ДОДАТОК А
ОПИС ДІАГРАМ КЛАСІВ

На малюнку А.1 буде наведена діаграма класів програми, при отримані данних від датчиків задимлення і протікання, на малюнку 2 – діаграма основних класів додатки, що працює на мікроконтролері Arduino Uno. Клас RCSwitch відповідає безпосередньо за прийом і передачу даних. Клас WiringPi організовує роботу зі сторонньою бібліотекою WiringPi і виконує первинну настройку обладнання.

send – точка входу додатки, єдиний метод main виконує ініціалізацію додатки і запуск нескінченного циклу опитування датчиків протікання і задимлення.

RCSwitch – реалізує прийом і дешифрування повідомлень.

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

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

- getReceivedValue – метод, який повертає отримане повідомлення;

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

- send – метод, що забезпечує відправку повідомлення з RaspberryPi.

Повідомлення може бути закодовано як однієї змінної типу long, так і масивом символів char []

WiringPi – виконує прив'язку пинов GPIO до змінних типу int. Це дозволяє отримувати повідомлення з мережі простий перевіркою значень змінних.

- wiringPiSetup – метод, що виконує настройку інтерфейсу GPIO.

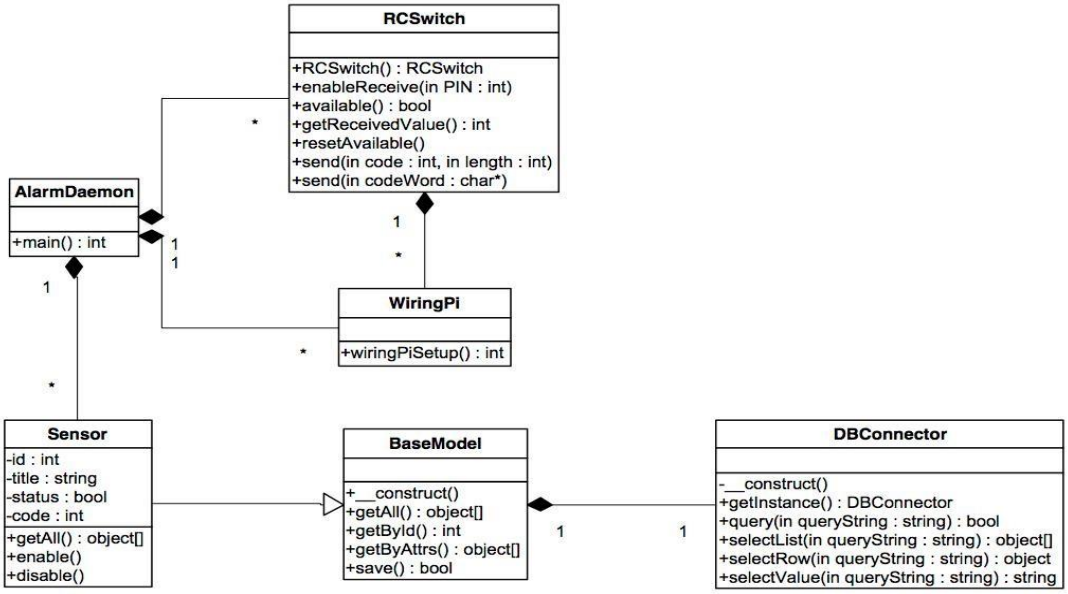

Рисунок А.1 – Діаграма класів програми, обробній повідомлення отримані від датчиків задимлення і протікання

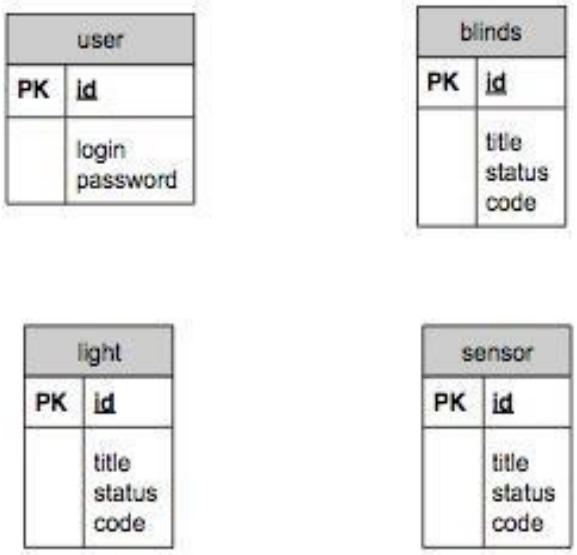

Рисунок А.2 – Модель сутність–зв'язок бази даних системи автоматизації житлових приміщень

