ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ
КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему: «Інформаційна система автоматизованого оповіщення та управління евакуацією людей із будівель»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
(код, найменування спеціальності)
освітньо-професійної програми Інформаційні системи та технології
(назва)

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

_____ Микола КЛИМОВИЧ
(підпис) *Ім'я, ПРІЗВИЩЕ здобувача*

Виконала: здобувача вищої освіти гр. ІСД-41
Микола КЛИМОВИЧ
Ім'я, ПРІЗВИЩЕ

Керівник: Ольга ЖИДКА
Ім'я, ПРІЗВИЩЕ

Рецензент:

_____
Ім'я, ПРІЗВИЩЕ

Київ 2026

ДЕРЖАВНИЙ УНІВЕРСИТЕТ
ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ
Навчально-науковий інститут інформаційних технологій

Кафедра Інформаційних систем та технологій
Ступінь вищої освіти Бакалавр
Спеціальність 126 Інформаційні системи та технології
Освітньо-професійна програма Інформаційні системи та технології

ЗАТВЕРДЖУЮ
Завідувач кафедрою ІСТ
_____Каміла СТОРЧАК
«_____» _____2026 р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
Климовичу Миколі Олександровичу
(прізвище, ім'я, по батькові здобувача)

1. Тема кваліфікаційної роботи: Інформаційна система автоматизованого оповіщення та управління евакуацією людей із будівель.
керівник кваліфікаційної роботи Ольга ЖИДКА
(Ім'я, ПРІЗВИЩЕ)
затверджені наказом Державного університету інформаційно-комунікаційних технологій від «20» лютого 2026р. №51
2. Строк подання кваліфікаційної роботи «01» червня 2026 р.
3. Вихідні дані до кваліфікаційної роботи:
    1. Нормативна база: ДСТУ, ДБН.
    2. Технологічний стек: Python, СУБД, HTML, CSS, JS.
    3. Математичний апарат: теорія графів, модифікований алгоритм Дейкстри.
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
    1. Аналіз предметної області, існуючих систем та нормативних вимог.
    2. Проєктування архітектури системи, визначення структури та взаємодії.
    3. Опис програмної реалізації.
    4. Експериментальна оцінка ефективності розробленого рішення.
5. Перелік графічного матеріалу: *презентація*
6. Дата видачі завдання «20» лютого 2026 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Підбір технічної літератури	20.02 – 28.02.2026	
2	Аналіз предметної області, існуючих систем та нормативних вимог	01.03 – 13.03.2026	
3	Проєктування архітектури системи, визначення структури та взаємодії	14.03 – 26.03.2026	
4	Проєктування бази данних	27.03 – 08.04.2026	
5	Опис програмної реалізації	08.04 – 28.04.2026	
6	Експериментальна оцінка ефективності розробленого рішення	29.04 – 13.05.2026	
7	Розробка демонстраційних матеріалів	14.05 – 20.05.2026	
8	Оформлення бакалаврської роботи	21.05 – 01.06.2026	

Здобувач вищої освіти _____ Микола КЛИМОВИЧ
(підпис) *(Ім'я, ПРІЗВИЩЕ)*

Керівник
кваліфікаційної роботи _____ Ольга ЖИДКА
(підпис) *(Ім'я, ПРІЗВИЩЕ*

РЕФЕРАТ

Текстова частина кваліфікаційної роботи на здобуття освітнього ступеня бакалавра: 63 стор., 4 рис., 7 табл., 15 джерел.

Мета роботи – підвищення ефективності та рівня безпеки евакуації людей за рахунок розробки інформаційної системи автоматизованого оповіщення та управління, яка динамічно обчислює та адаптує маршрути руху із застосуванням алгоритмів пошуку на графах.

Об'єкт дослідження – процеси управління евакуацією людей із будівель та споруд під час виникнення надзвичайних ситуацій.

Предмет дослідження – методи, алгоритми та програмні засоби динамічного розрахунку безпечних евакуаційних маршрутів і автоматизованого оповіщення користувачів.

