

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук та технологій

(повне найменування факультету)

Кафедра комп'ютерних систем та мереж

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавра

(ступінь вищої освіти)

на тему «Розробка системи автоматизованого управління сервісними
функціями житлового приміщення»

(назва теми)

Виконав: студент 4 курсу, групи КНТ-510сп

Спеціальності _____

123 «Комп'ютерна інженерія»

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерна інженерія

ФЕДОРОВ Я.Ю.

(ПРИЗВИЩЕ та ініціали)

Керівник

СКРУПСЬКИЙ С.Ю.

(ПРИЗВИЩЕ та ініціали)

Рецензент

КАЧАН О.І.

(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет Комп'ютерних наук і технологій

Кафедра Комп'ютерні системи та мережі

Ступінь вищої освіти бакалавр

Спеціальність 123 «Комп'ютерна інженерія»

(код і найменування)

Освітня програма (спеціалізація) Комп'ютерна інженерія

(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри Кудерметов Р.К.

« _____ » _____ 2023 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Федоров Ярослав Юрійович

(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка системи автоматизованого управління сервісними функціями житлового приміщення

керівник проєкту (роботи) к. т. н., доцент, СКРУПСЬКИЙ Степан Юрійович,

(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від «24» березня 2023 року № 71

2. Строк подання студентом проєкту (роботи) 25 травня 2023 року

3. Вихідні дані до проєкту (роботи) IoT пристрої: Arduino Uno, модуль RYLR896, перетворювач FT232RL, Raspberry Pi 3 B+;

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз технічного завдання

Розробка програмної частини системи

Загальне функціонування системи

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень, кількість слайдів, плакатів)

Пл1 – Структура пакету;

Пл2 – Діаграма класів системи;

Пл3 – Діаграма послідовностей процесу взаємодії користувача;

Пл4 – Зібрана система;

6. Консультанти розділів проекту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
	СКРУПСЬКИЙ С.Ю., к. т. н., доцент		
Нормоконтроль	ЩЕРБАК Н.В., ст. викл.		

7. Дата видачі завдання «8» травня 2023 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
	Аналіз технічного завдання	1 тиждень	виконано
	Аналіз структури роботи сигналізацій для автотранспорту	1 тиждень	виконано
	Вибір відповідної технології передачі інформації	1 тиждень	виконано
	Підбір комплектуючих елементів для створення системи, схеми підключення	2 тиждень	виконано
	Написання програмного коду	3 тиждень	виконано
	Проектування програмних засобів	3 тиждень	виконано
	Реалізація проекту	3 тиждень	виконано
	Проходження нормоконтролю	4 тиждень	виконано
	Захист роботи	4 тиждень	виконано

Студент(ка)


 (підпис)
Ярослав ФЕДОРОВ

(Ім'я ПРИЗВИЩЕ)

Керівник проекту (роботи)


 (підпис)
Степан СКРУПСЬКИЙ

(Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

ПЗ: 60 с., 20 рис., 1 табл., 13 джерел.

ARDUINO, RASPBERRY PI, АВТОМАТИЗАЦІЯ УПРАВЛІННЯ ПРИМІЩЕННЯМИ, РОЗУМНИЙ БУДИНОК

Результат роботи – реалізований базовий функціонал системи автоматизованого управління сервісними функціями житлового приміщення можливостями подальшого розширення, визначено напрямки подальшого розвитку.

Метою даної роботи є розробка системи управління "розумним будинком", яка дозволить автоматизувати повсякденні завдання, керуючи як пристроями та датчиками своєї розробки, так і пристроями сторонніх виробників у режимі "чорного ящика".

Завдання, які були вирішені в рамках цього проекту:

- аналіз існуючих систем та готових рішень, виявлення їх переваг та недоліків;
- розробка універсального захищеного протоколу управління пристроями «розумного дому»;
- реалізація програмного комплексу, управителя власними пристроями та пристроями сторонніх виробників.

ЗМІСТ

Вступ.....	6
1 Аналіз технічного завдання.....	7
1.1 Огляд існуючих технологій	7
1.2 Вимогу до продукту що розробляється.....	17
1.3 Вимоги до захищеного каналу системи	19
2 Розробка програмної частини системи	22
2.1 Розробка структури програмного комплексу.....	22
2.2 Реалізація протоколу взаємодії між пристроями системи	25
2.3 Опис алгоритму шифрування.....	29
2.4 Реалізація для RaspberryPi.....	33
2.5 Реалізація мікроконтролера.....	41
3 Загальне функціонування системи	43
3.1 Опис реалізації програми	43
3.2 Реалізація програми	52
Висновки	58
Перелік джерел посилання	59

Перелік графічного матеріалу:

Пл1 – Структура пакету;

Пл2 – Діаграма класів системи;

Пл3 – Діаграма послідовностей процесу взаємодії користувача;

Пл4 – Зібрана система;

ВСТУП

У сучасному світі з кожним роком все збільшується тенденція до спрощення та автоматизації повсякденних завдань. У побут сучасної людини щільно увійшли технології віддаленого та безконтактного управління.

Ці технології допомагають заощаджувати час, а й дозволяють не залежати від місцезнаходження. Наприклад, ви хочете завжди бути впевненим у збереженні свого майна у свою відсутність, або не хочете вставати з дивана, щоб увімкнути або вимкнути світло. В цьому вам допоможуть автоматизовані системи. Зростання популярності автоматизованих систем управління сервісними функціями житлових приміщень, таких як «розумний будинок», обумовлене прагненням людини до комфорту та зручності. Додатковою привабливістю є безпека, чи то протипожежна система, чи сигналізація з дистанційним оповіщенням.

«Розумний будинок» є сучасним інструментом підвищення комфорту та рівня життя, тому що частина процесів відбувається автоматично, а рештою можна управляти віддалено, що робить її актуальною для вивчення та вдосконалення [1].

Метою даної роботи є розробка системи управління "розумним будинком", яка дозволить автоматизувати повсякденні завдання, керуючи як пристроями та датчиками своєї розробки, так і пристроями сторонніх виробників у режимі "чорного ящика".

Завдання, які були вирішені в рамках цього проекту:

- аналіз існуючих систем та готових рішень, виявлення їх переваг та недоліків;
- розробка універсального захищеного протоколу управління пристроями «розумного дому»;
- реалізація програмного комплексу, управителя власними пристроями та пристроями сторонніх виробників.

1 АНАЛІЗ ТЕХНІЧНОГО ЗАВДАННЯ

1.1 Огляд існуючих технологій

У цьому розділі будуть розглянуті існуючі на ринку реалізації ідеї системи «розумний дім». У проаналізованих системах виявимо переваги та недоліки.

Концепція "розумного будинку" стала дедалі популярнішою в сучасному світі, пропонуючи нові можливості для автоматизації та поліпшення життя людей. Системи розумного будинку дають змогу керувати різними аспектами житлового простору, як-от освітлення, опалення, безпека та розваги, за допомогою інтеграції сучасних технологій. У цій статті ми розглянемо наявні на ринку реалізації ідеї системи "розумний дім" і представимо деякі з найпопулярніших платформ і рішень.

Одним із лідерів на ринку систем розумного будинку є Apple з їхньою платформою HomeKit. HomeKit пропонує широкий спектр можливостей для автоматизації будинку, даючи змогу керувати освітленням, термостатами, розетками, замками і багатьом іншим через пристрої Apple, як-от iPhone, iPad і HomePod. HomeKit також забезпечує високий рівень безпеки та шифрування даних, щоб забезпечити захист вашого будинку та особистої інформації.

Google Assistant і платформа Google Home пропонують зручне управління розумним будинком за допомогою голосових команд. Вони інтегруються з різними пристроями, такими як Nest-термостати, розумні колонки та пристрої освітлення. Google Assistant може виконувати різні завдання, як-от керування пристроями, налаштування розкладу, програвання музики та відповіді на запитання. За допомогою Google Home ви можете створити сцени і автоматизовані дії для підвищення комфорту та енергоефективності у вашому домі.

Amazon Alexa і серія пристроїв Echo пропонують широкий набір функцій для розумного будинку. За допомогою голосових команд ви можете керувати освітленням, термостатами, замками, системами безпеки та іншими пристроями.

Alexa також інтегрується з іншими пристроями та сервісами, як-от розумні колонки Sonos, системи відеоспостереження Ring і багато іншого. Ви можете налаштувати автоматизацію сцен і режимів, щоб ваш будинок функціонував оптимально відповідно до ваших уподобань.

Samsung SmartThings - це ще одна платформа, що надає можливості для управління розумним будинком. Вона підтримує широкий спектр пристроїв, включно з освітленням, термостатами, датчиками руху, дверними замками та багатьма іншими. SmartThings дає змогу створювати автоматизовані сценарії, які активуються на основі певних умов або розкладу. Ви також можете керувати своїм розумним будинком через додаток SmartThings на смартфоні або планшеті.

Xiaomi Mi Home - це китайська платформа розумного будинку, що пропонує доступні рішення для автоматизації будинку. Вона підтримує широкий спектр пристроїв, включно з освітленням, розумними розетками, камерами відеоспостереження, датчиками та іншими пристроями. Xiaomi Mi Home пропонує простий інтерфейс і можливість управління вашим розумним будинком через додаток на смартфоні. Вона також інтегрується з іншими платформами, як-от Google Assistant і Amazon Alexa, що дає змогу вам використовувати їхні функції для зручнішого керування вашим розумним будинком.

Існує безліч рішень на ринку для реалізації ідеї системи "розумний дім". Вони пропонують різні функції та інтеграцію з різними пристроями. Вибір конкретної платформи залежить від ваших потреб, уподобань і сумісності з вашими пристроями. Важливо провести дослідження і вибрати відповідне рішення, яке відповідає вашим вимогам щодо автоматизації та управління вашим розумним будинком.

«Розумний будинок» технічно є сукупністю систем пристроїв та датчиків, об'єднаних у загальну структуру деякою загальною шиною та керованою центром управління. Ці системи бувають наступних видів:

- система мікроклімату (опалення, вентиляція, кондиціонування, зволоження тощо);
- система безпеки (охоронна, пожежна сигналізація, система доступу,

контроль витоків газу та води, відеоспостереження);

- система електроживлення (резервні системи, контроль навантаження
- електромережі, система освітлення, система управління навантажувальними елементами тощо);

- система зв'язку (телефон, локальна мережа, інтернет, SMS оповіщення);
- система віддаленого керування.

На даний момент найбільш широко використовуються такі перспективні технології об'єднання на загальній шині систем та пристроїв «розумного дому»:

- LanDriver – універсальна платформа побудови шинних систем контролера та модулів, підключених між собою шиною (стандарт RS-485). До модулів підключається кероване обладнання. Орієнтована на промислове користування.

- EIB/KNX – Система EIB розподілена, управління здійснюється у межах пристроїв. Пристрої обмінюються інформацією по шині EIB відповідно до власного протоколу. Система, побудована на EIB, автономна і залежить від працездатності центрального контролера.

- AMX розробляє програмно-апаратні засоби віддаленого керування медіасистемою, системою відеоспостереження та широкого спектру датчиків. Протоколи передачі даних закриті. Спочатку застосовувалася власна шина передачі даних, у новій лінійці обладнання застосовуються стандартні протоколи Ethernet та Wi-Fi, є частково розвинені технології сполучення із системами EIB, LON та інших.

- Z-wave, бездротова технологія передачі даних, розроблена для домашньої автоматизації. У технології Z-wave застосовуються малопотужні та мініатюрні радіомодулі, що вбудовуються у побутову техніку. У основі технології лежить пориста технологія, у якій кожен вузол є приймачем і передавачем, тобто. у разі виникнення перешкоди сигнал піде через сусідні вузли мережі, що у радіусі дії. Ще однією перевагою є мале енергоспоживання, що разом із малими розмірами дозволяє вбудовувати Z-wave у різні побутові прилади.

Кожна з перелічених вище систем використовує обмін даних свої чи

сторонні протоколи, залежно від поставлених до системи завдань.

Варто зазначити, що більшість систем та технологій автоматизації приміщень закриті.

Розглянемо деякі варіанти готових систем, заснованих на перерахованих вище технологіях.

NetPing. Компанія Alentis Electronics є розробником і виробником пристрою моніторингу навколишнього середовища NetPing. Основна сфера застосування – віддалений контроль та моніторинг пристроїв у будинку та офісі. Використовує як базовий протокол Ethernet.

Завдання, які вирішуються за допомогою пристрою NetPing:

- віддалене керування електроживленням;
- управління безпекою та відстеження надзвичайних подій, використовуючи датчики диму, протікання води, витік газу, антивандальні системи, управління камерами відеоспостереження;
- управління мікрокліматом за допомогою датчиків температури, вологості та управління кондиціонером через інфрачервоний порт;
- керування АТС по порту RS-232;
- дистанційна зміна налаштувань залежно від ситуації;
- надсилання повідомлень про неполадки або інші важливі події за допомогою SMS, електронної пошти;
- доступ до системи в реальному часі через HTTP або SNMP;
- управління освітленням та іншими побутовими приладами за розкладом;



Рисунок 1.1 - Загальна схема системи NetPing

Пристрої NetPing дозволяють підключити до 16 датчиків на один пристрій. Завдяки вбудованому Web-серверу контроль та керування здійснюється через браузер.

NetPing 2/PWR-220 v2

Датчики температуры

[ГЛАВНАЯ](#) | [НАСТРОЙКИ](#) | [УПРАВЛЕНИЕ 220V](#) | [СТОРОЖ](#) | [ТЕМПЕРАТУРА](#) | [ВВОД-ВЫВОД](#) | [ЖУРНАЛ](#)

Параметр	Датчик 1	Датчик 2	Датчик 3	Датчик 4	Датчик 5	Датчик 6	Датчик 7	Датчик 8
памятка (до 16 симв.)								
текущая температура, °C	18	0	0	0	0	0	0	0
статус	в норме	сбой	сбой	сбой	сбой	сбой	сбой	сбой
верхн. граница нормы, °C	40	60	60	60	60	60	60	60
нижн. граница нормы, °C	10	10	10	10	10	10	10	10
посылка trap сообщений								
t° поднялась выше нормы	☐	☐	☐	☐	☐	☐	☐	☐
t° вошла в норму	☐	☐	☐	☐	☐	☐	☐	☐
t° опустилась ниже нормы	☐	☐	☐	☐	☐	☐	☐	☐
период посылки (10-9999с, 0=выкл)	10	0	0	0	0	0	0	0

Рисунок 1.2 - Контрольная панель датчиков температуры системы NetPing

Можна використовувати сторонні програми моніторингу, наприклад Zabbix, Nagios та PRTG Network, який рекомендує виробник NetPing. Перевага PRTG Network полягає у зручнішому інтерфейсі програми, можливість вести докладну статистику та мобільну версію програми (Android та iOS).