На малюнку А.2 продемонстрована діаграма класів фонової служби, що працює на апаратній платформі RaspberryPi. Клас RCSwitch відповідає за прийом і первинну обробку повідомлень, отриманих приймачем бездротового зв'язку. Клас WiringPi організовує роботу зі сторонньою

бібліотекою WiringPi і виконує первинну настройку обладнання. Класи Sensor, BaseModel і DBConnector забезпечують збереження отриманих даних в базу даних.

AlarmDaemon – точка входу додатки, єдиний метод main виконує ініціалізацію додатки і запуск нескінченного циклу прослуховування повідомлень від датчиків протікання і задимлення.

На малюнку А.3 продемонстрована діаграма класів фонової служби, що працює на апаратній платформі RaspberryPi. Клас RCSwitch відповідає за прийом і первинну обробку повідомлень, отриманих приймачем бездротового зв'язку. Клас WiringPi організовує роботу зі сторонньою бібліотекою WiringPi і виконує первинну настройку обладнання. Класи Sensor, BaseModel і DBConnector забезпечують збереження отриманих даних в базу даних.

AlarmDaemon – точка входу додатки, єдиний метод main виконує ініціалізацію додатки і запуск нескінченного циклу прослуховування повідомлень від датчиків протікання і задимлення.

WiringPi – виконує прив'язку пинов GPIO до змінних типу int. Це дозволяє отримувати повідомлення з мережі простий перевіркою значень змінних:

- wiringPiSetup – метод, що виконує настройку інтерфейсу GPIO RCSwitch – реалізує прийом і дешифрування повідомлень;
- enableReceive – метод, що перемикає бібліотеку RCSwitch на прийом даних;
- available – метод, який перевіряє наявність нового необробленого повідомлення;
- getReceivedValue – метод, який повертає отримане повідомлення;
- resetAvailable – метод, що перемикає додаток в стан очікування отримання нового повідомлення;
- send – метод, що забезпечує відправку повідомлення з RaspberryPi;

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

- __construct – метод, не започатковано підключення до бази даних через клас DBConnector;

- getAll – метод, який повертає список всіх записів;

- getById – метод, який повертає запис для вказаного ID;

- getByAttrs – метод, який повертає список записів, які відповідають параметрам пошуку;

- save – метод, який реалізує збереження об'єкта в базу даних. Якщо в об'єкта заданий первинний ключ, то запис оновиться, якщо первинний ключ не заданий, то буде створена нова запис.

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

- id – атрибут, містить ідентифікатор запису з таблиці sensor;

- title – атрибут, містить назву датчика;

- status – атрибут, містить статус датчика (в стані тривоги або спокою);

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

- getAll – метод, повертає список всіх датчиків з таблиці sensor;

- enable – метод, позначає датчик як включений (в стані тривоги);

- disable – метод, позначає датчик як вимкнений (в стані спокою).

DBConnector – клас, що забезпечує підключення до бази даних, реалізує шаблон проектування – одинак.

- __construct – метод, инициализирует підключення до бази даних;

- getInstance – метод, повертає екземпляр даного класу;

- query – метод, виконує SQL запит на вставку або оновлення;

- selectList – метод, повертає список записів, які відповідають параметрам пошуку;

Систему можна розділити на 3 основні складові частини: веб–інтерфейс, СУБД і інтерфейси взаємодії з бездротовими пристроями.

Роботу веб–інтерфейсу забезпечують наступні класи:

RequestParser – клас, що виконує обробку та маршрутизацію http–запитів:

- parseUrl – метод, виконує розбір http–запиту і викликає

відповідний метод зазначеного в запиті контролера.

BaseController – базовий клас для контролерів.

- render – метод, виконує рендеринг сторінки з передачею всіх необхідних змінних в html–шаблон;

IndexController – клас, який реалізує роботу головної сторінки веб–інтерфейсу.

