Вінницький національний технічний університет
(повне найменування вищого навчального закладу)

Факультет інтелектуальних інформаційних технологій та автоматизації
(повне найменування інституту, назва факультету (відділення))

Кафедра комп'ютерних наук
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна технологія управління безпекою розумного будинку»

Виконала: студентка 2-го курсу, групи 1КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
Капченко К. Г.
(прізвище та ініціали)

Керівник: PhD, професор каф. КН
Савчук Т. О.
(прізвище та ініціали)
«07» 12 2023 р.

Опонент: д.т.н, професор, зав. каф. АІІТ
Бісікало О. В.
(прізвище та ініціали)
«07» 12 2023 р.

Допущено до захисту
Завідувач кафедри КН
д.т.н., проф. Яровий А. А.
«08» 12 2023 р.

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра Комп'ютерних наук
Рівень вищої освіти другий (магістерський)
Галузь знань – 12 «Інформаційні технології»
Спеціальність – 122 «Комп'ютерні науки»
Освітньо-професійна програма – «Системи штучного інтелекту»

ЗАТВЕРДЖУЮ
Завідувач кафедри КН
проф., д.т.н. Яровий А.А.
"29" 08 2023 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ
Капченко Карині Григорівні

1. Тема роботи: Інформаційна технологія управління безпекою розумного будинку.
Керівник роботи: Савчук Тамара Олександрівна, PhD, професор кафедри КН.
Затверджені наказом вищого навчального закладу від 18.09.23 № 247

2. Термін подання студентом роботи 13.11.2023 р.

3. Вихідні дані до роботи:
Потужність множини користувачів – не менше 3; потужність множини правил – не менше 5; потужність множини факторів – не менше 8; потужність множини висновків – не менше 5; мінімальна кількість записів в базі даних – 30; мова програмування – об'єктно-орієнтована; середовище розробки з можливістю використовувати розширення; база даних – реляційна.

4. Зміст текстової частини (перелік питань які потрібно розробити):
Вступ; Сучасний рівень розвитку управління безпекою розумного будинку; Розробка методу та структури інформаційної технології управління безпекою розумного будинку; Розробка модулів інформаційної технології управління безпекою розумного будинку; Аналіз результатів тестування інформаційної технології управління безпекою розумного будинку; Висновки; Перелік використаних джерел; Додатки.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень):
UML-діаграма активності для управління безпекою розумного будинку; Структура інформаційної технології управління безпекою розумного будинку; UML-діаграма алгоритму функціонування модуля авторизації (Вхід у систему); UML-діаграма алгоритму функціонування модуля авторизації (Реєстрація);

UML-діаграма удосконаленого алгоритму аналізу безпеки розумного будинку; UML-діаграма алгоритму функціонування модуля сценаріїв безпеки; UML-діаграма алгоритму функціонування модуля аналітики безпеки; UML-діаграма алгоритму функціонування модуля виконання сценарію безпеки.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	Савчук Т. О., проф. каф. КН		
4	Адлер О. О., доц. каф. ЕПВМ		

7. Дата видачі завдання 29.08.2023р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Сучасний рівень розвитку управління безпекою розумного будинку	01.09.23	07.09.23	
2	Розробка методу та структури інформаційної технології управління безпекою розумного будинку	08.09.23	24.09.23	
3	Розробка модулів інформаційної технології управління безпекою розумного будинку	25.09.23	10.10.23	
4	Аналіз результатів тестування інформаційної технології управління безпекою розумного будинку	11.10.23	15.10.23	
5	Економічна частина	16.10.23	25.10.23	
6	Оформлення додатків	26.10.23	01.11.23	
7	Розробка інструкції користувача	02.11.23	06.11.23	
8	Оформлення матеріалів до захисту БДР	07.11.23	10.11.23	

Студентка _____ Капченко К. Г.
(підпис) (прізвище та ініціали)

Керівник роботи _____ Савчук Т. О.
(підпис) (прізвище та ініціали)

АНОТАЦІЯ

УДК 004.4

Капченко К. Г. Інформаційна технологія управління безпекою розумного будинку. Магістерська кваліфікаційна робота зі спеціальності 122 «Комп'ютерні науки», освітня програма «Системи штучного інтелекту». Вінниця: ВНТУ, 2023. 112 с.

На укр. мові. Бібліогр.: 29 назв; рис.: 29; табл. 12.

Робота присвячена розробці інформаційної технології управління безпекою розумного будинку. Метою роботи є підвищення швидкості реакції на виявлену загрози безпеки. Проведено аналіз сучасної математичної моделі та сучасних засобів управління безпекою розумного будинку, виявлено їх переваги та недоліки в результаті чого було визначено необхідність створення інформаційної технології управління безпекою розумного будинку, що забезпечить підвищення швидкості реакції на виявлену загрозу. В результаті чого було виконано постановку задачі.

Удосконалено математичну модель процесу управління безпекою, розроблено алгоритм для управління безпекою розумного будинку. Сформовано структуру інформаційної технології управління безпекою розумного будинку. Обґрунтовано вибір мови програмування та середовища розробки інформаційної технології управління безпекою розумного будинку, а також обрано систему управління базою даних. Відповідно до структурної схеми та алгоритму для управління безпекою розумного будинку було розроблено інформаційну технологію управління безпекою розумного будинку.

В результаті тестування визначено, що розроблена інформаційна технологія управління безпекою розумного будинку відповідає поставленим вимогам та мету досягнуто.

Ключові слова: інформаційна технологія, управління безпекою, розумний будинок, аналіз, безпека, сценарій безпеки.

ABSTRACT

Kapchenko K. G. Information technology for smart home security management. Master's thesis in the specialty 122 «Computer sciences», education program «Artificial intelligence systems». Vinnytsia: VNTU, 2023. 108 p.

In Ukrainian language. Bibliogr.: 29 titles; fig 29; table 12.

The work is devoted to the development of information technology for managing the security of a smart home. The aim of the work is to increase the speed of response to identified security threats. An analysis of the modern mathematical model and modern means of managing the security of a smart home was carried out, their advantages and disadvantages were identified, as a result of which the need to create an information technology for managing the security of a smart home was determined, which will increase the speed of response to the identified threat. As a result, the task was accomplished.

The mathematical model of the security management process was improved, an algorithm for managing the security of a smart home was developed. The structure of the information technology for managing the security of a smart home is formed. The choice of the programming language and development environment for the information technology of smart home security management is substantiated, and the database management system is selected. In accordance with the structural diagram and algorithm for managing the security of a smart home, an information technology for managing the security of a smart home has been developed.

As a result of testing the developed software tool, it was determined that the information technology for managing the security of a smart home meets the requirements and the goal is achieved.

Keywords: information technology, security management, smart home, analysis, security, security scenario.

ЗМІСТ

ВСТУП 4
1 СУЧАСНИЙ РІВЕНЬ РОЗВИТКУ УПРАВЛІННЯ БЕЗПЕКОЮ РОЗУМНОГО БУДИНКУ 8
1.1 Управління безпекою розумного будинку 8
1.2 Аналіз сучасних засобів управління безпекою розумного будинку 11
1.3 Постановка задачі 18
1.4 Висновок до розділу 1 18
2 РОЗРОБКА МЕТОДУ ТА СТРУКТУРИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ БЕЗПЕКОЮ В РОЗУМНОМУ БУДИНКУ 20
2.1 Розробка удосконаленої математичної моделі процесу управління безпекою розумного будинку 20
2.2 Розробка узагальненого алгоритму для управління безпекою розумного будинку 22
2.3 Розробка структури інформаційної технології управління безпекою розумного будинку 25
2.4 Висновок до розділу 2 27
3 РОЗРОБКА МОДУЛІВ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ БЕЗПЕКОЮ В РОЗУМНОМУ БУДИНКУ 28
3.1 Обґрунтування вибору мови програмування 28
3.2 Обґрунтування вибору середовища розробки інформаційної технології управління безпекою розумного будинку 32
3.3 Розробка інтерфейсу інформаційної технології управління безпекою розумного будинку 33
3.4 Розробка функціонування модулів інформаційної технології управління безпекою розумного будинку 40
3.4.1 Модуль авторизації 40
3.4.2 Модуль аналізу безпеки 43
3.4.3 Модуль сценаріїв безпеки 48
3.4.4 Модуль аналітики безпеки 51
3.4.5 Модуль виконання сценарію безпеки 53
3.5 Тестування інформаційної технології управління безпекою розумного будинку 56
3.6 Висновок до розділу 3 64

4 ЕКОНОМІЧНА ЧАСТИНА 65

4.1 Проведення комерційного та технологічного аудиту інформаційної технології управління безпекою розумного будинку 65

4.2 Розрахунок витрат на здійснення науково-дослідної роботи 66

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором 73

4.4 Висновок до розділу 4 77

ВИСНОВКИ 79

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ 81

ДОДАТКИ 84

Додаток А (обов’язковий) ..... Результат перевірки на плагіат в онлайн-системі UNICHECK 85

Додаток Б (обов’язковий) Лістинг програми 86

Додаток В (обов’язковий) ІЛЮСТРАТИВНА ЧАСТИНА 101

Додаток Г (довідниковий) Інструкція користувача 107

ВСТУП

Актуальність теми

У сучасному цифровому віці, коли технології стають неодмінною частиною нашого повсякденного життя, розумні будинки займають центральне місце серед інноваційних рішень. Завдяки швидкому розвитку технологій, у наш час все більше людей стають зацікавленими в використанні розумних систем для контролю за своїми будинками. Розумний будинок, який інтегрує в себе різні пристрої, надає зручність, комфорт та безпеку перетворюючи звичайний будинок в інтелектуальну житлову площу.

Розумний будинок, обладнаний інформаційною технологією управління, відкриває безліч можливостей для автоматизації та контролю. Проте, через стрімкий прогрес технологій та зростання інтересу до розумних систем, ми стикаємося з необхідністю забезпечення безпеки і конфіденційності в цифровому середовищі. Безпека є однією з найважливіших складових розумного будинку, і охоплює багато аспектів. Розумний будинок може бути оснащений системами відеоспостереження, давачами пожежі, витоку газу або води, а також системою безпеки з автоматичним оповіщенням. Ці системи забезпечують захист будинку і мешканців, а також можуть повідомляти про потенційну небезпеку.

Питання управління безпекою розумного будинку залишається актуальним, так як безпека займає головну роль в житті людини та збереженні її майна. Використання програмних засобів з управління безпекою в розумному будинку характеризуються зручним способом перегляду відео з камер спостереження, отримання сповіщень про керування системами освітленням і термоконтролем. Проте, їх використання для забезпечення повноцінної безпеки розумного будинку потребує удосконалень щодо покращення безпеки самого додатку, а також розширення можливостей до автоматизації.

Актуальність інформаційної технології управління безпекою розумного будинку полягає у потребі створення інструменту, який забезпечить надійне

управління розумним будинком. Сьогодні на ринку представлено достатньо програм, завдяки яким можна здійснювати управління розумним будинком. Проте, сучасні додатки мають певні недоліки з управлінням безпекою. Зважаючи на це, та на потребу використання розумних систем та додатків по управлінню ними, створення інформаційної технології управління безпекою розумного будинку є актуальним.

Зв'язок роботи з науковими програмами, планами, темами

Магістерська кваліфікаційна робота виконана відповідно до напряму наукових досліджень кафедри комп'ютерних наук Вінницького національного технічного університету 22 К1 «Розробка прикладних інтелектуальних інформаційних технологій та систем» та плану наукової та навчально-методичної роботи кафедри.

Мета дослідження

Метою дослідження є підвищення швидкості реакції на загрозу безпеці розумного будинку та розширення функціональних можливостей управління розумним будинком.

Об'єктом дослідження є процес управління безпекою розумного будинку.

Предметом дослідження є технологія управління безпекою розумного будинку.

Задачі дослідження

Для досягнення поставленої мети, необхідно сформувати та вирішити наступні задачі:

1. Обґрунтувати актуальність розробки інформаційної технології управління безпекою розумного будинку.
2. Провести аналіз сучасних засобів управління безпекою розумного будинку.
3. Розробити удосконалену математичну модель процесу управління безпекою розумного будинку.
4. Розробити структуру інформаційної технології управління розумним будинком.

5. Розробити алгоритм аналізу безпеки розумного будинку.
6. Розробити алгоритми функціонування складових інформаційної технології управління безпекою розумного будинку.
7. Створити тестовий набір даних для використання інформаційної технології.
8. Провести тестування розробленої інформаційної технології на відповідність поставленим вимогам.

Методи дослідження

Під час розробки використовувались наступні методи дослідження, а саме метод аналізу – для аналізу існуючих рішень управління безпекою розумного будинку; архітектури веб-продуктів; моделювання та візуалізації; теорії множин – для формування математичної моделі та теорії алгоритмів – для формування алгоритмів роботи платформи.

Наукова новизна

1. Удосконалено математичну модель процесу управління безпекою розумного будинку, яка на відміну від існуючих, дозволяє підвищити швидкість реагування на загрозу, за рахунок опрацювання вектору факторів впливу на стан безпеки.
2. Розроблено інформаційну технологію управління безпекою розумного будинку, яка на відміну від відповідних існуючих засобів, визначає не тільки загрози, а й вразливості засобу безпеки, що дозволяє завчасно попередити виникнення загрози, а також має розширений функціонал, що дозволяє завчасно створювати власнику будинку набір сценаріїв безпеки та забезпечує швидку реакцію на виявлені загрози.

Практичне значення одержаних результатів полягає у розробці алгоритму аналізу безпеки розумного будинку, алгоритму функціонування модулів інформаційної технології управління розумним будинком, структури інформаційної технології управління безпекою розумного будинку, а також у розробці інформаційної технології управління безпекою розумного будинку.

Апробація результатів роботи

Результати роботи були апробовані на XI Міжнародній науково-практичній конференції «SIENCE AND INNOVATION OF MODERN WORLD» (м. Лондон, Велика Британія, 2023р.) та на науково-технічній конференції факультету інтелектуальних інформаційних технологій та автоматизації (Вінниця, Україна, 2023р.).

Публікації

В результаті виконання магістерської кваліфікаційної роботи було опубліковано 2 тез доповідей: «Структура інформаційної технології управління безпекою розумного будинку» на XI Міжнародній науково-практичній конференції «SIENCE AND INNOVATION OF MODERN WORLD» (м. Лондон, Велика Британія, 2023р.) [1] та «Розробка алгоритму аналізу безпеки розумного будинку» на науково-технічній конференції факультету інтелектуальних інформаційних технологій та автоматизації (Вінниця, Україна, 2023р.) [2].

Отримано свідоцтво про реєстрацію авторського права на комп’ютерну програму «Інформаційна технологія управління безпекою розумного будинку» [3]. Крім того, підготовлено і направлено до редакційної колегії статтю у фаховий журнал категорії «Б» – Таврійський науковий вісник. Серія: Технічні науки.

1 СУЧАСНИЙ РІВЕНЬ РОЗВИТКУ УПРАВЛІННЯ БЕЗПЕКОЮ РОЗУМНОГО БУДИНКУ

1.1 Управління безпекою розумного будинку

Розумний будинок – це інноваційне технологічне рішення, що поєднує автоматизацію та інтеграцію різних систем, пристроїв та компонентів в житловому просторі задля покращення безпеки та комфорту. У розумному будинку використовуються передові сенсори, мережі зв’язку, розподілені системи керування та програмне забезпечення для того, щоб створити інтелектуальну інфраструктуру. Технологія розумного будинку використовується для різних цілей: підвищення комфорту, тепло та енергозбереження, забезпечення безпеки [4]. Одна з ключових особливостей розумного будинку - це його здатність до автоматизації. За допомогою розумних давачів і програмного забезпечення, будинок може реагувати на змінні умови навколишнього середовища, розпізнавати присутність мешканців і адаптувати свої функції відповідно.

З огляду на зростання популярності розумних будинків і збільшення кількості підключених пристроїв, забезпечення безпеки стає однією з основних проблем, які вимагають рішень. За даними Statista, прогнозується, що до 2025 року кількість підключених пристроїв у розумних будинках світової площі досягне понад 75 мільярдів. Це означає, що все більше людей обирають розумні технології для своїх будинків, і безпека є однією з найважливіших потреб для користувачів [5].

Також, за даними Cybersecurity Ventures, витрати на кібербезпеку світового ринку досягнуть $270 мільярдів до 2026 року. Це свідчить про зростання усвідомленості про ризики кібератак та несанкціонованого доступу до розумних будинків. За даними MarketsandMarkets, світовий ринок систем відеоспостереження досягне понад $74 мільярдів до 2025 року так, як

встановлення систем відеоспостереження є одним з ключових компонентів безпеки розумного будинку [6].

Безпека є однією з найважливіших характеристик розумного будинку. Вона охоплює багато важливих аспектів, які слід враховувати при використанні сучасних технологій для автоматизації та управління будинком таких, як:

1. Системи відеоспостереження.
2. Системи безпеки з автоматичним оповіщенням.
3. Давачі безпеки.
4. Автоматична сигналізація.
5. Віддалений доступ та моніторинг [7].

Оскільки розумний будинок підключений до Інтернету та використовує різноманітні пристрої, такі як камери відеоспостереження, системи безпеки, давачі руху та інші, важливо забезпечити адекватний рівень захисту для збереження конфіденційності, уникнення несанкціонованого доступу та запобігання можливим кібератакам.

Розумний будинок може забезпечувати інтерактивну комунікацію з власником через програмні засоби або спеціальні панелі керування. За допомогою них можна віддалено контролювати системи безпеки, отримувати сповіщення та статуси, а також взаємодіяти з давачами і пристроями будинку. Програмне рішення для управління безпекою розумного будинку може забезпечити надійність його захисту, а також спростити сам процес управління. Особливості використання програмних засобів для управління безпекою розумного будинку можуть включати наступні аспекти:

1. Централізоване керування: Програмні засоби для управління безпекою дозволяють користувачам керувати всіма аспектами безпеки свого розумного будинку з одного централізованого інтерфейсу. Це включає контроль доступу, відеоспостереження, системи оповіщення, давачі безпеки та інші функції.

2. Віддалений доступ: Багато програмних засобів надають можливість віддаленого доступу до системи безпеки розумного будинку через мобільні пристрої або комп'ютери. Це дозволяє користувачам контролювати та моніторити безпеку будинку навіть на відстані.
3. Системи оповіщення: Програмні засоби можуть мати вбудовані системи оповіщення, які надсилають повідомлення користувачеві про виникнення подій або несправностей у системі безпеки. Це може включати сповіщення про вторгнення, пожежі, витоку газу, зламу системи та інші небезпеки.
4. Інтеграція з іншими системами: Деякі програмні засоби можуть інтегруватися з іншими системами розумного будинку, такими як системи освітлення, опалення, автоматизації тощо. Це дозволяє створювати злагоджені сценарії безпеки, наприклад, автоматичне включення світла при виявленні руху або заблокування дверей при активації сигналізації.
5. Візуалізація та аналітика: Деякі програмні засоби надають візуалізацію даних з систем безпеки, що дозволяє користувачам отримувати зрозумілу та зручну інформацію про стан безпеки свого будинку. Також можуть бути наявні аналітичні інструменти, які допомагають виявляти аномальні поведінки, розпізнавати образи, аналізувати дані з давачів тощо.
6. Налаштування та персоналізація: Багато програмних засобів дозволяють користувачам налаштовувати параметри безпеки свого будинку згідно з власними потребами. Це може включати розклади активації системи, налаштування зон безпеки, керування давачами та інші налаштування.
7. Кібербезпека: Програмні засоби для управління безпекою розумного будинку повинні бути забезпечені високим рівнем кібербезпеки. Це означає застосування шифрування даних, захист від несанкціонованого доступу та постійне оновлення програмного забезпечення з метою усунення вразливостей системи [8].

Проте, особливості використання програмних засобів для управління безпекою розумного будинку можуть варіюватися в залежності від конкретного програмного засобу та його функціональності.

Таким чином, актуальне створення відповідної інформаційної технології, яка забезпечить надійне управління розумним будинком, за рахунок вчасного виявлення вразливостей та загроз безпеці та швидкого реагування на них.

1.2 Аналіз сучасних засобів управління безпекою розумного будинку

Ринок сучасних засобів управління безпекою розумного будинку постійно розширюється, і на ньому доступні різноманітні програмні засоби з різною функціональністю. Деякі з них є універсальними, проте мають певні недоліки. Розглянемо приклади існуючих популярних засобів для управління безпекою розумного будинку.

Samsung SmartThings – це додаток для з'єднання різних пристроїв і домашніх розумних систем в єдину екосистему. Цей додаток для управління системою розумного будинку надає можливість керувати безпекою та різними аспектами розумного будинку. Він підтримує інтеграцію з різноманітними пристроями та давачами, дозволяє налаштовувати режими безпеки та отримувати сповіщення про події. Додаток підтримує такі опції:

- установка таймера для включення та вимикання освітлення за межами будинку, в певних кімнатах;
- налаштування системи водопостачання, нагріву води;
- визначення тривалості роботи побутових приладів, їх базових функцій;
- система сигналів в разі некоректної роботи холодильника, сушарки, сигналізації;
- перевірка стану вікон і дверей, щоб уникнути поломки, утворення щілин і дестабілізації температури;
- виставлення списку завдань для ідеальної атмосфери романтичного вечора, щоденного відходу до сну, ранкового розкладу [9].

Переваги SmartThings полягають можливості інтеграції з великою кількістю пристроїв та розширюваності. SmartThings підтримує широкий спектр різних розумних пристроїв, включаючи освітлення, термостати, давачі, камери

та інші. Це дає вам можливість збільшити автоматизацію вдома та керувати різними пристроями з одного місця. Також платформа SmartThings є відкритою для розробників, що означає, що сторонні розробники можуть інтегрувати нові пристрої. Це дозволяє розширювати можливості платформи. Проте SmartThings має певні недоліки:

1. Нестабільність: Деякі користувачі повідомляють про нестабільну роботу додатку SmartThings, такі як помітні затримки або відмови працювати. Це зменшує швидкість реакції.
2. Складність налаштування: В деяких випадках налаштування SmartThings можуть бути складними, особливо для новачків. Вимагається розуміння різних налаштувань і інтерфейсів для успішного налаштування й автоматизації системи.
3. Залежність від хмарної інфраструктури: SmartThings використовує хмарну інфраструктуру для з'єднання та керування пристроями. Це означає, що для користування потрібне надійне підключення до Інтернету. Якщо інтернет з'єднання відсутнє або недостатньо стабільне, це може призвести до зниження функціональності.
4. Нестабільна робота: помітні затримки або відмови працювати, що зменшує швидкість реакції та відсутнє забезпечення оновлень безпеки та слідкування за загрозами.

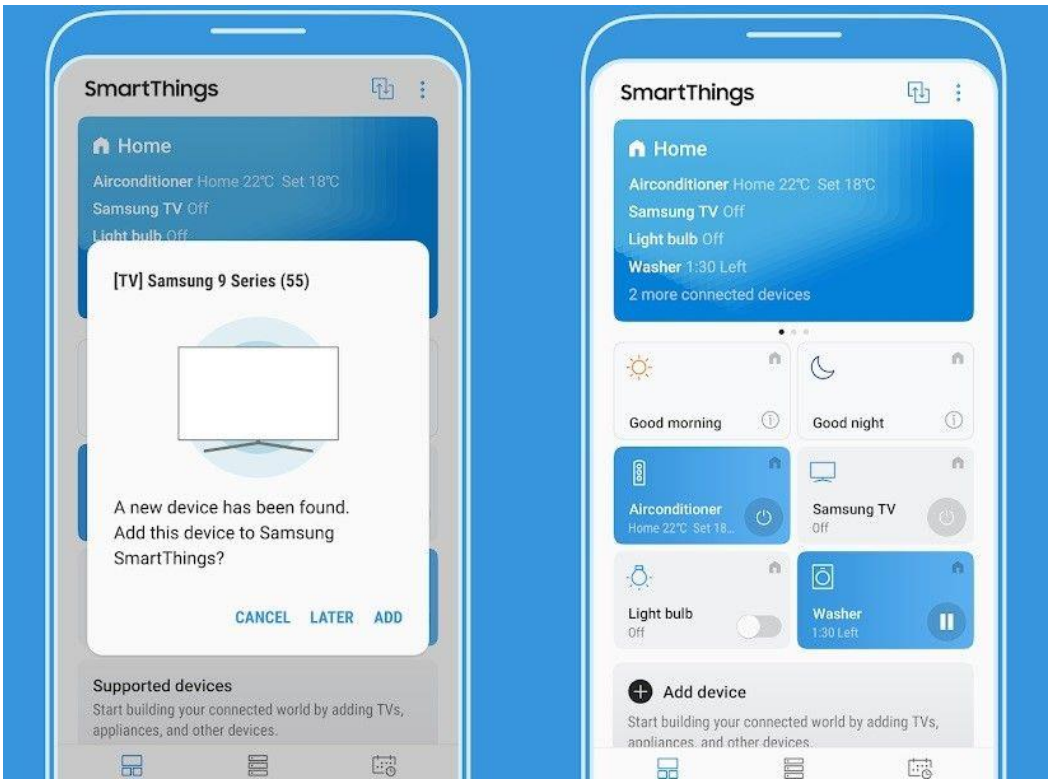

Рисунок 1.1 – Інтерфейс додатку SmartThings

Google Home – додаток від Google, який пропонує відеоспостереження, давачі руху, а також можливість керування системами розумного будинку через мобільний додаток або голосовими командами. Він інтегрується з іншими пристроями розумного будинку, такими як термостати і освітлення [10]. Переваги даного додатку:

1. Інтеграція з екосистемою Google: Google Home повністю інтегрований з екосистемою Google, що означає, що ви можете використовувати його для доступу до різних послуг Google, таких як Google Assistant, Google Calendar, Google Maps та інших.
2. Широкий вибір сумісних пристроїв: Google Home підтримує велику кількість різних пристроїв, що працюють з технологією "розумного дому". Ви можете підключити до нього розумні давачі, камери, освітлення, термостати та багато інших.
3. Голосове керування: Google Home має потужний голосовий асистент Google Assistant, що дозволяє керувати вашим розумним будинком

просто за допомогою голосових команд. Ви можете запитувати погоду, контролювати підключені пристрої, ставити таймери та багато іншого.

Недоліками Google Home є висока вартість, обмежена сумісність, залежність від інтернету, а також те, що цей додаток не має добре налаштованої і автоматизованої системи оновлень, яка регулярно перевіряє наявність нових загроз безпеці.

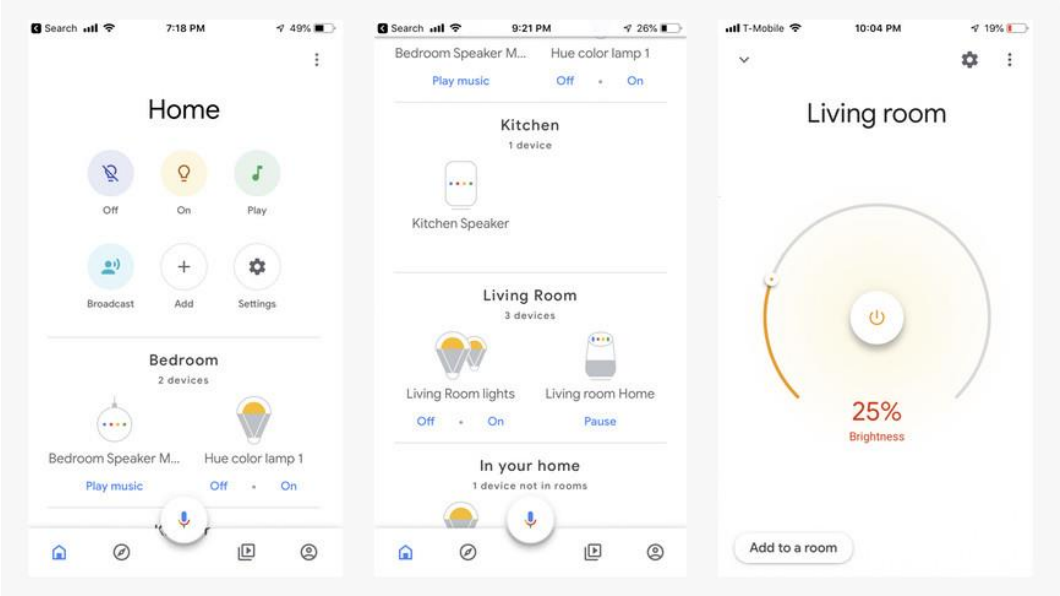

Рисунок 1.2 – Інтерфейс додатку Google Nest Secure

Vivint Smart Home є комплексною системою домашньої автоматизації, яка надає широкий спектр функцій та пристроїв для забезпечення комфорту, безпеки та енергоефективності. Вона включає в себе сигналізацію, камери відеоспостереження, давачі руху та інші пристрої. Користувачі можуть керувати системою через мобільний додаток або голосові команди [11]. Перевагами Vivint Smart Home є:

1. Інтегрована безпека: Vivint Smart Home пропонує комплексні рішення безпеки, включаючи систему безпеки з давачами руху, давачами дверей та вікон, камерами відеоспостереження та вуглекислотними давачами. Ви можете моніторити та керувати безпекою вашого дому з

використанням мобільного додатку або віддалено через хмарну платформу.