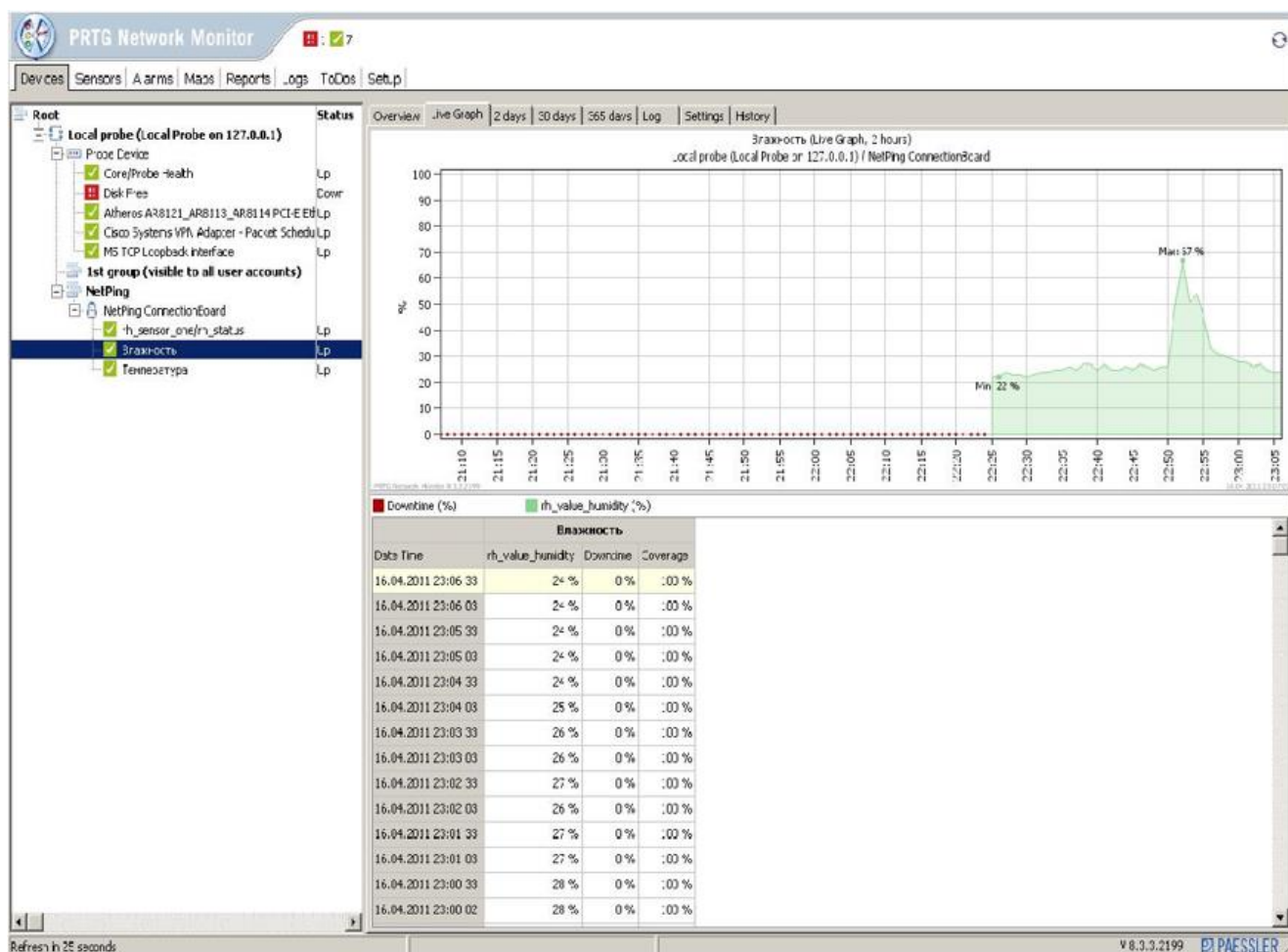


Рисунок 1.3 - Моніторинговий додаток PRTG Network, підключений до датчиків системи NetPing

OpenRemote – програмний комплекс, що забезпечує автоматизацію житлових та комерційних приміщень. OpenRemote дозволяє створити мобільний додаток для розумного будинку без програмування, при цьому можна використовувати різні існуючі протоколи зв'язку - EIB/KNX, AMX, Z-wave. Простими словами це кроссплатформенний конструктор, в якому можна створити інтерфейс майбутнього мобільного додатка. Контролери, які можуть бути використані: AMX, KNX, Beckhoff, Lutron, Z-Wave, 1- Wire, MiCasaVerde Vera, EnOcean, xPL, Insteon, X10, Infrared, Russound, GlobalCache, IRTrans, XBMC, VLC, Samsung SmartTV, panStamps, Denon AVR, Marantz AVR, FreeBox, MythTV, RaZBerry та інші.

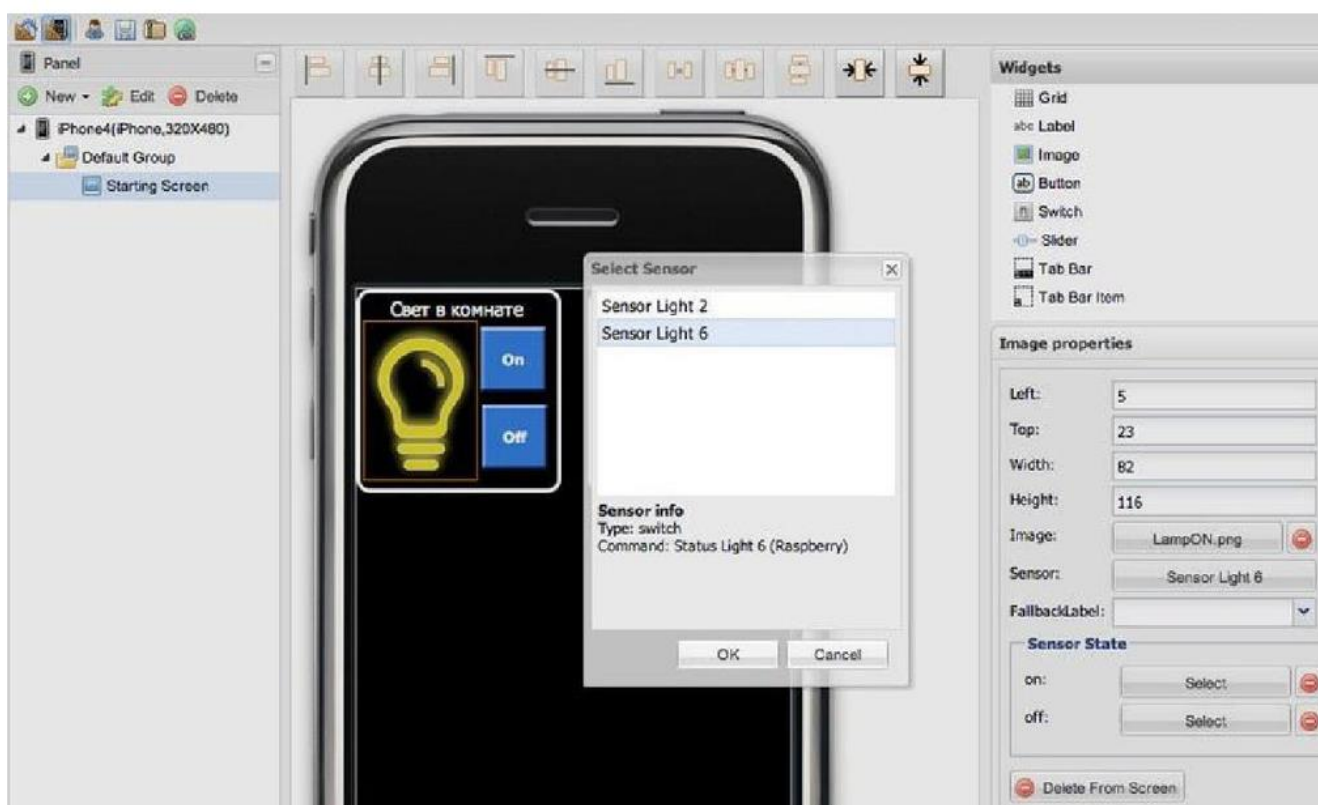


Рисунок 1.4 - Процес створення мобільного додатка системи OpenRemote

Home Sapiens. Інтелектуальна система з голосовим управлінням, є програмним комплексом. У комплект не входить обладнання, але при цьому забезпечена максимальна сумісність із комп'ютерним залізом. Забезпечено інтеграцію із системами Z-wave, Gira, ZigBee, x10, C-bus, що дозволяє керувати освітленням, побутовою електронікою, системою опалення та іншими підсистемами. Основний упор йде на голосове управління та зручний інтерфейс.



Рисунок 1.5 - Інтерфейс керування Home Sapiens

MajorDoMo – це відкрита програмна платформа для автоматизації домашніх процесів. Використовує Ethernet. Дана система кросплатформова не вимоглива до ресурсів комп'ютера. Може бути використана без модулів (датчиків) як персональний органайзер. Завдання, які вирішуються за допомогою MajorDoMo:

- система безпеки;
- система мікроклімату;
- медіасистема;
- органайзер.



Рисунок 1.6 - вікно контролю системи MajorDoMo

Fibaro – система автоматизації будівель, заснована на бездротовій технології передачі даних Z-Wave. Мініатюрні модулі можуть бути встановлені за будь-яким вимикачем світла або в побутовому приладі, тобто відпадає необхідність монтажу кабельних каналів зв'язку. Завдяки бездротовій технології передачі даних пристрою Fibaro можна демонтувати та переносити на нове місце. Висока інтеграція коїться з іншими системами. Мозком системи Fibaro є Home Center 2. Інтерфейс надає простий контроль над групами пристроїв, що відповідають за функції – опалення, кондиціонування, освітлення тощо.



Рисунок 1.7 - інтерфейс HomeCenter2 системи управління розумним будинком від Fibaro

Проаналізувавши вищезазначені рішення, ми можемо виділити кілька основних проблем, властивих існуючим системам[3]:

- у більшості випадків - закритість протоколів обміну даними, що призводить до неможливості використовувати датчики та пристрої однієї системи в іншій;
- майже всі рішення вимагають прокладання кабельних мереж для

інтеграції в існуючу структуру приміщень та приєднання до прокладених на етапі будівництва комунікацій;

– ринкова вартість всіх без винятку рішень починається від 50000 рублів навіть у базовій комплектації та без урахування монтажних та пуско-налагоджувальних робіт.

1.2 Вимоги до продукту що розробляється

Враховуючи вищевикладені висновки з аналізу систем, що існують на ринку, ми можемо сформулювати список вимог до створюваної системи «розумного будинку»:

- бездротовий зв'язок між усіма пристроями та датчиками, що вимагає мінімального втручання у конструкції приміщення;
- простота та масштабованість протоколу зв'язку для можливості використання пристроїв як своєї, так і сторонньої розробки;
- захищеність протоколу зв'язку від перешкод та спроб стороннього впливу;
- максимальна дешевизна та надійність використовуваних компонентів.

Концепція розумного будинку стала дедалі популярнішою в сучасному світі. Вона пропонує можливості автоматизації та інтеграції різних аспектів житлового простору, даючи змогу керувати освітленням, опаленням, безпекою та розвагами. При розробці продукту в якості розумного будинку необхідно визначити вимоги, щоб забезпечити високу функціональність, зручність використання і безпеку. У цій статті ми розглянемо основні вимоги до розроблюваного продукту в якості розумного будинку.

Один з головних аспектів розумного будинку - це можливість інтеграції з різними пристроями і системами. Продукт має бути сумісний з широким спектром пристроїв, такими як розумні розетки, освітлення, термостати, системи безпеки та

розваг. Він повинен підтримувати стандарти зв'язку і протоколи, такі як Wi-Fi, Bluetooth, Zigbee, Z-Wave, щоб забезпечити сумісність з іншими пристроями на ринку. Інтеграція з популярними голосовими помічниками, такими як Amazon Alexa, Google Assistant або Apple Siri, також є важливою вимогою.

Продукт має бути легким у використанні та мати інтуїтивно зрозумілий інтерфейс. Користувачі повинні мати можливість легко налаштовувати і керувати різними функціями розумного будинку. Мобільні додатки, панелі управління і голосові команди повинні бути доступні і прості у використанні. Можливість створення сценаріїв і автоматизованих дій для певних ситуацій, як-от вхід або вихід із дому, також є важливою вимогою.

Безпека та захист особистої інформації є критично важливими вимогами для розумного будинку. Продукт повинен забезпечувати високий рівень захисту від несанкціонованого доступу і злому. Підключення і передача даних повинні бути шифрованими для запобігання витоку інформації. Також необхідно забезпечити можливість керування доступом і налаштування дозволів для користувачів, щоб запобігти несанкціонованому управлінню розумним будинком.

Продукт повинен мати можливість розширення і додавання нових пристроїв і функцій у майбутньому. Користувачі можуть захотіти збільшити можливості свого розумного будинку, тому важливо, щоб продукт був гнучким і масштабованим. Він повинен підтримувати можливість додавання нових модулів і пристроїв, а також інтеграцію з новими технологіями і стандартами, які з'являться в майбутньому.

Продукт має бути енергоефективним і екологічно дружнім. Він повинен мати можливість оптимізації енергоспоживання в будинку, керувати освітленням і опаленням відповідно до присутності людей, часу доби та інших факторів. Також важливо передбачити можливість моніторингу енергоспоживання та надання звітів користувачеві, щоб він міг ухвалювати рішення щодо енергозбереження.

Розробка продукту як розумного будинку вимагає врахування низки ключових вимог. Інтеграція і сумісність, зручність використання та інтерфейс, безпека і конфіденційність, розширюваність і гнучкість, а також

енергоефективність - все це фактори, які мають бути враховані під час розроблення розумного будинку. Розуміння і задоволення цих вимог допоможе створити продукт, який забезпечить комфорт, безпеку та ефективність у житловому просторі користувача.

Оскільки програмний комплекс «розумного будинку» безпосередньо управляє системами безпеки (охоронна сигналізація, приводи замків та водопровідних кранів та інші пристрої), однією з основних проблем при використанні бездротових технологій в організації зв'язку між пристроями та датчиками є захищеність каналу зв'язку. Будь-які дані, що передаються «по повітрю», можуть бути перехоплені та використані зловмисником у своїх цілях за допомогою недорогої апаратури. Відповідно, оскільки фізично захиститися від такого перехоплення неможливо, необхідно використовувати методи захисту, які роблять перехоплені дані марними. Таким способом захисту є шифрування даних[6].

Також не варто забувати про сучасну насиченість радіоефіру різноманітними даними на різних частотах, які, по суті, є для нас перешкодами та їх необхідно враховувати та фільтрувати, підтримувати стабільний канал зв'язку в умовах «забрудненого» ефіру. Протокол обміну повинен вирішувати проблему нестабільного зв'язку та підтримувати канал, а у разі неможливості – програма має відповідним чином зреагувати на зникнення зв'язку.

1.3 Вимоги до захищеного каналу системи

Розумний будинок - це сучасна концепція, яка пропонує автоматизацію та інтеграцію різних аспектів житлового простору, включно з освітленням, опаленням, безпекою та розвагами. При розробці розумного будинку важливо врахувати вимоги до захищеного каналу, який забезпечить безпечне передавання

даних і комунікацію між різними пристроями та системами. У цій статті ми розглянемо основні вимоги до захищеного каналу системи в розумному будинку.

Однією з основних вимог до захищеного каналу є шифрування даних. Усі дані, що передаються між пристроями та системами в розумному будинку, мають бути зашифровані для запобігання несанкціонованому доступу та підміни інформації. Шифрування може здійснюватися з використанням симетричних або асиметричних алгоритмів шифрування, таких як AES або RSA. Це гарантує, що тільки авторизовані пристрої зможуть розшифрувати і прочитати передані дані.

Іншою важливою вимогою є автентифікація та авторизація користувачів і пристроїв у розумному будинку. Користувачі повинні мати можливість ідентифікувати себе перед доступом до системи і отримати відповідні дозволи на управління пристроями і функціями. Це може бути реалізовано за допомогою паролів, біометричної ідентифікації, двофакторної аутентифікації та інших методів. Також важливо надати можливість управління дозволами та налаштуваннями доступу, щоб користувачі могли контролювати рівень доступу до свого розумного будинку.

Захист від атак є важливою вимогою для захищеного каналу в розумному будинку. Система має бути захищена від різних видів атак, таких як атаки переповнення буфера, відмова в обслуговуванні (DDoS), фішинг, злом паролів та інші. Цього можна досягти за допомогою механізмів виявлення та запобігання вторгненням, фільтрації трафіку, регулярного оновлення програмного забезпечення та застосування найкращих практик безпеки.

Фізична безпека також є важливим аспектом захищеного каналу в розумному будинку. Пристрої та інфраструктура, пов'язані з розумним будинком, повинні бути захищені від фізичного доступу та несанкціонованого використання. Це може бути досягнуто за допомогою фізичного блокування, відеоспостереження, контролю доступу та інших методів фізичної безпеки.