- index – метод, що виконує генерацію головної сторінки.

IRController – клас, відповідає за обробку http запитів, пов'язаних з IR–пристроями.

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

- sendIrCode – метод, отримує на вхід код натиснутої кнопки пульта дистанційного керування, з подальшою передачею його ir–приймача.

BlindsController – клас, відповідає за обробку http запитів, пов'язаних зі шторами.

- getBlindsStatuses – метод, повертає список штор і жалюзі, керованих системою, з їх поточним статусом – відкритий / закритий;

- open – метод, обробляє запит на відкриття штор;

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

датчиками задимлення і протікання.

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

LightController – клас, відповідає за обробку http–запитів, пов'язаних з управлінням світловими приладами:

- getLightsStatuses – метод, повертає список світлових приладів з їх поточним статусом – включений / виключений;

- on – метод, обробляє запит на включення приладу освітлення;

- off – метод, обробляє запит на відключення обладнання освітлення;

AdminController – клас, забезпечує настройку системи;

- index – метод, що виконує генерацію головної сторінки адміністрування;

- login – метод, що виконує генерацію сторінки з формою для логіна;

- addDevice – метод, який реалізує можливість додавання світлових приладів;

View – клас, який відповідає за генерацію web–сторінок з html–шаблонів:

- __construct – конструктор, напівчіт на вхід назву шаблону;

- show – метод, що виконує генерацію сторінки з шаблону і висновок на екран;

За роботу з базою даних відповідають наступні класи:

DBConnector – клас, що виконує підключення і 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, виконує SQL запит до бази;

- selectList – метод, виконує SQL запит на SELECT, повертає список рядків;

- selectRow – метод, виконує SQL запит на SELECT, повертає один рядок;

- selectValue – метод, виконує SQL запит на SELECT, повертає одне значення;

BaseModel – базовий клас для ActiveRecord моделей.

- __construct – конструктор;

- getAll – метод, повертає список ActiveRecord об'єктів, відповідних всім записам з таблиці;

- getById – метод, повертає ActiveRecord об'єкт для зазначеного первинного ключа;

- getByAttrs – метод, повертає список ActiveRecord об'єктів, які відповідають параметрам вибірки;

- save – метод, виконує збереження ActiveRecord об'єкта в базі. Якщо первинний ключ був зазначений, то запис в таблиці оновлюється, в іншому випадку – створюється новий запис;

Blinds – клас, що забезпечує роботу з таблицею blinds.

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

- title – атрибут, назва пристрою;

- status – атрибут, стан пристрою – відкрито або закрито;

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

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

- open – метод, виставляє поле status = 1 для рядка з зазначеним id;

- close – метод, виставляє поле status = 0 для рядка з зазначеним id.

Light – клас, що забезпечує роботу з таблицею light:

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

- title – атрибут, назва пристрою;

- status – атрибут, стан пристрою – увімкнене;

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

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

- lightOn – метод, виставляє поле status = 1 для рядка з зазначеним id;

- lightOff – метод, виставляє поле status = 0 для рядка з зазначеним id;

Sensor – клас, що забезпечує роботу з таблицею sensor:

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

- title – атрибут, назва пристрою;

- status – атрибут, стан пристрою – активовано або в немає;

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

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

- enable – метод, виставляє поле status = 1 для рядка з зазначеним id;

- disable – метод, виставляє поле status = 0 для рядка з зазначеним id;

User – клас, що забезпечує роботу з таблицею user:

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

- password – атрибут, пароль користувача;

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

- tryLogin – метод, виконує перевірку коректності зазначеного логіна і пароля шляхом пошуку відповідного рядка в базі;

Робота з бездротовими пристроями забезпечується класами:

WirelessRemote – базовий клас, який реалізує роботу бездротовими пристроями через консольну утиліту send. Реалізує шаблон проектування – одинак:

- __construct – конструктор;

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

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

передані бездротових пристроїв;

LightRemote – клас, який реалізує управління світловими приладами.

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

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

BlindsRemote – клас, який реалізує управління шторами або жалюзі.

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

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

send – консольна утиліта, виконує ретрансляцію запитів з класу WirelessRemote бездротових пристроїв через пряму роботу з бездротовим передавачем 433MHz.