Короткий зміст роботи: Проведено порівняльний аналіз існуючих комерційних та наукових аналогів, досліджено чинні нормативні вимоги та державні стандарти. Виявлено ключові проблеми та недоліки сучасних систем. Розроблено загальну архітектуру інформаційної системи та визначено її функціональну модель. Описано процеси взаємодії компонентів за допомогою діаграм моделювання. Спроєктовано реляційну базу даних. Наведено обґрунтування вибору технологічного стеку, що включає мову програмування Python, мікрофреймворк Flask та СУБД SQLite. Проведено порівняльну характеристику створеного рішення з базовими аналогами та визначено економічну доцільність і практичну цінність його впровадження.

КЛЮЧОВІ СЛОВА: ІНФОРМАЦІЙНА СИСТЕМА, АВТОМАТИЗОВАНОГО ОПОВІЩЕННЯ, УПРАВЛІННЯ ЕВАКУАЦІЄЮ,

АЛГОРИТМ ДЕЙКСТРИ, ДИНАМІЧНИЙ МАРШРУТ, ВЕБ-ЗАСТОСУНОК, FLASK, SQL

ЗМІСТ

ВСТУП

Актуальність теми. У сучасних умовах стрімкого розвитку урбанізації, збільшення кількості багатоповерхових будівель, торговельних центрів, офісних комплексів та інших об’єктів масового перебування людей особливої актуальності набуває проблема забезпечення безпеки та своєчасної евакуації у разі виникнення надзвичайних ситуацій. Пожежі, техногенні аварії, витоки небезпечних речовин або інші критичні події можуть становити серйозну загрозу для життя та здоров’я людей, особливо за умов недостатньо ефективного оповіщення та організації евакуаційних процесів.

Традиційні системи оповіщення, які базуються на звукових сигналах або простих інформаційних повідомленнях, часто не забезпечують достатнього рівня ефективності, оскільки не враховують реальний стан будівлі, розташування людей та динаміку поширення небезпеки.

У зв’язку з цим актуальним є створення інтелектуальних інформаційних систем, здатних автоматично аналізувати ситуацію, обробляти дані з різноманітних датчиків та формувати оптимальні маршрути. Таким чином, необхідність підвищення рівня безпеки людей у будівлях шляхом автоматизації процесів оповіщення та впровадження інтелектуальних підходів до аналізу надзвичайних ситуацій є дуже важливим питанням, що робить дану тему актуальною.

Мета роботи - розробка інформаційної системи автоматизованого оповіщення та управління евакуацією людей із будівель, яка забезпечує оперативне інформування користувачів та визначення безпечних маршрутів виходу з урахуванням поточної ситуації.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- проаналізувати предметну область систем оповіщення та евакуації;
- дослідити існуючі рішення та визначити їхні недоліки;
- сформулювати вимоги до інформаційної системи;
- спроєктувати архітектуру системи та модель даних;

- розробити алгоритм визначення оптимального маршруту евакуації;
- реалізувати програмну частину системи;
- провести тестування та оцінити ефективність розробленого рішення.

Об'єкт дослідження - процеси оповіщення та евакуації людей у будівлях під час надзвичайних ситуацій.

Предмет дослідження - методи, моделі та програмні засоби створення інформаційної системи автоматизованого управління евакуацією.

Методи дослідження. Включають аналіз і узагальнення наукових джерел, методи системного аналізу, моделювання процесів, алгоритмічні методи пошуку оптимальних шляхів (зокрема алгоритми графів), а також методи програмної реалізації та тестування програмного забезпечення.

Наукова новизна одержаних результатів. У ході дослідження запропоновано підхід до побудови системи управління евакуацією, який полягає в інтеграції даних з датчиків (диму, температури, руху) в реальному часі з алгоритмами пошуку на графах, що дозволяє динамічно перераховувати та оптимізувати маршрути евакуації залежно від зміни умов надзвичайної ситуації.