2. Автоматизація розумного дому: Vivint Smart Home дозволяє автоматизувати різні аспекти вашого дому, такі як освітлення, термостати, замки на дверях та інші пристрої. Ви можете створювати розумні сценарії, розклади та режими, щоб оптимізувати енергоспоживання, підвищити комфорт та заощадити час.
3. Професійна установка та підтримка: Vivint Smart Home пропонує професійну установку та підтримку, що забезпечує правильне налаштування пристроїв та гарантує їх надійну роботу. Крім того, ви можете отримувати технічну підтримку та консультації від експертів Vivint.

Проте Vivint Smart Home має характерні недоліки: високу вартість, залежність від інтернету, обмеження в сумісності зі сторонніми пристроями, та інші.

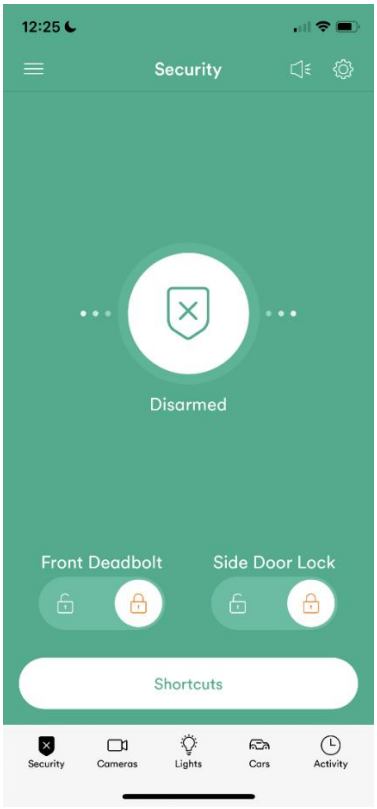

Рисунок 1.3 – Інтерфейс додатку Vivint Smart Home

Розглянемо алгоритм роботи програмного засобу з управління безпекою розумного будинку. Існуючі засоби з управління безпекою розумного будинку в основному працюють за схожим алгоритмом. Усі вони використовують ряд основних функцій, таких як:

1. Авторизація, що визначає, які користувачі чи пристрої мають доступ до різних систем та пристроїв в будинку.
2. Аналіз безпеки, який передбачає постійне відстеження та оцінку стану безпеки розумного будинку, що дозволяє виявляти загрози безпеці.
3. Відображення інформації про виявленні загрози, що передбачає інформування користувачів про виявлені загрози та їхній стан.

Такий функціонал допомагає розумному будинку забезпечувати ефективне управління безпекою та забезпечувати користувачам інформацію та контроль над станом безпеки. UML-діаграма алгоритму управління безпекою зображена на рис. 1.4.

Рисунок 1.4 – UML-діаграма алгоритму управління безпекою

Проте, даний алгоритм є узагальненим та потребує покращення. Таким чином, доцільна розробка інформаційної технології з удосконаленим алгоритмом управління безпекою розумного будинку, за рахунок наявності функціоналу створення сценарії безпеки, що дозволить власнику розумного будинку, завчасно прописати дії, які виконуватимуться у разі виявлення тієї чи іншої загрози безпеці, що підвищить швидкість реагування на загрозу безпеці.

Проаналізувавши існуючі засоби управління безпекою розумного будинку створено порівняльну таблицю 1.1, в якій наведено порівняння основних характеристик програмних засобів управління безпекою розумного будинку.

Таблиця 1.1 – Характеристика програмних засобів управління безпекою розумного будинку

Характеристика	Samsung SmartThings	Google Home	Vivint Smart Home
Зрозумілий інтерфейс	+	+	+
Сумісність з іншими пристроями	+	+	-
Інтеграція з системами	+	+	+
Незалежність від інтернету	-	-	-
Можливість створення сценаріїв безпеки	-	-	-
Автоматизація	-	-	+
Легкість налаштування	-	+	-

Враховуючи порівняльну характеристику існуючих програмних засобів управління безпекою розумного будинку, засіб управління безпекою розумного будинку Google Home може бути використаний для процесу управління безпекою розумного будинку, але він потребує покращень, за рахунок підвищення швидкості реакції на загрозу безпеці та розширення функціоналу. Таким чином, доцільним є створення відповідної інформаційної технології, яка надаватиме можливість користувачу створити набір сценаріїв з заходами безпеки та забезпечить надійне управління безпекою розумного будинку за рахунок виявлення потенційних загроз безпеці та підвищить швидкість реакції на загрозу.

1.3 Постановка задачі

Для розробки ефективної технології управління безпекою розумного будинку, яка буде враховувати стан безпеки, введемо такі позначення:

Нехай, $S_i(t)$ – i-ий стан безпеки в момент часу t, що приймає значення від 0 до 1.

Тоді, задача управління безпекою розумного будинку зводиться до визначення:

$$F(F_j(S_i(t))),$$

де $F(F_j(S_i(t)))$ – це множина сценаріїв безпеки за i-м станом $S_i(t)$.

Кожна функція $F_j(S_i(t))$ визначає конкретний j-й сценарій або набір дій, які користувач встановлює для технології управління безпекою розумного будинку за i-м станом $S_i(t)$ в разі виявлення потенційної загрози.

1.4 Висновок до розділу 1

В даному розділі було досліджено особливості використання програмних засобів для управління безпекою розумного будинку. Проведено аналіз сучасних засобів управління безпекою розумного будинку та наведено їх порівняльну характеристику. Також було описано основні функції існуючих програмних засобів управління безпекою розумного будинку. Враховуючи це, було визначено актуальність розробки інформаційної технології управління безпекою розумного будинку, яка забезпечить надійність захисту розумного будинку з урахуванням результатів аналізу безпеки розумного будинку, а також надаватиме можливість власнику розробити набір заходів безпеки.

2 РОЗРОБКА МЕТОДУ ТА СТРУКТУРИ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ БЕЗПЕКОЮ В РОЗУМНОМУ БУДИНКУ

2.1 Розробка удосконаленої математичної моделі процесу управління безпекою розумного будинку

Для удосконалення математичної моделі процесу управління безпекою розумного будинку введемо додаткові характеристики.

Нехай $S_i(t)$ це i-й стан безпеки будинку в момент часу t.

Тоді, стан безпеки можна описати з використанням двійкової логіки, а саме: $S_i(t) = 1$, якщо є потенційна загроза, та $S_i(t) = 0$, якщо загрози немає. $Sensor(t)$ $\{Sensor_1(t), Sensor_2(t), ..., Sensor_n(t)\}$ – вектор значень давачів, необхідних для аналізу стану безпеки, де $Sensor_i(t)$ представляє вимірювання i-го давача в момент часу t.

Таким чином, вектор факторів безпеки $V_i(t)$ у i-му стані безпеки може бути визначений як функція від значень різних давачів, які впливають на стан безпеки засобу:

$$V_i(t) = f(Sensor_1(t), Sensor_2(t), ..., Sensor_n(t)).$$

Нехай $A(S_i(t))$ буде алгоритмом аналізу стану безпеки в момент часу t. Він визначає, чи є потенційна загроза на основі отриманих даних. Таким чином, реакція на події залежить від аналізу, проведеного алгоритмом $A(S_i(t))$, і налаштувань користувача. Якщо $A(S_i(t))$ виявляє загрозу, і якщо користувачі налаштували сценарій $F_j(S_i(t))$, то відбувається відповідна реакція на загрозу.

Стан безпеки $S_i(t)$ може оновлюватися в момент часу $(t + dt)$, де dt – це інтервал часу протягом якого оновлюється стан безпеки на основі отриманих даних та виконаних сценаріїв.

Отже в кожний наступний момент часу з урахуванням факторів впливу g та відповідно обраних сценаріїв безпеки розумного будинку за i-м станом $S_i(t)$, визначатиметься як:

$$S_i(t+dt) = g(V_i(t), F_1(S_i(t)), F_2(S_i(t)), ..., F_n(S_i(t))),$$

де $g(V_i(t), F_1(S_i(t)), F_2(S_i(t)), ..., F_n(S_i(t)))$ – вектор факторів впливу на стан безпеки.

Запропонована математична модель дозволяє визначити взаємозв'язок між станом безпеки, користувацькими сценаріями та аналізом безпеки. Формули надають точну характеристику того, як сенсори та інші фактори впливають на стан безпеки. Зокрема, вона враховує багатоаспектність безпеки, оскільки стан безпеки визначається не лише одним, але різними давачами. Формули також конкретизують, як користувацькі сценарії впливають на реакцію технології на потенційні загрози так, як кожен сценарій $F_j(S_i(t))$ представляє собою визначений користувачем набір дій, які виконуються в залежності від стану безпеки та налаштувань користувача.

Оновлення стану безпеки $S_i(t+dt)$ визначається функцією g та враховує дані сенсорів та виконані користувацькі сценарії. Це надає можливість адаптуватися до змін у довкіллі та користувацьких уподобань. Така модель може бути адаптована та розширена відповідно до конкретних потреб та вимог користувача, надаючи гнучкість та персоналізацію в управлінні безпекою розумного будинку.

Таким чином, дана математична модель дозволить підвищити швидкість реагування на загрозу безпеці за рахунок постійного аналізу безпеки та реакції на події, що можуть її порушити.

2.2 Розробка узагальненого алгоритму для управління безпекою розумного будинку

Для удосконалення алгоритму управління розумним будинком доцільно додати до алгоритму функціонал, який дасть користувачеві можливість створення індивідуальних сценаріїв безпеки, які б зберігались в базі даних інформаційної технології, а також проведення аналізу безпеки розумного будинку, що дасть можливість виявлення вразливостей систем безпеки розумного будинку, виявлення загрози та визначення її типу. Після проведення аналізу безпеки розумного будинку та за умови ідентифікації певної загрози, буде виконуватись відповідний сценарій безпеки.

Під час розробки узагальненого алгоритму управління безпекою розумного будинку необхідно забезпечити виконання таких функцій:

1. Створення акаунту власника розумного будинку та ідентифікація:

Крок 1. Реєстрація користувача.

Користувач реєструє свій акаунт, надаючи особисті дані та обираючи унікальне ім'я користувача та пароль.

Крок 2. Підтвердження ідентифікації.

Після реєстрації відбувається перевірка і підтвердження ідентифікації користувача шляхом. Ідентифікація та аутентифікація - це перші кроки для забезпечення безпеки користувачів та їх даних в системі розумного будинку. Це дозволить зберігати та контролювати доступ до особистої інформації та функцій.

2. Створення індивідуальних сценаріїв безпеки та перегляд існуючих:

Крок 1. Додавання сценаріїв.

Користувач може створювати індивідуальні сценарії безпеки, де він визначає умови та дії для конкретних ситуацій. Наприклад, сценарій може передбачати активацію сигналізації, якщо двері відкриваються після певного часу.

Крок 2. Управління сценаріями.

Користувач може переглядати, редагувати та видаляти існуючі сценарії безпеки, а також вимикають або активувати їх за потреби.

Сценарії безпеки дають користувачеві можливість налаштовувати інформаційну технологію управління безпекою розумного будинку на свій смак і вимоги. Це робить технологію більш гнучкою та відповідною до конкретних потреб користувача.

3. Проведення аналізу безпеки розумного будинку:

Крок 1. Збір інформації про параметри безпеки.

Інформаційна технологія збирає дані від різних давачів та систем безпеки в розумному будинку, такі як стан дверей, вікон, давачі руху, димові давачі тощо.

Крок 2. Ідентифікація можливих загроз.

Інформаційна технологія аналізує результати та ідентифікує можливі загрози безпеці, які можуть виникнути на основі поточного стану безпеки розумного будинку.

Крок 3. Обчислення рівня безпеки.

Використовуючи зібрані дані, інформаційна технологія обчислює загальний рівень безпеки.

Аналіз безпеки дозволяє користувачам отримувати інформацію про стан системи розумного будинку, її потенційні вразливості та загрози в реальному часі, що допомагає приймати вчасні рішення та активувати відповідні сценарії безпеки.

4. Відображення результатів актуальної аналітики безпеки та виконання певного сценарію безпеки:

Крок 1. Відображення результатів.

Інформаційна технологія відображає користувачу результати аналізу безпеки, вказуючи на потенційні загрози та рівень безпеки.

Крок 2. Виконання сценаріїв безпеки.

Якщо існують активні сценарії безпеки, які вимагають реакції, інформаційна технологія автоматично активує відповідні дії для запобігання можливим загрозам. Відображення результатів та виконання сценаріїв безпеки допомагає користувачу зберігати контроль над безпекою свого розумного будинку та приймати вчасні заходи для запобігання небезпеці.

Функціонал розроблюваної інформаційної технології об’єднано в такі групи: авторизація, функціонал створення сценаріїв безпеки, функціонал аналізу безпеки розумного будинку, функціонал аналітики, функціонал виконання сценарію безпеки. На основі описаного функціоналу розроблено узагальнений алгоритм для управління безпекою розумного будинку, який включає у себе усі перераховані функції. UML-діаграма алгоритму наведена на рисунку 1.

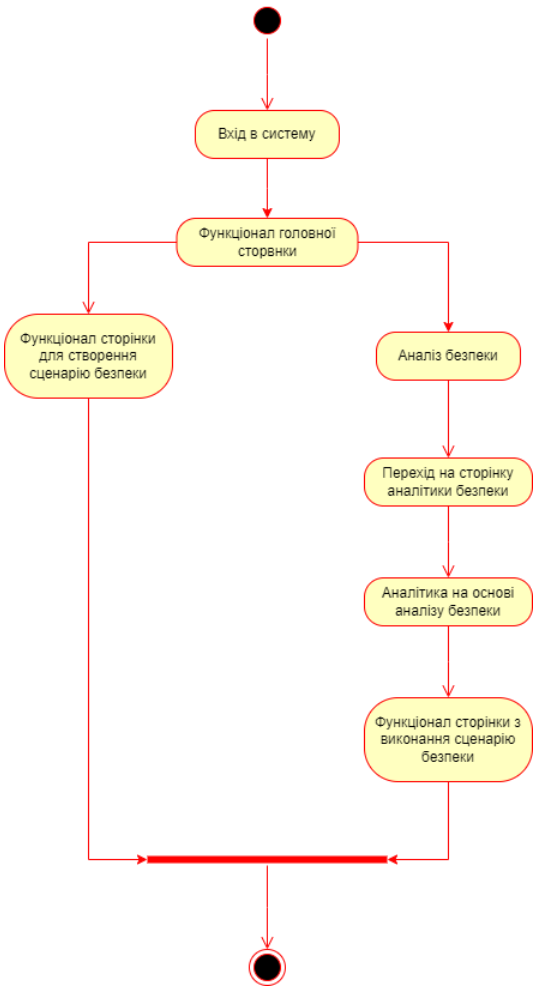

Рисунок 2.1. – UML-діаграма активності для управління безпекою розумного будинку

Таким чином, сформовано узагальнений алгоритм для управління безпекою розумного будинку з урахуванням зазначеного функціоналу, який забезпечить постійний аналіз безпеки та створення набору сценаріїв безпеки, що дозволить підвищити швидкість реакції на загрозу.

2.3 Розробка структури інформаційної технології управління безпекою розумного будинку

Реалізація запропонованого алгоритму для управління безпекою розумного будинку можлива за наявністю у складі інформаційної технології таких модулів:

1. Авторизація.
2. Аналіз безпеки.
3. Сценаріїв безпеки.
4. Аналітика безпеки.
5. Виконання сценарію безпеки.

Модуль авторизації отримує першочергову інформацію про користувача, який хоче ввійти до інформаційної технології та перевіряє чи є він поточним користувачем, відповідно до чого дає доступ до основного функціоналу або повертає помилку. Модуль сценаріїв безпеки дозволяє додавати сценарії безпеки шляхом отримання інформації від користувача через користувацький інтерфейс, редагувати та видаляти їх за потреби. Модуль аналізу безпеки отримує інформацію про поточний стан систем розумного будинку та проводить аналіз вразливостей. А також ідентифікує наявні загрози безпеці за рахунок отримання інформації з відповідних давачів або камер. Модуль аналітики безпеки отримує дані про стан систем розумного будинку та формує звіт про стан безпеки за певний проміжок часу, а також надає оцінку критичності небезпеки у разі її виявлення. Модуль виконання сценарію безпеки отримує інформацію про оцінку

критичності небезпеки, відповідно до чого запускає алгоритм дій прописаний в відповідному сценарію безпеки.

Структура інформаційної технології управління безпекою розумного будинку наведена на рисунку 2.2.

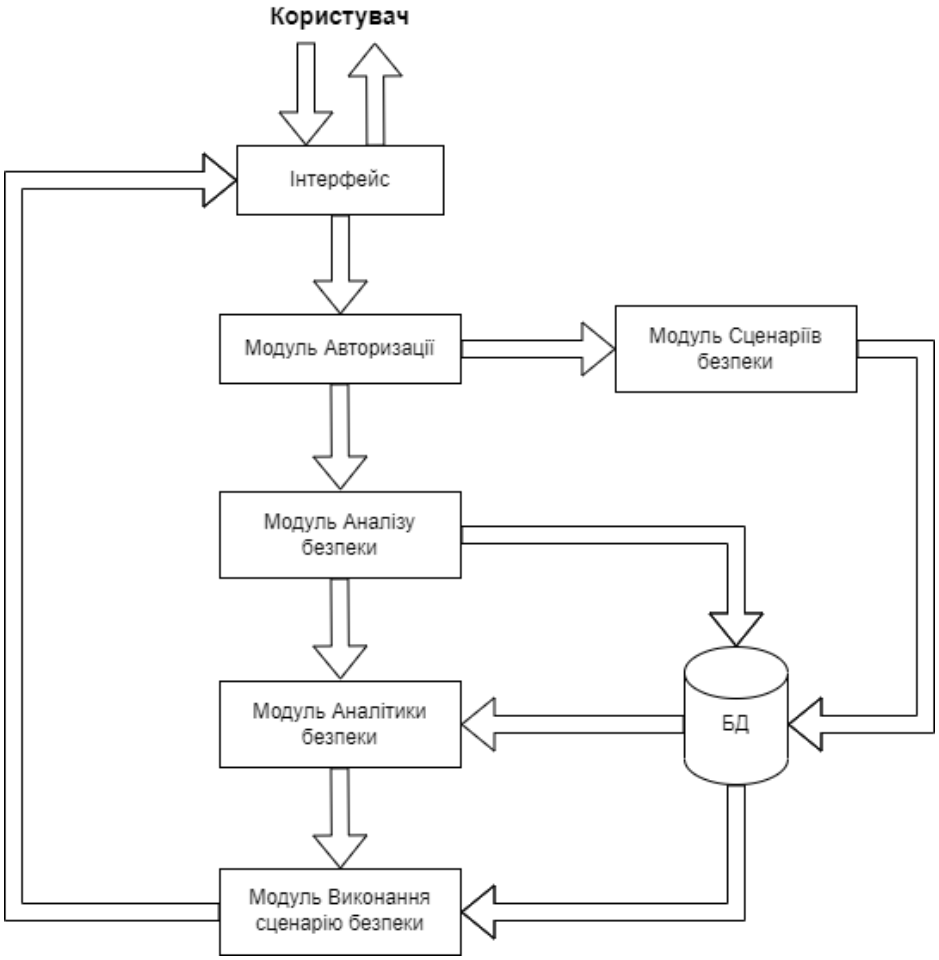

Рисунок 2.2 – Структура інформаційної технології управління безпекою розумного будинку

Взаємодія користувача з певним модулем відбувається через зрозумілий інтерфейс. Першим модулем з яким взаємодіє користувач є модуль «Авторизації» У користувача є можливість зареєструватись та увійти до інформаційної технології. Після входу до інформаційної технології користувач має можливість взаємодіяти з такими модулями, як: модуль «Сценаріїв безпеки», у якому необхідно створити власні сценарії безпеки, модуль «Аналіз безпеки»,

за допомогою якого буде відбуватись аналіз безпеки розумного будинку та виявлення можливих вразливостей безпеки та потенційних загроз безпеці розумного будинку, модуль «Аналітики безпеки», який буде надавати можливість спостерігати за аналітичними даними зібраними в ході роботи інформаційної технології, а також модуль «Виконання сценарію безпеки». Модулі «Сценарії безпеки», «Аналіз безпеки» та «Аналітика безпеки» будуть взаємодіяти з базою даних для відправки та отримання певних необхідних даних.

Отже, запропонована структура інформаційної технології управління безпекою розумного будинку розширить функціонал аналізу безпеки в існуючих додатках з управління безпекою розумного будинку, а також функціонал реагування на виявлену загрозу безпеці розумного будинку за рахунок наявності модулів «Сценарії безпеки», «Аналіз безпеки» та «Виконання сценарію безпеки».

2.4 Висновок до розділу 2

Розроблено удосконалену математичну модель процесу управління безпекою розумного будинку, яка дасть можливість визначити взаємозв'язок між станом безпеки, користувацькими сценаріями та алгоритмом аналізу. Сформовано узагальнений алгоритм для управління безпекою розумного будинку. Розроблено структуру інформаційної технології управління безпекою розумного будинку розширить функціонал аналізу безпеки в існуючих додатках з управління безпекою розумного будинку, а також функціонал реагування на виявлену загрозу безпеці розумного будинку за рахунок наявності модулів «Сценарії безпеки», «Аналіз безпеки» та «Виконання сценарію безпеки», та забезпечить надійне управління безпекою розумного будинку за рахунок своєчасного виявлення потенційних загроз безпеці та створеного набору сценаріїв власником будинку за допомогою чого вчасно відбуватиметься ідентифікація загрози та реагування на неї.

3 РОЗРОБКА МОДУЛІВ ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ БЕЗПЕКОЮ В РОЗУМНОМУ БУДИНКУ

3.1 Обґрунтування вибору мови програмування

Для розробки інформаційної технології управління безпекою розумного будинку необхідно створити клієнт-серверну архітектуру. Ця архітектура дозволить ефективно керувати та моніторити аспекти безпеки розумного будинку через взаємодію між клієнтськими пристроями та центральним сервером. Вибір мови програмування для реалізації клієнт-серверної частини є ключовим, оскільки вона відповідає за забезпечення зручного та безпечного взаємодії користувача та забезпечення комунікацій. Вибір мови програмування повинен враховувати такі фактори, як масштаб проекту, безпека, доступність бібліотек та фреймворків, а також зручність розробки та підтримки програмного забезпечення. Розглянемо декілька варіантів мов програмування, які можуть реалізувати серверну та клієнтську частину інформаційної технології управління безпекою розумного будинку.

Для інформаційної технології управління безпекою розумного будинку розглянемо такі варіанти мов програмування, як: C++, C#, PHP, JavaScript, Java, та оберемо найліпший варіант.

C++ – мова програмування загального призначення з підтримкою кількох парадигм програмування, таких як об’єктно-орієнтованої, узагальненої, процедурної та інших. Мова C++ включає в себе можливість створення класів і об'єктів для об'єктно-орієнтованого програмування, підтримку шаблонів (templates), що дозволяють створювати загальні алгоритми та структури даних, інкапсуляцію, наслідування та поліморфізм для підвищення модульності [12].

C# (C Sharp) – об’єктно орієнтована мова програмування з безпечною системою типізації для платформи.NET. Синтаксис C# близький до C++ і Java. Мова має строгу статичну типізацію, підтримує поліморфізм, перевантаження

операторів, вказівники на функції-члени класів, атрибути, події, властивості, винятки, коментарі у форматі XML [13].

PHP (Personal Home Page Tools) – скриптова мова програмування, яка була створена для генерації HTML-сторінок на стороні web-сервера. Основні особливості PHP включають в себе вбудовану підтримку для роботи з веб-серверами, що дозволяє легко обробляти HTTP-запити і відправляти HTTP-відповіді та підтримку сесій, що дозволяє зберігати стан інформації між різними HTTP-запитами [14].

JavaScript (JS) – динамічна, об’єктно-орієнтована прототипна мова програмування, яка найчастіше використовується для створення сценаріїв web-сторінок. Являється скриптовою мовою програмування з динамічною типізацією. Мова програмування JavaScript включає в себе здатність маніпулювати DOM (Document Object Model) - структурою веб-сторінки, що дозволяє динамічно змінювати вміст та структуру сторінки, роботу з AJAX (Asynchronous JavaScript and XML), що дозволяє асинхронно взаємодіяти з сервером і оновлювати сторінки без перезавантаження, а також велику кількість вбудованих об'єктів і функцій, які спрощують роботу з рядками, числами, масивами, датами і багатьма іншими типами даних [15].

Java – це об'єктно-орієнтована мова програмування, яка створює програмне забезпечення для кількох платформ. Має менше низькорівневих можливостей для роботи з апаратним забезпеченням, що зменшує швидкість роботи. Java включає в себе різні механізми безпеки, включаючи обмеження доступу до ресурсів системи та автоматичне управління пам'яттю для запобігання витокам пам'яті та іншим проблемам [16].

Обрана мова програмування має забезпечувати в першу чергу маштабованість, швидкодію та надійність. Таким чином, обрана мова програмування має бути суворо типізованою, надійною та забезпечувати маштабованість. Також має забезпечувати безпеку та доступність фреймворків. Порівняння мов програмування наведено у таблиці 3.1.

Таблиця 3.1 – Порівняння мов програмування для реалізації інформаційної технології управління безпекою розумного будинку

Мова	Маштабованість	Швидкодія	Надійність	Безпека	Сувора типізація	Доступність фреймворків
C++	-	-	+	+	+	+
C#	+	-	+	+	+	+
PHP	-	-	-	+	+	-
JavaScript	+	+	+	+	+	+
Java	-	-	-	+	+	-

Аналізуючи порівняльну таблицю, дійдено висновку, що JavaScript є найкращим вибором для розробки інформаційної технології управління безпекою розумного будинку. Дана мова програмування забезпечує маштабованість, швидкодію, є суворо типізованою та надійною, а також забезпечує доступність до фреймворків. Для розробки серверної частини буде використовуватись платформа NodeJS з пакетом суворої типізації TypeScript. Ця платформа призначена для виконання високопродуктивних мережевих застосунків та окрім роботи із серверними скриптами для web-застосунків також використовується для створення клієнт-серверних програм [17]. З використанням фреймворку NestJS процес реалізації інформаційної технології буде спрощено так, як NestJS сприяє побудові додатків у вигляді модулів, що спрощує організацію коду і дозволяє розробникам розділяти функціональність на невеликі, незалежні частини [18].

Розглянемо мову JavaScript та її доповнений та типізований варіант – TypeScript.

TypeScript — мова програмування, що позиціонується як засіб розробки вебзастосунків, що розширює можливості JavaScript. Особливості TypeScript включають в себе те, що TypeScript дозволяє вказувати типи для змінних, функцій, параметрів та інших об'єктів у коді. Це допомагає виявляти помилки на ранніх етапах розробки і поліпшує надійність програми. А також, TypeScript

підтримує об'єктно-орієнтований підхід до програмування, що дозволяє створювати класи, інтерфейси та успадкування та дозволяє організовувати код у модулі для полегшення керування структурою додатку [19].

Порівняння цих мов програмування з урахуванням суворої типізації, швидкодії, та надійності, наведено у таблиці 3.2.

Таблиця 3.2 - Порівняльна характеристика мов JavaScript та Typescript.

Мова	Швидкодія	Сувора типізація	Надійність
JavaScript	+	-	-
Typescript	+	+	+

Таким чином, враховуючи порівняльну характеристику, визначено, що TypeScript має більше переваг над JavaScript, а отже краще підійде для розробки клієнтської частини інформаційної технології управління безпекою розумного будинку.

Для оптимізації процесу розробки рекомендовано використовувати фреймворки, які надають готовий функціонал для розширення, що сприяє прискоренню розробки. Для створення клієнтської частини вибрали Angular, який надає розробникам потужні інструменти та компоненти для створення веб-додатків. Angular – фреймворк вищої версії розроблений на TypeScript з відкритим кодом. Він найчастіше використовується для створення великих програмних засобів, так як має чітко-визначену архітектуру та використовує компонентний підхід. Angular налічує в собі майже 95% готового функціоналу, що значно пришвидшуватиме процес розробки клієнтської частини програмного засобу [20].

При виборі системи управління базами даних (СУБД) слід звертати увагу на фактори, такі як швидкодія та надійність. Враховуючи ці аспекти, розумно віддавати перевагу реляційним базам даних, оскільки саме цей тип баз даних відзначається високою ступенем надійності. Реляційна база даних — база даних, заснована на реляційній моделі даних [21]. Для реалізації було обрано

PostgreSQL – об’єктно-реляційну систему керування базами даних. Перевагою цієї системи є те, що PostgreSQL може бути розширено користувачем для власних потреб практично в будь-якому аспекті [22].

3.2 Обґрунтування вибору середовища розробки інформаційної технології управління безпекою розумного будинку

При виборі середовища розробки інформаційної технології управління безпекою розумного будинку, важливо враховувати обрану мову програмування, оскільки вона визначає вибір інструментів, які найкраще підходять для створення даної інформаційної технології.

Інформаційна технологія управління безпекою розумного будинку – web-застосунок, таким чином, доцільно використовувати середовище розробки Microsoft Visual Studio. Це інтегроване середовище розробки (IDE) від компанії Microsoft, призначене для створення різних типів програмних додатків, включаючи web-застосунок, із зручним інтерфейсом та широким функціоналом [23].