IRRemote – базовий клас, 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, повертає ім'я файлу зображення пульта дистанційного керування, для використання в веб–інтерфейсі;

- sendIRCommand – метод, відправляє IR–команду, використовуючи консольну утиліту IRSend з пакета LIRC.

IRConfigParser – клас, що виконує розбір конфігураційних файлів для різних пристроїв зі складу проекту LIRC:

- __construct – конструктор;

- getKeysList – метод, повертає список кнопок пульта дистанційного керування для заданого пристрою;

- getCodeByKeyName – метод, повертає код, що ідентифікує кнопку пульта дистанційного керування по її назві;

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

- powerOn – метод, виконує включення телевізора;

- powerOff – метод, виконує вимкнення телевізора;

- volumePlus – метод, виконує збільшення гучності;

- volumeMinus – метод, виконує зменшення гучності;

- channel – метод, перемикає канал на телевізорі на заданий;

- channelPlus – метод, перемикає канал на наступний;

- channelMinus – метод, перемикає канал на попередній;

IRSend – консольна утиліта, виконує ретрансляцію запитів з класу IRRemote через IR–передавач.

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

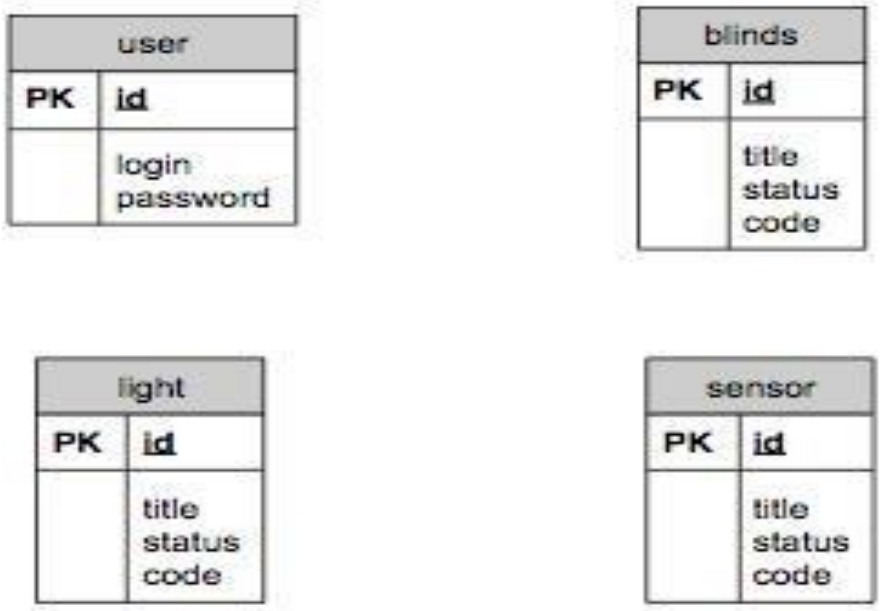

Рисунок А.3 – модель сутність–зв'язок бази даних системи автоматизації житлових приміщень

На малюнку А.4 продемонстрована ER–модель бази даних системи автоматизації житлових приміщень. База даних містить наступні команди:

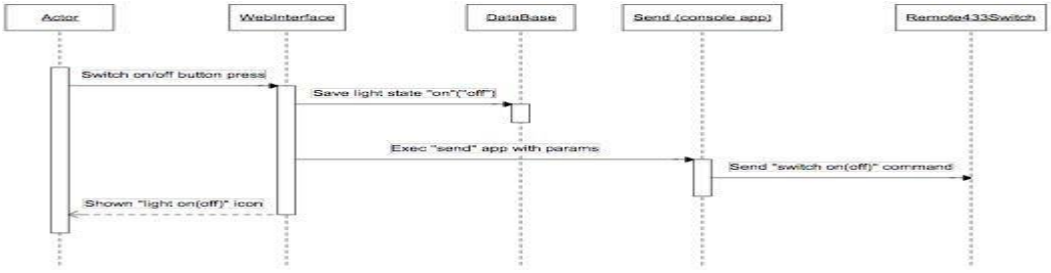

Рисунок А.4 – діаграма послідовностей процесу взаємодії користувача з системою для управління пристроями по бездротовому протоколу на частоті 433MHz

- id – унікальний ідентифікатор користувача, сурогатний первинний ключ;

- login – логін (ім'я) користувача;

- password – пароль користувача.

blinds – містить інформацію про шторах і жалюзі, відкриття і закриття яких автоматизовано з використанням сервоприводів.