Практична значущість одержаних результатів. Полягає у можливості використання розробленої системи для підвищення ефективності евакуації людей у будівлях різного призначення, що дозволяє зменшити ризики для життя та здоров'я у надзвичайних ситуаціях.

Апробація результатів та публікації. Основні положення і результати роботи доповідались на науково-практичних конференціях «МІЖДИСЦИПЛІНАРНІ НАУКОВІ ДОСЛІДЖЕННЯ ТА ПЕРСПЕКТИВИ ЇХ РОЗВИТКУ», та «ПРОБЛЕМИ ТА ПЕРСПЕКТИВИ РЕАЛІЗАЦІЇ ТА ВПРОВАДЖЕННЯ МІЖДИСЦИПЛІНАРНИХ НАУКОВИХ ДОСЯГНЕНЬ», з поданням тез на теми «ІНФОРМАЦІЙНА СИСТЕМА АВТОМАТИЗОВАНОГО ОПОВІЩЕННЯ ТА УПРАВЛІННЯ ЕВАКУАЦІЄЮ ЛЮДЕЙ ІЗ БУДІВЕЛЬ» та «ПРОГРАМНІ СИСТЕМИ ТА АРХІТЕКТУРА: СУЧАСНІ ПІДХОДИ ДО ПРОЄКТУВАННЯ ТА РОЗРОБКИ».

РОЗДІЛ 1. АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Поняття систем оповіщення та евакуації

Системи оповіщення та управління евакуацією є невід'ємною складовою комплексної безпеки будівель і споруд, особливо тих, що передбачають масове перебування людей. Їх основною функцією є своєчасне інформування людей про виникнення небезпечної ситуації та організація безпечного і швидкого покидання зони ризику.

Під системою оповіщення розуміють сукупність технічних засобів і програмних рішень, призначених для передачі сигналів тривоги та інформаційних повідомлень у разі виникнення надзвичайної ситуації. Такі системи можуть використовувати різні канали комунікації, зокрема звукові сигнали, голосові повідомлення, світлові індикатори, текстові повідомлення на екранах або мобільних пристроях.

Система управління евакуацією, у свою чергу, є більш складною підсистемою, яка не лише повідомляє про небезпеку, але й координує дії людей під час евакуації. Вона включає алгоритми визначення оптимальних маршрутів, аналіз ситуації в реальному часі та можливість адаптації до змін умов, таких як поширення пожежі або задимлення.

Сучасні системи оповіщення та евакуації зазвичай інтегруються в загальну інфраструктуру безпеки будівлі та взаємодіють з іншими підсистемами, такими як:

- системи пожежної сигналізації;
- системи відеоспостереження;
- системи контролю доступу;
- датчики диму, температури та руху;
- автоматизовані системи управління будівлею (BMS).

Залежно від рівня складності та функціональних можливостей, системи оповіщення та евакуації можна умовно поділити на декілька типів:

1. **Локальні системи оповіщення** — забезпечують передачу сигналу тривоги в межах окремого приміщення або будівлі, зазвичай у вигляді звукової сирени або світлового сигналу.
2. **Централізовані системи** — передбачають наявність єдиного центру управління, який здійснює контроль за станом об’єкта та активує оповіщення у разі необхідності.
3. **Автоматизовані системи** — здатні самостійно реагувати на сигнали від датчиків і запускати процедури оповіщення без участі людини.
4. **Інтелектуальні системи** — використовують алгоритми аналізу даних та можуть адаптувати маршрути евакуації залежно від розвитку ситуації.

Основними компонентами систем оповіщення та евакуації є:

- **датчики** (джерела інформації про небезпеку);
- **контролери** (пристрої обробки сигналів);
- **засоби оповіщення** (динаміки, дисплеї, світлові табло);
- **програмне забезпечення** (для аналізу та управління процесами);
- **інтерфейс користувача** (для операторів або диспетчерів).

Особливу роль відіграє швидкість реагування системи, оскільки навіть незначна затримка може призвести до серйозних наслідків. Ефективна система повинна забезпечувати мінімальний час між виявленням небезпеки та передачею сигналу оповіщення.