Важливою вимогою до захищеного каналу в розумному будинку є регулярне оновлення програмного забезпечення та підтримка. Розробники мають забезпечити регулярні оновлення системи, щоб усунути вразливості та

забезпечити безпеку переданих даних. Також важливо надати підтримку користувачів, щоб вони могли розв'язувати проблеми, що виникають, і отримувати актуальну інформацію про безпеку.

Захищений канал є ключовим елементом системи розумного будинку. Вимоги до захищеного каналу охоплюють шифрування даних, автентифікацію та авторизацію, захист від атак, фізичну безпеку, а також оновлення та підтримку системи. Врахування цих вимог допоможе створити безпечне та захищене середовище для передавання даних і комунікації в розумному будинку.

З допомогою шифрування повинні забезпечуватися три стани безпеки інформації: конфіденційність (приховання інформації від сторонніх осіб), цілісність (запобігання зміні даних під час їх передачі) та ідентифікованість (автентифікація приймача та передавача).

Тільки при дотриманні цих умов ми можемо вважати, що з'єднання є безпечним.

За методами шифрування інформації поділяється на симетричне та несиметричне.

У симетричних криптосистемах для шифрування і розшифровки використовується один і той же ключ. Звідси назва – симетричні.

Алгоритм та ключ вибирається заздалегідь і відомий обом сторонам. Збереження ключа в секретності є важливим завданням для встановлення та підтримки захищеного каналу зв'язку. У зв'язку з цим виникає проблема початкової передачі ключа (синхронізації ключів). Крім того, існують методи криптоатак, що дозволяють так чи інакше дешифрувати інформацію не маючи ключа або за допомогою його перехоплення на етапі узгодження. У цілому нині ці моменти є проблемою криптостойкості конкретного алгоритму шифрування є аргументом під час виборів конкретного алгоритму.

Недоліками симетричного шифрування є проблема передачі ключа співрозмовнику та неможливість встановити справжність чи авторство тексту

У системах з асиметричним шифруванням використовуються два ключі - відкритий і закритий, пов'язані певним математичним чином один з одним.

Відкритий ключ передається по відкритому (тобто незахищеному, доступному для спостереження) каналу і використовується для шифрування повідомлення та перевірки ЕЦП. Для розшифровки повідомлення та для генерації ЕЦП використовується секретний ключ.

Ця схема вирішує проблему симетричних схем, пов'язану з початковою передачею ключа іншій стороні. Якщо в симетричних схемах зломисник перехопить ключ, то він зможе як «слухати», так і вносити правки в інформацію, що передається. В асиметричних системах іншій стороні передається відкритий ключ, який дозволяє шифрувати, але не розшифровувати інформацію. Отже вирішується проблема симетричних систем, що з синхронізацією ключів[6].

Оскільки в системі, що створюється, використовуються мікроконтролери Arduino Uno, що не мають великих обчислювальних потужностей, присутня необхідність використання потокового шифрування, а також немає необхідності в постійній зміні ключа, для реалізації захисту каналу був обраний алгоритм симетричного шифрування в режимі простої заміни. З докладнішим описом роботи алгоритму ми ознайомимося у розділі 2.3

2 РОЗРОБКА ПРОГРАМНОЇ ЧАСТИНИ СИСТЕМИ

2.1 Розробка структури програмного комплексу

Розумні будинки стають дедалі популярнішими, оскільки вони пропонують зручність, безпеку та енергоефективність. Одним із ключових аспектів розробки розумного будинку є створення програмного комплексу, який керує та координує роботу різних пристроїв і систем у будинку. У цій статті розглянемо основні аспекти розробки структури програмного комплексу розумного будинку.

Першим кроком у розробці структури програмного комплексу розумного будинку є визначення архітектури системи. Архітектура визначає організацію компонентів програмного комплексу, їхню взаємодію та основні принципи роботи.

У розумному будинку може використовуватися різні архітектурні підходи, такі як централізована, розподілена або гібридна архітектура. Кожен із цих підходів має свої переваги та недоліки, і вибір залежить від конкретних потреб і вимог розумного будинку.

Програмний комплекс розумного будинку складається з різних компонентів, які виконують певні функції. Важливо визначити й описати ці компоненти, щоб зрозуміти їхню роль у системі та взаємозв'язки між ними. Деякі з основних компонентів програмного комплексу розумного будинку включають:

Керуючий модуль: здійснює загальне управління і координацію роботи системи, ухвалює рішення на основі отриманих даних і керує пристроями в будинку.

Інтерфейс користувача: надає користувачеві можливість взаємодії з системою, управління пристроями та налаштуванням параметрів.

Сенсори та датчики: збирають дані про стан довкілля, такі як температура, освітлення, рух та інші параметри.

Актuatorи: пристрої, які виконують певні дії на основі команд від керуючого модуля, наприклад, увімкнення або вимкнення освітлення, відчинення та зачинення вікон і дверей та інші дії.

Система безпеки: забезпечує захист розумного будинку від несанкціонованого доступу, включно із системами відеоспостереження, контролю доступу та сигналізації.

Мережева інфраструктура: забезпечує зв'язок і взаємодію між компонентами програмного комплексу, а також із зовнішніми системами і сервісами.

Сховище даних: використовується для збереження та аналізу даних, зібраних сенсорами та іншими пристроями.

Для забезпечення сумісності та взаємодії між різними пристроями і системами в розумному будинку необхідно визначити відповідні протоколи і стандарти. Протоколи визначають правила обміну даними і командами між пристроями, а стандарти забезпечують узгодженість і сумісність різних пристроїв і систем. Деякі з найпоширеніших протоколів і стандартів для розумних будинків

включають Zigbee, Z-Wave, Bluetooth, Wi-Fi, MQTT та інші.

Важливим аспектом розробки структури програмного комплексу розумного будинку є забезпечення безпеки. Розумний будинок містить безліч пристроїв, які можуть бути вразливими для атак і несанкціонованого доступу. Тому необхідно передбачити заходи безпеки, як-от шифрування даних, аутентифікація користувачів, контроль доступу та моніторинг активності. Також важливо регулярно оновлювати програмне забезпечення і стежити за новими вразливостями та загрозами безпеці.

Розумний будинок може бути інтегрований з іншими зовнішніми системами і сервісами, такими як хмарні платформи, системи безпеки, системи енергозбереження та інші. Розроблення структури програмного комплексу має враховувати можливість взаємодії з такими системами і забезпечувати сумісність та інтеграцію.

Розробка структури програмного комплексу розумного будинку є важливим етапом створення сучасної та ефективної системи управління будинком. Правильна організація компонентів, вибір протоколів і стандартів, забезпечення безпеки та інтеграція із зовнішніми системами є ключовими факторами успіху. Однак кожен розумний дім має свої особливості та вимоги, тому розробка структури програмного комплексу має бути індивідуальною та адаптованою під конкретні потреби та цілі користувача. З урахуванням цих аспектів, розробники можуть створювати інноваційні та функціональні розумні будинки, які забезпечують комфорт, безпеку та енергоефективність для мешканців.

Проектування системи будувалося на переліку бізнес-процесів, які має реалізовувати ця система. Узагальнено можна виділити такі:

- перегляд списку пристроїв та їх стану;
- взаємодія з пристроями, що мають 2 стани - включено/вимкнено або відкрито/закрито;
- отримання повідомлень від датчиків;
- керування іг-пристроями;
- додавання нових пристроїв через панель адміністрування.

Реалізована система складається з:

- веб-інтерфейсу (хост-сервер на RaspberryPi);
- утиліти для надсилання повідомлень бездротовим каналом (RaspberryPi);
- утиліти для надсилання команд через ir-передавач (RaspberryPi);
- сервісу, що слухає повідомлення, отримані бездротовим каналом (RaspberryPi);
- програми, що знімає показання з датчика задимлення (Arduino Uno);
- програми, що знімає показання з датчика затоплення (Arduino Uno);
- програми, що управляє сервоприводами для відкриття та закриття штор (Arduino Uno);

Користувач взаємодіє із системою лише через веб-інтерфейс.

2.2 Реалізація протоколу взаємодії між пристроями системи

Розумні будинки стають дедалі популярнішими, і їхня функціональність зростає з кожним роком. У розумному будинку різні пристрої, як-от освітлення, опалення, системи безпеки та інші, повинні взаємодіяти між собою, щоб забезпечувати зручність, енергоефективність і безпеку для користувачів. Реалізація протоколу взаємодії між пристроями є важливим аспектом розробки системи розумного будинку. У цій статті розглянемо основні аспекти реалізації такого протоколу.

Першим кроком у реалізації протоколу взаємодії між пристроями системи розумного будинку є вибір відповідного протоколу. Існує безліч протоколів, таких як Zigbee, Z-Wave, Bluetooth, Wi-Fi, MQTT та інші, які можуть використовуватися для взаємодії між пристроями. Кожен протокол має свої особливості, переваги та недоліки, і вибір протоколу залежить від конкретних вимог і цілей системи розумного будинку. Важливо вибрати протокол, який забезпечує надійність,

ефективність передавання даних, безпеку та сумісність із широким спектром пристроїв.

Після вибору протоколу необхідно реалізувати комунікаційний шар, який забезпечує передачу даних і взаємодію між пристроями. Комунікаційний шар може бути реалізовано на різних рівнях, включно з фізичним рівнем, рівнем каналу передавання даних і рівнем прикладних протоколів. Важливо забезпечити надійну передачу даних, управління потоком інформації, обробку помилок та інші аспекти, необхідні для ефективної взаємодії пристроїв.

Одним із ключових елементів реалізації протоколу взаємодії є протокол керування пристроями. Цей протокол визначає команди та повідомлення, якими пристрої можуть обмінюватися для контролю та управління один одним. Протокол управління може включати такі функції, як увімкнення і вимкнення пристроїв, регулювання параметрів, отримання статусу та інші. Важливо визначити стандартний набір команд і повідомлень, щоб забезпечити сумісність і взаємодію між різними пристроями.

Безпека є важливим аспектом у розумних будинках, тому необхідно реалізувати протокол безпеки для захисту даних і запобігання несанкціонованому доступу до пристроїв і систем. Протокол безпеки може включати такі функції, як автентифікація користувачів, шифрування даних, контроль доступу та моніторинг активності. Важливо вибрати надійні алгоритми та методи безпеки, які відповідають сучасним стандартам і вимогам.

Після реалізації протоколу взаємодії необхідно провести тестування та оптимізацію системи. Тестування дає змогу перевірити роботу протоколу на різних сценаріях та умовах, виявити можливі проблеми та помилки, і внести відповідні виправлення. Оптимізація спрямована на поліпшення продуктивності, ефективності та надійності протоколу. Важливо проводити тестування та оптимізацію на всіх етапах розробки й уважно відстежувати зворотний зв'язок користувачів для подальшого вдосконалення протоколу.

Реалізація протоколу взаємодії між пристроями є невід'ємною частиною розробки системи розумного будинку. Правильний вибір протоколу, реалізація

комунікаційного шару, протоколу керування пристроями, протоколу безпеки, а також тестування та оптимізація, забезпечують ефективну та надійну взаємодію між пристроями. Розробники систем розумного будинку повинні приділити достатню увагу цьому аспекту, щоб створити інноваційні та функціональні рішення, які задовольняють потреби користувачів і забезпечують комфорт, безпеку та енергоефективність у їхніх будинках.

Для обміну даними між різними пристроями системи необхідно розробити протокол.

Вимоги до протоколу:

- необхідно, щоб передача даних відбувалася не окремими значеннями, а набором величин, що описуватимуть, від якого модуля прийшли дані, якому модулю призначаються, що з ними робити і, власне, самі дані (формувати пакет даних);

- при всьому вищеописаному бажано, щоб розмір одного пакета був мінімальним. Цим ми зменшимо кількість пошкоджених пакетів під час передачі;

Також, використовуючи пакети ми можемо вказувати їхню контрольну суму, що дозволить легко обчислити пошкоджений пакет і попросити віддалений пристрій повторити його передачу.

Крім того, можна шифрувати дані під час передачі для підвищення безпеки.

Для кожного типу пристроїв використовується система команд. Таким чином, один і той же номер команди для різних пристроїв може мати різне значення.

Після реалізації протоколу взаємодії необхідно провести тестування та оптимізацію системи. Тестування дає змогу перевірити роботу протоколу на різних сценаріях та умовах, виявити можливі проблеми та помилки, і внести відповідні виправлення. Оптимізація спрямована на поліпшення продуктивності, ефективності та надійності протоколу. Важливо проводити тестування та оптимізацію на всіх етапах розробки й уважно відстежувати зворотний зв'язок користувачів для подальшого вдосконалення протоколу.

Таблиця 2.1 - Структура пакету

Назва	Тип поля	Опис
device_id	unsigned int8	Номер пристрою, з якого відправляється пакет. максимальне кількість пристроїв у системі - 256
destination_id	unsigned int8	Номер пристрою-одержувача
packet_id	unsigned int16	Ідентифікатор пакету. У нього записується випадкове число яке ідентифікує пакет як унікальний
command	unsigned int8	Номер команди. Інструкція, що робити з даними пакета

Для кожного типу пристроїв використовується система команд. Таким чином, один і той же номер команди для різних пристроїв може мати різне значення.

Протокол взаємодії RaspberryPi з вимикачами:

Включити світло:

command = 1

Вимкнути світло:

command = 0

Протокол взаємодії RaspberryPi із шторами/жалюзі:

Відкрити штори:

command = 1

Закрити штори:

command = 0

Протокол взаємодії датчиків протікання/задимлення з RaspberryPi:

Відбулася виняткова ситуація (задимлення або протікання):

command = 1

Виняткова ситуація більше не спостерігається:

command = 0

2.3 Опис алгоритму шифрування

Системи розумного будинку стають дедалі популярнішими, пропонуючи зручність, безпеку та енергоефективність. Однак, зі зростанням функціональності і підключених пристроїв, виникає питання про захист конфіденційності та безпеки даних, що передаються і зберігаються в розумному будинку. Один із важливих аспектів безпеки в розумних будинках - це використання алгоритмів шифрування для захисту даних від несанкціонованого доступу. У цій статті буде представлено опис алгоритму шифрування, який може бути застосований в розумних будинках для забезпечення безпеки даних.

Цілі шифрування в розумному будинку. Основна мета використання алгоритмів шифрування в розумних будинках - це забезпечення конфіденційності даних, що передаються між пристроями, і захист від несанкціонованого доступу до інформації. Шифрування дає змогу перетворити дані в зашифрований вигляд, який можна зрозуміти і розшифрувати тільки за допомогою правильного ключа. Це дає змогу запобігти можливості перехоплення та читання даних третіми особами.

Основні алгоритми шифрування. У розумних будинках можна використовувати різні алгоритми шифрування для захисту даних. Деякі з найпоширеніших алгоритмів шифрування включають в себе.

Алгоритм Advanced Encryption Standard (AES): Цей алгоритм шифрування використовує симетричний ключ для зашифрування і розшифрування даних. AES

є одним із найнадійніших алгоритмів і широко застосовується в різних системах безпеки.

RSA: RSA - це асиметричний алгоритм шифрування, який використовує пару ключів, що складається з публічного і приватного ключів. Публічний ключ використовується для шифрування даних, а приватний ключ використовується для розшифрування. RSA забезпечує високий ступінь безпеки, але може бути менш ефективним під час обробки великих обсягів даних.

Elliptic Curve Cryptography (ECC): ECC - це алгоритм шифрування, заснований на математичній теорії еліптичних кривих. Він забезпечує високий ступінь безпеки під час використання коротших ключів порівняно з іншими алгоритмами, що робить його ефективним для пристроїв з обмеженими обчислювальними ресурсами.

Процес шифрування в розумному будинку зазвичай включає такі етапи. Генерація ключів: На початку процесу необхідно згенерувати ключі для шифрування і розшифрування даних. Ключі можуть бути симетричними (однаковими для шифрування і розшифрування) або асиметричними (відмінними для шифрування і розшифрування).