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title – назва пристрою;

- status – стан пристрою – відкрито або закрито;

- code – код пристрою, використовується в протоколі для ідентифікації пристрою.

light – містить інформацію про світлових приладах, доступних для автоматичного управління:

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title – назва пристрою;

- status – стан пристрою – увімкнене;

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

sensor – містить інформацію про датчиках задимлення і протікання і їх стані:

- id – унікальний ідентифікатор пристрою, сурогатний первинний ключ;

- title – назва пристрою;

- status – стан пристрою – активовано чи ні;

- code – код пристрою, використовується в протоколі для ідентифікації

Схема взаємодії користувача з системою для управління пристроями через IR–порт

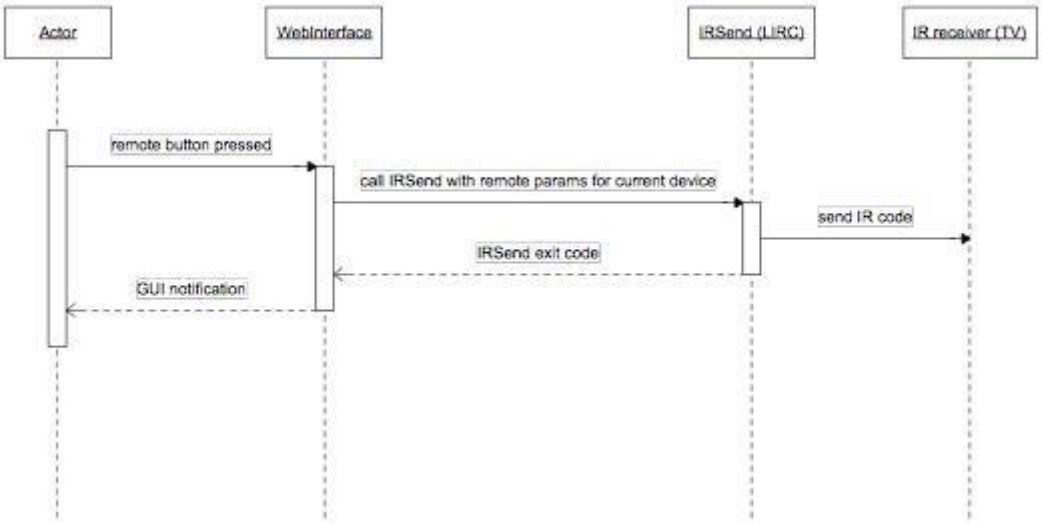

Рисунок А.5 – діаграма послідовностей процесу взаємодії користувача з системою для управління пристроями через IR–порт

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮІК.ХХХХХХ.ХХХ ПЗ	Пояснювальна записка	75 с.
		Графічні матеріали	
2	Таблиця 1	Структура пакета	
3	Рисунок 1.1 – 1.6	Реалізації ідеї системи «розумний дім»	Включено в ПЗ
4	Рисунок 2.1 – 2.22	Схеми архітектури Raspberry Pi.	Включено в ПЗ
5	Рисунок 3.1 – 3.5	Схеми підключення RS–485	Включено в ПЗ
6	Рисунок 3.6 – 3.8	Платформа Arduino, ESP NODEMCU	Включено в ПЗ
		Носії інформації	
7	ГЮІК.ХХХХХХ.ХХХ КД	Компакт-диск (CD)	1 шт.

Змін.	Арк.	Номер докум.	Підп.	Дата	ГЮІК.ХХХХХХ.ХХХ ВД			
Розроб.		Устьянов М.С.			Система керування розумним будинком на основі платформи Raspberry Pi Відомість атестаційної роботи	Літ	Аркуш	Аркушів
Перевір.		Безсонов О.О.				У	1	1
						ХНУРЕ Кафедра КІТС		
Н.контр.		Калмикова В.І						
Затв.		Руденко О.Г.						