Крім того, важливим аспектом є зрозумілість та доступність інформації для користувачів. Оповіщення має бути чітким, однозначним і таким, що не викликає паніки. У сучасних системах дедалі частіше застосовуються голосові повідомлення з інструкціями щодо дій, а також візуальні підказки у вигляді схем евакуації.

Таким чином, системи оповіщення та управління евакуацією є складними інформаційно-технічними комплексами, які поєднують апаратні та програмні компоненти з метою забезпечення безпеки людей. Їх розвиток спрямований на підвищення рівня автоматизації, адаптивності та інтеграції з іншими системами

«розумної будівлі».

1.2 Аналіз існуючих рішень

На сучасному етапі розвитку технологій існує значна кількість систем оповіщення та управління евакуацією, які застосовуються у житлових, комерційних та промислових будівлях. Вони постійно вдосконалюються, переходячи від простих сигналізацій до інтелектуальних систем, інтегрованих у концепцію «розумних будівель» (Smart Building).

Найбільш поширеними є **традиційні системи пожежної сигналізації**, які виконують функцію виявлення небезпеки та оповіщення людей. Такі системи базуються на використанні датчиків диму, температури або ручних кнопок тривоги. При спрацюванні вони активують звукові та світлові сигнали, що інформують людей про необхідність евакуації. Центральним елементом є панель управління, яка обробляє сигнали та запускає відповідні сценарії реагування .

Основною перевагою традиційних систем є їхня простота, надійність і відносно низька вартість. Вони широко застосовуються у невеликих будівлях, де достатньо базового рівня безпеки. Однак такі рішення мають суттєві недоліки, зокрема:

- відсутність детальної інформації про характер небезпеки;
- неможливість адаптації до змін ситуації;
- відсутність індивідуалізованих маршрутів евакуації;
- високий ризик паніки через відсутність чітких інструкцій.

Наступним етапом розвитку стали **системи голосового оповіщення (Voice Evacuation Systems)**. На відміну від звичайних сигналів, вони використовують гучномовці для передачі голосових інструкцій, які пояснюють людям, як діяти у надзвичайній ситуації. Наприклад, система може повідомити точне місце виникнення пожежі та вказати безпечний вихід. Це значно знижує рівень паніки та підвищує ефективність евакуації .

Перевагами таких систем є:

- чіткість і зрозумілість повідомлень;
- можливість зонального оповіщення;
- зменшення часу евакуації;
- інтеграція з системами пожежної сигналізації.

Разом з тим, навіть голосові системи мають обмеження, оскільки вони не завжди враховують реальний стан будівлі в момент небезпеки та не здатні динамічно змінювати маршрути евакуації.

У сучасних умовах активно розвиваються **інтелектуальні системи евакуації на основі IoT (Internet of Things)**. Вони використовують мережі датчиків для збору інформації про стан середовища (дим, температура, заповненість приміщень) та аналізують ці дані у реальному часі. На основі отриманої інформації система може визначати найбезпечніші маршрути евакуації та коригувати їх залежно від розвитку ситуації .

Такі системи мають низку суттєвих переваг:

- адаптивність до змін середовища;
- можливість побудови динамічних маршрутів;
- інтеграція з відеоспостереженням і аналітикою;
- підвищення загальної ефективності управління евакуацією.

Однак вони також характеризуються складністю реалізації, високою вартістю та потребою у надійній інфраструктурі передачі даних.

Окрему категорію становлять **системи моделювання та симуляції евакуації**, які використовуються для прогнозування поведінки людей у надзвичайних ситуаціях. Такі системи дозволяють оцінити час евакуації, визначити вузькі місця у будівлі та оптимізувати планування евакуаційних шляхів. Вони базуються на математичних моделях руху людей та алгоритмах оптимізації .

Крім того, сучасні дослідження спрямовані на використання **штучного інтелекту та машинного навчання** для прогнозування розвитку надзвичайних ситуацій та автоматичного прийняття рішень. Наприклад, системи можуть

аналізувати дані з датчиків і передбачати поширення пожежі, що дозволяє заздалегідь визначити найбезпечніші маршрути евакуації .