Visual Studio Code – середовище, яке орієнтоване на створення, редагування та зневадження сучасних web-застосунків [24]. Вибір середовища розробки, такого як Visual Studio Code, для розробки інформаційної технології управління безпекою розумного будинку на мові JavaScript має численні обґрунтовані переваги:

1. Безкоштовність і відкритий код: VS Code є безкоштовним і відкритим програмним забезпеченням, що робить його доступним для будь-якого розробника без витрат на ліцензії.
2. Підтримка мови JavaScript: VS Code має вбудовану підтримку JavaScript, яка включає функції, такі як автодоповнення, перевірка синтаксису та відлагодження, що сприяє зручності розробки.
3. Розширюваність: VS Code дозволяє розширювати його функціонал за допомогою різних розширень та плагінів. Розширення, пов'язані з

JavaScript та Angular, можуть значно полегшити роботу розробників та надати додатковий функціонал.

4. Крос-платформенність: VS Code підтримується на різних операційних системах, включаючи Windows, macOS та Linux, що дозволяє розробникам використовувати одне середовище незалежно від платформи.
5. Інтеграція з Git: VS Code має вбудовану підтримку системи контролю версій Git, що дозволяє командам розробників ефективно керувати версіями коду та спільно працювати над проектом.
6. Велика спільнота та підтримка: VS Code має велику та активну спільноту користувачів, що означає наявність багатьох онлайн-ресурсів, плагінів і підтримки.
7. Легка налаштуваність: VS Code дозволяє розробникам налаштовувати робоче середовище на свій смак, вибираючи необхідні розширення, теми та інші параметри.

Загалом, Visual Studio Code є раціональним вибором для розробки інформаційної технології управління безпекою розумного будинку на мові JavaScript через свою зручність, розширюваність та підтримку мови.

3.3 Розробка інтерфейсу інформаційної технології управління безпекою розумного будинку

Створення зручного користувацького інтерфейсу – це один із ключових кроків у розробці інформаційної технології управління безпекою розумного будинку. Важливо, щоб інтерфейс цієї технології був легким та інтуїтивно зрозумілим для користувачів, що спростить їх взаємодію з інформаційною технологією. Простий та інтуїтивний інтерфейс полегшує користувачам взаємодію з інформаційною технологією. Це дозволяє користувачам швидше розуміти, як взаємодіяти з нею та отримувати бажану інформацію. Також

зрозумілий інтерфейс допомагає уникати помилок при взаємодії з розроблюваною інформаційною технологією.

Інтерфейс користувача – це засіб зручної взаємодії користувача з інформаційною системою [25]. Інтерфейс користувача грає важливу роль у забезпеченні зручності та ефективності користувачів при використанні програми. Один з типів інтерфейсу користувача – це графічний інтерфейс. Це тип інтерфейсу, який дає змогу користувачам взаємодіяти з електронними пристроями через графічні компоненти екрану. У графічних системах інтерфейс користувача, втілюється багатовіконним режимом, змінами кольору, розміру, видимості (прозорість, напівпрозорість, невидимість) вікон, їхнім розташуванням, сортуванням елементів вікон, гнучкими налаштуваннями як самих вікон, так і окремих їх елементів (файли, теки, ярлики, шрифти тощо), доступністю багатокористувацьких налаштувань [26].

Для створення інтерфейсу інформаційної технології управління безпекою розумного будинку використано векторний сервіс розробки інтерфейсів та прототипування з можливістю організації спільної роботи, який працює як і в браузері так і на десктопі користувача – Figma [27]. Це популярний онлайн-інструмент для дизайну та співпраці над проектами, який має численні переваги:

1. Онлайн доступ: Figma доступний у веб-браузері, що дозволяє робити дизайн та спільно працювати над проектами з будь-якого місця, де є Інтернет.
2. Зручність: Figma має потужний набір інструментів для створення інтерфейсів, включаючи можливість створювати векторні графічні об'єкти, прототипи та анімацію.
3. Система компонентів: Figma дозволяє створювати компоненти та бібліотеки для подальшого використання, що спрощує роботу над дизайном та забезпечує консистентність.
4. Збереження та доступ до проектів: Всі зміни автоматично зберігаються, і користувачі мають постійний доступ до проектів через хмарне збереження.

На рис. 3.1 – 3.9. наведено скріншоти розробленого інтерфейсу інформаційної технології управління безпекою розумного будинку.

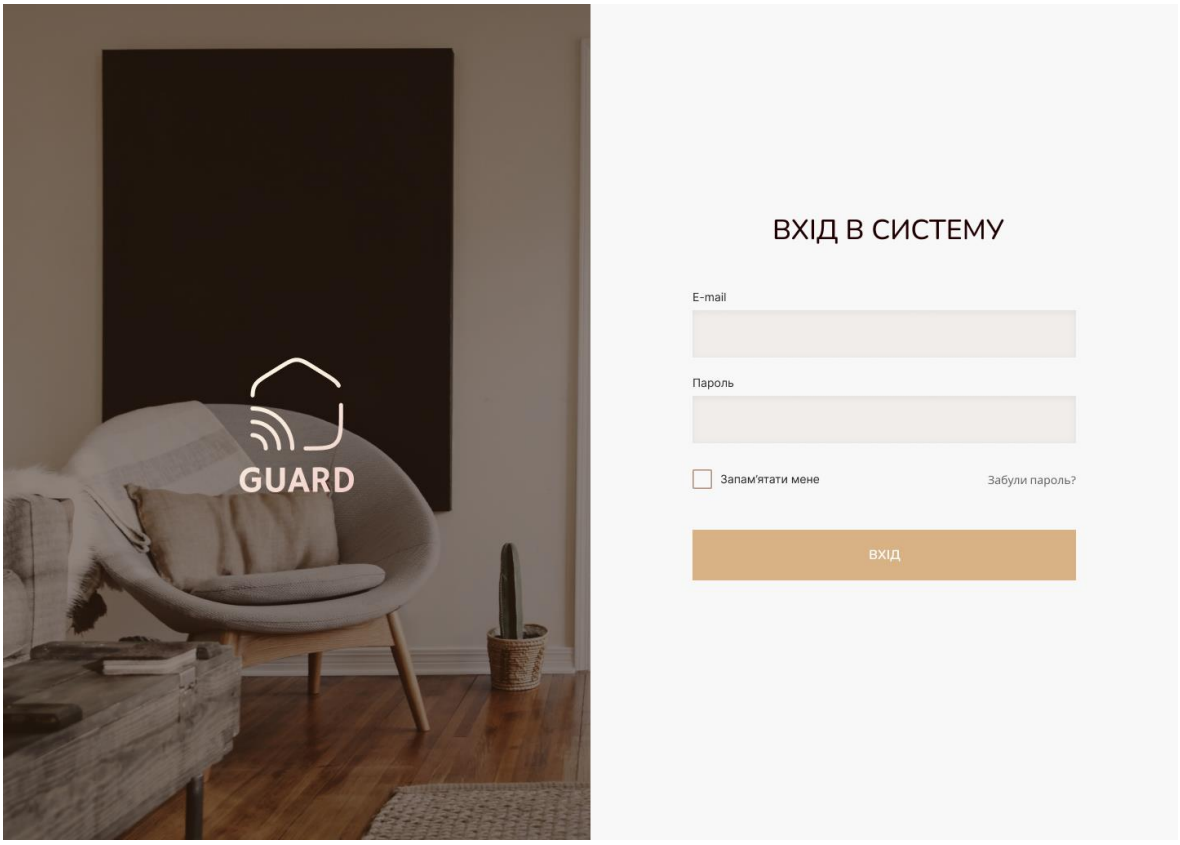

Рисунок 3.1 – Сторінка «Вхід»

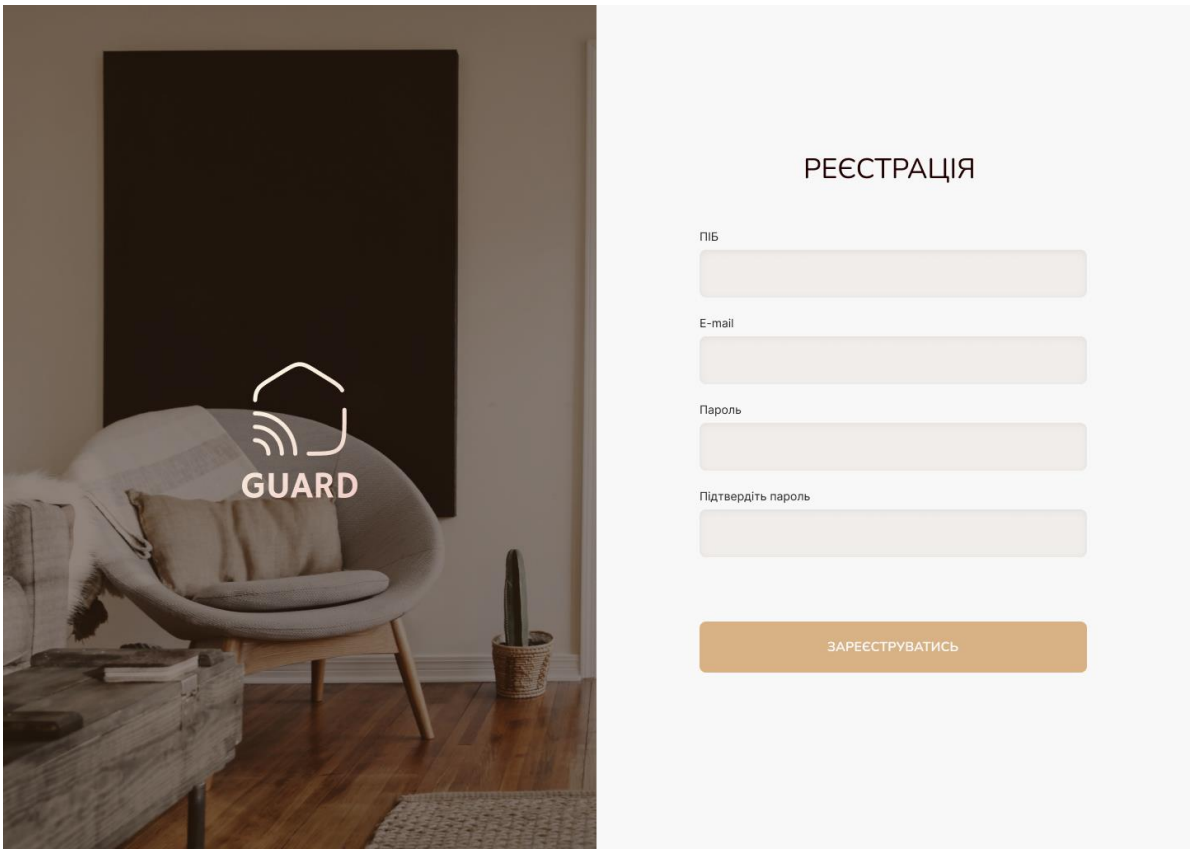

Рисунок 3.2 – Сторінка «Реєстрація»

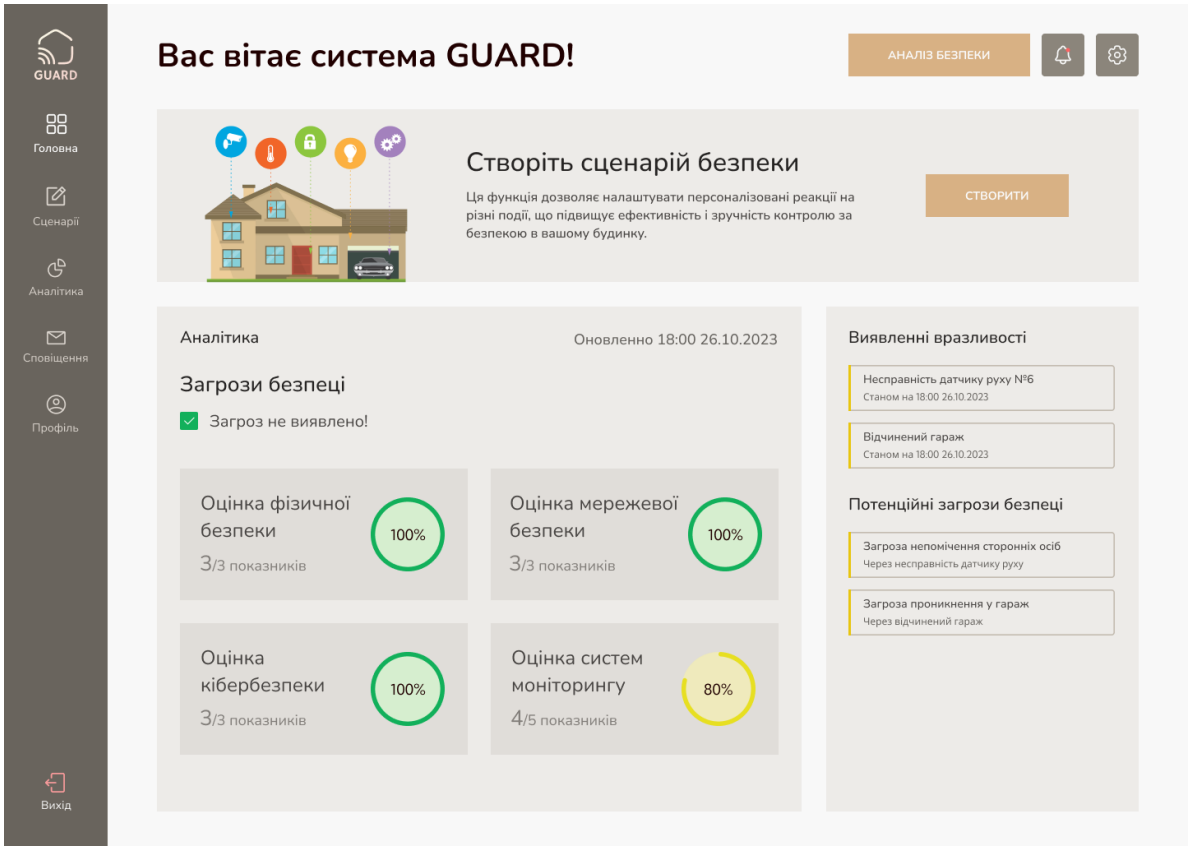

Рисунок 3.3 – Головна сторінка інформаційної технології

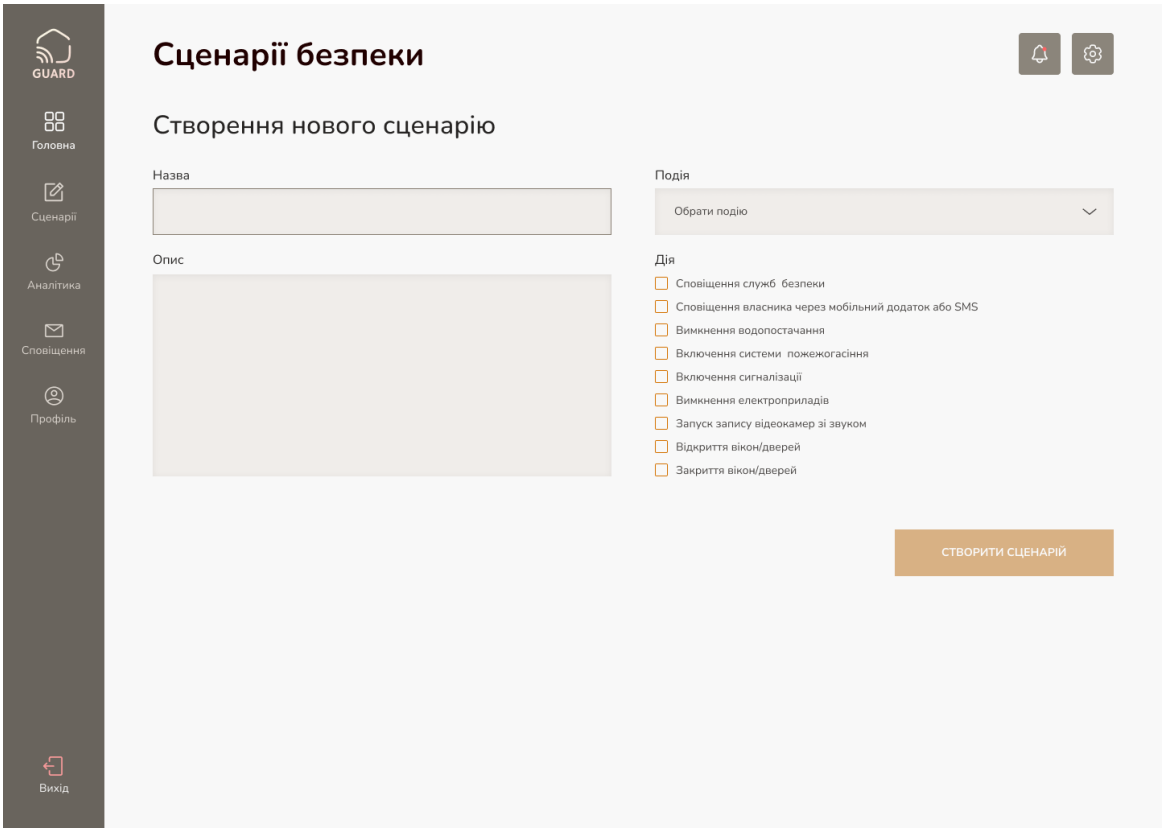

Рисунок 3.4 – Сторінка створення нового сценарію безпеки

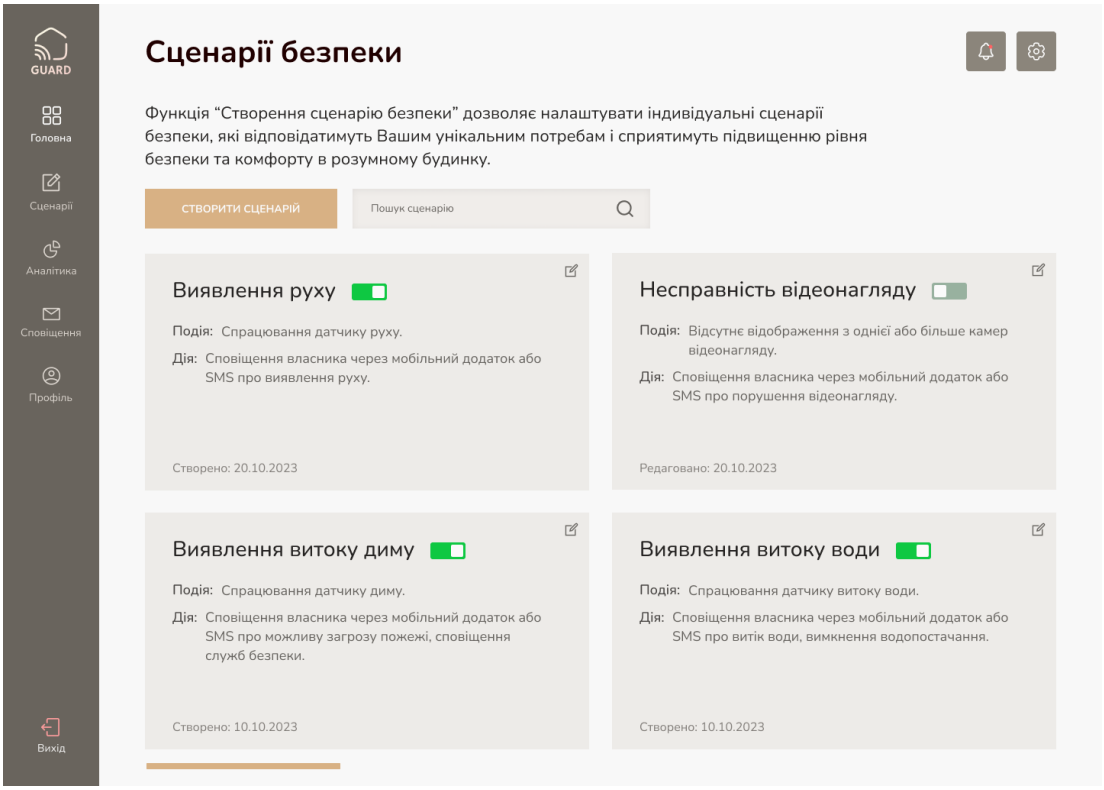

Рисунок 3.5 – Сторінка «Сценарії безпеки»

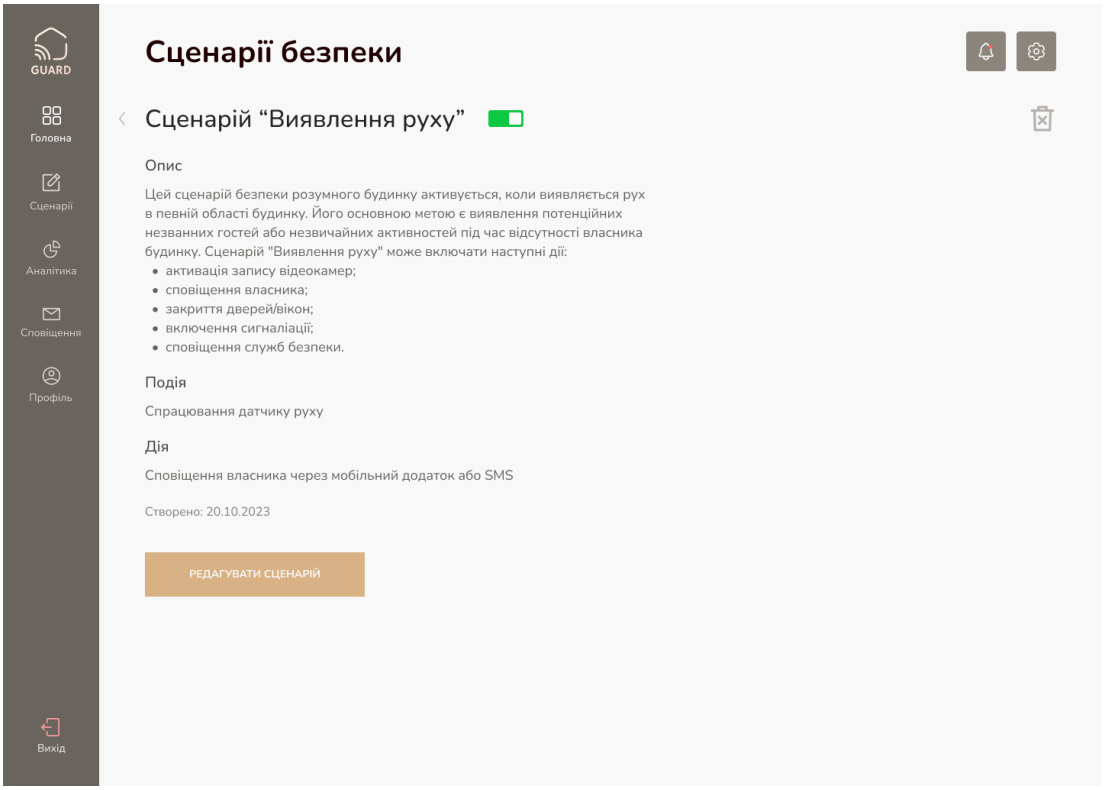

Рисунок 3.6 – Сторінка перегляду сценарію

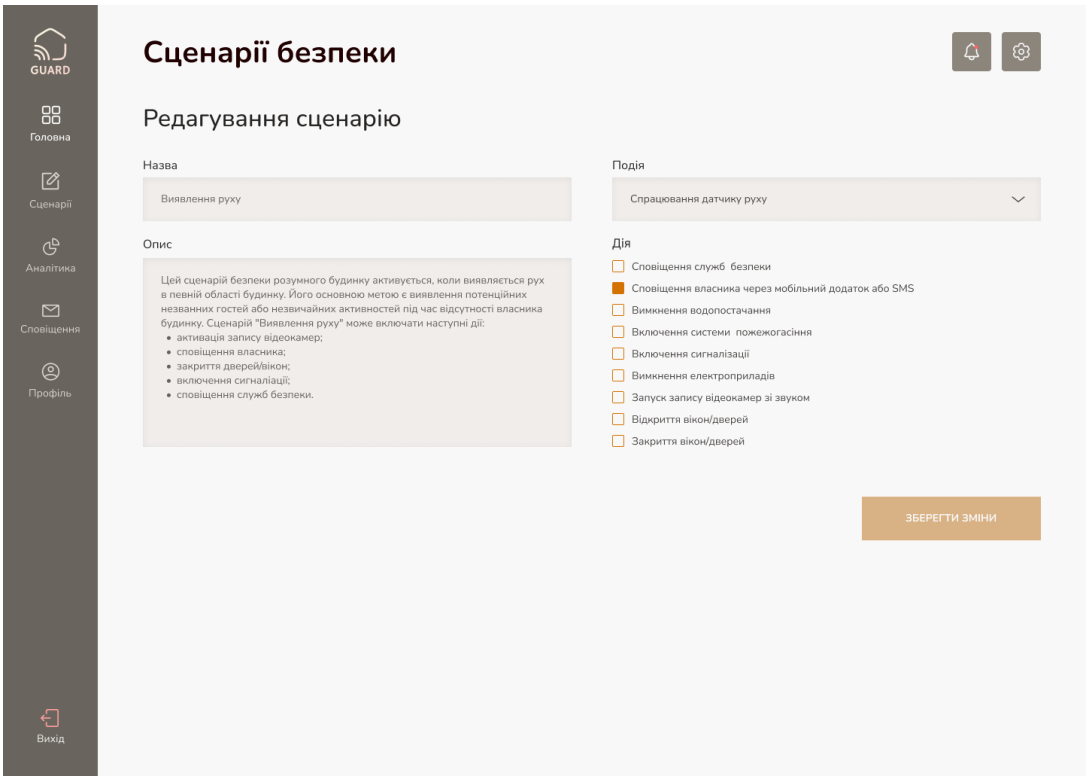

Рисунок 3.7 – Сторінка «Редагування сценарію»

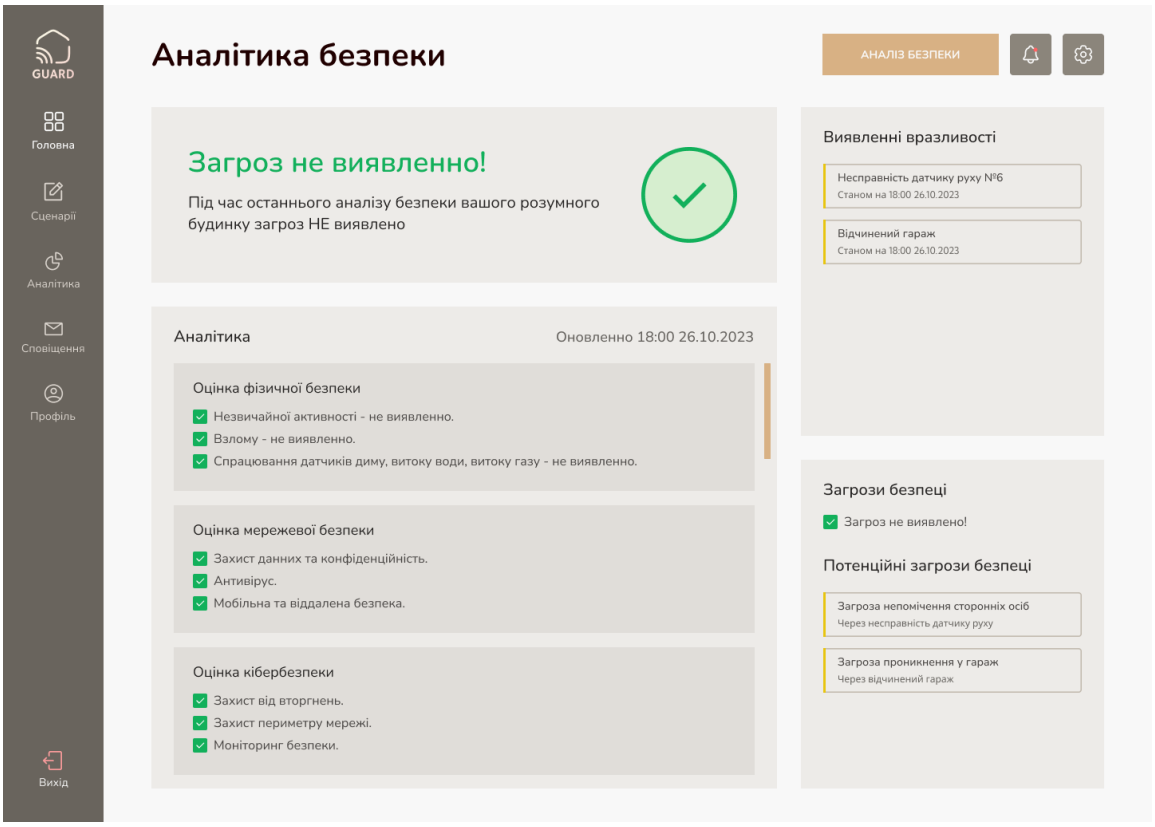

Рисунок 3.8 – Сторінка перегляду аналітики безпеки

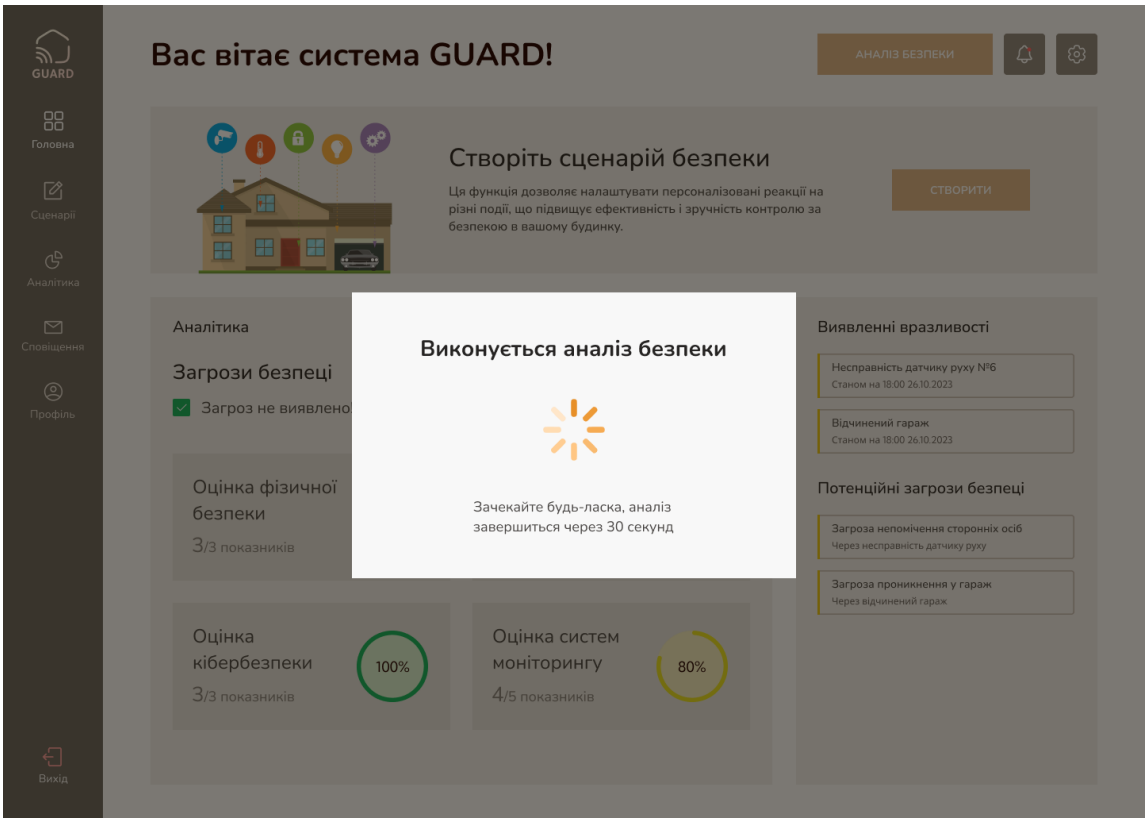

Рисунок 3.9 – Сторінка очікування виконання аналізу безпеки

3.4 Розробка функціонування модулів інформаційної технології управління безпекою розумного будинку

3.4.1 Модуль авторизації

Для того, щоб користувач мав можливість використовувати інформаційну технологію управління безпекою розумного будинку та виконувати необхідні дії, важливо пройти процес авторизації, який включає в себе реєстрацію користувача та вхід. Залежно від того, чи користувач уже зареєстрований, під час авторизації відбувається виконання конкретного функціоналу. Для входу до інформаційної технології функціонал має містити такі кроки:

Крок 1. Введення електронної адреси, яку користувач використовував при реєстрації.

