинов А. Л. Розробка та дослідження ймовірнісних моделей оцінювання якості інформаційно-управляючих систем / А. Л. Литвинов // Комунальне господарство міст. : Серія: Технічні науки та архітектура. – Харків : ХНУМГ, 2016. – вип. 126, – С. 22 – 26.
29. Литвинов А. Л. Теорія систем масового обслуговування : навч. посібник / А. Л. Литвинов ; Харків. нац. ун-т міськ. госп-ва ім. О. М. Бекетова. – Харків : ХНУМГ ім. О. М. Бекетова, 2018. – 141 с.
30. Логічні пристрої «Розумного» будинку: [Електронний ресурс]: // sutem – Режим доступу до ресурсу: http://sutem.com.ua/7smartbus.php
31. Міночкін А.І., Скрипник Л.В., Шаціло П.В. Спеціальні інформаційні системи і бази даних ВІТІ К. 2004р. – 523 с.
32. Николайчук Я. М. Проектування спеціалізованих комп'ютерних систем : навч. посібник / Я. М. Николайчук, Н. Я. Возна, І. Р. Пітух. - Тернопіль : ТзОВ "Терно-граф", 2010. - 392 с.

ДОДАТОК

Лістінг програмного забезпечення

```
#include <iostream>
#include <string>
#include <vector>
#include <Windows.h>
#include <Serial.h>
#include <SerialDataReceivedEventArgs.h>
#include <SerialDataReceivedEventHandler.h>

using namespace System;
using namespace System::Collections::Generic;
using namespace System::ComponentModel;
using namespace System::Data;
using namespace System::Drawing;
using namespace System::IO;
using namespace System::IO::Ports;
using namespace System::Linq;
using namespace System::Text;
using namespace System::Threading::Tasks;
using namespace System::Windows::Forms;

namespace Arduino
{
    public ref class Form1 : public Form
    {
    public:
        Form1()
        {
            InitializeComponent();

            notifyIcon1->Visible = false;
            notifyIcon1->MouseClick += gcnew MouseEventHandler(this,
&Form1::notifyIcon1_MouseClick);
            this->Resize += gcnew System::EventHandler(this,
&Form1::Form1_Resize);

            // Відкриваємо порт і задаємо швидкість 9600 бод
            String^ portName = "COM1"; // Замініть "COM1" на відповідний порт
            int baudRate = 9600;

            SerialPort^ serialPort1 = gcnew SerialPort(portName, baudRate);
            serialPort1->DataReceived += gcnew
SerialDataReceivedEventHandler(this, &Form1::serialPort1_DataReceived);
            serialPort1->Open();
        }

    private:
        void serialPort1_DataReceived(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ vlag = serialPort1->ReadLine();
```

```
            this->BeginInvoke(gcnew LineReceivedEvent(this,
&Form1::LineReceived), vlag);
        }

        void serialPort1_DataReceived1(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ temp = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent1(this,
&Form1::LineReceived1), temp);
        }

        void serialPort1_DataReceived2(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            // Ваш код обробки події отримання даних
        }

        // Додайте інші методи та події за необхідністю
    };
}
     #include <iostream>
#include <string>
#include <vector>
#include <Windows.h>
#include <Serial.h>
#include <SerialDataReceivedEventArgs.h>
#include <SerialDataReceivedEventHandler.h>

using namespace System;
using namespace System::Collections::Generic;
using namespace System::ComponentModel;
using namespace System::Data;
using namespace System::Drawing;
using namespace System::IO;
using namespace System::IO::Ports;
using namespace System::Linq;
using namespace System::Text;
using namespace System::Threading::Tasks;
using namespace System::Windows::Forms;

namespace Arduino
{
    public ref class Form1 : public Form
    {
    public:
        Form1()
        {
            InitializeComponent();

            notifyIcon1->Visible = false;
            notifyIcon1->MouseClick += gcnew MouseEventHandler(this,
&Form1::notifyIcon1_MouseClick);
            this->Resize += gcnew System::EventHandler(this,
&Form1::Form1_Resize);

            // Відкриваємо порт і задаємо швидкість 9600 бод
            String^ portName = "COM1"; // Замініть "COM1" на відповідний порт
            int baudRate = 9600;
```

```
            SerialPort^ serialPort1 = gcnew SerialPort(portName, baudRate);
            serialPort1->DataReceived += gcnew
SerialDataReceivedEventHandler(this, &Form1::serialPort1_DataReceived);
            serialPort1->Open();
        }

    private:
        void serialPort1_DataReceived(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ vlag = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent(this,
&Form1::LineReceived), vlag);
        }

        void serialPort1_DataReceived1(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ temp = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent1(this,
&Form1::LineReceived1), temp);
        }

        void serialPort1_DataReceived2(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ svet = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent2(this,
&Form1::LineReceived2), svet);
        }

        void serialPort1_DataReceived3(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ mowe = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent3(this,
&Form1::LineReceived3), mowe);
        }

        void serialPort1_DataReceived4(Object^ sender,
SerialDataReceivedEventArgs^ e)
        {
            String^ relestats = serialPort1->ReadLine();
            this->BeginInvoke(gcnew LineReceivedEvent4(this,
&Form1::LineReceived4), relestats);
        }

        // Запис вологості
        delegate void LineReceivedEvent(String^ vlag);
        void LineReceived(String^ vlag)
        {
            textBox1->Text = vlag;
            String^ path = "График_влажности.txt";
            String^ date = DateTime::Now.ToString();
            // Створення файлу і запис в нього
            StreamWriter^ sw = File::AppendText(path);
            sw->WriteLine(vlag);
            sw->WriteLine(date);
```

```
            sw->Close();
        }
    else
#include <iostream>
#include <fstream>
#include <string>
#include <windows.h>

HWND textBox3, textBox4, textBox5;
HWND notifyIcon1;
std::string svet, move, rele;

void SetText(HWND textBox, const std::string& text) {
    SendMessageA(textBox, WM_SETTEXT, 0, (LPARAM)text.c_str());
}

void WriteToFile(const std::string& path, const std::string& text) {
    std::ofstream file(path, std::ios::app);
    if (file.is_open()) {
        file << text << std::endl;
        file.close();
    }
}

LRESULT CALLBACK Form1Proc(HWND hwnd, UINT msg, WPARAM wParam, LPARAM lParam) {
    switch (msg) {
        case WM_SIZE:
            if (wParam == SIZE_MINIMIZED) {
                ShowWindow(hwnd, SW_HIDE);
                SetWindowTextA(notifyIcon1, "Visible");
            }
            break;

        case WM_USER + 1:
            SetText(textBox3, "Достаточно освещения");
            svet = "Достаточно освещения";
            WriteToFile("График_sveta.txt", svet + " " + __TIMESTAMP__);
            break;

        case WM_USER + 2:
            move = std::to_string(lParam);
            if (lParam == 1) {
                SetText(textBox4, "Движение");
            } else {
                SetText(textBox4, "Нет_Движения");
            }
            WriteToFile("График_move.txt", textBox4 + " " + __TIMESTAMP__);
            break;

        case WM_USER + 3:
            rele = std::to_string(lParam);
            if (lParam == 1) {
                SetText(textBox5, "Включено");
            } else {
                SetText(textBox5, "Выключено");
            }
            WriteToFile("График_rele.txt", textBox5 + " " + __TIMESTAMP__);
            break;
```

```
        case WM_CLOSE:
            ShowWindow(hwnd, SW_HIDE);
            SetWindowTextA(notifyIcon1, "Visible");
            return 0;

        case WM_DESTROY:
            PostQuitMessage(0);
            return 0;
    }

    return DefWindowProc(hwnd, msg, wParam, lParam);
}

int main() {
    // Create the main form window
    HINSTANCE hInstance = GetModuleHandle(NULL);
    const char* className = "Form1Class";

    WNDCLASS wc = {};
    wc.lpfnWndProc = Form1Proc;
    wc.hInstance = hInstance;
    wc.lpszClassName = className;

    if (!RegisterClass(&wc)) {
        std::cerr << "Failed to register window class" << std::endl;
        return 1;
    }

    HWND formHandle = CreateWindowEx(
        0, className, "Form1", WS_OVERLAPPEDWINDOW, CW_USEDEFAULT,
CW_USEDEFAULT,
        CW_USEDEFAULT, CW_USEDEFAULT, NULL, NULL, hInstance, NULL);

    if (formHandle == NULL) {
        std::cerr << "Failed to create main window" << std::endl;
        return 1;
    }

    // Create other controls and assign their handles
    textBox3 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 10, 200, 20, formHandle, NULL, hInstance, NULL);