Незважаючи на значний розвиток технологій, існуючі рішення мають ряд спільних проблем:

- недостатня адаптивність у реальному часі;
- залежність від людського фактору;
- складність інтеграції різних підсистем;
- обмежена доступність для невеликих об’єктів;
- висока вартість впровадження інтелектуальних систем.

Таким чином, аналіз існуючих рішень показує, що сучасні системи оповіщення та евакуації поступово переходять від простих сигналізацій до інтелектуальних комплексів, здатних аналізувати ситуацію в реальному часі. Проте існує потреба у створенні більш універсальних, доступних і адаптивних інформаційних систем, що і обумовлює доцільність розробки власного програмного рішення у межах даної курсової роботи.

1.3 Нормативні вимоги та стандарти

Забезпечення ефективної роботи систем оповіщення та управління евакуацією людей із будівель регламентується низкою нормативно-правових актів, стандартів і будівельних норм. Дотримання цих вимог є обов’язковим при проєктуванні, впровадженні та експлуатації відповідних систем, оскільки вони безпосередньо впливають на рівень безпеки людей у надзвичайних ситуаціях.

В Україні основними нормативними документами у сфері пожежної безпеки є державні будівельні норми (ДБН), правила пожежної безпеки та стандарти, гармонізовані з міжнародними вимогами. Одним із ключових документів є **ДБН В.1.1-7:2016 «Пожежна безпека об’єктів будівництва»**, який встановлює загальні вимоги до забезпечення пожежної безпеки, включаючи організацію евакуації людей та систем оповіщення.

Згідно з цим документом, кожна будівля повинна бути обладнана системами, що забезпечують:

- своєчасне виявлення пожежі або іншої небезпеки;
- оперативне оповіщення людей;
- безпечні та доступні шляхи евакуації;
- мінімізацію ризику виникнення паніки.

Особлива увага приділяється **часу евакуації**, який повинен бути мінімальним і відповідати встановленим нормативам. Шляхи евакуації повинні бути чітко позначені, освітлені та не мати перешкод, що можуть ускладнити рух людей.

Крім ДБН, важливу роль відіграють **Правила пожежної безпеки в Україні (НАПБ А.01.001)**, які визначають організаційні та технічні заходи щодо запобігання пожежам і забезпечення безпечної евакуації. Вони регламентують:

- порядок дій персоналу у разі пожежі;
- вимоги до систем оповіщення;
- обов’язковість проведення інструктажів і навчань;
- необхідність регулярної перевірки працездатності систем.

У міжнародній практиці широко застосовуються стандарти, розроблені такими організаціями, як **ISO (International Organization for Standardization)** та **NFPA (National Fire Protection Association)**. Зокрема, стандарт **NFPA 72 (National Fire Alarm and Signaling Code)** визначає вимоги до систем пожежної сигналізації та оповіщення, включаючи:

- типи сигналів оповіщення;
- вимоги до гучності та зрозумілості повідомлень;
- принципи зонального оповіщення;
- інтеграцію з іншими системами безпеки.

Також важливим є стандарт **ISO 7240**, який регламентує вимоги до систем пожежної сигналізації, включаючи технічні характеристики обладнання, методи тестування та критерії надійності.

У контексті систем евакуації значну роль відіграють вимоги до **евакуаційного освітлення та навігації**, які забезпечують орієнтацію людей у просторі навіть за умов обмеженої видимості. Вони передбачають використання світлових покажчиків, планів евакуації та аварійного освітлення.

Окрім цього, сучасні стандарти передбачають інтеграцію систем оповіщення з інформаційними технологіями. Це включає:

- використання автоматизованих систем управління;
- передачу повідомлень на мобільні пристрої;
- застосування цифрових платформ для моніторингу ситуації;
- можливість дистанційного керування системою.