Крок 2. Введення паролю, який користувач вказував при реєстрації.

Крок 3. Натиснення кнопки "Вхід".

Для реєстрації нового користувача функціонал передбачає наступні етапи:

1. Введення прізвища, ім’я та по батькові.
2. Введення електронної адреси.
3. Створення та підтвердження паролю.
4. Натискання кнопки "Зареєструватись".

UML-діаграма алгоритмів функціонування модуля авторизації зображена на рис. 3.10-3.11.

Фрагмент лістингу який відповідає за вхід (повний варіант лістингу наведений в додатку Б):

```
import { Controller, Post, UseGuards, Request, Body } from '@nestjs/common';
import { LocalAuthGuard } from './guards/localauth.guard';
import { JwtAuthGuard } from './guards/jwtauth.guard';
import { AuthService } from './auth.service';
import { CreateUserDto } from '../user/dto/CreateUser.dto';

@Controller()
```

```
export class AuthController {
  constructor(private authService: AuthService) {}

  @UseGuards(LocalAuthGuard)
  @Post('auth/login')
  public async login(@Request() req)
```

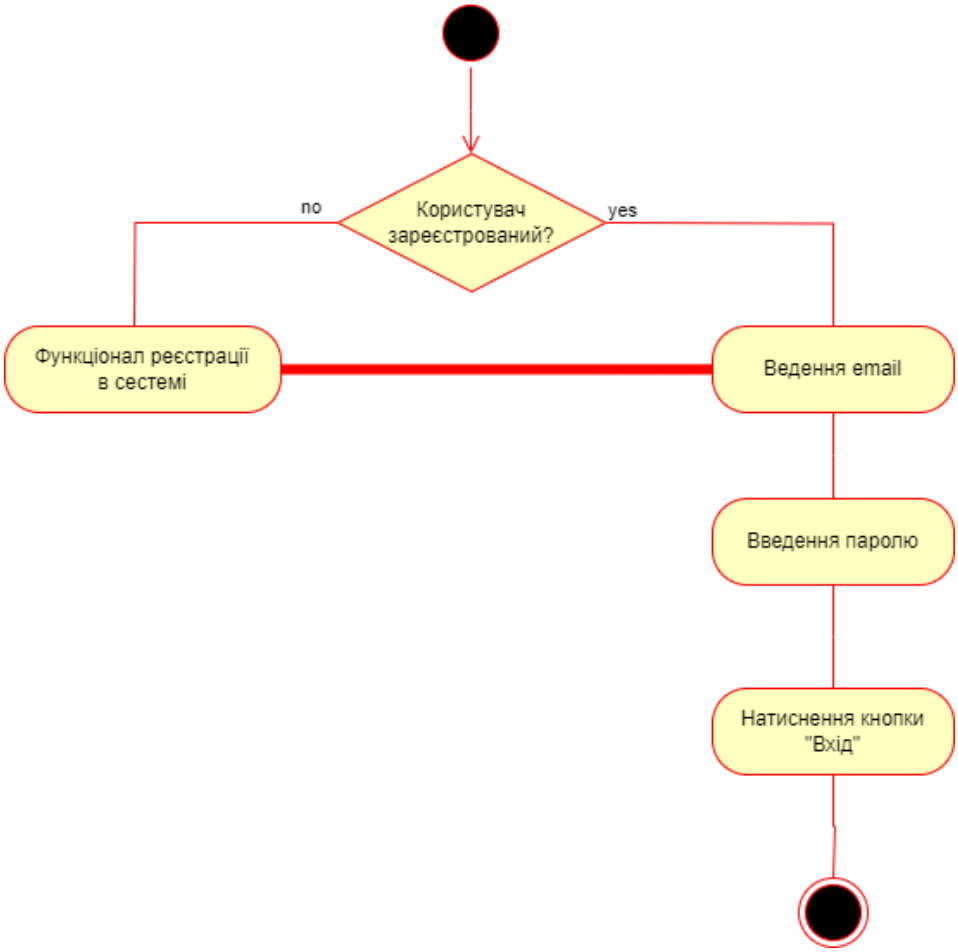

Рисунок 3.10 – UML-діаграма алгоритму функціонування модуля авторизації (Вхід)

Фрагмент лістингу який відповідає за реєстрацію (повний варіант лістингу наведений в додатку Б):

```
  @Post('auth/registration')
  public async registration(@Body() createUserDto: CreateUserDto) {
    return this.authService.registration(createUserDto)
```

```
  }
}
  public async registration(createUserDto: CreateUserDto) {
    return await this.userService.createUser(createUserDto);
  }
}
```

Рисунок 3.11 – UML-діаграма алгоритму функціонування модуля авторизації (Реєстрація)

Отже, розроблено алгоритм функціонування модуля авторизації інформаційної технології управління безпекою розумного будинку, який відповідає за реєстрацію та вхід.

3.4.2 Модуль аналізу безпеки

Створення удосконаленого алгоритму аналізу безпеки розумного будинку є актуальним завданням з численними практичними обґрунтуваннями. В першу чергу, розумні будинки стають все більш популярними, і багато людей і компаній впроваджують ці технології в своїх будинках і офісах. Завдяки цьому, збільшується потреба в безпеці. По-друге, у сучасному світі постійно зростає кількість кіберзагроз і фізичних загроз для безпеки будинків. Також є нагальна потреба в ефективній реакції. Ефективна реакція на потенційні загрози вимагає точного та швидкого аналізу і прийняття рішень. Удосконалений алгоритм аналізу дозволить реалізувати цю реакцію, сприяючи збереженню життя та майна.

Аналіз безпеки розумного будинку здатен мінімізувати ризики несанкціонованого доступу та досягти більш високого рівня захисту [10], за рахунок таких кроків:

1) Аудит безпеки: Проведення аудиту безпеки дозволяє ідентифікувати потенційні вразливості та ризики в системах розумного будинку. Може включати перевірку додатків на вразливості, аналіз мережевої безпеки.
2) Шифрування даних: Забезпечення шифрування даних може запобігти несанкціонованому доступу та перехопленню даних. Використання сильних шифрувальних алгоритмів та протоколів допомагає забезпечити конфіденційність та цілісність даних.
3) Аутентифікація та авторизація: Реалізація надійних методів аутентифікації допомагає перевірити ідентичність користувача перед наданням доступу до інформаційної технології. Окрім того, важливо належно керувати правами доступу та авторизувати користувачів для запобігання несанкціонованому використанню.

Існуючі додатки для управління безпекою в розумному будинку такі як «Google Home» та «Vivint Smart Home» характеризуються зручним способом перегляду відео з камер спостереження, отримання сповіщень про керування системами освітленням і термоконтролем. Проте, їх використання для

забезпечення повноцінної безпеки розумного будинку потребує удосконалень щодо покращення безпеки самого додатку, наприклад, шляхом удосконалення механізмів аутентифікації та захисту персональних даних користувачів, а також розширення можливостей до автоматизації.

Процес аналізу безпеки розумного будинку передбачає оцінювання за такими основними критеріями:

1. Підключення відеоспостереження: Цей критерій включає оцінку системи відеоспостереження в розумному будинку. Для забезпечення безпеки, система відеоспостереження повинна мати відеокамери, розташовані в стратегічних місцях, які покривають ключові зони. Важливо оцінити якість відеозапису, можливість перегляду в реальному часі та можливості зберігання записів. Крім того, важливо перевірити, чи існує захист від несанкціонованого доступу до відео.
2. Підключення до служби моніторингу безпеки: Оцінка підключення до служби моніторингу безпеки передбачає перевірку можливості підключення розумного будинку до професійної служби моніторингу. Це може включати сповіщення про події безпеки, які автоматично передаються до центру моніторингу, і можливість надсилати кваліфіковану допомогу або викликати екстрені служби при потребі.
3. Давачі пожежі (витоку газу, диму, води): Оцінка наявності давачів пожежі в розумному будинку включає перевірку наявності давачів витоку газу, диму та води. Ці давачі мають спрацювати при виявленні небезпеки і надіслати сповіщення користувачеві та/або активувати автоматичну сигналізацію для швидкого реагування на потенційну небезпеку.
4. Розпізнання чужої присутності: Оцінка системи розпізнання чужої присутності включає перевірку наявності давачів руху та інших пристроїв, які можуть виявляти недозволену активність у будинку. Такі системи можуть активувати сигналізацію або сповіщення, якщо вони виявляють незвичайну активність, що може свідчити про несанкціонований доступ до будинку.

5. Автоматична сигналізація: Оцінка наявності автоматичної сигналізації передбачає перевірку системи, яка може активувати сигналізацію при виявленні небезпечної ситуації, такої як злам або вторгнення. Це може включати виклик служб безпеки або екстрені служби, якщо потрібно.

Удосконалений алгоритм аналізу безпеки розумного будинку базується на зниженні ризиків несанкціонованого доступу або виходу з ладу систем таких, як відеоспостереження, давачі пожежі і тд., та забезпеченні надійного захисту будинку. Основою для аналізу безпеки розумного будинку є використання статистичних методів, що дозволяють об'єктивно оцінити стан безпеки будинку і знизити вплив суб'єктивних факторів.

Алгоритм складатиметься з таких кроків:

Крок 1.

Отримання інформації про поточний стан безпеки розумного будинку у вигляді технічних характеристик та функціональних можливостей компонентів розумного будинку, що пов'язані з безпекою, такі як відеокамери, давачі руху, системи тривоги, давачі пожежі, тощо та визначення їх поточного стану.

Крок 2.

Аналіз вразливостей – виявлення потенційних вразливостей безпеки розумного будинку, за рахунок визначення поточного стану компонентів розумного будинку та отримання списку потенційних вразливостей таких, як слабкі місця у захисті будинку, незахищеність мережі, слабкі паролі, несправність давачів чи відеоспостереження і тд. Включаючи:

- Оцінку фізичної безпеки;
- Оцінку мережевої безпеки;
- Оцінку кібербезпеки;
- Оцінку систем моніторингу.

Крок 3.

Розробка заходів безпеки у вигляді набору заходів безпеки у додатку для запобігання та мінімізації виявлених загроз і вразливостей. Це може

включати встановлення сильних паролів, шифрування комунікацій, використання двофакторної аутентифікації, оновлення програмного забезпечення, та регулярну перевірку на наявність вразливостей. А також набор сценаріїв при виявлені певної загрози безпеці, створений власником у додатку.

Крок 4.

Тестування безпеки за допомогою аудиту безпеки, основне ідея якого полягає в систематичному скануванні розумного будинку з метою виявлення потенційних вразливостей. Може включати: перевірку мережевої безпеки, перевірку наявності захисту від хакерських атак, перевірку справності давачів безпеки та інших компонентів системи. Це допоможе виявити потенційні слабкі місця і вразливості, які можуть бути використані зловмисниками.

Крок 5.

Ідентифікація загроз – визначення потенційних загроз безпеці будинку за рахунок отримання сповіщень про загрозу з відповідних давачів або камер. Це можуть бути фізичні загрози (наприклад, крадіжки, пожежі) або цифрові загрози (наприклад, хакерські атаки, злам системи). Ідентифікація загрози відбуватиметься з використанням алгоритмів машинного навчання із урахуванням попереднього аналізу вразливостей систем розумного будинку та зібраних даних з давачів безпеки та відеоспостереження.

Крок 6.

Механізм реагування на виявлену загрозу. Реагування відбуватиметься з урахуванням сценаріїв реагування. При виявлені загрози, власнику будинку та службі моніторингу безпеки надійде сповіщення, що демонструватиме потенційну загрозу, а також буде виконано відповідний сценарій для забезпечення безпеки.

З урахуванням означених кроків, побудовано удосконалений алгоритм функціонування модуля аналізу безпеки розумного будинку. UML-діаграму розробленого алгоритму зображено на рисунку 1.

Рисунок 3.12 – UML-діаграма удосконаленого алгоритму функціонування модуля аналізу безпеки

Фрагмент лістингу який відповідає за аналіз безпеки (повний варіант лістингу наведений в додатку Б):

```
@Injectable({
  providedIn: 'root'
})
export class SecurityAnalysisService {
  public getSystemStatus(): any {
    const systemStatus = this.fetchSystemStatusFromAPI();
    return systemStatus;
  }
```

```
  public analyzeVulnerabilities(systemStatus: any): any {
    const vulnerabilities = this.analyzeVulnerabilitiesForSystem(systemStatus);
    return vulnerabilities;
  }
  public developSecurityMeasures(vulnerabilities: any): any {
    const securityMeasures =
this.developSecurityMeasuresForVulnerabilities(vulnerabilities);
    return securityMeasures;
  }
  public testSecurity(securityMeasures: any): any {
    const securityTestResults = this.testSecurityMeasures(securityMeasures);
    return securityTestResults;
  }
  public identifyThreats(systemStatus: any): any {
    const threats = this.identifyThreatsBasedOnSystemStatus(systemStatus);
    return threats;
  }
```

Розроблено удосконалений алгоритм функціонування модуля аналізу безпеки розумного будинку, який дасть можливість виявлення вразливостей безпеки розумного будинку, виявлення загрози та визначення її типу.

3.4.3 Модуль сценаріїв безпеки

Для розширення функціоналу програмних засобів управління безпекою розумного будинку, інформаційна технологія вимагає наявності модуля сценаріїв безпеки. Цей модуль дозволить користувачам створювати, налаштовувати та керувати різними сценаріями безпеки в їхньому розумному будинку. Сценарії безпеки можуть включати в себе різні дії та реакції на події, такі як рух давача, відкриття дверей або вікон, виявлення диму або витоку газу тощо. Щоб забезпечити користувача можливістю виконання певних дій по управлінню сценаріями безпеки, необхідно щоб даний модуль забезпечував виконання таких функцій:

1. Створення нового сценарію: користувач повинен мати можливість створювати нові сценарії для налаштування різних послідовностей дій при виявленні загрози в розумному будинку.
2. Отримання переліку наявних сценаріїв: користувачу потрібно мати можливість переглядати список усіх створених сценаріїв для швидкого доступу та управління ними.
3. Отримання певного сценарію: користувач повинен мати можливість переглядати інформацію про окремий сценарій, включаючи його налаштування та умови активації.
4. Редагування та видалення сценарію: користувачу потрібно надавати можливість редагувати і змінювати існуючі сценарії безпеки, а також видаляти їх, якщо вони більше не потрібні або потребують змін.

Процес управління сценаріями безпеки в інформаційній технології управління безпекою розумного будинку забезпечується функціонуванням модуля сценаріїв безпеки, UML-діаграма якого зображена на рис. 3.13.

Фрагмент лістингу який відповідає за управління сценаріями безпеки (повний варіант лістингу наведений в додатку Б):

```
@Injectable({
  providedIn: 'root'
})
export class SecurityScenariosService {
  private baseUrl = 'https://your-security-api-url';

  constructor(private http: HttpClient) {}

  public createSecurityScenario(scenarioData: any): Observable<any> {
    const url = `${this.baseUrl}/security-scenarios`;
    return this.http.post(url, scenarioData);
  }
```

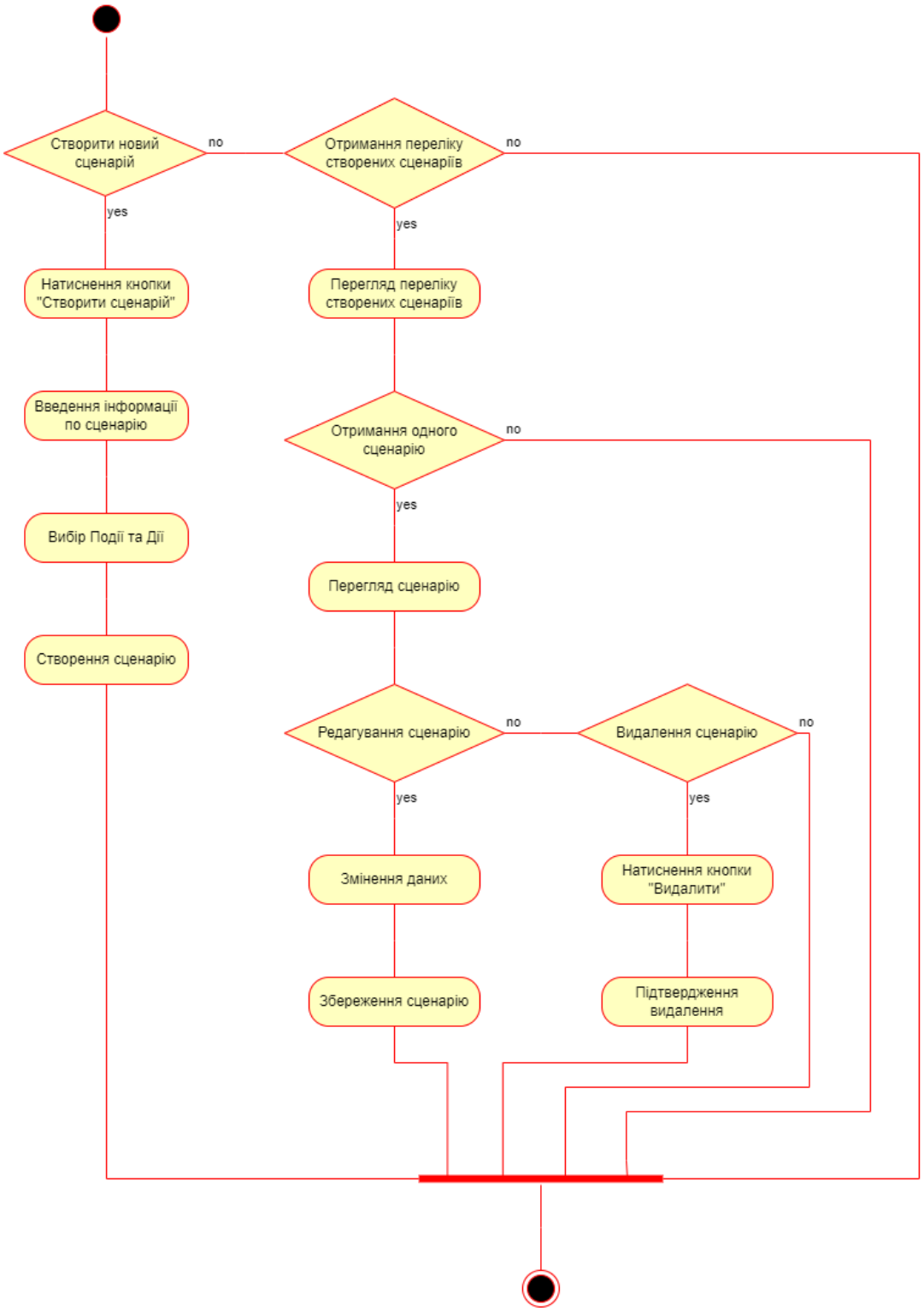

Рисунок 3.13 – UML-діаграма алгоритму функціонування модуля сценаріїв безпеки

Таким чином розроблено алгоритм функціонування модуля сценаріїв безпеки, завдяки якому користувач зможе налаштовувати власні послідовності та реакції на виявленні загрози безпеці розумного будинку.

3.4.4 Модуль аналітики безпеки

Також важливим функціоналом інформаційної технології управління безпекою розумного будинку – є функціонал модуля аналітики безпеки. Завдяки наявності даного модуля в інформаційній технології, власник зможе переглядати інформацію по безпеці розумного будинку, а саме оцінку різних складових безпеки за останнім проведеним аналізом, перелік ідентифікованих та/чи потенційних загроз, а також вразливостей. Процес обчислення середньої оцінки безпеки включає в себе кілька ключових кроків, що дозволяють отримати комплексну оцінку рівня безпеки розумного будинку. Зважаючи на наявність чотирьох критеріїв безпеки: фізична безпека, мережева безпека, кібербезпека та системи моніторингу, процес обчислення середньої оцінки можна розглядати наступним чином:

1. Отримання оцінок по всіх складових: Спочатку здійснюється збір оцінок, отриманих під час аналізу кожного критерію безпеки. Наприклад, для фізичної безпеки оцінюються аспекти, такі як стан дверей та вікон, справність системи сигналізації та доступ до приміщень. Для мережевої та кібербезпеки враховуються аспекти мережевої інфраструктури та програмного забезпечення. Системи моніторингу оцінюються за їхньою здатністю виявляти потенційні загрози та надавати інформацію про стан безпеки.
2. Групування оцінок за критеріями: Оцінки групуються відповідно до зазначених чотирьох критеріїв безпеки, кожна оцінка пов'язується з відповідним критерієм, до якого вона належить.
3. Обчислення середньої оцінки за кожним критерієм: Для кожного критерію безпеки проводиться обчислення середньої оцінки, що відображає рівень безпеки для цього аспекту.
4. Оцінка загальної безпеки: Остаточна середня оцінка безпеки розумного будинку обчислюється, враховуючи середні оцінки за кожним з чотирьох критеріїв. Ця оцінка відображає загальний рівень безпеки

розумного будинку та допомагає власнику отримати уявлення про стан безпеки в цілому.

Отримання комплексної оцінки безпеки відображено у відсотках. Це допомагає власникам розумного будинку легко оцінити рівень безпеки і зрозуміти, наскільки він високий або низький. А також такий підхід допомагає ефективно управляти та вдосконалювати систему безпеки, звертаючи увагу на конкретні аспекти, які можуть потребувати покращень.

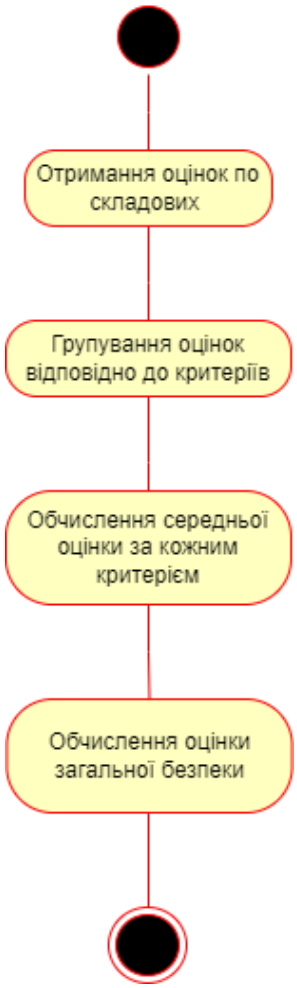

Рисунок 3.14 – UML-діаграма алгоритму функціонування модуля аналітики безпеки

Фрагмент лістингу який відповідає за аналітику безпеки (повний варіант лістингу наведений в додатку Б):

```
import { Injectable } from '@angular/core';

@Injectable()
export class SecurityAnalysisService {
 getSecurityScores(): Promise<SecurityScore[]>
 }

 groupScoresByCriteria(scores: SecurityScore[]): Map<string, number[]> {
  const groupedScores = new Map<string, number[]>();

  scores.forEach(score => {
   if (groupedScores.has(score.criteria)) {
    groupedScores.get(score.criteria).push(score.value);
   } else {
    groupedScores.set(score.criteria, [score.value]);
   }
  });

  return groupedScores;
 }
```

Розроблено алгоритм функціонування модуля аналітики безпеки, завдяки якому користувач зможе переглядати інформацію по безпеці розумного будинку та контролювати безпеку. Завдяки інформацію про вразливості та потенційні загрози, власник зможе вчасно реагувати на них.

3.4.5 Модуль виконання сценарію безпеки

Для того, щоб підвищити безпеку розумного будинку, і для того, щоб користувач мав змогу вчасно дізнатись про загрозу безпеці, інформаційна технологія управління безпекою будинку має налічувати функціонал реагування на загрозу, а саме виконання сценарію безпеки прописаного завчасно власником розумного будинку. За рахунок цього модуля, буде виконуватись та чи інша дія

реагування на певну ідентифіковану загрозу безпеці. Реагування на загрозу буде відбуватися за певним принципом:

1. Пошук відповідного сценарію безпеки в базі даних (БД): при виникненні загрози виконується пошук сценарію з подією в БД, де зберігаються всі прописані сценарії безпеки. Кожен сценарій містить інструкції щодо дій, які повинні бути виконані в разі ідентифікації певної загрози.
2. Виконання дії прописаної в даному сценарії: після знаходження відповідного сценарію, інформаційна технологія виконує інструкції, прописані в цьому сценарії. Це може включати в себе різні дії, такі як:
    - активація системи сигналізації та сповіщення служб безпеки або власника будинку;
    - запуск відеоспостереження та запис подій;
    - закриття та блокування дверей або вікон для запобігання несанкціонованому доступу;
    - вимикання або ізоляція певних систем для запобігання подальшим загрозам;
    - повідомлення власника будинку чи інших авторизованих осіб.

UML-діаграма алгоритму функціонування модуля виконання сценарію безпеки представлена на рис. 3.15.

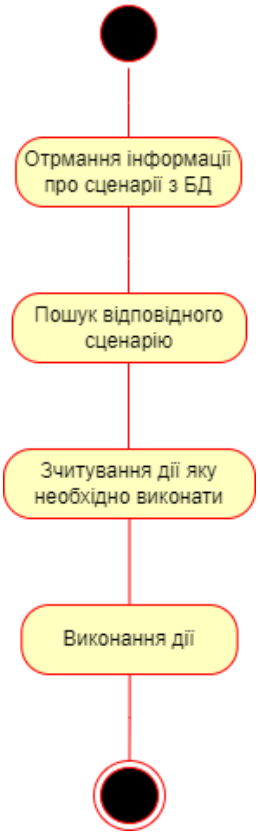

Рисунок 3.15 – UML-діаграма алгоритму функціонування модуля виконання сценарію безпеки

Фрагмент лістингу який відповідає за аналітику безпеки (повний варіант лістингу наведений в додатку Б):

```
import { Injectable } from '@nestjs/common';
import * as Twilio from 'twilio';

@Injectable()
export class TwilioService {
  private readonly client: Twilio.Twilio;

  constructor() {
    this.client = Twilio(process.env.TWILIO_ACCOUNT_SID,
process.env.TWILIO_AUTH_TOKEN);
  }

  async sendSms(to: string, body: string): Promise<void> {
```

```
    await this.client.messages.create({
      body,
      to,
      from: process.env.TWILIO_PHONE_NUMBER,
    });
  }
}

@Injectable()
export class SmsService {
  constructor(private readonly twilioService: TwilioService) {}

  async sendSmsToEmail(phoneNumber: string, email: string): Promise<void> {
    const message = You have a new SMS. Check your email: ${email};
    await this.twilioService.sendSms(phoneNumber, message);
  }
}
```

Таким чином, розроблений функціонал модулю виконання сценарію дозволяє реагувати на різні загрози безпеці та надає власникам розумного будинку контроль над тим, які заходи повинні бути вжиті у випадку потенційно небезпечних ситуацій.