    textBox4 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 40, 200, 20, formHandle, NULL, hInstance, NULL);
    textBox5 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 70, 200, 20, formHandle, NULL, hInstance, NULL);
    notifyIcon1 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 100, 200, 20, formHandle, NULL, hInstance, NULL);

    // Simulate events
    SendMessage(formHandle, WM_USER + 1, 0, 0); // Simulate event for textBox3
    SendMessage(formHandle, WM_USER + 2, 0, 1); // Simulate event for textBox4
    SendMessage(formHandle, WM_USER + 3, 0, 1); // Simulate event for textBox5

    // Main message loop
    MSG msg = {};
    while (GetMessage(&msg, NULL, 0, 0) > 0) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }
```

```
    return msg.wParam;
}
using namespace Gdiplus;

HWND label1, panel1, tabControl1, panel2;
HWND chart1, chart2, chart3;

void SetVisible(HWND control, bool visible) {
    ShowWindow(control, visible ? SW_SHOW : SW_HIDE);
}

void SetZoomableChartProperties(HWND chart) {
    SendMessage(chart, WM_USER + 1, 0, MAKELPARAM(0, 30)); // Zoom in X-axis
    SendMessage(chart, WM_USER + 2, 0, 1); // Enable cursor
    SendMessage(chart, WM_USER + 3, 0, 1); // Enable user selection
    SendMessage(chart, WM_USER + 4, 0, 1); // Make X-axis zoomable
    SendMessage(chart, WM_USER + 5, 0, 1); // Position scrollbar inside
}

void LoadChartData(HWND chart, const std::string& filePath) {
    std::ifstream file(filePath);
    if (file.is_open()) {
        SendMessage(chart, WM_USER + 6, 0, 0); // Clear chart points

        std::string Y, X;
        while (std::getline(file, Y) && std::getline(file, X)) {
            SendMessage(chart, WM_USER + 7, 0, MAKELPARAM(std::stoi(X),
std::stoi(Y))); // Add chart point
        }

        file.close();
    }
}

LRESULT CALLBACK Form1Proc(HWND hwnd, UINT msg, WPARAM wParam, LPARAM lParam) {
    switch (msg) {
        case WM_COMMAND:
            if (lParam == (LPARAM)label1) {
                // label1_Click event handler
            } else if (lParam == (LPARAM)демоToolStripMenuItem) {
                // демоToolStripMenuItem_Click event handler
                SetVisible(panel1, false);
                SetVisible(tabControl1, true);
                SetVisible(panel2, false);

                SetZoomableChartProperties(chart1);
                LoadChartData(chart1, "График_температуры.txt");

                SetZoomableChartProperties(chart2);
                LoadChartData(chart2, "График_влажности.txt");

                SetZoomableChartProperties(chart3);
                LoadChartData(chart3, "График_sveta.txt");
            } else if (lParam == (LPARAM)выходToolStripMenuItem) {
                // выходToolStripMenuItem_Click event handler
                PostMessage(hwnd, WM_CLOSE, 0, 0);
            }
            break;
```

```
        case WM_CLOSE:
            PostQuitMessage(0);
            return 0;
    }

    return DefWindowProc(hwnd, msg, wParam, lParam);
}

int main() {
    // Initialize GDI+
    ULONG_PTR gdiplusToken;
    GdiplusStartupInput gdiplusStartupInput;
    GdiplusStartup(&gdiplusToken, &gdiplusStartupInput, NULL);

    // Create the main form window
    HINSTANCE hInstance = GetModuleHandle(NULL);
    const char* className = "Form1Class";

    WNDCLASS wc = {};
    wc.lpfnWndProc = Form1Proc;
    wc.hInstance = hInstance;
    wc.lpszClassName = className;

    if (!RegisterClass(&wc)) {
        std::cerr << "Failed to register window class" << std::endl;
        return 1;
    }

    HWND formHandle = CreateWindowEx(
        0, className, "Form1", WS_OVERLAPPEDWINDOW, CW_USEDEFAULT,
CW_USEDEFAULT,
        CW_USEDEFAULT, CW_USEDEFAULT, NULL, NULL, hInstance, NULL);

    if (formHandle == NULL) {
        std::cerr << "Failed to create main window" << std::endl;
        return 1;
    }

    // Create other controls and assign their handles
    label1 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 10, 200, 20, formHandle, NULL, hInstance, NULL);
    panel1 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 40, 200, 20, formHandle, NULL, hInstance, NULL);
    tabControl1 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 70, 200, 20, formHandle, NULL, hInstance, NULL);