Важливим аспектом є також **надійність та відмовостійкість системи**. Нормативні документи вимагають, щоб системи оповіщення мали резервні джерела живлення, дублювання ключових компонентів та можливість роботи в аварійних умовах.

Разом з тим, аналіз нормативної бази показує, що більшість стандартів орієнтовані на традиційні системи оповіщення і не повною мірою враховують можливості сучасних інтелектуальних інформаційних систем. Зокрема, недостатньо регламентовано використання алгоритмів динамічного визначення маршрутів евакуації та аналізу даних у реальному часі.

Таким чином, нормативні вимоги та стандарти створюють основу для забезпечення безпеки людей у будівлях, визначаючи мінімально необхідний рівень функціональності систем оповіщення та евакуації. Водночас розвиток інформаційних технологій відкриває нові можливості для підвищення ефективності таких систем, що потребує подальшого вдосконалення нормативної бази та впровадження інноваційних підходів у практику проєктування.

1.4 Проблеми сучасних систем

Незважаючи на значний розвиток технологій у сфері безпеки будівель, сучасні

системи оповіщення та управління евакуацією мають ряд суттєвих недоліків, які знижують їх ефективність у реальних умовах надзвичайних ситуацій.

Однією з ключових проблем є **обмежена адаптивність систем**. Більшість традиційних рішень функціонує за заздалегідь визначеними сценаріями та не враховує динаміку розвитку подій. Наприклад, при виникненні пожежі система може активувати стандартний сигнал тривоги, не аналізуючи реальний стан будівлі, що призводить до неефективного розподілу потоків людей.

Іншою важливою проблемою є **відсутність інтелектуального аналізу даних**. У багатьох системах інформація з датчиків використовується лише для фіксації факту небезпеки, без подальшої обробки та прогнозування. Це не дозволяє визначити оптимальні маршрути евакуації з урахуванням таких факторів, як рівень задимлення, температура або щільність натовпу.

Суттєвим недоліком є також **залежність від людського фактору**. У ряді випадків прийняття рішень покладається на операторів або персонал, що може призводити до затримок або помилок через стресові умови. Крім того, користувачі часто не мають достатньої інформації про правильні дії під час евакуації, що спричиняє паніку та хаотичний рух.

Ще однією проблемою є **недостатня інформативність оповіщення**. Стандартні звукові сигнали або навіть голосові повідомлення не завжди забезпечують чітке розуміння ситуації. Вони, як правило, не містять персоналізованих рекомендацій і не враховують розташування конкретної людини в будівлі.

Важливим обмеженням є **відсутність динамічного управління маршрутами евакуації**. У більшості випадків маршрути визначаються статично на етапі проєктування будівлі і не змінюються залежно від розвитку небезпеки. Це може призводити до ситуацій, коли люди рухаються в небезпечному напрямку або потрапляють у зони з високим рівнем ризику.

Також слід відзначити проблему **складності інтеграції різних підсистем**. У багатьох будівлях системи пожежної сигналізації, відеоспостереження та управління

будівлею функціонують окремо, без єдиного центру обробки даних. Це ускладнює отримання цілісної картини ситуації та знижує ефективність реагування.

Не менш важливою є **висока вартість впровадження сучасних інтелектуальних систем**, що обмежує їх використання, особливо у невеликих об’єктах. Внаслідок цього більшість будівель продовжує використовувати застарілі рішення з обмеженим функціоналом.

Окремо варто зазначити проблему **надійності та відмовостійкості систем**. У разі відключення електроживлення або пошкодження обладнання система може частково або повністю втратити працездатність, що є критичним у надзвичайних ситуаціях.

Таким чином, сучасні системи оповіщення та евакуації, незважаючи на їх розвиток, мають низку суттєвих недоліків, серед яких ключовими є недостатня адаптивність, відсутність інтелектуального аналізу, залежність від людського фактору та обмежена інтеграція. Це обумовлює необхідність розробки нових інформаційних систем, які будуть більш гнучкими, ефективними та здатними працювати в умовах динамічних загроз.