Шифрування даних: Дані, які потрібно захистити, шифруються з використанням обраного алгоритму шифрування і ключа. Шифрування перетворює дані в незрозумілий вигляд, який не може бути прочитаний без відповідного ключа.

Розшифрування даних: Для отримання вихідних даних, зашифровані дані мають бути розшифровані з використанням правильного ключа. Розшифровка перетворює дані назад у вихідний вигляд, який може бути використаний для подальшої обробки або відображення.

Управління ключами є важливим аспектом алгоритму шифрування в розумних будинках. Ключі повинні бути надійно зберігатися і передаватися тільки між довіреними пристроями. Крім того, може бути необхідний механізм обміну ключами для встановлення безпечного каналу зв'язку між пристроями. Можливі різні методи обміну ключами, включно з попереднім обміном ключами,

використанням асиметричних алгоритмів для передавання симетричних ключів та інші.

Реалізація алгоритму шифрування для розумного будинку включає розробку програмного забезпечення або інтеграцію готових рішень. Важливо вибрати відповідну реалізацію, яка відповідає вимогам системи розумного будинку, забезпечує високий ступінь безпеки і сумісна з іншими пристроями і системами.

Алгоритми шифрування відіграють важливу роль у забезпеченні безпеки даних у системах розумного будинку. Вони дають змогу захистити конфіденційність даних і запобігти несанкціонованому доступу. Вибір і реалізація відповідного алгоритму шифрування, управління ключами та обмін ключами - це важливі аспекти розробки безпечних систем розумного будинку. Розробники та виробники розумних будинків повинні приділяти належну увагу цим аспектам і прагнути до створення надійних і безпечних рішень для користувачів розумних будинків.

Розглянемо докладніше алгоритм шифрування. Блоковий шифр з 256-бітним ключем і 32 циклами перетворення, оперуючий 64-бітними блоками.

Основа алгоритму шифру – мережа Фейстеля.

Для зашифровування у цьому режимі 64-бітний блок відкритого тексту спочатку розбивається на дві половини (молодші біти – А, старші біти – В). На і-му циклі використовується підключення K_i :

$$A_{i+1} = B_i \oplus f(A_i, K_i)$$

$$B_{i+1} = A_i$$

Для генерації підключень вихідний 256-бітний ключ розбивається на вісім 32-бітових блоків: $K_1 \dots K_8$.

Ключі $K_9 \dots K_{24}$ є циклічним $K_1 \dots K_8$ є повторення ключі $K_{25} \dots K_{32}$ є ключами $K_8 \dots K_1$.

Після виконання всіх 32 раундів алгоритму, блоки A_{33} та B_{33} склеюються (старшим блоком стає A_{33} , а молодшим – B_{33}) – результат є результатом роботи алгоритму.

Розшифрування виконується так само, як і зашифрування, але інвертується порядок підключень K_i .

Функція $f(A_i, K_i)$ обчислюється так:

A_i та K_i складаються за модулем 232. Результат розбивається на вісім 4-бітових підпоследовностей, кожна з яких надходить на вхід свого вузла таблиці заміни (у порядку зростання старшинства бітів), званого нижче S-блоком. Загальна кількість S-блоків алгоритму - вісім, тобто стільки ж, скільки підпоследовностей. Кожен S-блок є перестановкою чисел від 0 до 15 (конкретний вид S-блоків у стандарті не визначено). Перша 4-бітна підпоследовність попадає на вхід першого S-блоку, друга - на вхід другого і т.д.

Якщо вузол S-блоку виглядає так:

1, 15, 13, 0, 5, 7, 10, 4, 9, 2, 3, 14, 6, 11, 8, 12

на вході S-блоку 0, то на виході буде 1, якщо 4, то на виході буде 5, якщо на вході 12, то на виході 6 і т.д.

Виходи всіх восьми S-блоків поєднуються в 32-бітове слово, потім все слово циклічно зсувається вліво (до старших розрядів) на 11 бітів.

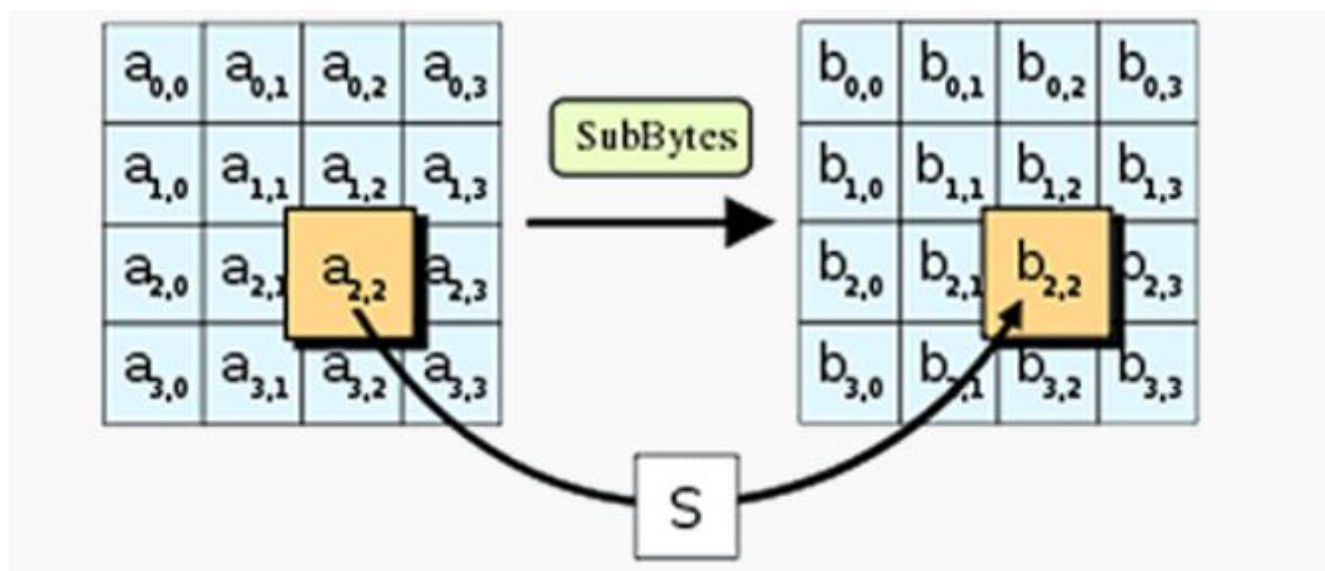


Рисунок 2.1 – Принцип роботи шифруючого алгоритму

2.4 Реалізація для RaspberryPi

Системи розумного будинку стають дедалі популярнішими, пропонуючи користувачам зручність, комфорт, безпеку та енергоефективність. Raspberry Pi - це одноплатний комп'ютер, який можна використовувати як основу для розробки та реалізації систем розумного будинку. У цій статті розглянемо, як можна реалізувати розумний будинок з використанням Raspberry Pi, а також основні компоненти і функціональність, які можна включити в таку систему.

Raspberry Pi є чудовим вибором для реалізації розумного будинку з кількох причин. По-перше, він є доступним і недорогим комп'ютером, що робить його доступним для широкого кола користувачів. По-друге, Raspberry Pi має достатню обчислювальну потужність і гнучкість для роботи з різними датчиками, пристроями та інтерфейсами, необхідними для розумного будинку. По-третє, Raspberry Pi має велике співтовариство розробників та ентузіастів, що забезпечує підтримку та ресурси для розроблення та реалізації проєктів розумного будинку.

Основні компоненти розумного будинку на Raspberry Pi. Розумний будинок на Raspberry Pi може містити такі основні компоненти. Управління освітленням: З використанням Raspberry Pi і відповідних модулів реле або силових ключів, можна реалізувати управління освітленням у будинку. Це може включати в себе автоматичне увімкнення і вимкнення освітлення за розкладом, контроль освітлення через мобільний додаток і навіть інтелектуальне освітлення, яке реагує на наявність людей або зміну освітленості.

Керування опаленням і кондиціонуванням повітря: Raspberry Pi можна використовувати для керування системами опалення та кондиціонування повітря в будинку. Це дає змогу автоматизувати регулювання температури в різних зонах будинку, керувати роботою опалювальних і кондиціонуючих пристроїв з будь-якої точки світу через інтернет і створювати розклади роботи систем.

Розумні розетки та керування електроприладами: За допомогою Raspberry Pi можна створити розумні розетки, які дадуть змогу керувати електроприладами з будь-якої точки будинку або віддалено через інтернет. Це дає змогу вмикати і вимикати прилади, регулювати потужність і навіть моніторити їхнє енергоспоживання.

Безпека і відеоспостереження: Raspberry Pi може бути використаний для реалізації системи безпеки і відеоспостереження в розумному будинку. За допомогою камер і датчиків руху можна створити систему моніторингу будинку, яка надсилатиме сповіщення на мобільні пристрої в разі виявлення руху або інших неправильних ситуацій.

Розумний датчик довкілля: Raspberry Pi можна використовувати для збору даних про погоду, якість повітря, рівень шуму та інші параметри довкілля. Ці дані можуть бути використані для ухвалення рішень щодо автоматичного керування будинком, наприклад, увімкнення/вимкнення систем вентиляції залежно від якості повітря або керування автоматичним поливом саду залежно від погоди.

Програмне забезпечення для розумного будинку на Raspberry Pi. Для реалізації розумного будинку на Raspberry Pi необхідно використовувати відповідне програмне забезпечення. Існує кілька платформ і фреймворків, які можна використовувати для розроблення застосунків і керування розумним будинком на Raspberry Pi, включно з Home Assistant, OpenHAB, Domoticz та іншими. Ці платформи забезпечують зручний інтерфейс для керування пристроями, налаштування автоматизації та моніторингу системи розумного будинку.

Інтеграція з іншими пристроями та системами. Розумний будинок на Raspberry Pi може бути інтегрований з іншими пристроями та системами, як-от мобільні додатки, голосові помічники (наприклад, Amazon Alexa або Google Assistant), системи розумного міста та інші. Це дає змогу користувачам керувати розумним будинком через різні інтерфейси та розширює функціональність і можливості системи.

Реалізація розумного будинку на Raspberry Pi пропонує широкий спектр можливостей для автоматизації та контролю різних аспектів домашнього середовища. Raspberry Pi є доступною і гнучкою платформою, яка може бути налаштована і розширена за допомогою різних модулів і датчиків. За умови правильного розроблення та інтеграції з програмним забезпеченням, розумний дім на Raspberry Pi може стати сучасним та інтелектуальним рішенням для підвищення комфорту, безпеки та енергоефективності в домашньому середовищі.

Опис веб-частини системи. Веб-частина системи, що здійснює взаємодію користувача і системи, реалізована на принципах концепції MVC – модель-вид-контролер. MVC це схема використання декількох шаблонів проектування, з допомогою яких модель програми, користувацький інтерфейс і взаємодія з користувачем розділені на три окремі компоненти таким чином, щоб модифікація одного з компонентів мала мінімальний вплив на інші.

Всі запити HTTP обробляються класом `RequestParser`. Цей клас виконує розбір запитів та передачу керування контролерам для подальшого виконання.

`IndexController` – контролер, який забезпечує роботу головної сторінки веб-інтерфейсу. Він генерує сторінку з описом всіх пристроїв підключених до систем та з можливістю керування ними.

`BlindsController` - контролер, що відповідає за обробку запитів, пов'язаних зі шторами (відкрити/закрити та отримати список пристроїв - штори або жалюзі). Цей контролер обробляє лише аїах-запити та повертає результати у вигляді json-рядків.

`SensorController` - контролер, що відповідає за обробку запитів, пов'язаних датчиками задимлення та протікання. Повертає json-рядок зі списком всіх датчиків та їх станами для подальшого відображення у веб-інтерфейсі.

`LightController` - контролер, що відповідає за обробку запитів, пов'язаних з керуванням світловими приладами. Приймає аїах-запити на увімкнення/вимкнення світлових приладів. Дозволяє повернути json-рядок зі списком всіх світлових приладів та їх станами – увімкнено або вимкнено.

IRController - контролер, що відповідає за обробку запитів, пов'язаних з IR-пристроями. Приймає аїах-запити з кодами кнопок іі-пультів для керування телевізорами, акустичними системами та іншою технікою з можливістю керування через іі-приймач.

Окремо варто виділити контролер AdminController, у ньому реалізовано адміністрування системи. Він дозволяє здійснювати авторизацію користувачів та редагувати пристрої, підключені до системи.

За створення сторінок з html-шаблонів відповідає клас View. Він отримує на вхід назву шаблону та список параметрів, необхідних для створення сторінки[12].

Класи-моделі системи: Blinds, Light, Sensor та User реалізують шаблон проектування ActiveRecord. Ці класи використовуються для отримання та збереження даних у таблицях з відповідними назвами.

Для надсилання повідомлень бездротовим пристроям реалізовано клас WirelessRemote. Він являє собою обгортку над утилітою send, описаної нижче.

Для надсилання повідомлень іі-пристроєм реалізовано клас IRRemote. Він являє собою обгортку над утилітою irsend із пакета LIRC.

Опис бази даних. На рисунку 2.2 продемонстровано ER-модель бази даних системи автоматизації житлових приміщень. База даних використовується для зберігання облікових записів користувачів, які мають доступ до системи та інформацію про поточні стани пристроїв, підключених до системи.

З більшістю пристроїв немає зворотного зв'язку, таким чином для того, щоб завжди знати, в якому стані вони зараз знаходяться (ввімкнено/вимкнено), необхідно запам'ятовувати цей стан відразу після його зміни. Також таблиця sensor використовується для зворотного зв'язку між датчиками затоплення/задимлення та веб-інтерфейсом. Так фоновий сервіс, що слухає повідомлення від датчиків, оновлює їх стани в таблиці sensor, а PHP-скрипт періодично читає ці значення і відображає в веб-інтерфейсі.

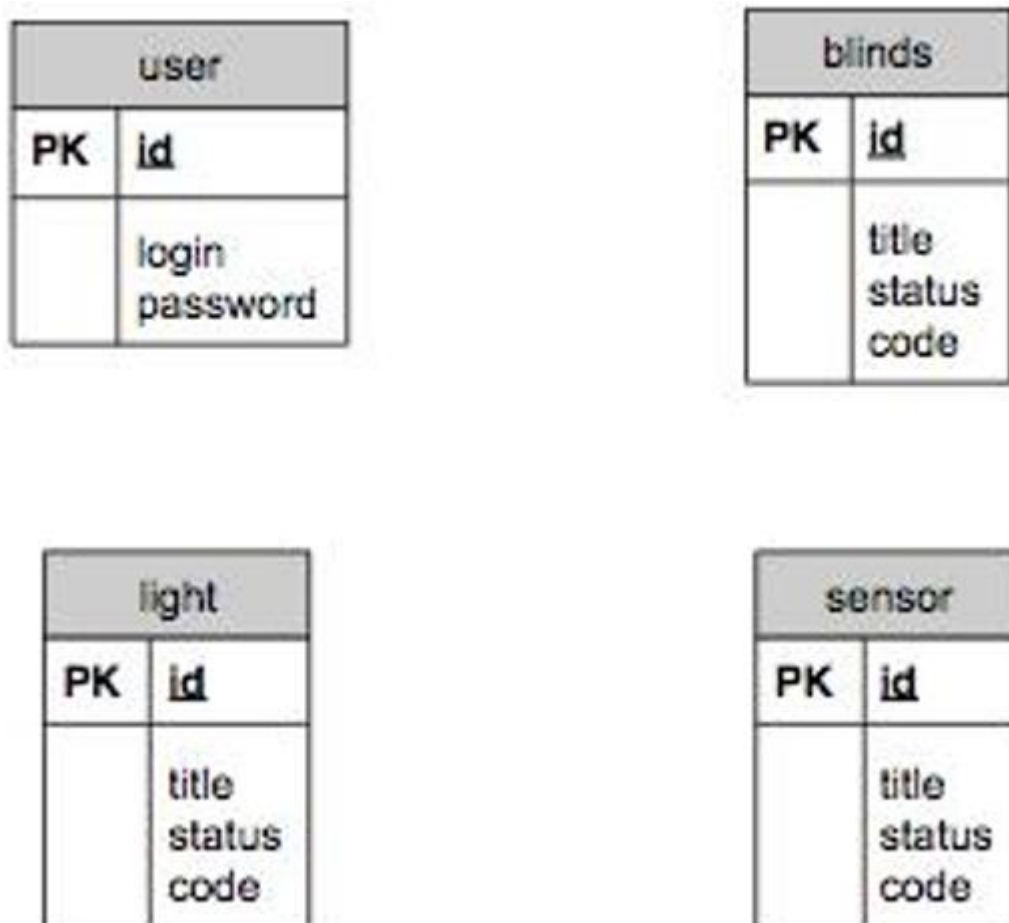


Рисунок 2.2 - Модель сутність-зв'язок бази даних системи автоматизації житлових приміщень

Утиліта Send. Для надсилання повідомлень на вимикачі та реле (штори/жалюзі, приводи водопровідних кранів та замків, інші сервоприводи) використовується консольна утиліта send, написана мовою C++.