    panel2 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 100, 200, 20, formHandle, NULL, hInstance, NULL);
    chart1 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 130, 200, 20, formHandle, NULL, hInstance, NULL);
    chart2 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 160, 200, 20, formHandle, NULL, hInstance, NULL);
    chart3 = CreateWindowEx(0, "STATIC", "", WS_CHILD | WS_VISIBLE,
        10, 190, 200, 20, formHandle, NULL, hInstance, NULL);

    // Simulate events
    SendMessage(formHandle, WM_COMMAND, 0, (LPARAM)label1); // Simulate
label1_Click event
```

```
    MSG msg = {};
    while (GetMessage(&msg, NULL, 0, 0) > 0) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    // Clean up GDI+
    GdiplusShutdown(gdiplusToken);

    return msg.wParam;

using namespace Gdiplus;

HWND panel1, tabControl1, panel2;
HWND textBox3, panel1, dataGridView2, dataGridView1;

void SetVisible(HWND control, bool visible) {
    ShowWindow(control, visible ? SW_SHOW : SW_HIDE);
}

void LoadDataGrid(HWND dataGridView, const std::string& filePath) {
    std::ifstream file(filePath);
    if (file.is_open()) {
        SendMessage(dataGridView, WM_USER + 6, 0, 0); // Clear data grid

        std::string header, row;
        std::getline(file, header);
        std::string[] col = System.Text.RegularExpressions.Regex.Split(header, "
");
        for (int c = 0; c < col.Length; c++) {
            SendMessage(dataGridView, WM_USER + 7, 0, col[c]); // Add column to
data grid
        }

        while (std::getline(file, row)) {
            std::string[] value =
System.Text.RegularExpressions.Regex.Split(row, " ");
            SendMessage(dataGridView, WM_USER + 8, 0, value); // Add row to data
grid
        }

        file.close();
    }
}

LRESULT CALLBACK Form1Proc(HWND hwnd, UINT msg, WPARAM wParam, LPARAM lParam) {
    switch (msg) {
        case WM_COMMAND:
            if (lParam == (LPARAM)поточнийСтанToolStripMenuItem) {
                // поточнийСтанToolStripMenuItem_Click event handler
                SetVisible(panel1, true);
                SetVisible(tabControl1, false);
                SetVisible(panel2, false);
            } else if (lParam == (LPARAM)textBox3) {
                // textBox3_TextChanged event handler
            } else if (lParam == (LPARAM)panel1) {
                // panel1_Paint event handler
            } else if (lParam == (LPARAM)Form1_Load) {
                // Form1_Load event handler
```

```
                SetVisible(panel1, true);
                SetVisible(tabControl1, false);
                SetVisible(panel2, false);
            } else if (lParam ==
(LPARAM)состояниеРелеИДатчикаДвиженияToolStripMenuItem) {
                // состояниеРелеИДатчикаДвиженияToolStripMenuItem_Click event
handler
                SetVisible(panel1, false);
                SetVisible(tabControl1, false);
                SetVisible(panel2, true);

                LoadDataGrid(dataGridView2, "График_move.txt");

                LoadDataGrid(dataGridView1, "График_rele.txt");
            } else if (lParam == (LPARAM)проПрограммуToolStripMenuItem) {
                // проПрограммуToolStripMenuItem_Click event handler
                MessageBox(NULL, L"Программа розроблена для роботи з контролером
Arduino UNO, підключення за допомогою порта COM3 і дозволяє відображати і\n"
                    L"зберігати на комп'ютері дані про температуру, освітлення,
вологості та стан реле і рух", L"Про програму", MB_OK);
            } else if (lParam == (LPARAM)проАвтораToolStripMenuItem) {
                // проАвтораToolStripMenuItem_Click event handler
                Form2 pro_avtora = new Form2();
                pro_avtora.Show();
            } else if (lParam == (LPARAM)comboBox1) {
                // comboBox1_SelectedIndexChanged event handler
            } else if (lParam == (LPARAM)button1) {
                // button1_Click event handler
            }
            break;
        case WM_CLOSE:
            DestroyWindow(hwnd);
            break;
        case WM_DESTROY:
            PostQuitMessage(0);
            break;
        default:
            return DefWindowProc(hwnd, msg, wParam, lParam);
    }
    return 0;
}

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR
lpCmdLine, int nCmdShow) {
    // Register window class
    WNDCLASSEX wc = {};
    wc.cbSize = sizeof(WNDCLASSEX);