1.5 Постановка задачі

Проведений аналіз предметної області, існуючих рішень та нормативних вимог показав, що сучасні системи оповіщення та управління евакуацією не повною мірою відповідають вимогам ефективності, адаптивності та інтелектуального управління в умовах динамічних надзвичайних ситуацій. Основними проблемами є відсутність автоматизованого аналізу даних у реальному часі, статичність маршрутів евакуації та недостатній рівень інтеграції між різними підсистемами безпеки.

У зв’язку з цим виникає необхідність розробки інформаційної системи, яка забезпечить:

- автоматичне виявлення небезпечних ситуацій;
- оперативне та зрозуміле оповіщення користувачів;

- визначення безпечних маршрутів евакуації з урахуванням поточних умов;
- можливість адаптації до змін у середовищі;
- інтеграцію з іншими системами будівлі.

Метою розробки є створення інформаційної системи автоматизованого оповіщення та управління евакуацією людей із будівель, яка дозволить підвищити ефективність евакуаційних процесів та мінімізувати ризики для життя і здоров’я людей.

Для досягнення поставленої мети необхідно вирішити такі основні задачі:

- виконати аналіз предметної області та визначити основні вимоги до системи;
- розробити архітектуру інформаційної системи;
- спроєктувати модель даних та структуру бази даних;
- розробити алгоритм визначення оптимального маршруту евакуації з урахуванням небезпечних факторів;
- реалізувати програмну частину системи (серверну та клієнтську);
- забезпечити взаємодію з датчиками або їх імітацію;
- реалізувати інтерфейс користувача для візуалізації процесу евакуації;
- провести тестування системи та оцінити її ефективність.

У межах даної роботи передбачається створення прототипу інформаційної системи, який демонструє основні функціональні можливості, зокрема:

- генерацію сигналу тривоги;
- відображення карти будівлі;
- побудову маршруту евакуації;
- реагування на зміну умов (наприклад, блокування проходу).

Об’єктом дослідження є процес евакуації людей із будівель у надзвичайних ситуаціях.

Предметом дослідження є методи, алгоритми та програмні засоби побудови інформаційної системи автоматизованого оповіщення та управління евакуацією.

Очікуваним результатом роботи є програмно реалізована система, яка

забезпечує підвищення ефективності евакуації за рахунок автоматизації прийняття рішень та використання алгоритмів оптимізації маршрутів.

Таким чином, постановка задачі визначає основні напрями дослідження та розробки, що будуть розглянуті у наступних розділах курсової роботи.

РОЗДІЛ 2. ПРОЄКТУВАННЯ СИСТЕМИ

2.1 Архітектура системи

Архітектура інформаційної системи автоматизованого оповіщення та управління евакуацією є ключовим етапом проєктування, оскільки вона визначає структуру системи, взаємодію її компонентів та принципи обробки даних. Від правильно обраної архітектури залежить ефективність, масштабованість, надійність та можливість подальшого розвитку системи.

У межах даної роботи пропонується використання **багаторівневої клієнт-серверної архітектури**, яка забезпечує розподіл функцій між окремими компонентами системи та дозволяє ефективно обробляти дані в реальному часі.

Основними рівнями системи є:

1. Рівень збору даних (датчики)

Цей рівень включає пристрої, які здійснюють моніторинг стану середовища в будівлі. До них належать:

- датчики диму;
- температурні датчики;
- датчики руху;
- ручні кнопки тривоги.

Дані з датчиків передаються до центрального сервера для подальшої обробки. У рамках програмної реалізації допускається використання імітації роботи датчиків.

2. Серверний рівень (Backend)

Серверна частина системи є основним елементом обробки інформації. Вона виконує такі функції:

- прийом даних від датчиків;
- аналіз стану системи;
- визначення наявності небезпеки;
- запуск сценаріїв оповіщення;

- обчислення оптимальних маршрутів евакуації;
- взаємодія з базою даних.

На цьому рівні реалізується основна логіка системи, включаючи алгоритми пошуку найкоротшого або найбезпечнішого шляху (наприклад, алгоритм Дейкстри або A*).