3.5 Тестування інформаційної технології управління безпекою розумного будинку

Тестування програмного забезпечення - це процес під час якого перевіряється відповідність заявлених до продукту вимог до реально реалізованої функціональності, що відбувається шляхом спостереження роботою системи в штучно створених ситуаціях [17].

Під час тестування функціоналу інформаційної технології управління безпекою розумного будинку було виконано тестування усіх функцій, враховуючи вихідні данні, з метою перевірки правильності відповідей.

Для перевірки коректності роботи модуля авторизації, відкриємо сторінку реєстрації та заповнимо форму, рисунок 3.16.

Введемо необхідні данні для реєстрації користувача:

ПІБ: Капченко Карина Григорівна

E-mail: Email_Test@gmail.com

Пароль: qweFGtuo87

Підтвердження паролю: qweFGtuo87

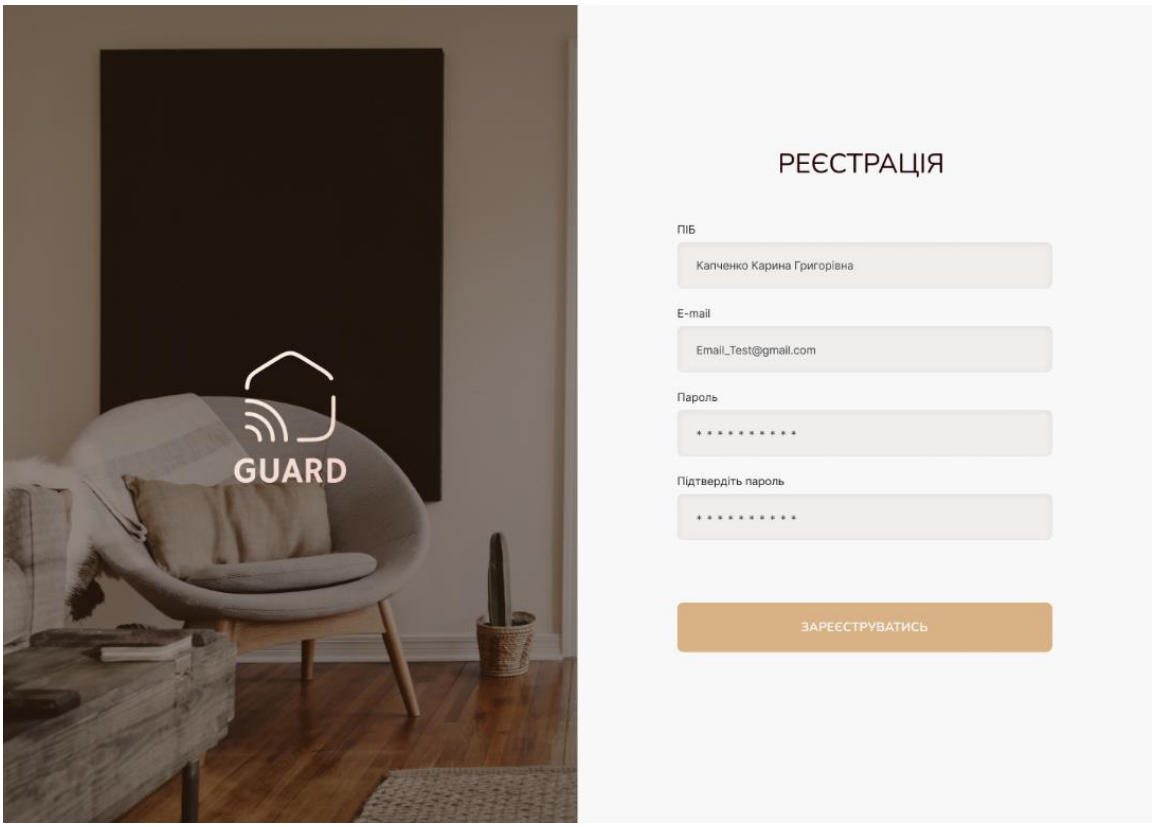

Рисунок 3.16 – Заповнення форми реєстрації

Наступним етапом є автентифікація, для цього використаємо e-mail та пароль, який використовувались під час реєстрації та натиснемо на кнопку “Вхід”. На рисунку 3.17 зображена заповнена форма входу до інформаційної технології, та на рисунку 3.18 – результат його виконання.

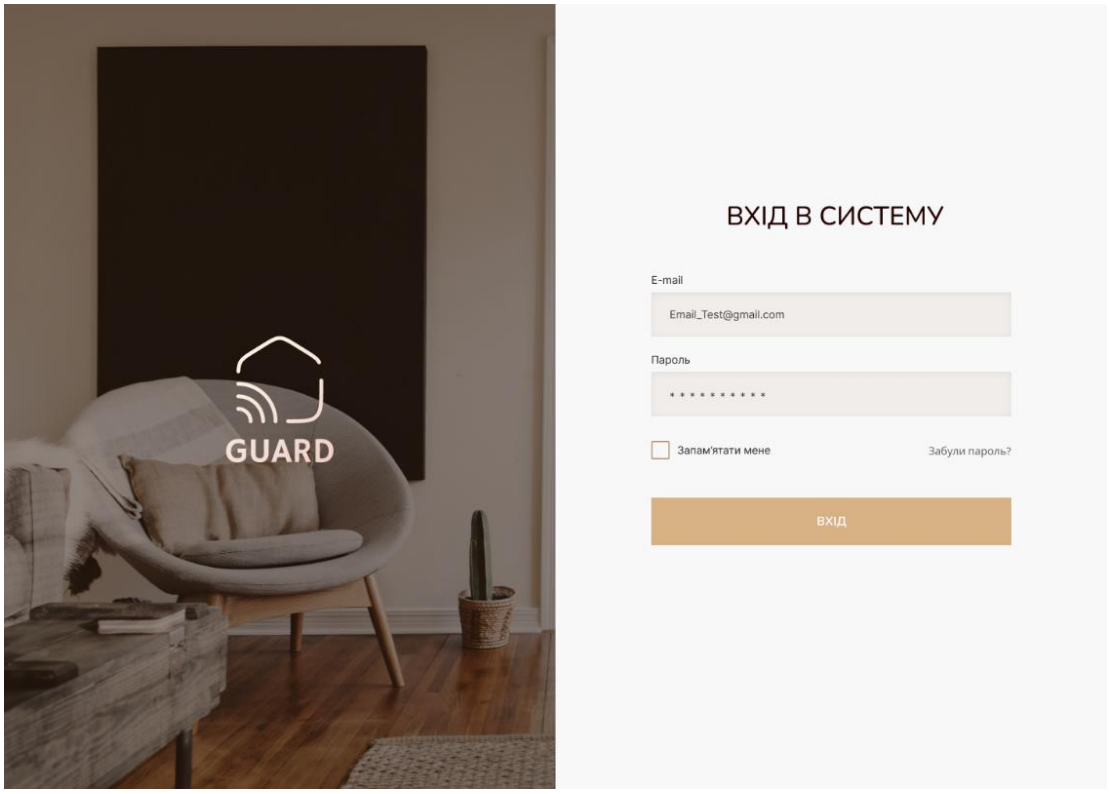

Рисунок 3.17 – Заповнення форми входу

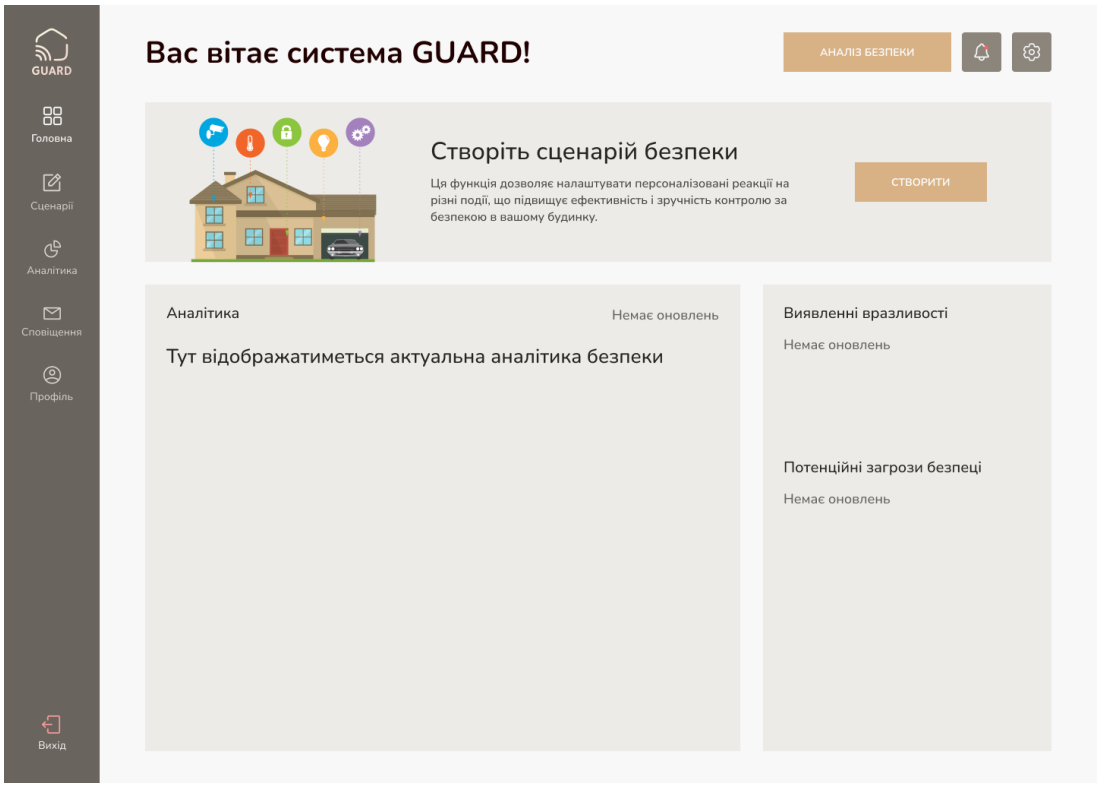

Рисунок 3.18 – Результат входу використовуючи коректні данні

Після входу до інформаційної технології, необхідно протестувати можливість створення сценаріїв безпеки. Нехай користувач хоче створити *n*-ну кількість сценаріїв. Для цього йому необхідно перейти на вкладку «Сценарії», натиснути кнопку «Створити новий сценарій» та заповнити форму створення сценарію. На рисунку 3.19 зображено заповнення форми створення сценарію безпеки. Для перевірки коректності створення нового сценарію, перейдемо на сторінку з сценаріями і перевіримо відображення нового сценарію. На рисунку 3.20 зображено список усіх створених сценаріїв.

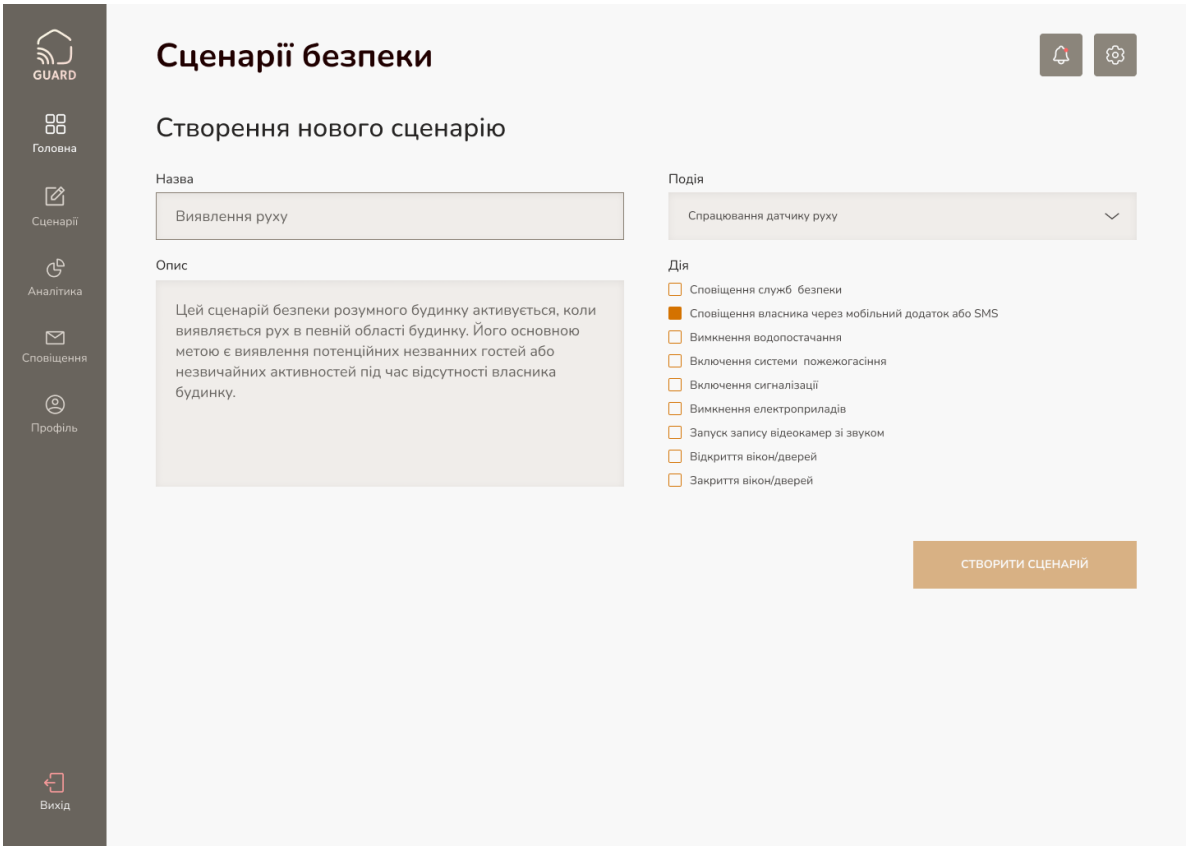

Рисунок 3.19 – Заповнення форми створення сценарію

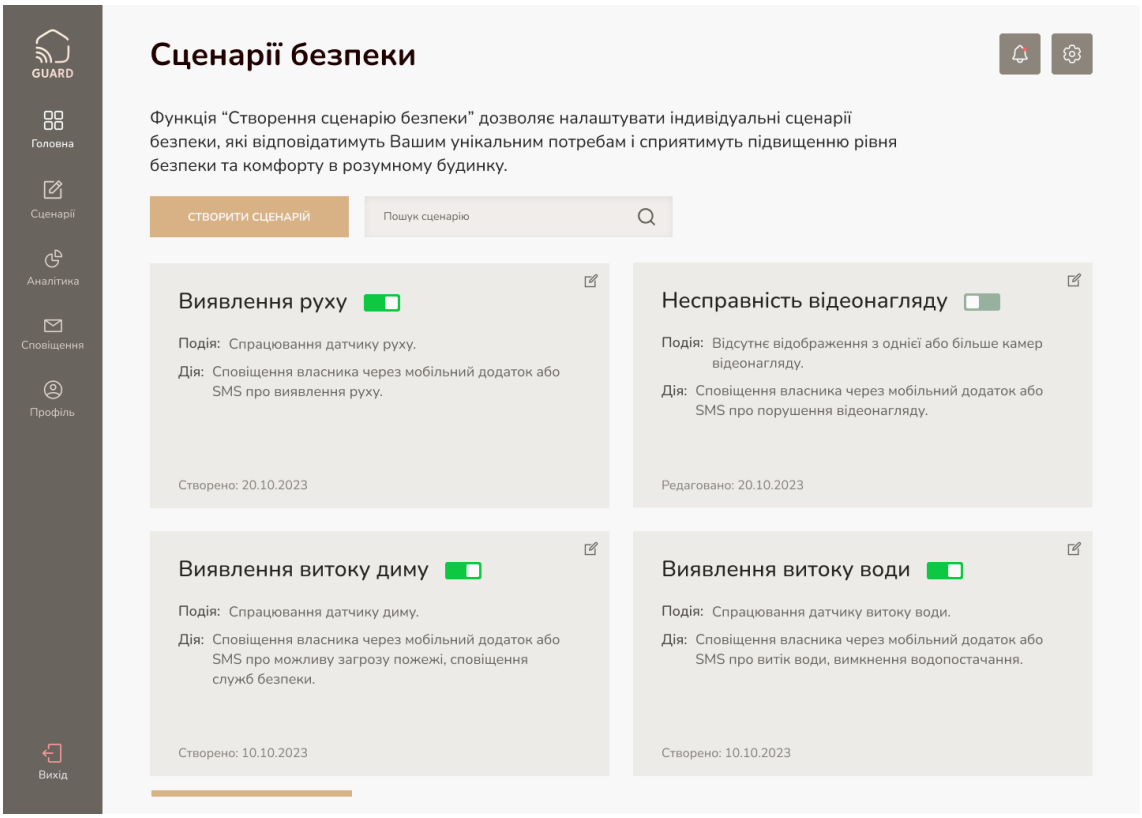

Рисунок 3.20 – Список створених сценаріїв безпеки

Для перевірки коректності відображення детальної інформації про сценарій, необхідно перейти на сторінку обраного сценарію. Результат виконання даної операції зображений на рисунку 3.21.

Щоб перевірити реагування на загрозу безпеці, необхідно виконати такі дії: запуск аналізу безпеки, а після його виконання – перегляд інформації у вкладці «Аналітика». На рисунку 3.22 зображено виконання аналізу безпеки, а на рисунку 3.23 – аналітика безпеки, враховуючи вразливості та потенційні загрози, що свідчить про правильність виконання цієї функції.

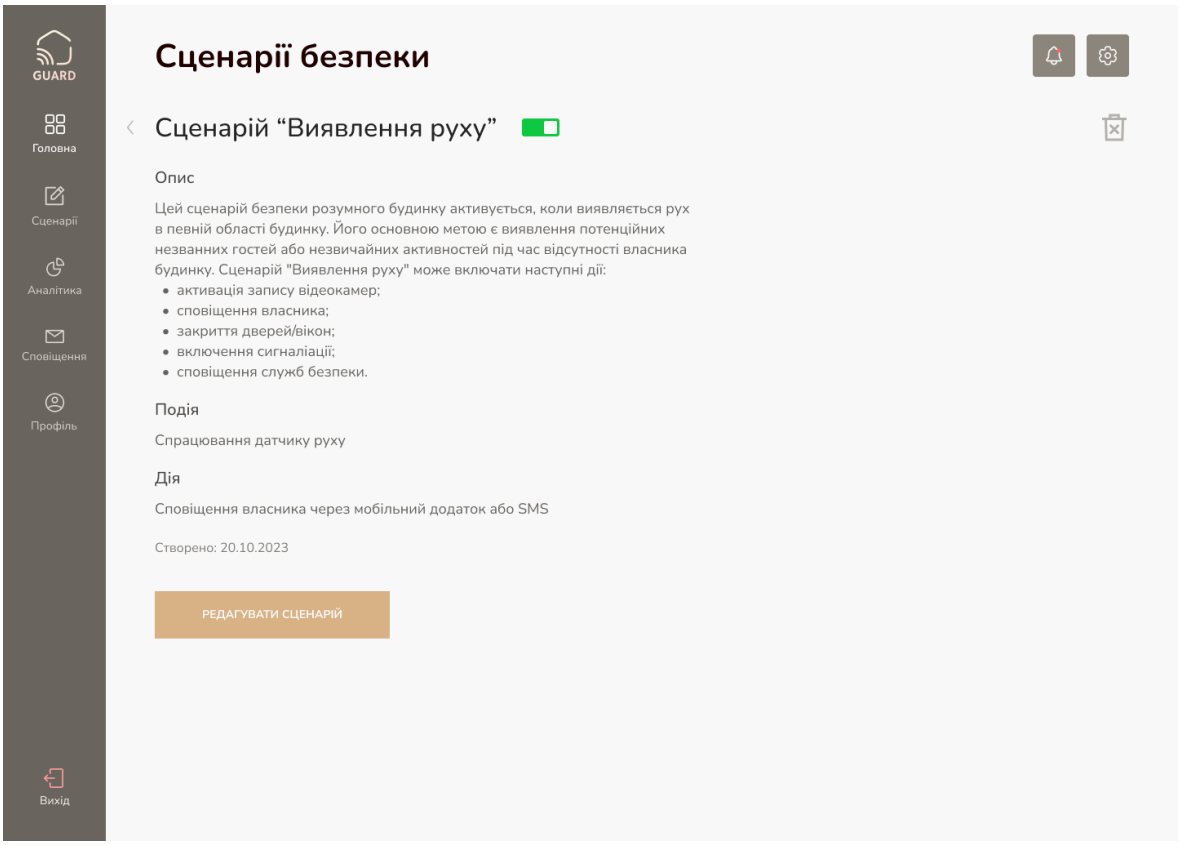

Рисунок 3.21 – Перегляд детальної інформації про сценарій

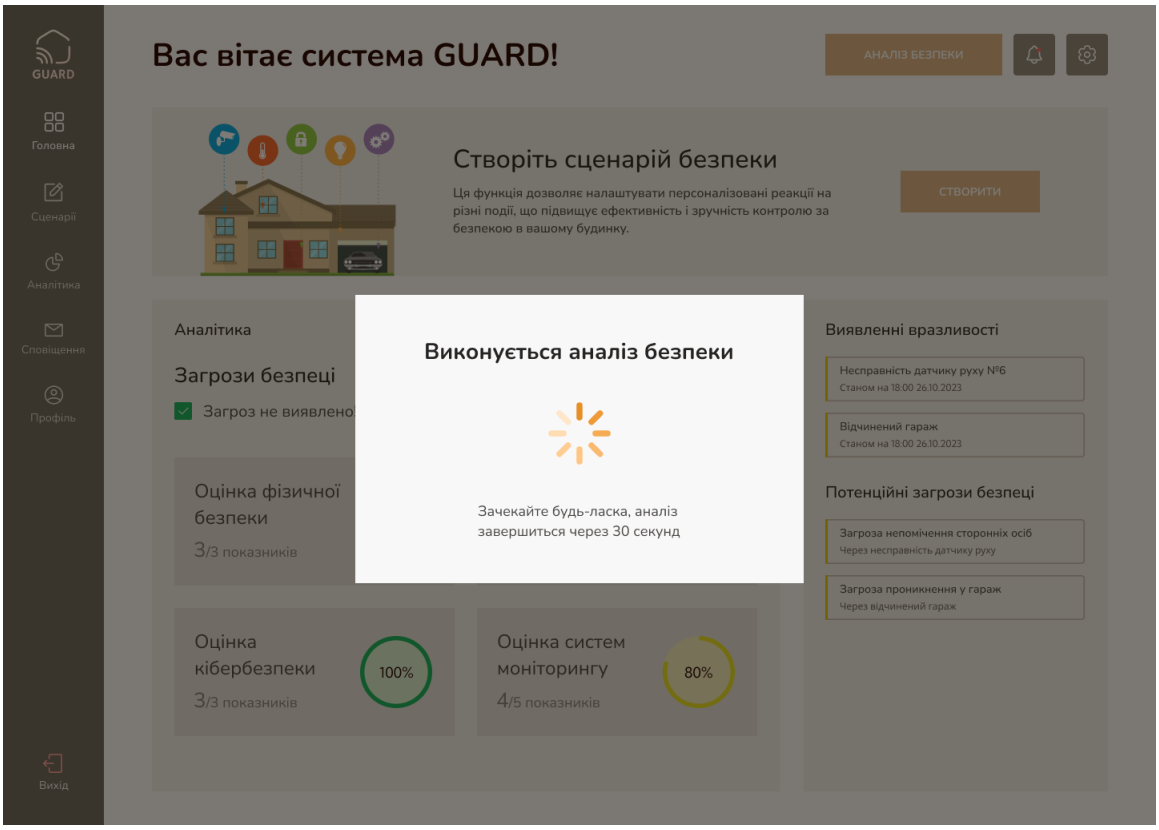

Рисунок 3.22 – Виконання аналізу безпеки

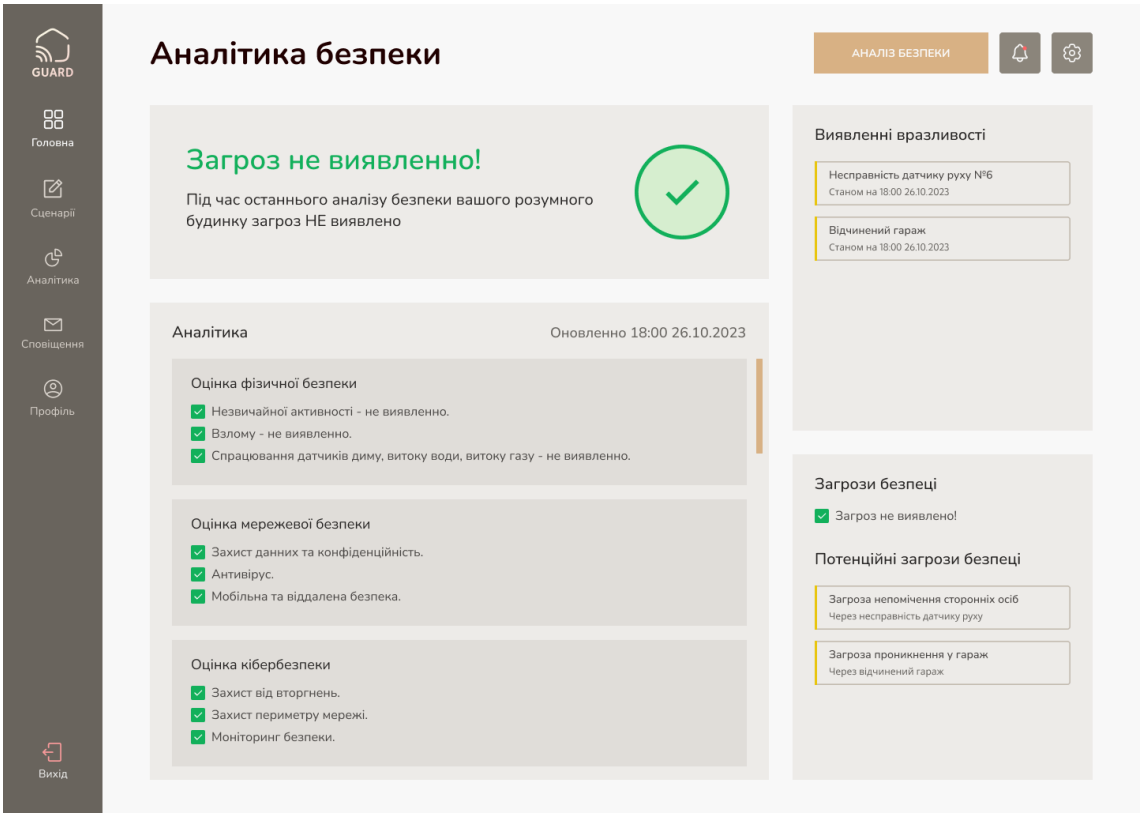

Рисунок 3.23 – Перегляд аналітики безпеки

Під час тестування інформаційної технології управління безпекою розумного будинку було виконано ряд досліджень: проведено тестування понад 30 користувачів, при цьому кожен користувач створив мінімум по 5 сценаріїв безпеки. Проведено реєстрацію і ідентифікацію кожного користувача; протестовано запуск аналізу безпеки; перевірено коректність та швидкість реагування за сценарієм для понад 10 видів загроз, а також виконання відповідних дій вказаних у сценарії. Результати тестування часу реакції на загрозу та часу забезпечення безпеки розумного будинку у порівнянні з обраним аналогом представлено у таблиці 3.3, а порівняння функціональних можливостей запропонованої інформаційної технології та засобу Google Home – у таблиці 3.4.

Таблиця 3.3 – Часові характеристики реакції на загрозу та забезпечення безпеки розумного будинку

Критерій	Інформаційна технологія	Google Home
Часовий інтервал від виявлення загрози до реакції на загрозу	500мс-3с	5с-8с
Час забезпечення безпеки розумного будинку	16с	20с

Таблиця 3.4 – Порівняльна характеристика функціоналу розробленої інформаційної технології

Критерій	Інформаційна технологія	Google Home
Швидкість роботи	+	+
Виведення аналітики безпеки	+	-
Можливість створення сценаріїв безпеки	+	-

Отже, запропонована інформаційна технологія характеризується розширеним функціоналом засобу безпеки за рахунок можливості створення власником сценаріїв безпеки та удосконаленого аналізу безпеки, який дозволяє виявляти не тільки загрози, а й вразливості засобу безпеки. Таким чином, це дозволило підвищити швидкість реакції на виявлену загрозу більше, ніж на 50%, а забезпечення безпеки розумного будинку – на 20%.