RaspberryPi є апаратним інтерфейсом GPIO, до якого здійснюється підключення бездротового контролера 433MHz.

Акронім GPIO розшифровується як General Purpose Input Output. Це низькорівневий інтерфейс введення-виводу прямого керування. Через цей інтерфейс Raspberry може слухати та віддавати команди будь-якому зовнішньому пристрої.

Щоб можна було з програми працювати з пінами GPIO, використовується стороння бібліотека WiringPi.

Ця бібліотека виконує прив'язку пінів GPIO до змінних типу `int` мови C++. Це дозволяє отримувати повідомлення з мережі простою перевіркою значень змінних. Надсилання повідомлень до мережі виконується так само за допомогою подачі сигналів на піни GPIO через зміну значень відповідних змінних через певні проміжки часу (модуляція). Модуляція даних здійснюється через тимчасові затримки у зміні рівня напруги на пінах GPIO

Витяг з коду:

```

– подати напругу: digitalWrite(this->nTransmitterPin, HIGH);
– зробити затримку на nHighPulses мікросекунд:
delayMicroseconds(nHighPulses);
– зняти напругу: digitalWrite(this->nTransmitterPin, LOW);
– зробити затримку на nLowPulses мікросекунд:
delayMicroseconds(nLowPulses).
```

Утиліта `send` приймає такі параметри:

- `--code` – код пристрою, для якого призначено повідомлення;
- `--command` - командне повідомлення, яке буде передано на пристрій.

Загальна логіка роботи утиліти `send` полягає в наступному:

- зберігаємо на вхід код пристрою та команду, призначену для цього пристрою;
- передаємо ці параметри методу `send()` бібліотеки `RCSwitch`, яка, у свою чергу, кодує це повідомлення для передачі по широкомовному бездротовому протоколу.

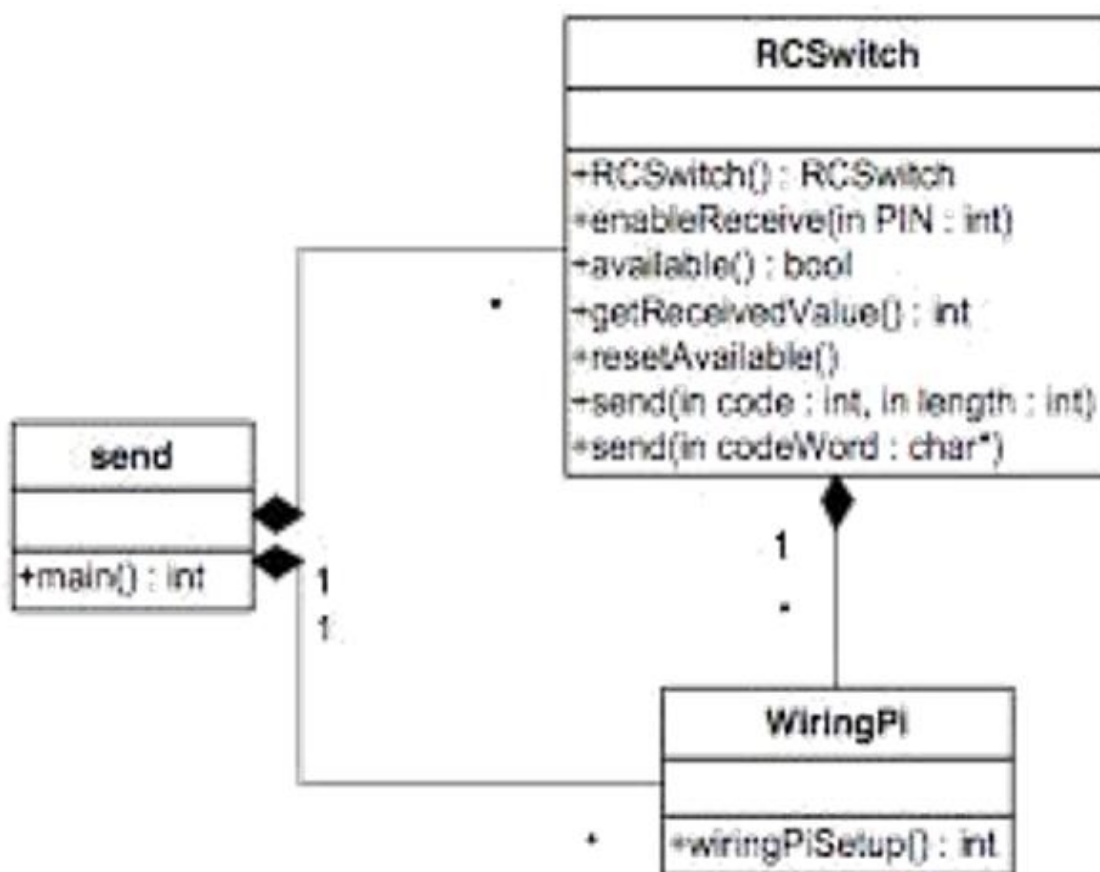


Рисунок 2.3 - діаграма класів утиліти send

IRSend. Для управління пристроями з IR-приймачем використовується утиліта irsend, реалізована у проекті LIRC GNU. Irsend працює з інфрачервоним передавачем, використовуючи який відправляє команди для керування електроприладами, такими як телевізори, акустичні системи тощо.

Слухаючий фоновий сервіс .

Для отримання повідомлень від датчиків задимлення та протікання використовується фоновий сервіс, що слухає зміни на GPIO інтерфейсі RaspberryPi.

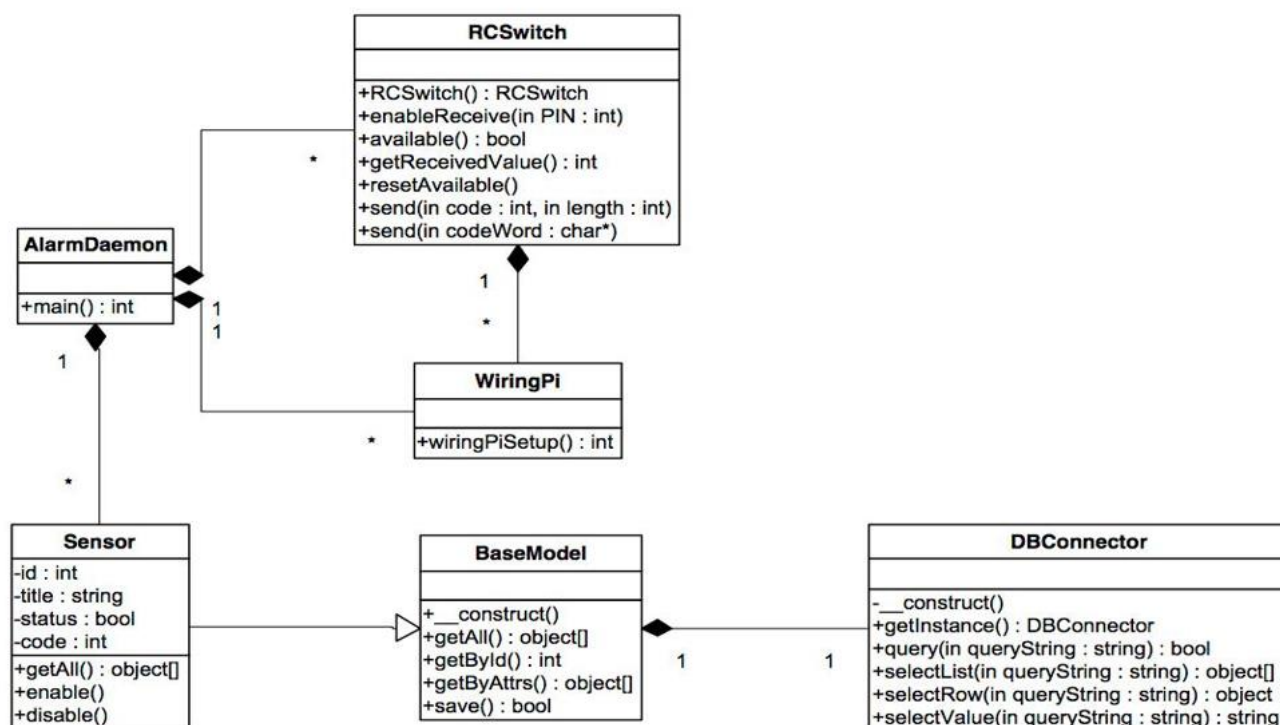


Рисунок 2.4 – діаграма класів фонового сервісу

На рисунку 2.4 продемонстровано діаграму класів фонової служби, що працює на апаратній платформі RaspberryPi. Клас RCSwitch відповідає за прийом та первинну обробку повідомлень, отриманих приймачем бездротового зв'язку. Клас WiringPi організовує роботу зі сторонньою бібліотекою WiringPi та виконує первинне налаштування обладнання [10]. Він виконує прив'язку пінів GPIO до змінних типу int. Це дозволяє отримувати повідомлення з мережі простою перевіркою значень змінних. Класи Sensor, BaseModel та забезпечують збереження отриманих даних у базу даних, реалізуючи шаблон проектування active record. DBConnector - клас-одинак, що виконує безпосереднє підключення до бази даних.

Алгоритм роботи сервісу можна описати наступним ланцюжком дій:

- датчик протікання або задимлення надсилає команду, що сталася виняткова ситуація;
- у функції main файлу AlarmDaemon.cpp виконується нескінченний цикл, в якому перевіряється наявність нових повідомлень, що надійшли з GPIO;
- якщо надійшло повідомлення про аварійну ситуацію з будь-яким датчиком, цей стан датчика зберігається в базі даних в таблиці sensor;

– якщо пізніше надійшло повідомлення від датчика про припинення аварійної ситуації, статус цього датчика також оновиться в таблиці sensor.

Веб-інтерфейс працює з цією ж базою даних та виконує періодичну перевірку статусів датчиків. Якщо один із датчиків має статус “аварійна ситуація”, це відобразиться у веб-інтерфейсі для інформування користувача.

2.5 Реалізація мікроконтролера

Передавач для датчиків задимлення/протечки. Як платформа для датчиків використовується Arduino Uno, з підключеним по serial port сенсором. У безкінечному циклі виконується опитування сенсора, якщо сталася аварійна ситуація, то формується повідомлення для утиліти RCSwitch, яка виконує надсилання цього повідомлення бездротовим протоколом для RaspberryPi.

Приймач для сервоприводу. Також у нескінченному циклі отримуємо повідомлення від RCSwitch, отримані команди передаємо на сервоприводи.

Приймач для вимикача. Також у нескінченному циклі отримуємо повідомлення від RCSwitch, отримані команди передаємо вимикачу.

Опис реалізації програми оповіщення про задимлення та протікання для мікроконтролера

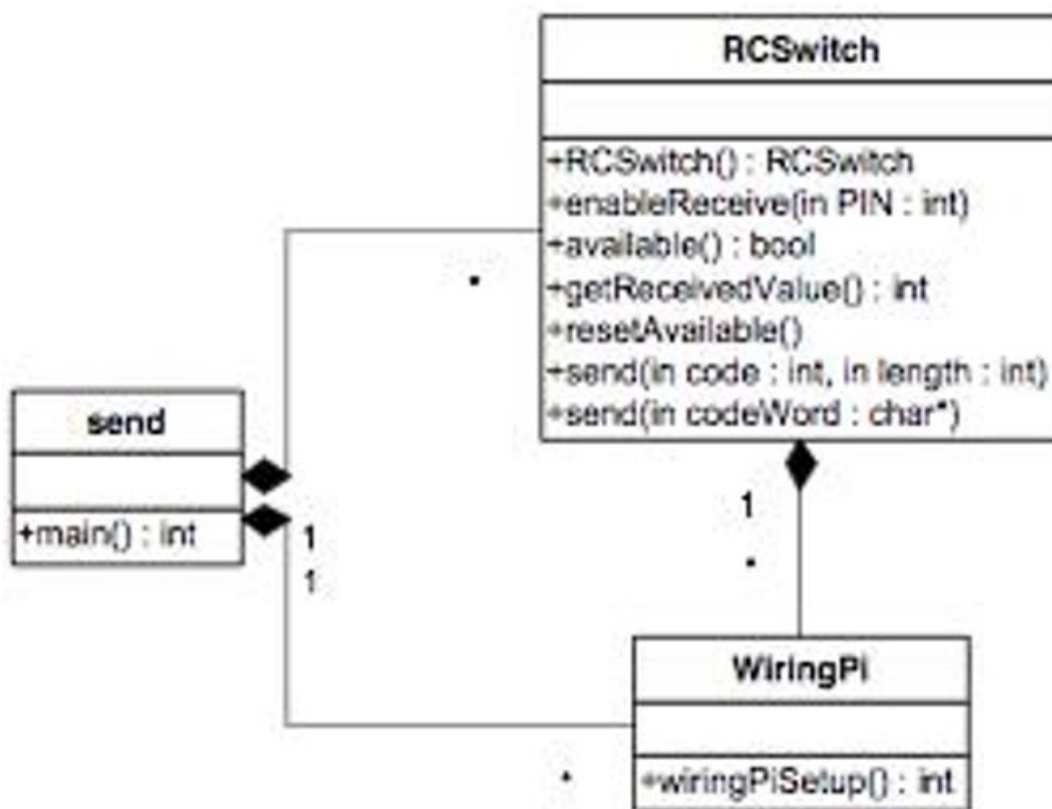


Рисунок 2.5 - діаграма класів програми оповіщення про задимлення та протікання для Arduino Uno

На рисунку 2.5 наведена діаграма основних класів програми, що працює на мікроконтролер Arduino Uno. Клас **RCSwitch** відповідає безпосередньо за прийом та передачу даних. Клас **WiringPi** організовує роботу зі сторонньою бібліотекою **WiringPi** та виконує первинне налаштування обладнання.

send - точка входу програми, єдиний метод `main` виконує ініціалізацію програми та запуск нескінченного циклу опитування датчиків протікання та задимлення.

RCSwitch - реалізує прийом та дешифрацію повідомлень.

- `enableReceive` - метод, що перемикає бібліотеку **RCSwitch** на прийом даних;
- `available` – метод, що перевіряє наявність нового необробленого повідомлення;
- `getReceivedValue` - метод, що повертає отримане повідомлення;

– `resetAvailable` - метод, що перемикає програму в стан очікування отримання нового повідомлення;

– `send` - метод, який би надсилення повідомлення з RaspberryPi.

Повідомлення може бути закодовано як однією змінною типу `long`, і масивом символів `char[]`

`WiringPi` - виконує прив'язку пінів GPIO до змінних типу `int`. Це дозволяє отримувати повідомлення з мережі простою перевіркою значень змінних.

– `wiringPiSetup` – метод, що виконує налаштування інтерфейсу GPIO.