    wc.lpfnWndProc = Form1Proc;
    wc.hInstance = hInstance;
    wc.lpszClassName = L"Form1Class";
    RegisterClassEx(&wc);

    // Create main window
    HWND formHandle = CreateWindowEx(0, L"Form1Class", L"Form1",
WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, NULL, NULL,
hInstance, NULL);

    if (formHandle == NULL) {
```

```
        std::cerr << "Не вдалося створити головне вікно" << std::endl;
        return 1;
    }

    // Create other controls and assign their handles
    panel1 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 40, 200, 20, formHandle, NULL, hInstance, NULL);
    tabControl1 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 70, 200, 20, formHandle, NULL, hInstance, NULL);
    panel2 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 100, 200, 20, formHandle, NULL, hInstance, NULL);
    textBox3 = CreateWindowEx(0, L"EDIT", L"", WS_CHILD | WS_VISIBLE,
        10, 220, 200, 20, formHandle, NULL, hInstance, NULL);
    panel1 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 250, 200, 20, formHandle, NULL, hInstance, NULL);
    dataGridView2 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 280, 200, 20, formHandle, NULL, hInstance, NULL);
    dataGridView1 = CreateWindowEx(0, L"STATIC", L"", WS_CHILD | WS_VISIBLE,
        10, 310, 200, 20, formHandle, NULL, hInstance, NULL);

    // Simulate events
    SendMessage(formHandle, WM_COMMAND, 0,
(LPARAM)поточнийСтанToolStripMenuItem); // Simulate
поточнийСтанToolStripMenuItem_Click event

    MSG msg = {};
    while (GetMessage(&msg, NULL, 0, 0) > 0) {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    return msg.wParam;
}
   void openSerialPort()
{
    try
    {
        std::string portName = comboBox1.Text;
        HANDLE serialHandle = CreateFile(portName.c_str(), GENERIC_READ |
GENERIC_WRITE, 0, NULL, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, NULL);

        if (serialHandle == INVALID_HANDLE_VALUE)
        {
            throw std::exception("Failed to open serial port.");
        }

        DCB dcbSerialParams = { 0 };
        dcbSerialParams.DCBlength = sizeof(dcbSerialParams);

        if (!GetCommState(serialHandle, &dcbSerialParams))
        {
            throw std::exception("Failed to get serial port state.");
        }

        dcbSerialParams.BaudRate = CBR_9600;
        dcbSerialParams.ByteSize = 8;
        dcbSerialParams.StopBits = ONESTOPBIT;
        dcbSerialParams.Parity = NOPARITY;
        dcbSerialParams.fDtrControl = DTR_CONTROL_ENABLE;
```

```
        if (!SetCommState(serialHandle, &dcbSerialParams))
        {
            throw std::exception("Failed to set serial port parameters.");
        }

        // Continue with other data received event registrations or operations

        CloseHandle(serialHandle);
    }
    catch (std::exception& ex)
    {
        MessageBoxA(NULL, ex.what(), "Error", MB_OK | MB_ICONERROR);
    }
}

void panel2_Paint(HWND hWnd, HDC hdc)
{
    // Paint event code for panel2
}

int main()
{
    // Windows Forms Application initialization code

    try
    {
        openSerialPort();
    }
    catch (std::exception& ex)
    {
        MessageBoxA(NULL, ex.what(), "Error", MB_OK | MB_ICONERROR);
    }

    // Event registration and other code

    return 0;
}

void loop() {
  rel = 0;
  mowe = 0;
  float h = dht.readHumidity();
  float t = dht.readTemperature();

  if (isnan(t) || isnan(h)) {
  } else {
    display.clearDisplay();
    display.setTextSize(1);
    display.setCursor(0, 10);
    display.print("Temp=");
    display.println(t);
    display.setCursor(0, 20);
    display.print("Volog=");
    display.println(h);
    display.display();

    if (t >= 27) {
      digitalWrite(Relay, LOW);
```

```
      rel = 1;
    } else {
      digitalWrite(Relay, HIGH);
      rel = 0;
    }
  }

  int value = analogRead(sensePin);
  value = constrain(value, 600, 900);
  int ledLevel = map(value, 900, 600, 255, 0);
  analogWrite(ledPin, ledLevel);
  long now = millis();

  if (digitalRead(pirPin) == HIGH) {
    mowe = 1;
    if (now > (lastSend + minSecsBetweenEmails * 1000)) {
      display.setTextSize(2);
      display.setCursor(15, 30);
      display.println("MOVEMENT");
      display.display();
      lastSend = now;
    }
  }

  sentParams(t, h, ledLevel, mowe, rel);
  delay(1000);
}
```