3.6 Висновок до розділу 3

Виконано обґрунтування вибору мови програмування, середовища розробки та СУБД для розробки інформаційної технології управління безпекою розумного будинку. Розроблено сучасний, інтуїтивно зрозумілий інтерфейс. Реалізовано функціонування усіх модулів інформаційної технології таких, як: модуль «Авторизація», модуль «Сценарії безпеки», модуль «Аналіз безпеки», модуль «Аналітика безпеки», модуль «Виконання сценарію безпеки». Виконано тестування функціонування інформаційної технології управління безпекою розумного будинку відповідно до вихідних даних та визначено відповідність меті дослідження.

4 ЕКОНОМІЧНА ЧАСТИНА

4.1 Проведення комерційного та технологічного аудиту інформаційної технології управління безпекою розумного будинку

Метою проведення комерційного і технологічного аудиту є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності, тобто під час виконання магістерської кваліфікаційної роботи.

Для проведення комерційного та технологічного аудиту залучаємо 3-х незалежних експертів, якими є провідні викладачі випускової або спорідненої кафедри. Оцінювання науково-технічного рівня розробки та її комерційного потенціалу здійснюємо із застосуванням п’ятибальної системи оцінювання за 12-ма критеріями, а результати зводимо до таблиці 4.1.

Таблиця 4.1 – Результати оцінювання науково-технічного рівня і комерційного потенціалу засобу поляриметричного аналізу оптично активних рідни

Критерії	Експерти		
	Савчук Т.О.	Озеранський В.С.	Колесницький О.К.
	Бали, виставлені експертами		
Технічна здійсненність концепції	2	2	2
Ринкові переваги (наявність аналогів)	3	3	3
Ринкові переваги (ціна продукту)	4	2	3
Ринкові переваги (технічні властивості)	3	2	2
Ринкові переваги (експлуатаційні витрати)	2	2	2
Ринкові перспективи (розмір ринку)	3	3	2
Ринкові перспективи (конкуренція)	2	2	3
Практична здійсненність (наявність фахівців)	2	3	2
Практична здійсненність (наявність фінансів)	3	3	3

Продовження таблиці 4.1

Практична здійсненність (необхідність нових матеріалів)	3	3	2
Практична здійсненність (термін реалізації)	3	3	2
Практична здійсненність (розробка документів)	2	2	2
Сума балів	32	30	28
Середньоарифметична сума балів, СБ	30		

За результатами розрахунків, наведених в таблиці 1 робимо висновок про те, що науково-технічний рівень та комерційний потенціал інформаційної технології управління безпекою розумного будинку – середній.

4.2 Розрахунок витрат на здійснення науково-дослідної роботи

Витрати на оплату праці. Належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці, також будь-які види грошових і матеріальних доплат, які належать до елемента «Витрати на оплату праці».

Основна заробітна плата дослідників. Витрати на основну заробітну плату дослідників ($З_о$) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \sum_{i=1}^{к} \frac{М_{ni} \cdot t_i}{Т_р},$$

де к – кількість посад дослідників, залучених до процесу дослідження; $М_{пі}$ – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.; $Т_р$ – число робочих днів в місяці; приблизно $Т_р$ = (21…23) дні; $t_і$ – число робочих днів роботи розробника (дослідника).

Зроблені розрахунки зводимо до таблиці 2.

Таблиця 4. 2 – Витрати на заробітну плату дослідників

Посада	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату, грн.
Керівник	30 000	1486	60	89160
Розробник	23 000	1095	140	153300
Всього:	242460			

Основна заробітна плата робітників. Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт розраховують за формулою:

$$З_р = \sum_{і=1}^{n} С_і \cdot t_і,$$

де $С_і$ – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год; $t_і$ – час роботи робітника на виконання певної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду С і можна визначити за формулою:

$$С_і = \frac{М_м \cdot К_і \cdot К_с}{Т_р \cdot t_{зм}},$$

де $М_м$ – розмір прожиткового мінімуму працездатної особи або мінімальної місячної заробітної плати (залежно від діючого законодавства), у 2023 році $М_м$=6700 грн; $К_і$ – коефіцієнт міжкваліфікаційного співвідношення для

встановлення тарифної ставки робітнику відповідного розряду; $К_с$ – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати; $Т_р$ – середня кількість робочих днів в місяці, приблизно $Т_р = 21…23$ дні; $t_{зм}$ – тривалість зміни, год.

Таблиця 4.3 – Витрати на заробітну плату робітників

Найменування робіт	Трудомісткість, н-год.	Розряд роботи	Погодинна тарифна ставка	Тариф. коеф.	Величина, грн.
Аналіз сучасних систем управління безпекою розумного будинку	120	4	50,6	1,27	6072
Проектування алгоритму та структури інформаційної технології управління безпекою розумного будинку	260	5	54,2	1,36	14092
Реалізація інформаційної технології управління безпекою розумного будинку	680	6	57,8	1,45	39304
Апробація та впровадження результатів дослідження	100	3	47,1	1,18	4710
Всього					64178

Додаткова заробітна плата. Додаткова заробітна плата $З_д$ всіх розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховується як (10…12)% від суми основної заробітної плати всіх розробників та робітників, тобто:

$$З_д = 0{,}12 \cdot (З_о + З_р) = 0{,}12 \cdot (153300 + 64178) = 26097 \text{ грн.}$$

Відрахування на соціальні заходи. Нарахування на заробітну плату $Н_{зп}$ розробників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою:

$$Н_{зп} = \beta \cdot (З_о + З_р + З_д) =$$
$$= 0{,}22 \cdot (153300 + 64178 + 26097) = 53587 \text{ грн.}$$

де $З_о$ – основна заробітна плата розробників, грн.; $З_р$ – основна заробітна плата робітників, грн.; $З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.; β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, % (приймаємо для 1-го класу професійності ризику 22%).

Розрахунок витрат на матеріали. Витрати на матеріали М, що були використані під час виконання даного етапу роботи, розраховуються за формулою:

$$М = \sum_{1}^{n} Н_і \cdot Ц_і \cdot К_і,$$

де $Н_і$ – кількість матеріалів і-го виду, шт.; $Ц_і$ – ціна матеріалів і-го виду, грн.; $К_і$ – коефіцієнт транспортних витрат, $К_і$ = (1,1…1,15); n – кількість видів матеріалів.

Таблиця 4.4 – Матеріали, що використані на розробку

Найменування матеріалів	Ціна за одиницю, грн.	Витрачено	Вартість витрачених комплектуючих, грн.
Офісний папір	249	1	249
Набір канцелярії	200	1	200
Картридж для принтера	3125	1	3125
Флешка на 64GB	319	1	319
Всього, з врахуванням коефіцієнта транспортних витрат			4477

Програмне забезпечення. До балансової вартості програмного забезпечення входять витрати на його інсталяцію, тому ці витрати беруться додатково в розмірі 10…12% від вартості програмного забезпечення. Балансову вартість програмного забезпечення розраховують за формулою:

$$В_{прг} = \sum_{1}^{к} Ц_{іпрг} \cdot С_{прг.і} \cdot К_{і},$$

де $Ц_{іпрг}$ – ціна придбання програмного забезпечення і-го виду, грн.; $С_{прг.і}$ – кількість одиниць програмного забезпечення відповідного виду, шт.; $К_{і}$ – коефіцієнт, що враховує інсталяцію, налагодження програмного забезпечення, $К_{і}$ = (1,1…1,12); к – кількість видів програмного забезпечення.

Таблиця 4.5 – Витрати на придбання програмного забезпечення

Найменування програмного забезпечення	Ціна за одиницю, грн.	Витрачено	Вартість програмного забезпечення, грн.
Ліцензійна ОС та Microsoft Office	2800	1	2800
Всього, з врахуванням коефіцієнта інсталяції та налагодження			3080

Амортизація обладнання. Амортизація обладнання, комп'ютерів та приміщень, які використовувались під час (чи для) виконання даного етапу роботи.

У спрощеному вигляді амортизаційні відрахування А в цілому бути розраховані за формулою:

$$А = \frac{Ц_{б}}{Т_{в}} \cdot \frac{t}{12},$$

де $Ц_{б}$ – загальна балансова вартість всього обладнання, комп'ютерів, приміщень тощо, що використовувались для виконання даного етапу роботи, грн.; t – термін

використання основного фонду, місяці; $Т_в$ – термін корисного використання основного фонду, роки.

Таблиця 4.6 – Амортизаційні відрахування за видами основних фондів

Найменування	Балансова вартість, грн.	Строк корисного використання, років	Термін використання, місяців	Сума амортизації, грн.
Ноутбук ASUS X570UD-E4037	27039	5	3	1352
WI-FI Роутер	749	5	3	37,5
Миша	500	2	3	62,5
Принтер	3500	5	3	175
Всього	1627			

Витрати на електроенергію для науково-виробничих цілей. Витрати на силову електроенергію $В_е$, якщо ця стаття має суттєве значення для виконання даного етапу роботи, розраховуються за формулою:

Таблиця 4.7 – Витрати на електроенергію

Найменування обладнання	Потужність, кВт	Тривалість годин роботи
Ноутбук ASUS X570UD-E4037	0,12	1120
WI-FI Роутер	0,006	900
Принтер	0,3	7
Освітлення	0,03	330

$$Ве = \sum \frac{W_i \cdot t_i \cdot Ц_е \cdot К_{впі}}{ККД} =$$

$$= \frac{0{,}12 \cdot 1120 \cdot 7{,}5 \cdot 0{,}95}{0{,}94} + \frac{0{,}006 \cdot 900 \cdot 7{,}5 \cdot 0{,}95}{0{,}94} + \frac{0{,}3 \cdot 7 \cdot 7{,}5 \cdot 0{,}95}{0{,}94}$$

$$+ \frac{0{,}03 \cdot 330 \cdot 7{,}5 \cdot 0{,}95}{0{,}94} = 1151 \text{ грн.},$$

W_i – встановлена потужність обладнання, кВт; t_i – тривалість роботи обладнання на етапі дослідження, год.; Це – вартість 1 кВт електроенергії, грн.; $К_{впі}$ – коефіцієнт використання потужності; ККД – коефіцієнт корисної дії обладнання.

Інші витрати. До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуються як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$Ів = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%} = (153300 + 64178) \cdot \frac{50}{100} = 108739 \text{ грн.},$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати».

Накладні (загальновиробничі) витрати. До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов’язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов’язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов’язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуються як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$Внзв = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%} = (153300 + 64178) \cdot \frac{100}{100} = 217478 \text{ грн.},$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати».

Витрати на проведення науково-дослідної роботи. Витрати на проведення науково-дослідної роботи розраховуються як сума всіх попередніх статей витрат за формулою:

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + В_{прг} + А_{обл} + В_е +$$
$$+ І_в + В_{нзв} = 153300 + 64178 + 26097 + 53587 + 4477 + 3080 + 1627$$
$$+ 1151 + 108739 + 217478 = 633714$$

Загальні витрати. Загальні витрати ЗВ на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховуються за формулою:

$$ЗВ = \frac{В_{заг}}{\eta} = \frac{633714}{0{,}9} = 704127 \text{ грн.},$$

де η – коефіцієнт, що характеризує етап виконання науково-дослідної роботи. Оскільки, якщо науково-технічна розробка знаходиться на стадії впровадження, то η=0,9.

4.3 Розрахунок економічної ефективності науково-технічної розробки за її можливої комерціалізації потенційним інвестором

В ринкових умовах узагальнюючим позитивним результатом, що його може отримати потенційний інвестор від можливого впровадження результатів тієї чи іншої науково-технічної розробки, є збільшення у потенційного інвестора величини чистого прибутку.

В даному випадку відбувається розробка засобу, тому основу майбутнього економічного ефекту буде формувати: ΔN – збільшення кількості споживачів,

яким надається відповідна інформаційна послуга в аналізовані періоди часу; N – кількість споживачів, яким надавалась відповідна інформаційна послуга у році до впровадження результатів нової науково-технічної розробки; $Ц_б$ – вартість послуги у році до впровадження інформаційної системи; $\pm\Delta Ц_о$ – зміна вартості послуги (зростання чи зниження) від впровадження результатів науково-технічної розробки в аналізовані періоди часу.

Можливе збільшення чистого прибутку у потенційного інвестора ΔПі для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження та комерціалізації науково-технічної розробки, розраховується за формулою:

$$\Delta П = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N_i)_i \cdot \lambda \cdot \rho \cdot \left(1 - \frac{\vartheta}{100}\right),$$

де ±ΔЦ – зміна основного якісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай, таким показником може бути зміна ціни реалізації одиниці нової розробки в аналізованому році (відносно року до впровадження цієї розробки); $\pm\Delta Ц_о$ може мати як додатне, так і від'ємне значення (від'ємне – при зниженні ціни відносно року до впровадження цієї розробки, додатне – при зростанні ціни); N – основний кількісний показник, який визначає величину попиту на аналогічні чи подібні розробки у році до впровадження результатів нової науково-технічної розробки; $Ц_о$ – основний якісний показник, який визначає ціну реалізації нової науково-технічної розробки в аналізованому році; $Ц_б$ – основний якісний показник, який визначає ціну реалізації існуючої (базової) науково-технічної розробки у році до впровадження результатів; ΔN – зміна основного кількісного показника від впровадження результатів науково-технічної розробки в аналізованому році. Зазвичай таким показником може бути зростання попиту на науково-технічну розробку в аналізованому році (відносно року до впровадження цієї розробки); λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану

вартість. У 2023 році ставка податку на додану вартість становить 20%, а коефіцієнт λ =0,8333; ρ – коефіцієнт, який враховує рентабельність інноваційного продукту (послуги). Рекомендується брати ρ =0,2...0,5; ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2023 році ϑ =18%.

Очікуваний термін життєвого циклу розробки 1 рік, тому:

$$\Delta П = ((135000 - 60000) \cdot 8000 - (8000 - 7500) \cdot 60000) \cdot 0{,}8333 \cdot 0{,}3 \cdot \left(1 - \frac{18}{100}\right) = 112176000 \text{ грн.}$$

Далі розраховують приведену вартість збільшення всіх чистих прибутків ПП, що їх може отримати потенційний інвестор від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t} = \frac{112176000}{(1+0{,}1)^1} = 101978182 \text{ грн.},$$

де ΔП – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн.; Т – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки (приймаємо Т=1 рік); τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ =0,05…0,15; t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

Далі розраховують величину початкових інвестицій PV , які потенційний інвестор має вкласти для впровадження і комерціалізації науково-технічної розробки. Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ = 7 \cdot 704127 = 4928889 \text{ грн.}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію. Це можуть бути витрати на підготовку приміщень, розробку технологій, навчання персоналу, маркетингові заходи тощо; зазвичай kінв=2...5, але може бути і більшим; ЗВ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, грн.

Тоді абсолютний економічний ефект $Е_{абс}$ або чистий приведений дохід для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки становитиме:

$$Е_{абс} = ПП - PV = 101978182 - 4928889 = 97049293 \text{ грн.},$$

де ПП – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, грн.; PV – теперішня вартість початкових інвестицій, грн.

Оскільки $Е_{абс}>0$, то можемо припустити про потенційну зацікавленість інвесторів у розробці.

Для остаточного прийняття рішення з цього питання необхідно розрахувати внутрішню економічну дохідність Ев або показник внутрішньої норми дохідності вкладених інвестицій та порівняти її з так званою бар'єрною ставкою дисконтування, яка визначає ту мінімальну внутрішню економічну дохідність, нижче якої інвестиції в будь-яку науково-технічну розробку вкладати буде економічно недоцільно.

Внутрішня економічна дохідність інвестицій Ев, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, розраховується за формулою:

$$Е_в = \sqrt[Т_ж]{1 + \frac{Еабс}{PV}} = \sqrt{1 + \frac{97049293}{4928889}} = 4{,}53,$$

де $Т_ж$ – життєвий цикл розробки, роки.

Визначимо бар'єрну ставку дисконтування $\tau_{мін}$, тобто мінімальну внутрішню економічну дохідність інвестицій, нижче якої кошти у впровадження науково-технічної розробки та її комерціалізацію вкладатися не будуть.

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$ визначається за формулою:

$$\tau_{мін} = d + f = 0{,}9 + 0{,}5 = 1{,}4,$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2023 році в Україні d =0,9...0,12; f – показник, що характеризує ризикованість вкладення інвестицій; зазвичай величина f =0,05...0,5, але може бути і значно вищою.

Оскільки $Е_в$=4,53>$\tau_{мін}$=1,4, то потенційний інвестор може бути зацікавлений у фінансуванні впровадження науково-технічної розробки та виведенні її на ринок, тобто в її комерціалізації.

Далі розраховуємо період окупності інвестицій $Т_о$, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки:

$$Т_о = \frac{1}{Е_в} = \frac{1}{4{,}53} = 0{,}22 \text{ року.}$$

Оскільки $Т_о$=0,22<1…3-х років, то це свідчить про комерційну привабливість науково-технічної розробки і може спонукати потенційного інвестора профінансувати впровадження цієї розробки та виведення її на ринок.

4.4 Висновок до розділу 4

Врезультаті, в даному розділі було проаналізовано економічну частину роботи. Під час виконання було проведено комерційний та технологічний аудит,

в якому трьома незалежними експертами було сформовано таблицю оцінювання (таблиця 4.1) комерційного потенціалу. За результатами дослідження з'ясувалося, що розробка має рівень комерційного потенціалу та і науково-технічного рівня є середнім. Такий рівень показника досягнуто в результаті підвищенні швидкості реакції на виявлену загрозу шляхом опрацювання вектору факторів впливу на стан безпеки. Було виконано прогнозування витрат на виконання НДР. На основі показників статей витрат розраховано загальні витрати на завершення НДР, які становлять 704127 грн. Після чого було розраховано період окупності, який дорівнює 0,22 року. Такий термін свідчить про комерційну привабливість проєкту.

ВИСНОВКИ

Під час написання магістерської кваліфікаційної роботи були вирішенні такі завдання.

Актуальність інформаційної технології управління безпекою розумного будинку полягає у потребі створення інструменту, який забезпечить надійне управління розумним будинком. Було досліджено особливості використання програмних засобів для управління безпекою розумного будинку та виявлено переваги та недоліки сучасних засобів управління безпекою розумного будинку. Враховуючи порівняльну характеристику існуючих програмних засобів управління безпекою розумного будинку, було визначено, що засіб управління безпекою розумного будинку Google Home може бути використаний для процесу управління безпекою розумного будинку, але потребує удосконалень, за рахунок підвищення швидкості реакції на загрозу безпеці та розширення функціоналу.

Визначено задачу підвищення швидкості реакції на виявлену загрозу, за рахунок використання відповідної інформаційної технології, яка дозволить користувачеві завчасно створювати набір сценаріїв безпеки, та забезпечить надійне управління безпекою розумного будинку за рахунок виявлення потенційних загроз. Виконано постановку задачі подальшого дослідження.

Розроблено удосконалену математичну модель управління безпекою розумного будинку, яка на відміну від існуючих дозволяє підвищити швидкість реагування на загрозу, за рахунок опрацювання вектору факторів впливу на стан безпеки.

Розроблено узагальнений алгоритм для управління безпекою розумного будинку. Також, відповідно до узагальненого алгоритму, було розроблено структуру інформаційної технології управління безпекою розумного будинку, до якої входять модуль «Авторизація», модуль «Сценарії безпеки», модуль «Аналіз безпеки», модуль «Аналітика безпеки» та модуль «Виконання сценарію безпеки».

Обґрунтовано вибір мови програмування, середовища розробки та СУБД для розробки інформаційної технології управління безпекою розумного будинку. Розроблено сучасний, зручний та інтуїтивно зрозумілий графічний інтерфейс. Після чого, реалізовано функціонування усіх блоків відповідно до узагальненого алгоритму та структури інформаційної технології. Таким чином, розроблено інформаційну технологію управління безпекою розумного будинку, яка на відміну від відповідних існуючих засобів, визначає не тільки загрози, а й вразливості засобу безпеки, що дозволяє завчасно попередити виникнення загрози, а також має розширений функціонал, що дозволяє завчасно створювати власнику будинку набір сценаріїв безпеки та забезпечує швидку реакцію на виявлені загрози.

Було проведено тестування для 30 користувачів, визначено 20 факторів впливу на стан безпеки, і як результат отримано понад 10 висновків. В результаті тестування визначено, що інформаційна технологія управління безпекою розумного будинку має розширений функціонал за рахунок можливості створення сценаріїв безпеки, яка дозволила підвищити швидкість реакції на виявлену загрозу більше, ніж на 50%, що свідчить про досягнення мети дослідження. Таким чином, це дозволило підвищити швидкість забезпечення безпеки розумного будинку – на 20%.

Проведено розрахунок витрат на розробку інформаційної технології, обраховано орієнтовну величину витрат, а також розраховано чистий прибуток. Розроблена інформаційна технологія управління безпекою розумного будинку є високо конкурентоспроможною. Період окупності складе приблизно 0,22 року.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Савчук Т. О. СТРУКТУРА ІНФОРМАЦІЙНОЇ ТЕХНОЛОГІЇ УПРАВЛІННЯ БЕЗПЕКОЮ РОЗУМНОГО БУДИНКУ / Т. О. Савчук, К. Г. Капченко. // Матеріали конференції «SIENCE AND INNOVATION OF MODERN WORLD»: Тез. доп. – *London, UK. 2023*.
2. Савчук Т. О. РОЗРОБКА АЛГОРИТМУ АНАЛІЗУ БЕЗПЕКИ РОЗУМНОГО БУДИНКУ / Т. О. Савчук, К. Г. Капченко. // Матеріали конференції «Науково-технічна конференція факультету інтелектуальних інформаційних технологій та автоматизації (2023)»: Тез. доп. – *м. Вінниця, Україна, 2023.*
3. Савчук Т. О. Комп’ютерна програма «Інформаційна технологія управління безпекою розумного будинку» / Савчук Т. О., Капченко К. Г. // Свідоцтво про реєстрацію авторського права на твір (АП) с202308055 Вх-47009/2023 від 28.11.2023
4. Розумний будинок − [Електронний ресурс]. – Режим доступу до ресурсу: https://oxorona.com/smart-home/
5. Безпека. − [Електронний ресурс]. – Режим доступу до ресурсу: https://www.smarthouse.ua/ua/bezopasnost.html.
6. Програмне забезпечення розумного будинку − [Електронний ресурс]. – Режим доступу до ресурсу: https://yalantis.com/blog/smart-home-automation-software/
7. Популярність розумних будинків − [Електронний ресурс]. – Режим доступу до ресурсу: https://www.statista.com/
8. Статистика − [Електронний ресурс]. – Режим доступу до ресурсу: https://cybersecurityventures.com/
9. Samsung SmartThings − [Електронний ресурс]. – Режим доступу до ресурсу: http://smartandyoung.com.ua/smart-things-shho-ce-za-programa-v-samsung
10. Google home − [Електронний ресурс]. – Режим доступу до ресурсу: https://home.google.com/welcome/

11. Vivint Smart Home − [Електронний ресурс]. – Режим доступу до ресурсу: https://www.vivint.com/packages/home-automation
12. C++ – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/C%2B%2B.
13. C Sharp – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/C_Sharp.
14. PHP – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/PHP.
15. JavaScript – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/JavaScript.
16. Java – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Java.
17. NodeJs – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Node.js.
18. NestJS – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/NestJS.
19. TypeScript – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/TypeScript.
20. Angular (фреймворк) – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Angular_(%D1%84%D1%80%D0%B5%D0%B9%D0%BC%D0%B2%D0%BE%D1%80%D0%BA).
21. Реляційна база даних – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Реляційна_база_даних.
22. PostgreSQL – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/PostgreSQL.
23. Microsoft Visual Studio – [Електронний ресурс] – Режим доступу до https://uk.wikipedia.org/wiki/Microsoft_Visual_Studio.
24. Visual Studio Code – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Visual_Studio_Code.

25. Інтерфейс користувача – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Інтерфейс_користувача.
26. Графічний інтерфейс користувача – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Графічний_інтерфейс_користувача.
27. Figma – [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/Figma.
28. Методичні вказівки до виконання магістерської кваліфікаційної роботи для студентів денної та заочної форм навчання спеціальності 122 - «Комп'ютерні науки» [Електронний ресурс] / Укладачі А. А. Яровий, О. К. Колесницький. – Вінниця: ВНТУ, 2023. – 58 с.
29. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт [Електронний ресурс] / Укладачі В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця: ВНТУ, 2021. – 47 с.

ДОДАТКИ

Додаток А (обов'язковий)

Результат перевірки на плагіат в онлайн-системі UNICHECK

ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: Інформаційна технологія управління безпекою розумного будинку

Тип роботи: магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ кафедра комп'ютерних наук, ФІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 84,9% Схожість 15,1%

Аналіз звіту подібності (відмітити потрібне):

✓ Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.

□ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

□ Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Ознайомлені з повним звітом подібності, який був згенерований системою Unicheck щодо роботи.

Автор роботи Капченко К.Г.

Керівник роботи Савчук Т.О.

Опис прийнятого рішення
Магістерську кваліфікаційну роботу допущено до захисту

Особа, відповідальна за перевірку Озеранський В.С.