3 ЗАГАЛЬНЕ ФУНКЦІОНУВАННЯ СИСТЕМИ

3.1 Опис реалізації програми

Опис реалізації програми, що обробляє повідомлення, отримані від датчиків задимлення та протікання

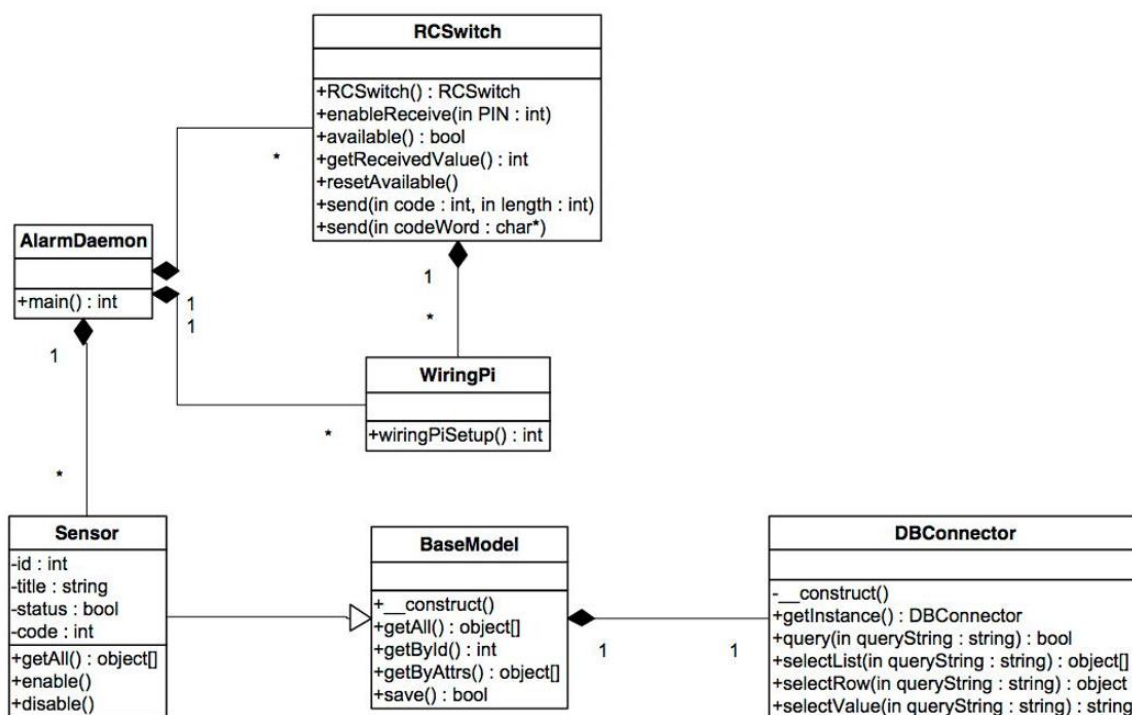


Рисунок 3.1 - Діаграма класів програми

На рисунку 3.1 продемонстровано діаграму класів фонові служби, що працює на апаратній платформі RaspberryPi. Клас RCSwitch відповідає за прийом та первинну обробку повідомлень, отриманих приймачем бездротового зв'язку. Клас WiringPi організовує роботу зі сторонньою бібліотекою WiringPi та виконує первинне налаштування обладнання. Класи Sensor, BaseModel та DBConnector забезпечують збереження отриманих даних у базі даних.

AlarmDaemon - точка входу програми, єдиний метод main виконує ініціалізацію програми та запуск нескінченного циклу прослуховування повідомлень від датчиків протікання та задимлення.

WiringPi - виконує прив'язку пінів GPIO до змінних типу int. Це дозволяє отримувати повідомлення з мережі простою перевіркою значень змінних.

- wiringPiSetup - метод, що виконує налаштування інтерфейсу GPIO
- RCSwitch - реалізує прийом та дешифрацію повідомлень.

- enableReceive - метод, що перемикає бібліотеку RCSwitch на прийом даних;

- available – метод, що перевіряє наявність нового необробленого повідомлення;

- getReceivedValue - метод, що повертає отримане повідомлення;

- resetAvailable - метод, що перемикає програму в стан очікування отримання нового повідомлення;

- send - метод, який би надсилення повідомлення з RaspberryPi.

BaseModel – базовий клас для класів-моделей, що реалізують роботу з базою даних через шаблон проектування – ActiveRecord:

- __construct - метод, який ініціалізує підключення до бази даних через клас DBConnector;

- getAll - метод, який повертає список усіх записів;

- getById - метод, який повертає запис для зазначеного ID;

- getByAttrs - метод, що повертає список записів, що відповідають параметрам пошуку;

- save - спосіб, реалізує збереження об'єкта до бази даних. Якщо в об'єкта

заданий первинний ключ, то запис оновиться, якщо первинний ключ не заданий, буде створено новий запис.

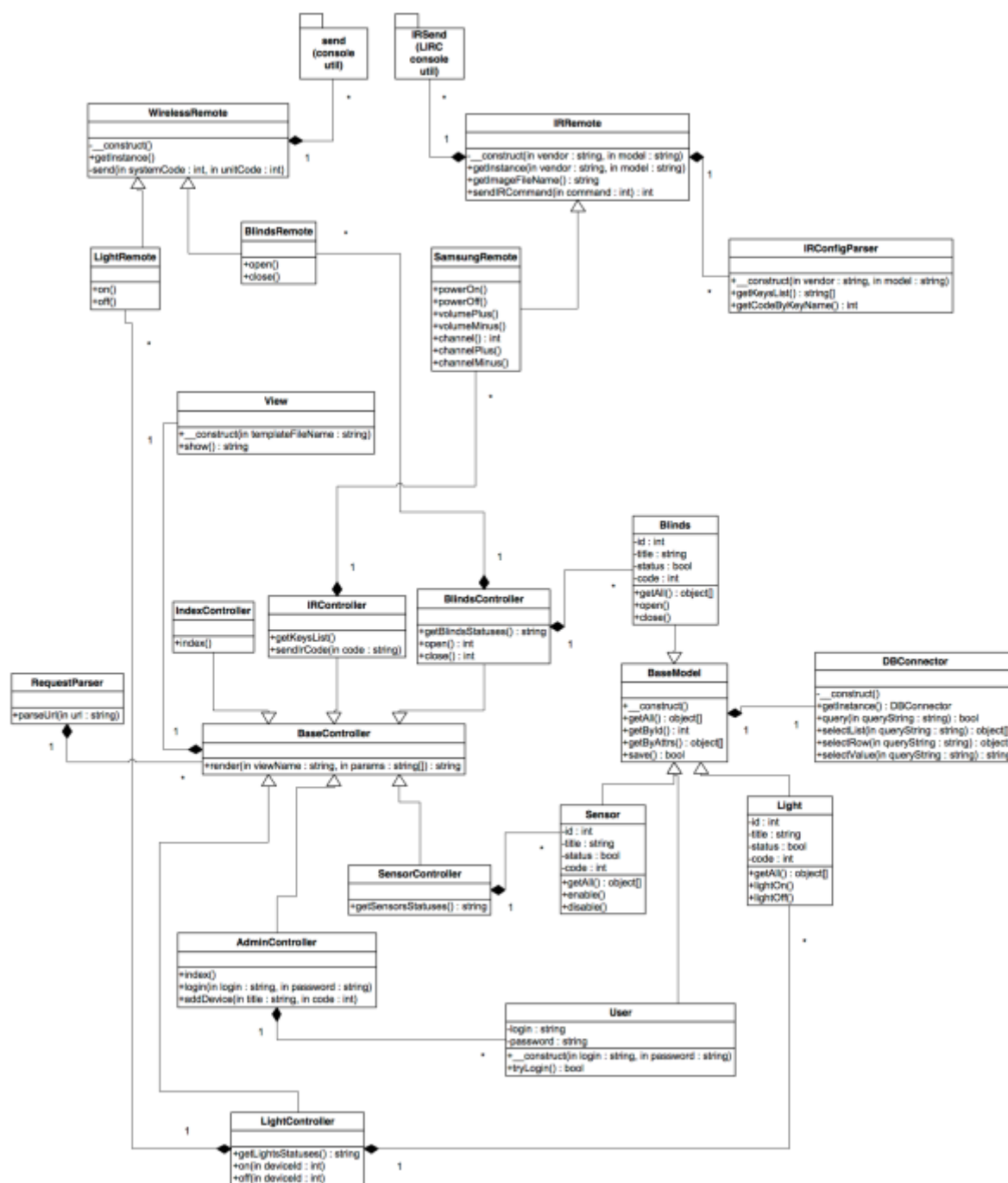
Sensor – дочірній клас BaseModel, забезпечує роботу з таблицею sensor:

- id - атрибут, що містить ідентифікатор запису з таблиці sensor;
- title - атрибут, що містить назву датчика;
- status - атрибут, що містить статус датчика (у стані тривоги або спокою);
- code - атрибут, що містить код датчика, що використовується для ідентифікації пристрою в протоколі;
- getAll - метод, що повертає список всіх датчиків з таблиці sensor;
- enable - метод, що позначає датчик як включений (в стані тривоги);
- disable - метод, що позначає датчик як вимкнений (у стані спокою).

DBConnector - клас, який забезпечує підключення до бази даних, реалізує шаблон проектування - одинак.

- __construct - метод, який ініціалізує підключення до бази даних;
- getInstance - метод, що повертає екземпляр даного класу;
- query - метод, який виконує SQL запит на вставку або оновлення;
- selectList - метод, що повертає список записів, що задовольняють параметрам пошуку;
- selectRow - метод, що повертає запис, що задовольняє параметрам пошуку;
- selectValue - метод, що повертає єдине значення, що задовольняє параметрам пошуку та зазначеному стовпцю.

Діаграма класів системи автоматизації житлових приміщень



Рисунку 3.2 – Діаграма класів системи автоматизації житлових приміщень

На рисунку 3.2 продемонстровано діаграму класів системи автоматизації житлових приміщень.

Систему можна розділити на 3 основні складові: веб-інтерфейс, СУБД та інтерфейси взаємодії з бездротовими пристроями.

Роботу веб-інтерфейсу забезпечують такі класи:

RequestParser - клас, що виконує обробку та маршрутизацію http-запитів.

- `parseUrl` - метод, виконує розбір `http`-запиту та викликає відповідний метод зазначеного у запиті контролера.

`BaseController` – базовий клас для контролерів.

- `render` - метод, що виконує рендеринг сторінки з передачею всіх необхідних змінних у `html`-шаблон;

- `IndexController` – клас, що реалізує роботу головної сторінки веб-інтерфейсу.

- `index` – метод, що виконує генерацію головної сторінки.

`IRController` - клас, що відповідає за обробку `http` запитів, пов'язаних з `IR`-пристроями.

- `getKeysList` - метод, що повертає список всіх допустимих кнопок пульта дистанційного керування;

- `sendIrCode` - метод, що отримує на вхід код натиснутої кнопки пульта ДК, з подальшою передачею його `ir`-приймачу.

`BlindsController` - клас, що відповідає за обробку `http` запитів, пов'язаних зі шторами:

- `getBlindsStatuses` - метод, що повертає список штор та жалюзі, керованих системою, з їх поточним статусом - відкритий/закритий;

- `open` - метод, який обробляє запит на відкриття штор;

- `close` – метод, що обробляє запит на закриття штор; `SensorController` - клас, що відповідає за обробку `http`-запитів, пов'язаних з

- датчиками задимлення та протікання.

- `getSensorsStatuses` - метод, що повертає список датчиків з їх поточним статусом - активний/неактивний.

`LightController` - клас, що відповідає за обробку `http`-запитів, пов'язаних з керуванням світловими приладами.

- `getLightsStatuses` - метод, що повертає список світлових приладів з їх поточним статусом - увімкнено/вимкнено;

- `on` - метод, що обробляє запит на включення приладу освітлення;

- off - метод, що обробляє запит на вимкнення приладу освітлення.

AdminController - клас, що забезпечує налаштування системи.

- index – метод, що виконує генерацію головної сторінки адміністрування;
- login - метод, що виконує генерацію сторінки з формою для логіну;
- addDevice – метод, що реалізує можливість додавання світлових приладів.

View - клас, що відповідає за генерацію веб-сторінок із html-шаблонів.

- __construct - конструктор, отримує на вхід назву шаблону;
- show - метод, що виконує генерацію сторінки із шаблону та виведення на екран.

За роботу з базою даних відповідають такі класи. DBConnector - клас, що виконує підключення та низькорівневі операції з базою даних. Реалізує шаблон проектування – одинак:

- __construct – конструктор, виконує підключення до СУБД;
- getInstance - статичний метод, який повертає екземпляр даного класу;
- query - метод, що виконує SQL запит до бази;
- selectList - метод, що виконує SQL запит на SELECT, повертає список рядків;
- selectRow - метод, що виконує SQL запит на SELECT, повертає один рядок;
- selectValue - метод, що виконує SQL запит на SELECT, повертає одне значення;

BaseModel - базовий клас для ActiveRecord моделей:

- __construct – конструктор;
- getAll - метод, що повертає список ActiveRecord об'єктів, що відповідають усім записам з таблиці;
- getById - метод, що повертає ActiveRecord об'єкт для зазначеного первинного ключа;
- getByAttrs - метод, що повертає список ActiveRecord об'єктів, що

задовольняють параметрам вибірки;

- save - метод, що виконує збереження ActiveRecord об'єкта в базі. Якщо первинний ключ було вказано, то запис у таблиці оновлюється, інакше - створюється новий запис.

Blinds – клас, який забезпечує роботу з таблицею blinds.

- id - атрибут, унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title - атрибут, назву пристрою;

- status - атрибут, стан пристрою - відкрито чи закрито;

- code - атрибут, код пристрою, використовується в протоколі для ідентифікації пристрою;

- getAll - метод, що повертає всі рядки з таблиці blinds;

- open - метод, що виставляє поле status = 1 для рядка із зазначеним id;

- close - метод, що виставляє поле status = 0 для рядка із зазначеним id.

Light – клас, який забезпечує роботу з таблицею light.

- id - атрибут, унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title - атрибут, назву пристрою;

- status - атрибут, стан пристрою - увімкнено або вимкнено;

- code - атрибут, код пристрою, використовується в протоколі для ідентифікації пристрою;

- getAll - метод, що повертає всі рядки з таблиці light;

- lightOn - метод, що виставляє поле status = 1 для рядка із зазначеним id;

- lightOff - метод, що виставляє поле status = 0 для рядка із зазначеним id.

Sensor – клас, який забезпечує роботу з таблицею sensor.

- id - атрибут, унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title - атрибут, назву пристрою;

- status - атрибут, стан пристрою - активоване або ні;

- code - атрибут, код пристрою, використовується в протоколі для ідентифікації пристрою;

- getAll - метод, що повертає всі рядки з таблиці sensor;

- enable - метод, що виставляє поле status = 1 для рядка із зазначеним id;

- disable - метод, що виставляє поле status = 0 для рядка із зазначеним id;

User - клас, який би роботу з таблицею user.

- login - атрибут, логін (ім'я) користувача;

- password - атрибут, пароль користувача;

- __construct - конструктор, створює об'єкт із зазначеним логіном та паролем;

- tryLogin - метод, що виконує перевірку коректності зазначеного логіну та пароля шляхом пошуку відповідного рядка в базі;

Робота з бездротовими пристроями забезпечується класами. WirelessRemote - базовий клас, що реалізує роботу бездротовими пристроями через консольну утиліту send. Реалізує шаблон проектування – одинак:

- __construct – конструктор;

- getInstance - статичний метод, який повертає екземпляр даного класу;

- send - метод, що викликає утиліту send з параметрами, які будуть

- передані бездротовим пристроям;

LightRemote - клас, що реалізує керування світловими приладами.

- on - викликати утиліту send з параметрами, необхідними включення пристрою;

- off - викликати утиліту send з параметрами, необхідними для вимкнення пристрою;

BlindsRemote – клас, що реалізує управління шторами або жалюзі.

- open - викликати утиліту send з параметрами, необхідними відкриття штор;

- close - викликати утиліту send із параметрами, необхідними для закриття штор;

Send - консольна утиліта, що виконує ретрансляцію запитів із класу WirelessRemote бездротовим пристроєм через пряму роботу з бездротовим передавачем 433MHz.

– IRRemote - базовий клас, що забезпечує роботу з пристроями, що приймають команди в інфрачервоному діапазоні. Реалізує шаблон проектування – одинак.

- __construct – конструктор;
- getInstance - статичний метод, який повертає екземпляр даного класу;
- getImageFileName - метод, що повертає ім'я файлу зображення пульта дистанційного керування, для використання у веб-інтерфейсі;

- sendIRCommand - метод, що відправляє IR-команду, використовуючи консольну утиліту IRSend з пакету LIRC.

IRConfigParser - клас, що виконує аналіз конфігураційних файлів для різних пристроїв зі складу проекту LIRC.

- __construct – конструктор;
- getKeysList - метод, що повертає список кнопок пульта ДУ для заданого пристрою;

- getCodeByKeyName - метод, що повертає код, що ідентифікує кнопку пульта дистанційного керування за її назвою.

SamsungRemote – приклад реалізації класу, що працює з пристроями конкретного виробника.

- powerOn - метод, що виконує включення телевізора;
- powerOff - метод, що виконує вимикання телевізора;
- volumePlus - метод, що виконує збільшення гучності;
- volumeMinus - метод, що виконує зменшення гучності;
- channel - метод, що перемикає канал на телевізорі на заданий;
- channelPlus - метод, що перемикає канал на наступний;
- channelMinus - метод, що перемикає канал на попередній;

3.2 Реалізація програми

IRSend - консольна утиліта, виконує ретрансляцію запитів із класу IRRemote через IR-передавач.

Модель сутність-зв'язок бази даних системи автоматизації житлових приміщень

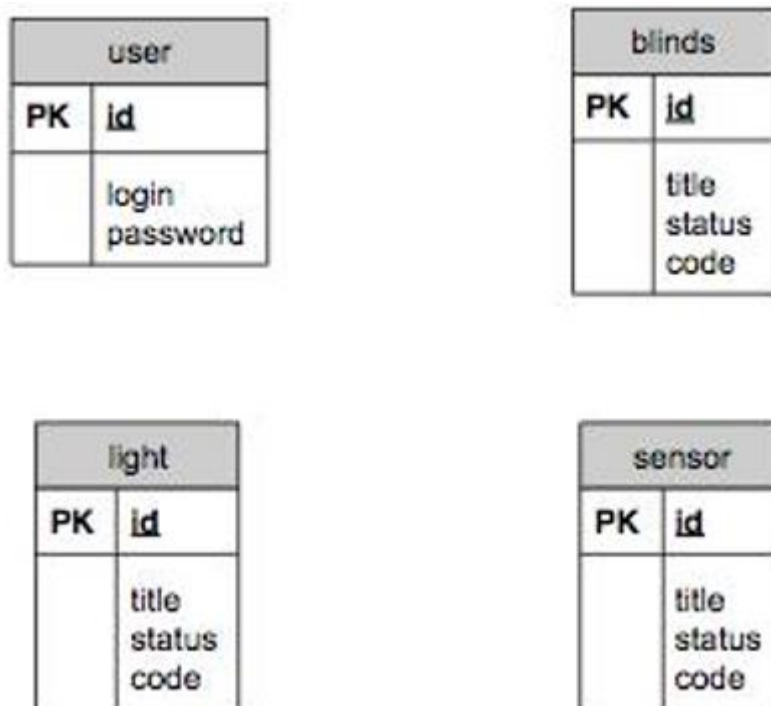


Рисунок 3.3 - модель сутність-зв'язок бази даних системи автоматизації житлових приміщень

На рисунку 3.3 продемонстровано ER-модель бази даних системи автоматизації житлових приміщень. База даних містить такі таблиці:

user - зберігає інформацію про профіль користувачів.

- id – унікальний ідентифікатор користувача, сурогатний первинний ключ;
- login – логін (ім'я) користувача;
- password – пароль користувача.

blinds - містить інформацію про штори та жалюзі, відкриття та закриття яких автоматизовано з використанням сервоприводів.

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;
- title – назва пристрою;
- status - стан пристрою - відкритий або закритий;
- code - код пристрою, який використовується в протоколі для ідентифікації пристрою.

light - містить інформацію про світлові прилади, доступні для автоматичного керування.

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;
- title – назва пристрою;
- status - стан пристрою - увімкнено або вимкнено;
- code - код пристрою, який використовується в протоколі для ідентифікації пристрою.

sensor - містить інформацію про датчики задимлення та протікання та їх стан.

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;
- title – назва пристрою;
- status - стан пристрою - активований чи ні;
- code - код пристрою, який використовується в протоколі для ідентифікації пристрою.

Схема взаємодії користувача з системою управління пристроями через IR-порт

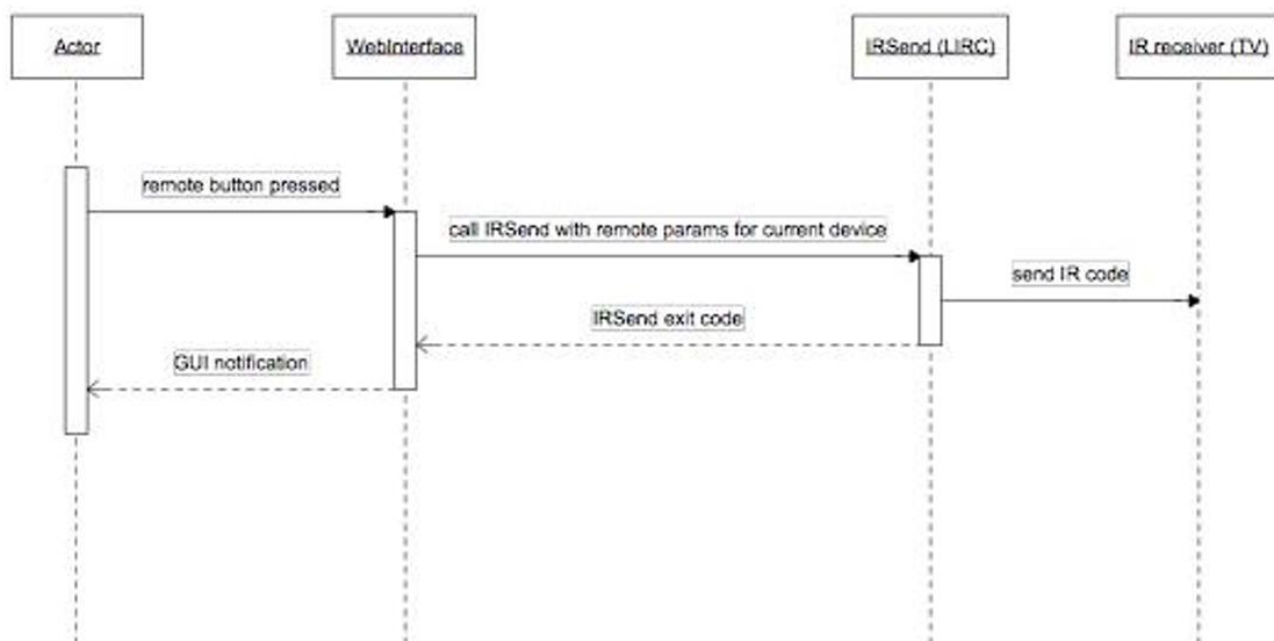


Рисунок 3.4 - діаграма послідовностей процесу взаємодії користувача із системою для керування пристроями через IR-порт

На рисунку 3.4 представлена діаграма послідовностей, що демонструє процес роботи з IR-пристроями.

Об'єкти системи:

- Actor – користувач системи.
- WebInterface - інтерфейс взаємодії користувача із системою, реалізований у вигляді сукупності web-сторінок.
- IRSender(LIRC) - модуль системи, що забезпечує відправку команд через інфрачервоний інтерфейс, використовується зі складу проекту з відкритим вихідним кодом – LIRC.
- IR receiver – приймач команд, відправлених в інфрачервоному діапазоні.

Схема взаємодії користувача із системою для керування пристроями за бездротовим протоколом на частоті 433MHz

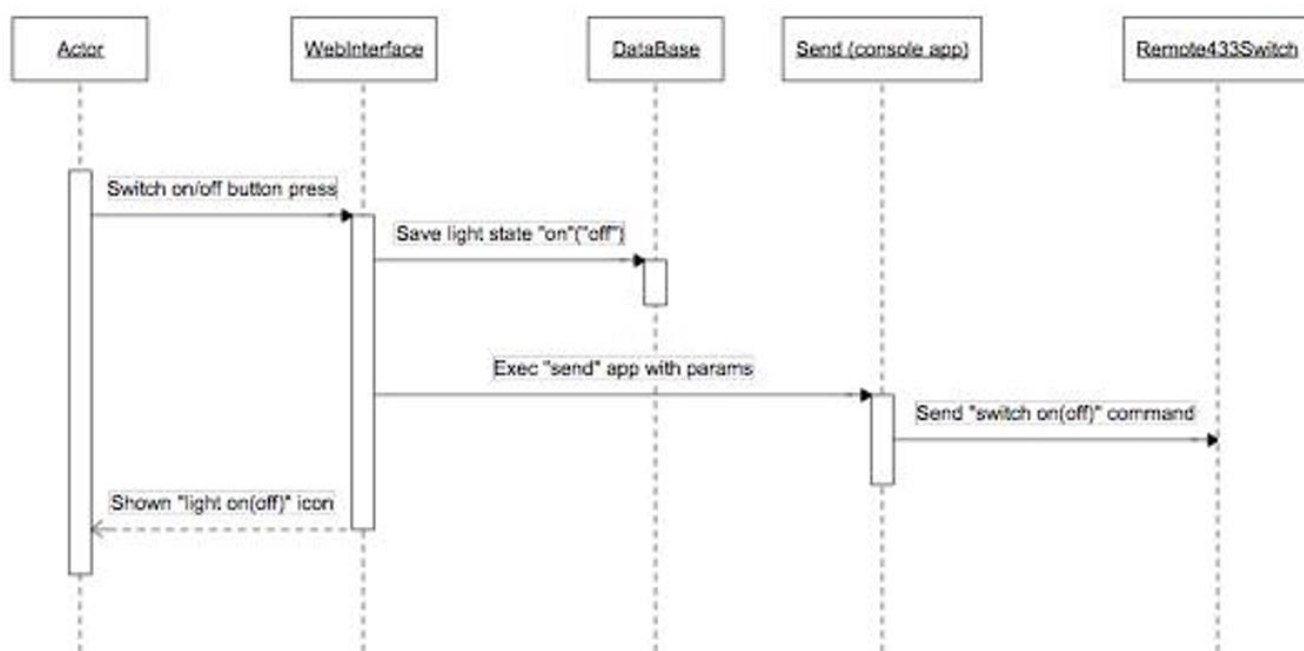


Рисунок 3.5 - Діаграма послідовностей процесу взаємодії користувача з системою для керування пристроями бездротовим протоколом на частоті 433MHz

Нарисунку 3.5 представлена діаграма послідовностей, що демонструє процес роботи з бездротовими пристроями.

Actor – користувач системи.

– WebInterface - інтерфейс взаємодії користувача із системою, реалізований у вигляді сукупності web-сторінок.

– DataBase - СУБД, що використовується для зберігання інформації про стани бездротових пристроїв.

– Send – модуль системи, що забезпечує відправлення команд на частоті 433MHz.

– Remote433Switch – приймач команд.

Схема взаємодії датчиків протікання та задимлення із системою для інформування користувача про позаштатну ситуацію

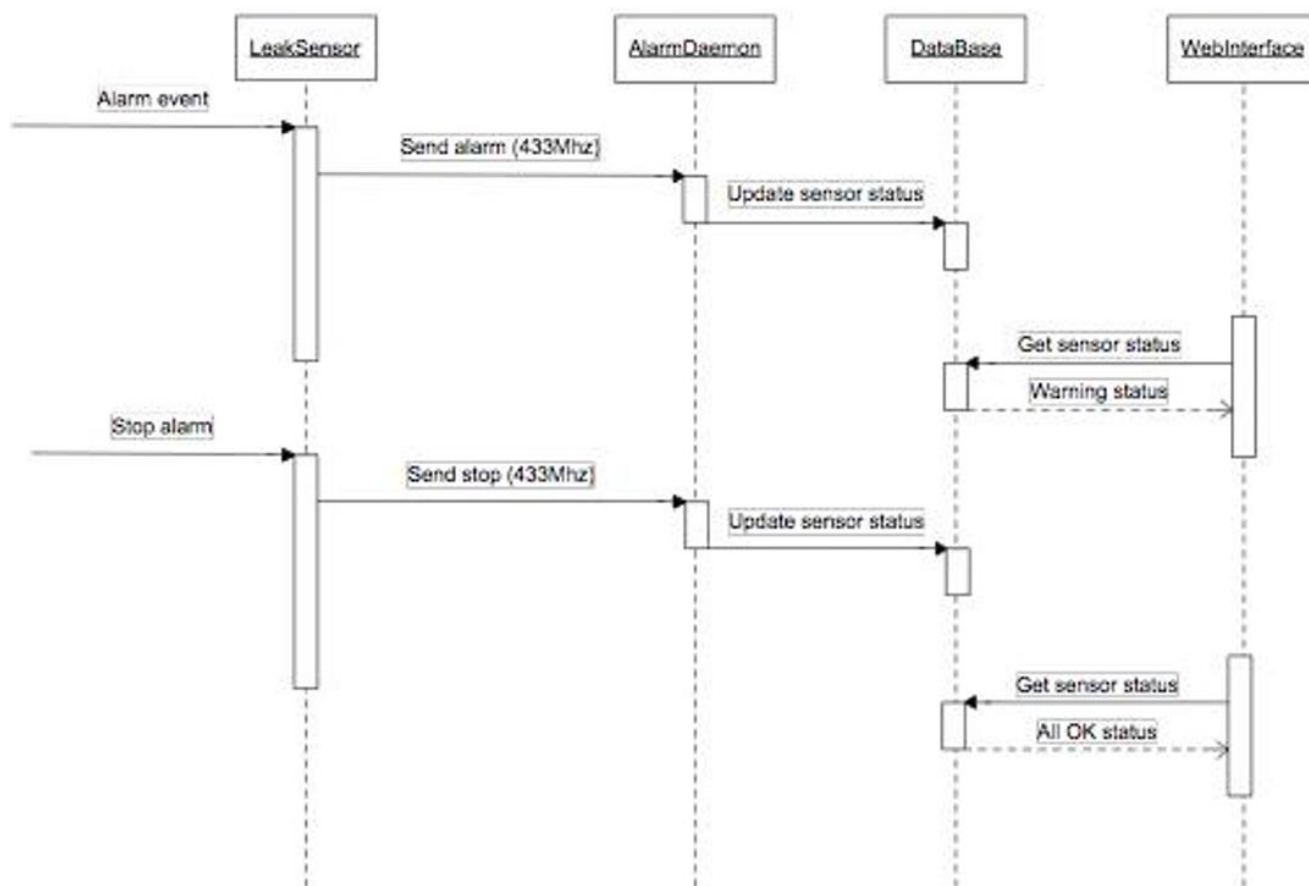


Рисунок 3.6 - діаграма послідовностей взаємодії датчиків протікання та задимлення із системою

На рисунку 3.6 представлена діаграма послідовностей, що демонструє процес роботи з датчиками задимлення та протікання.

LeakSensor - датчик протікання.

– AlarmDaemon - фоновий сервіс, слухає повідомлення про тривогу, отримані від датчиків протікання та задимлення.

– DataBase - СУБД, що використовується для зберігання інформації про стани бездротових пристроїв.