Додаток Б (обов'язковий)

Лістинг програми

```
import { Body, Controller, Delete, Get, Param, Patch, Post } from '@nestjs/common';
import { ScenarioService } from './scenario.service';
import { CreateScenarioDto } from './dto/CreateScenario.dto';

@Controller()
export class ScenarioController {

  constructor(private ticketService: ScenarioService) {}

  @Post('scenarios/create')
  public async createScenario(@Body() createScenarioDto: CreateScenarioDto) {
    return await this.ticketService.createScenario(createScenarioDto);
  }

  @Get('scenarios/all')
  public async getAllScenarios() {
    return await this.ticketService.getAllScenarios();
  }

  @Get('scenarios/get-count')
  public async getScenarioCount() {
    return await this.ticketService.getScenariosCount();
  }

  @Get('scenarios/:id')
  public async getScenarioById(@Param('id') id: string) {
    return await this.ticketService.getScenarioById(id);
  }

  @Get('scenarios/company/:companyId')
  public async getScenarioByCompanyId(@Param('companyId') id: string) {
    return await this.ticketService.getScenarioByCompanyId(id);
  }

  @Post('scenarios/:id/change-status')
  public async changeScenarioStatus(@Param('id') id: string) {
    return await this.ticketService.changeScenarioStatus(id);
  }

  @Patch('scenarios/:id')
  public async updateScenarioById(@Body() updateScenarioDto: Partial<CreateScenarioDto>, @Param('id') id: string)
{
    return await this.ticketService.updateScenarioById(updateScenarioDto, id);
  }

  @Delete('scenarios/:id')
  public async deleteScenarioById(@Param('id') id: string) {
    return await this.ticketService.deleteScenarioById(id);
  }
}

import { BaseEntity } from 'src/common/base.entity';
import { Column, Entity } from 'typeorm';
```

```
export interface IScenatrioAttributes {
  id: string;
  title: string;
  description: string;
  ticketCreationDate: string;
  isActive: boolean;
  task: string;
  reason: string;
}

@Entity({ name: 'Scenario' })
export class ScenarioEntity extends BaseEntity implements IScenatrioAttributes {
  @Column({ type: 'varchar', length: 150 })
  title: string;
  @Column({ type: 'timestamp without time zone', nullable: true })
  ticketCreationDate: string;

  @Column({ type: 'boolean', nullable: false, default: false })
  isActive: boolean;

  @Column({ type: 'varchar', length: 150 })
  task: string;

  @Column({ type: 'varchar', length: 150 })
  reason: string;

  @Column({ type: 'varchar', length: 2500 })
  description: string;
}

import { Module } from '@nestjs/common';
import { ScenarioEntity } from './scenario.entity';
import { TypeOrmModule } from '@nestjs/typeorm';

@Module({
  imports: [TypeOrmModule.forFeature([ScenarioEntity])],
  controllers: [ScenarioController],
  providers: [ScenarioService]
})
export class ScenarioModule {}

import { Injectable } from '@nestjs/common';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository } from 'typeorm';
import { CreateScenarioDto } from './dto/CreateScenario.dto';
import { ScenarioEntity } from './scenario.entity';

@Injectable()
export class ScenarioService {
  constructor(
    @InjectRepository(ScenarioEntity)
    private scenarioRepository: Repository<ScenarioEntity>,
  ) {}

  public async getScenariosCount() {
    return await this.scenarioRepository.count();
```

```
  }

  public async createScenario(createScenarioDto: CreateScenarioDto) {
    // const ticketScoreAnalyze = this.analyzeScenarioDescription(createScenarioDto.description);
    return await this.scenarioRepository.save({ ...createScenarioDto });
  }

  public async getAllScenarios() {
    return await this.scenarioRepository.find();
  }

  public async getScenarioById(id: string) {
    return await this.scenarioRepository.findOne({ where: { id } });
  }

  public async updateScenarioById(
    updateScenarioDto: Partial<CreateScenarioDto>,
    id: string,
  ) {
    return await this.scenarioRepository.update(id, updateScenarioDto);
  }

  public async getScenarioByCompanyId(companyId: string) {
    // return await this.scenarioRepository.findBy({ companyId });
  }

  public async deleteScenarioById(id: string) {
    return await this.scenarioRepository.delete(id);
  }

  public async changeScenarioStatus(id: string) {
    return await this.scenarioRepository.update(id, { isActive: true });
  }
}

import { BaseEntity } from 'src/common/base.entity';
import { Column, Entity } from 'typeorm';

export interface IUserAttributes {
  id: string;
  email: string;
  password: string;
  fullName: string;
}

@Entity({ name: 'User' })
export class UserEntity extends BaseEntity implements IUserAttributes {
  @Column({ type: 'varchar', length: 150 })
  email: string;
  @Column({ type: 'varchar', length: 300 })
  password: string;
  @Column({ type: 'varchar', length: 300, nullable: true })
  fullName: string;
}

import { Injectable } from '@nestjs/common';
```

```
import { IUserAttributes, UserEntity } from './user.entity';
import { InjectRepository } from '@nestjs/typeorm';
import { Repository, UpdateResult } from 'typeorm';
import { CreateUserDto } from './dto/CreateUser.dto';

@Injectable()
export class UserService {
  constructor(
    @InjectRepository(UserEntity)
    private userEntityRepository: Repository<UserEntity>
  ) {}

  async findByEmail(email: string): Promise<UserEntity> {
    return await this.userEntityRepository.findOne({
      where: {
        email,
      },
    });
  }

  public async patchUser(
    id: string,
    data: Omit<IUserAttributes, 'id'>,
  ): Promise<UpdateResult> {
    return await this.userEntityRepository.update(id, data);
  }

  public async createUser(createUserDto: CreateUserDto) {
    const user = this.userEntityRepository.create(createUserDto);
    return await this.userEntityRepository.save(user);
  }
}

import { Controller, Post, UseGuards, Request, Body } from '@nestjs/common';
import { LocalAuthGuard } from './guards/localauth.guard';
import { JwtAuthGuard } from './guards/jwtauth.guard';
import { AuthService } from './auth.service';
import { CreateUserDto } from '../user/dto/CreateUser.dto';

@Controller()
export class AuthController {
  constructor(private authService: AuthService) {}

  @UseGuards(LocalAuthGuard)
  @Post('auth/login')
  public async login(@Request() req) {
    return this.authService.login(req.user)
  }

  @Post('auth/registration')
  public async registration(@Body() createUserDto: CreateUserDto) {
    return this.authService.registration(createUserDto)
  }
}

import { Injectable } from '@nestjs/common';
```

```
import { UserService } from '../user/user.service';
import { JwtService } from '@nestjs/jwt';
import { IUserAttributes, UserEntity } from '../user/user.entity';
import { CreateUserDto } from '../user/dto/CreateUser.dto';
import { ConfigService } from '@nestjs/config';

@Injectable()
export class AuthService {
  private jwt_secret;
  constructor(
    private userService: UserService,
    private jwtService: JwtService,
    private configService: ConfigService
  ) {
    this.jwt_secret = this.configService.get('jwt_secret')
  }

  public async validateUser(email: string, password: string) {
    const user = await this.userService.findByEmail(email);

    if (user && user.password === password) {
      const { password, ...result } = user;
      return result;
    }

    return null;
  }

  public login(user: any) {
    const payload = { email: user.email, id: user.id };
    return {
      access_token: `Bearer ${this.jwtService.sign(payload, { secret: this.jwt_secret })}`,
    };
  }

  public async registration(createUserDto: CreateUserDto) {
    return await this.userService.createUser(createUserDto);
  }
}

import { PassportStrategy } from "@nestjs/passport";
import { AuthService } from "../auth.service";
import { Injectable, UnauthorizedException } from "@nestjs/common";
import { Strategy } from "passport-local";

@Injectable()
export class LocalStrategy extends PassportStrategy(Strategy) {
  constructor(private authService: AuthService) {
    super({ usernameField: 'email' });
  }

  validate(email: string, password: string) {
    const user = this.authService.validateUser(email, password);
    if(!user) {
      throw new UnauthorizedException();
    }
```

```
    return user
  }
}
import { Component } from '@angular/core';
import { Router } from '@angular/router';
import { TicketService } from '../../services/ticket.service';
import { ITicket } from '../../types/types';

@Component({
  selector: 'app-analytics',
  templateUrl: './analytics.component.html',
  styleUrls: ['./analytics.component.scss'],
})
export class AnalyticsComponent {
  public criterias: any = {
    PHYSICO: {},
    NETWORK: {},
    CYBER: {},
    MONITORING: {},
  };

  public defaultCriterias: any = [];
  public isError: boolean = false;
  constructor(private ticketService: TicketService, private router: Router) {}

  ngOnInit(): void {
    this.ticketService.getAllAnalytic().subscribe({
      next: (res: any) => {
        this.defaultCriterias = res.map((el: any) => {
          return { ...el, title: CRITERIA_TEXT[el.criteria] };
        });
        this.isError = this.defaultCriterias.find((el: any) => el.isActivated || !el.isEnabled)
        console.log(this.isError)
        res.forEach((element: any) => {
          if (element.group in this.criterias[element.type]) {
            this.criterias[element.type][element.group].items.push({
              ...element,
              title: CRITERIA_TEXT[element.criteria],
            });
            this.criterias[element.type][element.group].isGood = this.criterias[
              element.type
            ][element.group].isGood
              ? element.isEnabled && !element.isActivated
              : false;
          } else {
            this.criterias[element.type][element.group] = {
              title: GROUP_TEXT[element.group],
              items: [
                {
                  ...element,
                  title: CRITERIA_TEXT[element.criteria],
                },
              ],
              isGood: element.isEnabled && !element.isActivated,
            };
          }
```

```
        });
        console.log(this.criterias);
      },
    });
  }

  // public selectTicket(ticketId: string) {
  //   this.router.navigate([`/main/orders/${ticketId}`])
  // }

  public toCreate() {
    this.router.navigate([`/main/orders/create`]);
  }
}

const GROUP_TEXT: any = {
  BROKE: 'Взлом',
  DETECTORS: 'Спрацювання датчиків',
  STRANGE: 'Незвичайні активності',
  ANTIVIRUS: 'Антивірус',
  CONFIDENCE: 'Захист даних та конфіденційність',
  MOBILE: 'Мобільна та віддалена безпека',
  ENERGY: 'Моніторинг енергоспоживання',
  LIGHT: 'Моніторинг освітлення',
  TEMPERATURE: 'Моніторинг температури та вологості',
  MONITOR: 'Моніторинг безпеки',
  SECURE_CYBER: 'Захист від вторгнень',
  SECURE_NETWORK: 'Захист периметру мережі',
};

const CRITERIA_TEXT: any = {
  ENERGY_SYSTEM: 'Моніторинг енергосистеми',
  WINDOW_BREAK: 'Поломка замків дверей',
  SECURE_WIFI: 'Поломка WiFi',
  SECURE_PENT: 'Захист від вторгень',
  TEMPERATURE_DETECTOR: 'Датчик температури',
  VIDEO: 'Відеокамери',
  LIGHT_DETECTOR: 'Датчик світла',
  REMOTE_ACCESS: 'Віддалений доступ',
  GAS_DETECTOR: 'Датчик газу',
  MOVE_DETECTOR: 'Датчик руху',
  FIRE_DETECTOR: 'Датчик диму',
  UPDATES: 'Оновлення антивірусу',
  SECURE: 'Оновлення захисту',
  FILTER_TRAFIC: 'Фільтрація мережевого трафіку',
  SIGNALISATION: 'Сигналізація',
  WATER_DETECTOR: 'Датчик води',
  SIEM: 'Системи аналізу безпеки',
  REMOTE_CONTOL: 'Віддалене управління',
  IDENTIFY_PENT: 'Ідентифікація вторгнень',
  IDENTIFY: 'Ідентифікація',
  VPN: 'Віртуальні приватні мережі',
  DOOR_BREAK: 'Поломка замків дверей',
  SECURE_AUTH: 'Захист авторизації',
};
```

```
import { Component } from '@angular/core';
import { FormArray, FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { TicketService } from '../../services/ticket.service';
import { JwtService } from 'src/app/modules/shared/services/jwt.service';

@Component({
  selector: 'app-create-ticket',
  templateUrl: './create-ticket.component.html',
  styleUrls: ['./create-ticket.component.scss'],
})
export class CreateTicketComponent {
  private tokenData: any;
  public options: any[] = [];
  public createTicketForm!: FormGroup<{
    title: FormControl<string | null>;
    description: FormControl<string | null>;
    task: FormControl<string | null>;
    reason: FormControl<string | null>;
  }>;

  constructor(
    private router: Router,
    private ticketService: TicketService,
    private jwtService: JwtService
  ) {
    const token = localStorage.getItem('access_token')?.split(' ')[1];
    this.tokenData = this.jwtService.getTokenPayload(token as string);
  }

  ngOnInit(): void {
    this.ticketService.getAllAnalytic().subscribe({
      next: (res: any) => {
        this.options = res.map((element: any) => {
          return {key: CRITERIA_TEXT[element.criteria], value: element.criteria}
        });
        console.log(this.options)
      },
    });
    this.createTicketForm = new FormGroup({
      title: new FormControl('', [Validators.required]),
      reason: new FormControl('', [Validators.required]),
      description: new FormControl('', [Validators.required]),
      task: new FormControl('', [Validators.required]),
    });
  }

  public submitCreation() {
    console.log('this', this.createTicketForm.controls);
    const isFormValid = this.createTicketForm.valid;
    if (!isFormValid) {
      alert('Form invalid');
      return;
    }

    const formValue = this.createTicketForm.value;
```

```
    console.log(formValue);

    const createTicketPayload = {
      ...formValue,
    };

    this.ticketService.createTicket(createTicketPayload as any).subscribe({
      next: (response) => {
        console.log('RES', response);
      },
    });
  }
}

const CRITERIA_TEXT: any = {
  "ENERGY_SYSTEM": "Моніторинг енергосистеми",
  "WINDOW_BREAK": "Поломка замків дверей",
  "SECURE_WIFI": "Поломка WiFi",
  "SECURE_PENT": "Захист від вторгень",
  "TEMPERATURE_DETECTOR": "Датчик температури",
  "VIDEO": "Відеокамери",
  "LIGHT_DETECTOR": "Датчик світла",
  "REMOTE_ACCESS": "Віддалений доступ",
  "GAS_DETECTOR": "Датчик газу",
  "MOVE_DETECTOR": "Датчик руху",
  "FIRE_DETECTOR": "Датчик диму",
  "UPDATES": "Оновлення антивірусу",
  "SECURE": "Оновлення захисту",
  "FILTER_TRAFIC": "Фільтрація мережевого трафіку",
  "SIGNALISATION": "Сигналізація",
  "WATER_DETECTOR": "Датчик води",
  "SIEM": "Системи аналізу безпеки",
  "REMOTE_CONTOL": "Віддалене управління",
  "IDENTIFY_PENT": "Ідентифікація вторгнень",
  "IDENTIFY": "Ідентифікація",
  "VPN": "Віртуальні приватні мережі",
  "DOOR_BREAK": "Поломка замків дверей",
  "SECURE_AUTH": "Захист авторизації",
}

function transformDateToMS(date: string) {
  return new Date(date).getTime();
}

import { Component } from '@angular/core';
import { ITicket } from '../../types/types';
import { TicketService } from '../../services/ticket.service';
import { Router } from '@angular/router';

@Component({
  selector: 'app-created-tickets',
  templateUrl: './created-tickets.component.html',
  styleUrls: ['./created-tickets.component.scss'],
})
export class CreatedTicketsComponent {
```

```
  public scenarios!: ITicket[];

  constructor(private ticketService: TicketService, private router: Router) {}

  ngOnInit(): void {
    this.ticketService.getMainTickets().subscribe({
      next: (tickets: any) => {
        console.log('TICKETS', tickets);
        this.scenarios = tickets;
      },
    });
  }

  public selectTicket(ticketId: string, event: any) {
    console.log(event);
    if (
      event.srcElement.tagName == 'IMG' ||
      event.srcElement.nodeName == 'INPUT' ||
      event.srcElement.nodeName == 'LABEL'
    ) {
      return;
    }
    this.router.navigate([`/main/orders/${ticketId}`]);
  }

  public toCreate() {
    this.router.navigate([`/main/orders/create`]);
  }

  public toggleCheckBox(scenario: ITicket) {
    this.ticketService.updateScenario(scenario.id, {
      isActive: !scenario.isActive,
    }).subscribe({next: (res) => {
      console.log(res)
    }});
  }
}


import { Component, OnInit } from '@angular/core';
import { TicketService } from '../../services/ticket.service';
import { ITicket } from '../../types/types';
import { Router } from '@angular/router';

@Component({
  selector: 'app-main-tickets',
  templateUrl: './main-tickets.component.html',
  styleUrls: ['./main-tickets.component.scss'],
})
export class MainTicketsComponent implements OnInit {

  public defaultCriterias: any = [];
  public criterias: any = {
    PHYSICO: {},
    NETWORK: {},
    CYBER: {},
```

```
    MONITORING: {},
  };

  constructor(private ticketService: TicketService, private router: Router) {}

  ngOnInit(): void {
    this.ticketService.getAllAnalytic().subscribe({
      next: (res: any) => {
        this.defaultCriterias = res.map((el: any) => {
          return { ...el, title: CRITERIA_TEXT[el.criteria] };
        });
        res.forEach((element: any) => {
          if (Object.keys(this.criterias[element.type]).length) {
            this.criterias[element.type].total += 1
            if(element.isEnabled && !element.isActivated) {
              this.criterias[element.type].good += 1
            }
          } else {
            this.criterias[element.type]= {
              total: 1,
              good: element.isEnabled && !element.isActivated ? 1 : 0
            };
          }
        });
        console.log(this.criterias);
      },
    });
  }

  // public selectTicket(ticketId: string) {
  //   this.router.navigate([`/main/orders/${ticketId}`])
  // }

  public toCreate() {
    this.router.navigate([`/main/orders/create`])
  }
}


const CRITERIA_TEXT: any = {
  ENERGY_SYSTEM: 'Моніторинг енергосистеми',
  WINDOW_BREAK: 'Поломка замків дверей',
  SECURE_WIFI: 'Поломка WiFi',
  SECURE_PENT: 'Захист від вторгень',
  TEMPERATURE_DETECTOR: 'Датчик температури',
  VIDEO: 'Відеокамери',
  LIGHT_DETECTOR: 'Датчик світла',
  REMOTE_ACCESS: 'Віддалений доступ',
  GAS_DETECTOR: 'Датчик газу',
  MOVE_DETECTOR: 'Датчик руху',
  FIRE_DETECTOR: 'Датчик диму',
  UPDATES: 'Оновлення антивірусу',
  SECURE: 'Оновлення захисту',
  FILTER_TRAFIC: 'Фільтрація мережевого трафіку',
  SIGNALISATION: 'Сигналізація',
  WATER_DETECTOR: 'Датчик води',
```

```
  SIEM: 'Системи аналізу безпеки',
  REMOTE_CONTOL: 'Віддалене управління',
  IDENTIFY_PENT: 'Ідентифікація вторгнень',
  IDENTIFY: 'Ідентифікація',
  VPN: 'Віртуальні приватні мережі',
  DOOR_BREAK: 'Поломка замків дверей',
  SECURE_AUTH: 'Захист авторизації',
};


import { Component } from '@angular/core';
import { TicketService } from '../../services/ticket.service';
import { ActivatedRoute, Router } from '@angular/router';
import { ITicket } from '../../types/types';

@Component({
  selector: 'app-ticket',
  templateUrl: './ticket.component.html',
  styleUrls: ['./ticket.component.scss'],
})
export class TicketComponent {
  private ticketId!: string;
  public ticket!: ITicket;
  constructor(
    private ticketService: TicketService,
    private route: ActivatedRoute,
    private router: Router
  ) {
    this.route.params.subscribe({
      next: (params: any) => (this.ticketId = params.id),
    });
  }

  ngOnInit(): void {
    this.ticketService.getTicketById(this.ticketId).subscribe({
      next: (ticket: any) => {
        console.log('TICKET', ticket);
        this.ticket = ticket;
      },
    });
  }

  toEdit() {
    this.router.navigate([`/main/orders/update/${this.ticketId}`])
  }

  deleteScenario(id: string) {
    this.ticketService.deleteTicket(id).subscribe({
      next: (ticket: any) => {
        this.router.navigate(['/main/orders/my']);
      },
    });
  }
}

import { Component } from '@angular/core';
```

```
import { FormArray, FormControl, FormGroup, Validators } from '@angular/forms';
import { ActivatedRoute, Router } from '@angular/router';
import { TicketService } from '../../services/ticket.service';
import { JwtService } from 'src/app/modules/shared/services/jwt.service';

@Component({
  selector: 'app-create-ticket',
  templateUrl: './update-ticket.component.html',
  styleUrls: ['./update-ticket.component.scss'],
})
export class UpdateTicketComponent {
  private scenarioId!: string;
  private tokenData: any;
  public createTicketForm!: FormGroup<{
    title: FormControl<string | null>;
    description: FormControl<string | null>;
    task: FormControl<string | null>;
    reason: FormControl<string | null>;
  }>;

  constructor(
    private router: Router,
    private ticketService: TicketService,
    private jwtService: JwtService,
    private activeRoute: ActivatedRoute
  ) {
    const token = localStorage.getItem('access_token')?.split(' ')[1];
    this.tokenData = this.jwtService.getTokenPayload(token as string);
    this.activeRoute.params.subscribe({
      next: (params: any) => (this.scenarioId = params.id),
    });
  }

  ngOnInit(): void {
    this.createTicketForm = new FormGroup({
      title: new FormControl('', [Validators.required]),
      reason: new FormControl('', [Validators.required]),
      description: new FormControl('', [Validators.required]),
      task: new FormControl('', [Validators.required]),
    });
    this.ticketService.getTicketById(this.scenarioId).subscribe({
      next: (res: any) => {
        this.createTicketForm.patchValue({
          title: res?.title,
          reason: res?.reason,
          description: res?.description,
          task: res?.task,
        });
      },
    });
  }

  public submitCreation() {
    console.log('this', this.createTicketForm.controls);
    const isFormValid = this.createTicketForm.valid;
    if (!isFormValid) {
```

```
      alert('Form invalid');
      return;
    }

    const formValue = this.createTicketForm.value;

    console.log(formValue);

    const createTicketPayload = {
      ...formValue,
    };

    this.ticketService.updateScenario(this.scenarioId, createTicketPayload as any).subscribe({
      next: (response) => {
        this.router.navigate(['/main/orders/my'])
      },
    });
  }
}

function transformDateToMS(date: string) {
  return new Date(date).getTime();
}

import { Component, OnInit } from '@angular/core';
import { FormControl, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { AuthService } from 'src/app/modules/shared/services/auth.service';
import { ILoginPayload } from 'src/app/modules/shared/types/types';

@Component({
  selector: 'app-login',
  templateUrl: './login.component.html',
  styleUrls: ['./login.component.scss'],
})
export class LoginComponent implements OnInit {
  public loginForm!: FormGroup<{
    email: FormControl<string | null>;
    password: FormControl<string | null>;
  }>;

  constructor(private router: Router, private authService: AuthService) {}

  ngOnInit(): void {
    this.loginForm = new FormGroup({
      email: new FormControl('', [Validators.required, Validators.email]),
      password: new FormControl('', [
        Validators.required,
        Validators.minLength(4),
      ]),
    });
  }

  public submitLogin() {
    const isFormValid = this.loginForm.valid
    if(!isFormValid) {
```

```
      alert("Form invalid")
      return
    }
    this.authService.login(this.loginForm.value as ILoginPayload).subscribe({next: (response: any) => {
      console.log(response)
      localStorage.setItem('access_token', response.access_token)
      this.router.navigate(['/main/orders/orders']);
    }})
  }
}

import { Component, OnInit } from '@angular/core';
import { FormGroup, FormControl, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { AuthService } from 'src/app/modules/shared/services/auth.service';
import {
  ILoginPayload,
  IRegisterPayload,
} from 'src/app/modules/shared/types/types';

@Component({
  selector: 'app-signup',
  templateUrl: './signup.component.html',
  styleUrls: ['./signup.component.scss'],
})
export class SignupComponent implements OnInit {
  public registerForm!: FormGroup<{
    fullName: FormControl<string | null>;
    email: FormControl<string | null>;
    password: FormControl<string | null>;
    confirmPassword: FormControl<string | null>;
  }>;

  constructor(private router: Router, private authService: AuthService) {}

  ngOnInit(): void {
    this.registerForm = new FormGroup({
      fullName: new FormControl('', [Validators.required]),
      email: new FormControl('', [Validators.required, Validators.email]),
      password: new FormControl('', [
        Validators.required,
        Validators.minLength(4),
      ]),
      confirmPassword: new FormControl('', [
        Validators.required,
        Validators.minLength(4),
      ]),
    });
  }

  public submitRegister() {
    const isFormValid = this.registerForm.valid;
    if (!isFormValid) {
      alert('Form invalid');
      return;
    }
```

Додаток В (обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

ІНФОРМАЦІЙНА ТЕХНОЛОГІЯ УПРАВЛІННЯ БЕЗПЕКОЮ РОЗУМНОГО БУДИНКУ

Виконала: студентка 2-го курсу, групи 1КН-22м
спеціальності 122 «Комп'ютерні науки»
(шифр і назва напряму підготовки, спеціальності)
_____ Капченко К. Г.
(прізвище та ініціали)

Керівник: PhD, професор каф. КН
_____ Савчук Т. О.
(прізвище та ініціали)
«___» _____ 2023 р.

Опонент: д.т.н, професор каф. АІІТ
_____ Бісікало О. В.
(прізвище та ініціали)
«07» 12 2023 р.

Вінниця ВНТУ – 2023 рік

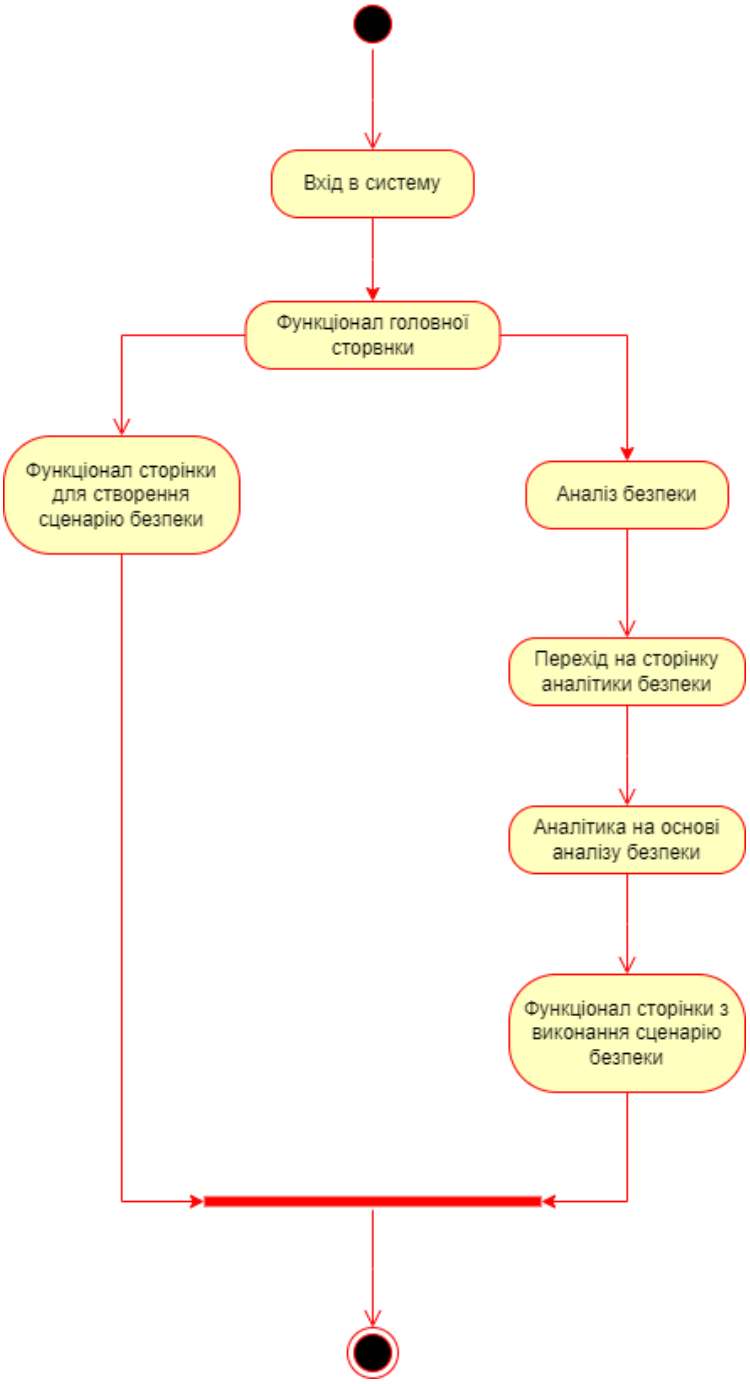

Рисунок Г.1 – UML-діаграма активності для управління безпекою розумного будинку

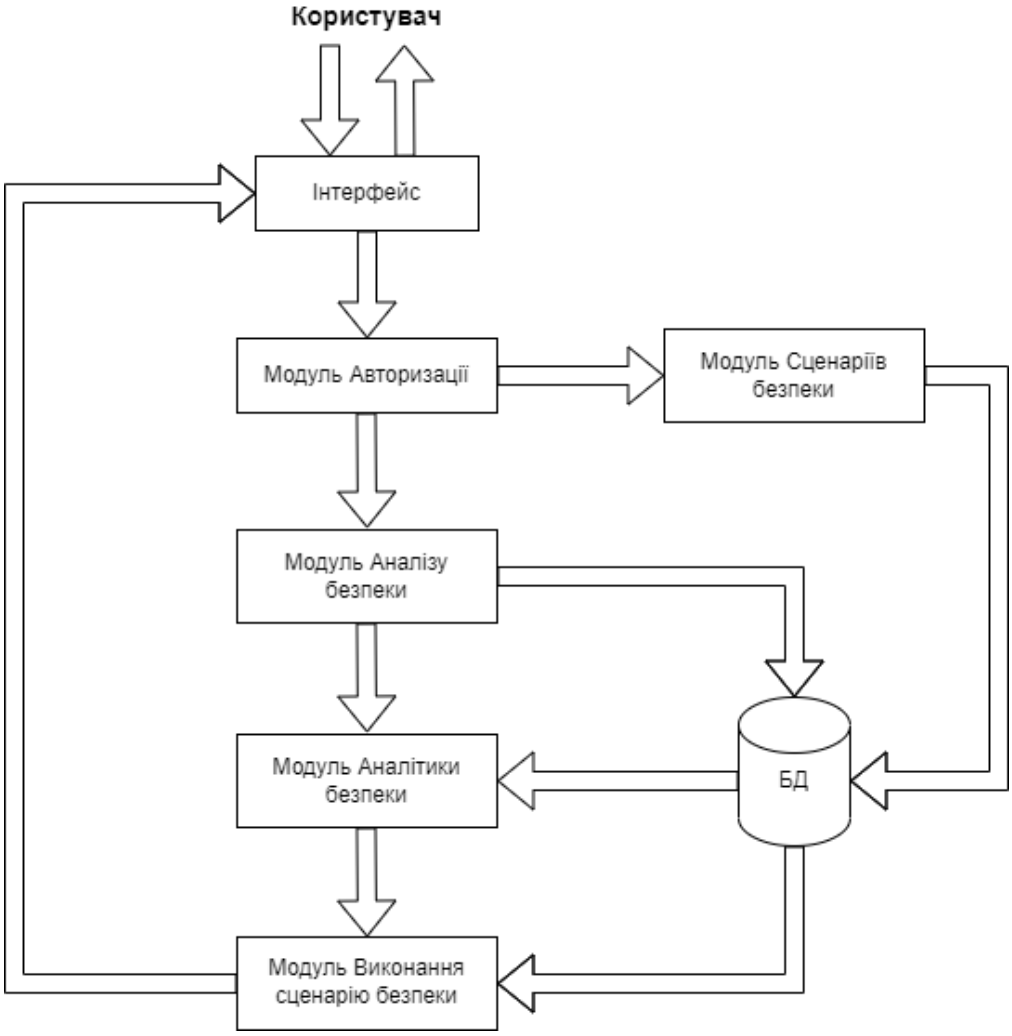

Рисунок Г.2 – Структура інформаційної технології управління безпекою розумного будинку

Рисунок Г.3 – UML-діаграма удосконаленого алгоритму функціонування модуля аналізу безпеки