– WebInterface - інтерфейс взаємодії користувача із системою, реалізований у вигляді сукупності web-сторінок.

Користувач системи може виконувати такі дії:

- вмикати/вимикати світло;
- відкривати/закривати штори або жалюзі;
- керувати приладами з IR-інтерфейсом.

Система взаємодіє з користувачем через повідомлення про події:

- задимлення;
- протікання.

Система взаємодіє з пристроями (керує ними):

- вимикачі світла;
- сервоприводи;
- IR-пристрої.

Датчики, що реєструють виникнення задимлення або протікання, взаємодіють із системою через надсилання повідомлень про:

- спрацьовування датчика реєстрації задимлення;
- спрацьовуванні датчика реєстрації протікання.

ВИСНОВКИ

У рамках цієї роботи було вирішено поставлене завдання, а саме – розробка системи управління «розумним будинком», що має базовий функціонал і можливості до розширення.

Крім того, можна шифрувати дані під час передачі для підвищення безпеки.

Для кожного типу пристроїв використовується система команд. Таким чином, один і той же номер команди для різних пристроїв може мати різне значення.

Алгоритми шифрування відіграють важливу роль у забезпеченні безпеки даних у системах розумного будинку. Вони дають змогу захистити конфіденційність даних і запобігти несанкціонованому доступу. Вибір і реалізація відповідного алгоритму шифрування, управління ключами та обмін ключами - це важливі аспекти розробки безпечних систем розумного будинку. Розробники та виробники розумних будинків повинні приділяти належну увагу цим аспектам і прагнути до створення надійних і безпечних рішень для користувачів розумних будинків.

Перевагами розробленої системи є те, що вона, по-перше, є універсальною для будь-яких пристроїв, що працюють у заданому частотному діапазоні, по-друге, захищеною від перешкод та зовнішнього втручання, по-третє, досить дешева для успішного впровадження навіть у складній економічній обстановці.

Надалі планується розширення наявного функціоналу у бік збільшення кількості типів підтримуваних пристроїв (зокрема, робота з іншими частотними діапазонами та протоколами зв'язку).

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. BCM2835 ARM Peripherals: Raspberry Pi Foundation [Електронний ресурс]- 2016. – Режим доступу:<http://www.raspberrypi.org/wp-content/uploads/2012/02/https://www.arduino.cc/en/Reference/HomePage>, вільний.
2. 10. Bell C. Початок Sensor Networks з Arduino and Raspberry Pi - Apress, 2013. - 372 p. - ISBN-10: 1430258241, ISBN-13: 978-1430258247
3. Dennis AK Raspberry Pi Home Automation with Arduino - Packt Publishing, 2013. - 176 p. - ISBN-10: 1230336241, ISBN-13: 679-1230258562
4. Arduino language reference: Arduino project [Електронний ресурс] - 2016. – Режим доступу:<https://www.arduino.cc/en/Reference/HomePage>, вільний.
5. BCM2835 ARM Peripherals: Raspberry Pi Foundation [Електронний ресурс] - 2016. – Режим доступу:<http://www.raspberrypi.org/wp-content/uploads/2012/02/https://www.arduino.cc/en/Reference/HomePage>, вільний.
6. Bell C. Початок Sensor Networks з Arduino and Raspberry Pi - Apress, 2013. - 372 p. - ISBN-10: 1430258241, ISBN-13: 978-1430258247
7. Dennis AK Raspberry Pi Home Automation with Arduino - Packt Publishing, 2013. - 176 p. - ISBN-10: 1230336241, ISBN-13: 679-1230258562
8. Effects of Available Sugar on the Reproductive Fitness and Vectorial Capacity of the Malaria Vector *Anopheles gambiae* (Diptera: Culicidae) 2001 Richard E. Gary, Jr. Woodbridge A. Foster. URL: <https://academic.oup.com/jme/article/38/1/22/1004998/Effects-of-Available-Sugar-on-the-Reproductive> (дата звернення 02.05.2023).
9. Consumers Want Healthy Foods - And Will Pay More For Them. Forbes. Nancy Gagliardi 2015. URL: <https://www.forbes.com/sites/nancygagliardi/2015/02/18/consumers-want-healthy-foods-and-will-pay-more-for-them/#68eaf0e875c5> (дата звернення 03.05.2023)
10. What Everyone Must Know About Industry 4.0. Bernard Marr. Forbes 2016

URL:<https://www.forbes.com/sites/bernardmarr/2016/06/20/what-everyone-must-know-about-industry-4-0/#b4c9ac7795f7>(дата звернення 03.05.2023).

11.Інфрачервоний датчик Sharp GP2Y0A21YK0F 2019

URL:http://electromicro.com/market/datchiki_i_sensory/infrakrasnyj_datchiMiller,
DR Security Information and Event Management (SIEM) implementation / DR
Information Technology. Information Security. Information Assurancy. Електронний
доступ:<http://www.isaca.org>. (дата звернення: 19.04.2023)

12.Jones JA An Introduction to FAIR / JA Jones - Trustees of Norwich
University, 2005. - 67 p.

13.Zadeh LA Fuzzy sets / LA Zadeh. - Information and Control, 1965. -
№8. - P. 338-353.

14.Zadeh LA On optimální kontrol i linear programming / LA Zadeh,
BH Whalen. - IRE Trans. Automaticcontrol, Ac-7, 1962. - P. 45-46.

15.Zimmermann H.-J. Fuzzy Sets, Decision Making and Expert Systems / H.-J.
Zimmermann. - Kluwer: Dordrecht, 1987. - 335 p.

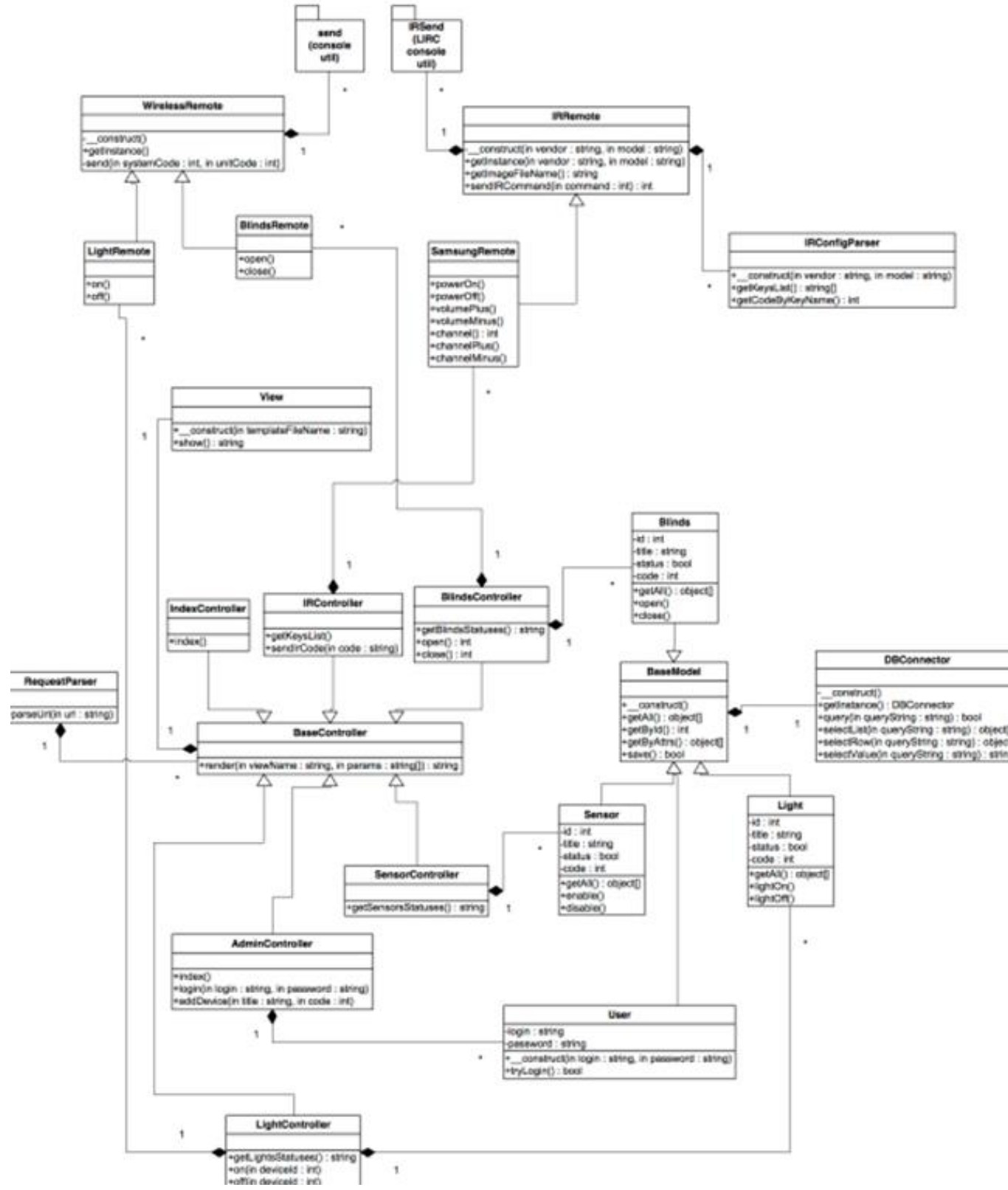
16. Arduino language reference: Arduino project [Електронний ресурс] -
2016. –Режим доступу:<https://www.arduino.cc/en/Reference/HomePage>,вільний.

Структура пакету

Назва	Тип поля	Опис
device_id	unsigned int8	Номер пристрою, з якого відправляється пакет. максимальне кількість пристроїв у системі - 256
destination_id	unsigned int8	Номер пристрою-одержувача
packet_id	unsigned int16	Ідентифікатор пакету. У нього записується випадкове число яке ідентифікуєт пакет як унікальний
command	unsigned int8	Номер команди. Інструкція, що робити з даними пакета

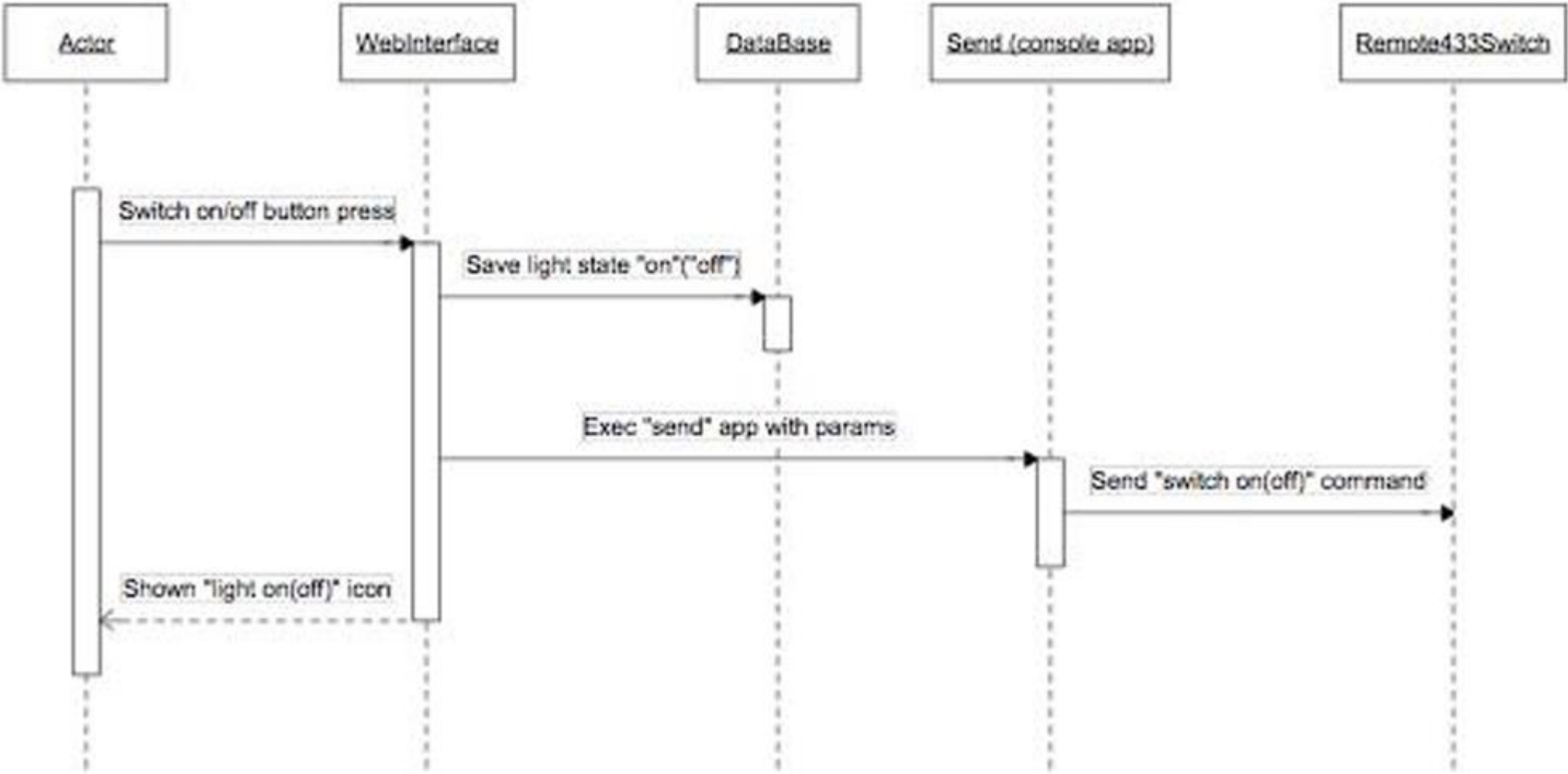
					13.02070849.51040 ПЛ1			
					Структура пакету Розробка системи автоматизованого управління сервісними функціями житлового приміщення	Лит.	Маса	Масит.
Зам.	Лист.	№ докум.	Підп.	Дата				
Розроб.		Федоров Я.Ю.						
Перев.		Скрупський С.Ю.						
Т.контр.						Лист 1		
Н.контр.		Щербак Н.В.				Листів 1		
Затв.		Кудерметов Р.К.				НУ «Запорізька політехніка» КНТ-510сп		

Діаграма класів системи



					13.02070849.51040 ПЛ2									
					Діаграма класів системи Розробка системи автоматизованого управління сервісними функціями житлового приміщення					Лит.		Маса	Масит.	
Зам.	Лист.	№ докум.	Підп.	Дата										
Розроб.		Федоров Я.Ю.												
Перев.		Скрупський С.Ю.												
Т.контр.														
Н.контр.		Щербак Н.В.								Лист 1		Листів 1		
Затв.		Кудерметов Р.К.								Лист 1		Листів 1		
					НУ «Запорізька політехніка»					КНТ-510сп				

Діаграма послідовностей процесу взаємодії користувача



					13.02070849.51040 ПЛЗ										
					Діаграма послідовностей процесу взаємодії користувача Розробка системи автоматизованого управління сервісними функціями житлового приміщення					Лит.		Маса		Масшт.	
Зам.	Лист.	№ докум.	Підп.	Дата											
Розроб.		Федоров Я.Ю.													
Перев.		Скрупський С.Ю.													
Т.контр.										Лист 1		Листів 1			
Н.контр.		Щербак Н.В.													
Затв.		Кудерметов Р.К.													
										НУ «Запорізька політехніка» КНТ-510сп					

Зібрана система



					13.02070849.51040 ПЛ4			
					Зібрана система Розробка системи автоматизованого управління сервісними функціями житлового приміщення	Лист.	Маса	Масит.
Зам.	Лист.	№ докум.	Підп.	Дата				
Розроб.		Федоров Я.Ю.						
Перев.		Скрупський С.Ю.						
Т.контр.						Лист 1	Листів 1	
Н.контр.		Щербак Н.В.				НУ «Запорізька політехніка» КНТ-510сп		
Затв.		Кудерметов Р.К.						