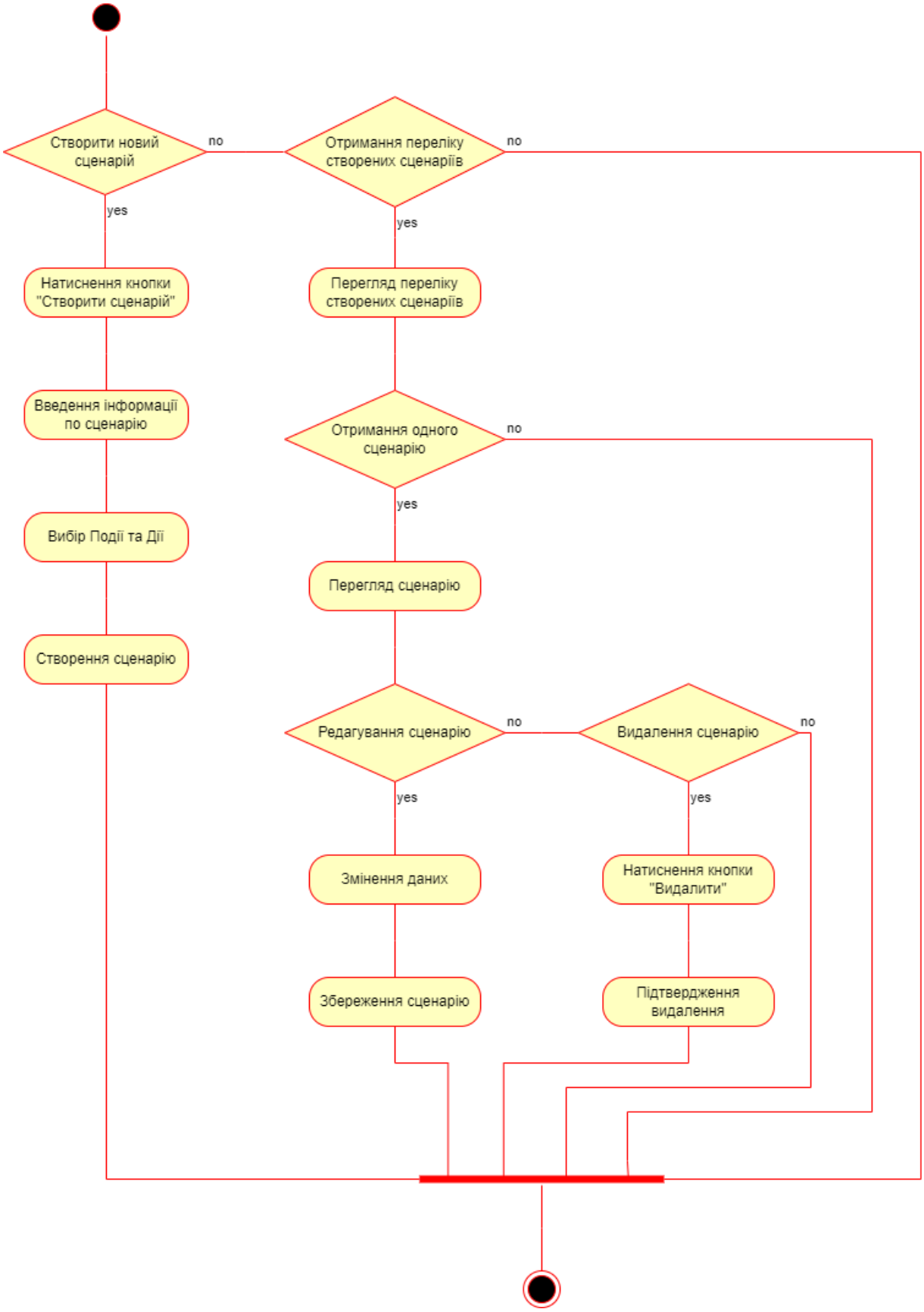

Рисунок Г.4 – UML-діаграма алгоритму функціонування модуля сценаріїв безпеки

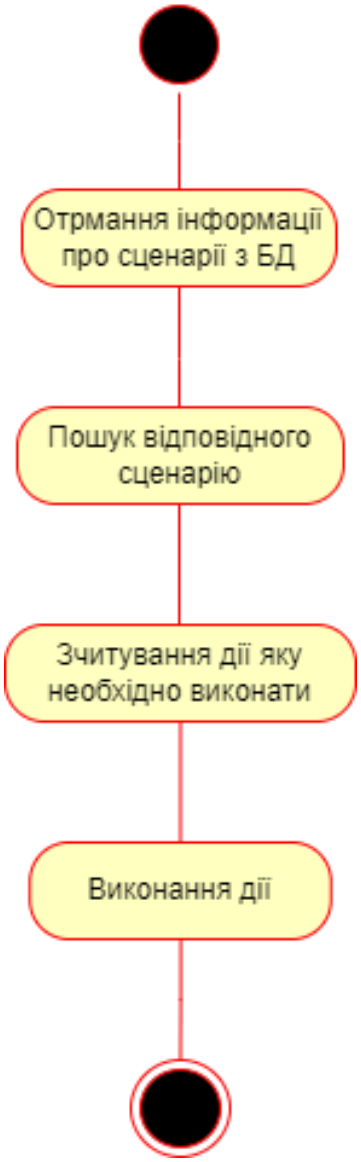

Рисунок Г.5 – UML-діаграма алгоритму функціонування модуля виконання сценарію безпеки

Додаток Г (довідниковий)

Інструкція користувача

Крок 1. Відкрити інформаційну технологію управління безпекою розумного будинку, зображену на рисунку Г.1.

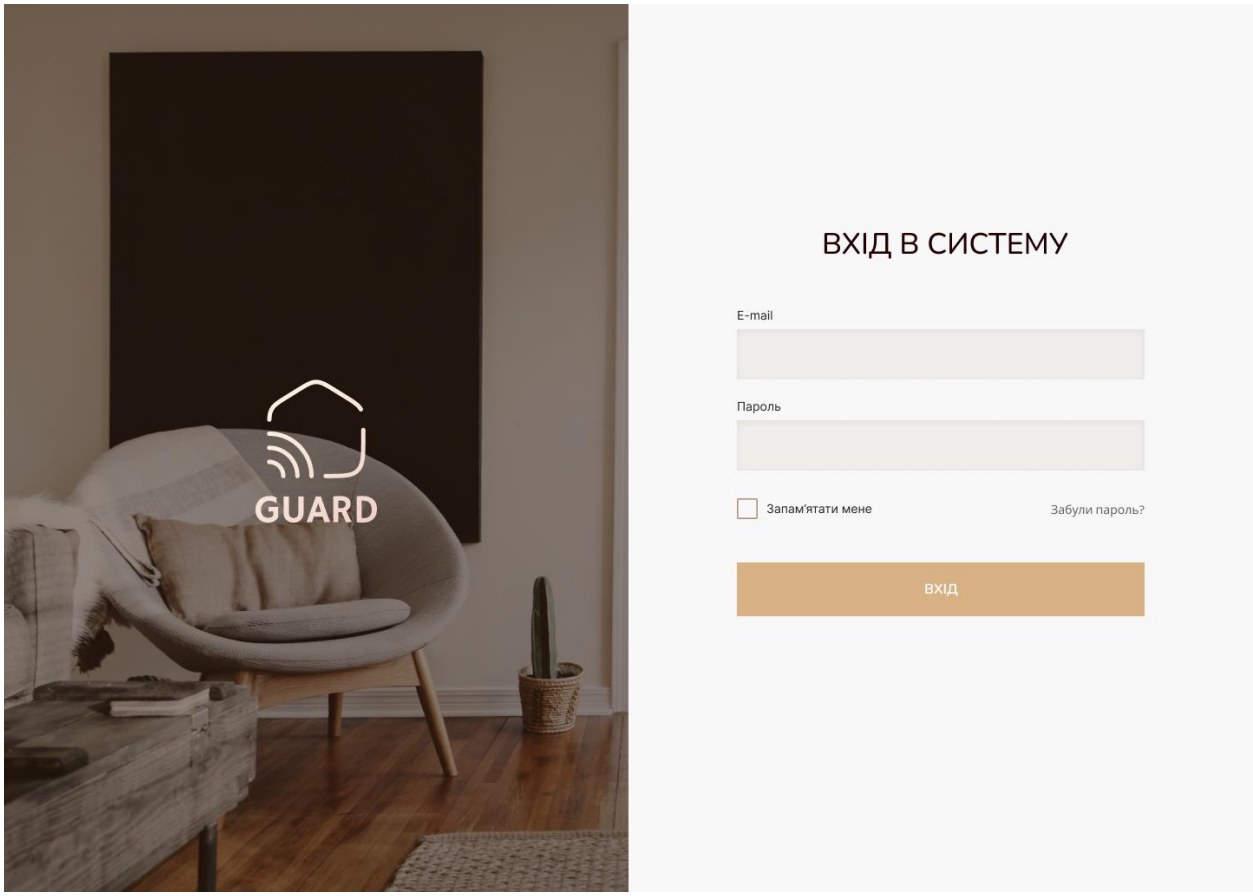

Рисунок Б.1 – Початкова сторінка інформаційної технології

Крок 2. Зареєструватись, якщо не має облікового запису, як зображено на рисунку Г.2.

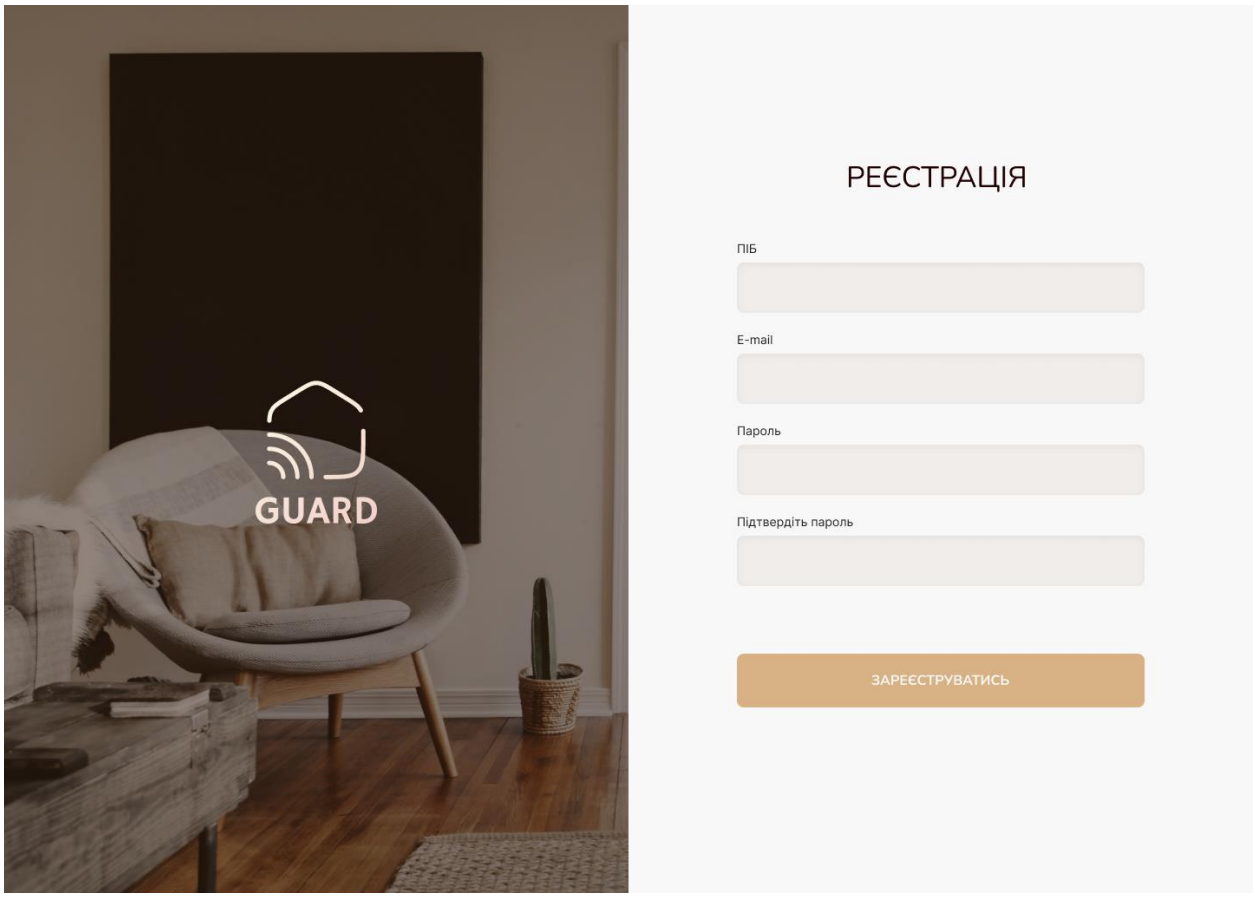

Рисунок Г.2 – Реєстрація користувача

Крок 3. Після успішної реєстрації та входу, користувач потрапляє на головну сторінку, на якій відображений аналітика безпеки розумного будинку, та вказана дата і час останнього аналізу. Користувач може переглянути інформацію про те, чи виявленні вразливості та загрози безпеці та перейти на сторінку з детальною інформацією по аналітиці. Також на головній сторінці є можливість запустити аналіз безпеки та перейти на сторінку створення сценарію безпеки. Головну сторінку інформаційної технології управління безпекою розумного будинку відображено на рисунку Г.3.

Рисунок Г.3 – Головна сторінка інтелектуального модулю

Крок 4. Натиснувши на кнопку створення сценарію, переходимо на сторінку «Створення нового сценарію», рисунок Г.4.

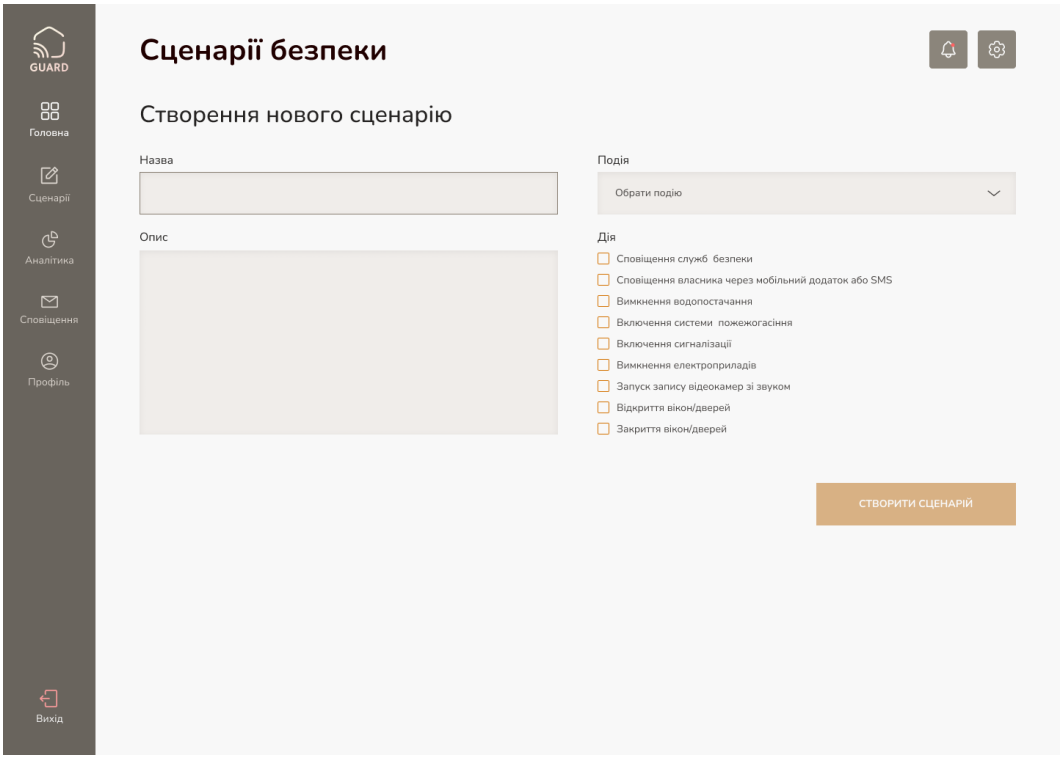

Рисунок Г.4 – Сторінка створення нового сценарію

Крок 5. З головної сторінки можна перейти на сторінку з аналітикою безпеки, зображену на рисунку Г.5.

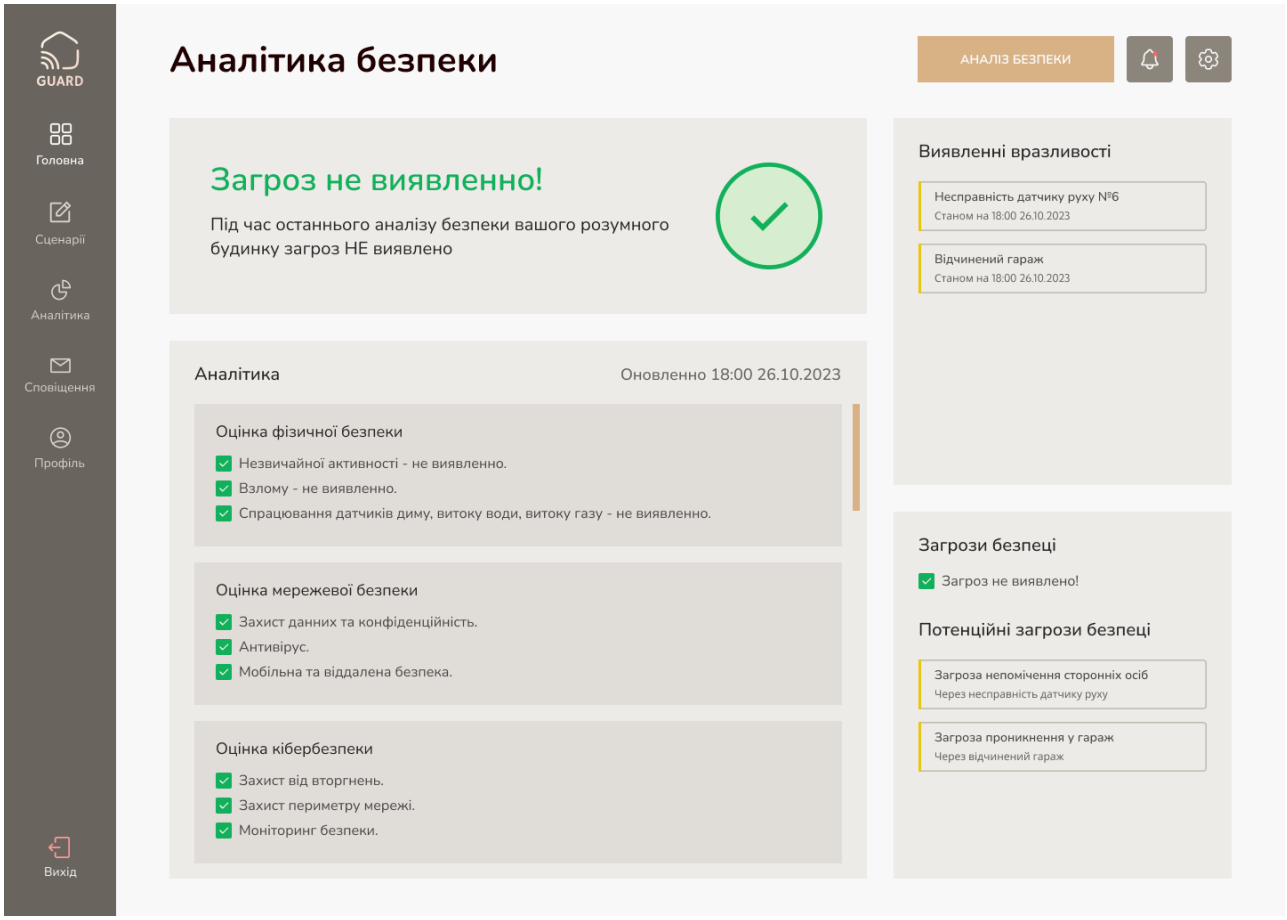

Рисунок Г.5 – Сторінка аналітики безпеки розумного будинку

Крок 6. Обравши в меню розділ «Сценарії» відбувається перехід на сторінку з існуючими сценаріями, де є можливість створити новий сценарій, редагувати існуючий. На рисунку Г.6 зображена сторінка сценаріїв безпеки.

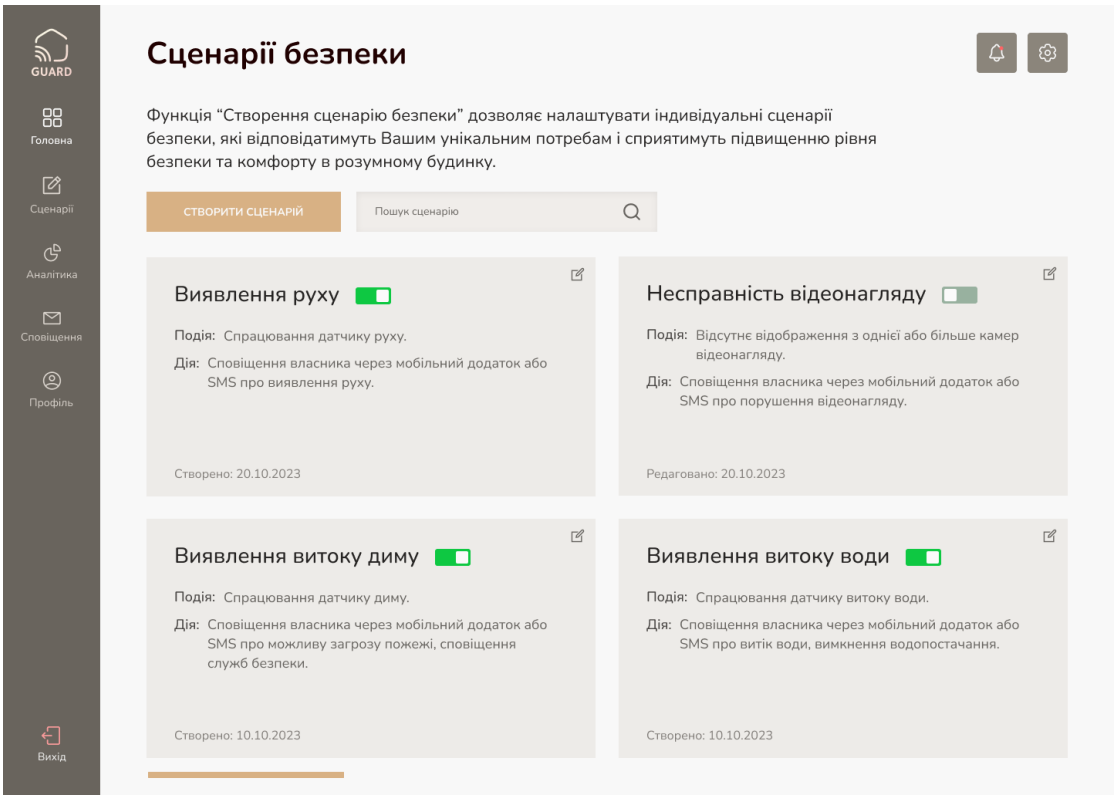

Рисунок Г.6 – Сторінка «Сценарії безпеки»

Крок 7. Обравши певний сценарій відбувається перехід на сторінку з інформацією по конкретному сценарію, інтерфейс якої зображено на рисунку Г.7.

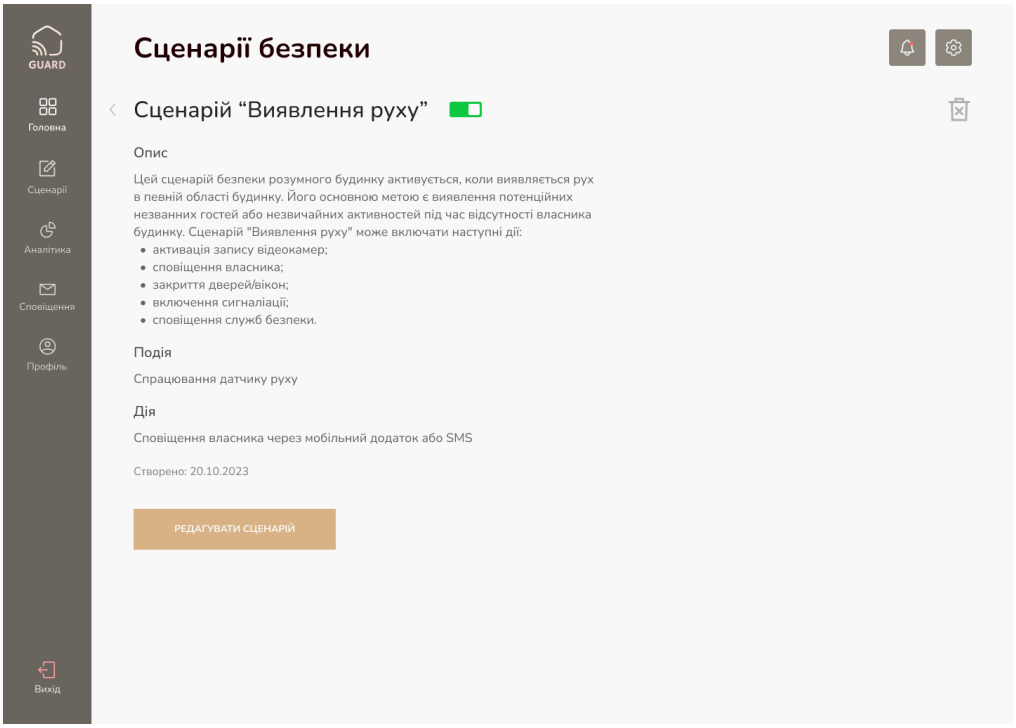

Рисунок Г.7 – Сторінка сценарію

Крок 8. Натиснувши кнопку «Редагувати сценарій» відбувається перехід на сторінку редагування сценарію. Сторінку редагування сценарію зображено на рисунку Г.8.

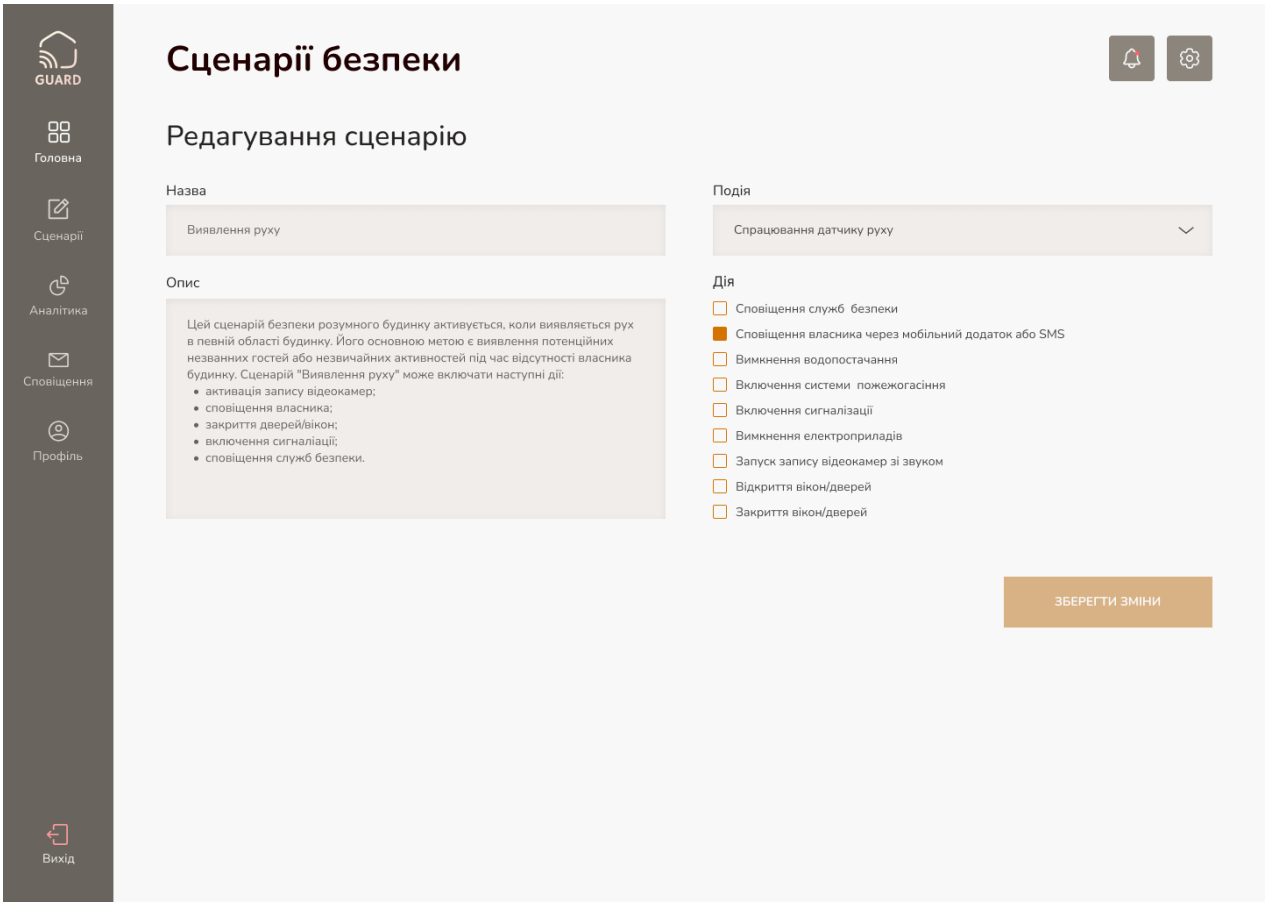

Рисунок Г.8 – Сторінка редагування сценарію