3. Рівень зберігання даних (Database)

База даних використовується для зберігання інформації, необхідної для функціонування системи, зокрема:

- дані про будівлю (плани, приміщення, виходи);
- інформація про датчики;
- історія подій та тривог;
- маршрути евакуації.

Використання реляційної бази даних (наприклад, PostgreSQL або SQLite) дозволяє забезпечити структурованість та швидкий доступ до даних.

4. Клієнтський рівень (Frontend)

Клієнтська частина забезпечує взаємодію користувача із системою. Вона може бути реалізована у вигляді веб-інтерфейсу та включає:

- відображення карти будівлі;
- візуалізацію маршруту евакуації;
- індикацію небезпечних зон;
- сповіщення про тривогу;
- інтерфейс для оператора.

Користувач отримує інформацію у зрозумілому вигляді, що дозволяє швидко приймати рішення у надзвичайній ситуації.

5. Рівень оповіщення (Notification Layer)

Цей рівень відповідає за передачу сигналів користувачам. Він може включати:

- звукові сигнали;
- текстові повідомлення;

- push-сповіщення;
- візуальні підказки на екрані.

Загальна схема архітектури

Узагальнено архітектуру системи можна представити у вигляді наступного ланцюга:

Датчики → Сервер обробки → База даних → Клієнтський інтерфейс → Користувач

Переваги обраної архітектури

Обрана клієнт-серверна архітектура має ряд переваг:

- **масштабованість** — можливість додавання нових датчиків і користувачів;
- **гнучкість** — зміна логіки системи без модифікації клієнтської частини;
- **надійність** — централізований контроль і обробка даних;
- **розширюваність** — можливість інтеграції з іншими системами (відеоспостереження, Smart Building);
- **ефективність** — обробка даних у реальному часі.

Таким чином, обрана архітектура інформаційної системи забезпечує ефективну взаємодію між її компонентами та дозволяє реалізувати ключові функції автоматизованого оповіщення та управління евакуацією. Використання багаторівневої клієнт-серверної моделі створює основу для подальшої реалізації програмного забезпечення та впровадження інтелектуальних алгоритмів обробки даних.

2.2 Функціональна модель системи

Функціональна модель інформаційної системи визначає перелік основних функцій, які система повинна виконувати, а також описує взаємодію між користувачами та системою. Для наочного представлення функціональних можливостей доцільно використовувати **діаграму варіантів використання (Use Case)**, яка дозволяє відобразити сценарії роботи системи.

У межах розроблюваної інформаційної системи автоматизованого оповіщення та управління евакуацією можна виділити кілька основних **акторів**:

- **Користувач (людина в будівлі)** — отримує повідомлення про небезпеку та маршрути евакуації;
- **Оператор (диспетчер)** — контролює роботу системи та може вручну запускати оповіщення;
- **Система датчиків** — автоматично передає інформацію про стан середовища;
- **Сервер системи** — виконує обробку даних та керує логікою роботи.

Основні функції системи

Функціональна модель включає такі ключові варіанти використання:

1. Моніторинг стану середовища

Система безперервно отримує дані від датчиків (дим, температура, рух) та аналізує їх для виявлення потенційної небезпеки.

2. Виявлення надзвичайної ситуації

На основі отриманих даних система визначає наявність загрози (наприклад, пожежі) та активує відповідний сценарій реагування.

3. Генерація сигналу тривоги

Після виявлення небезпеки система формує сигнал тривоги та передає його через канали оповіщення.

4. Оповіщення користувачів

Користувачі отримують повідомлення у вигляді:

- звукового сигналу;
- текстового повідомлення;
- візуального відображення на інтерфейсі.

5. Побудова маршруту евакуації

Система визначає оптимальний маршрут виходу з будівлі з урахуванням:

- розташування користувача;
- наявності небезпечних зон;

- доступності евакуаційних виходів.

6. Відображення маршруту

На інтерфейсі користувача відображається карта будівлі з позначеним маршрутом евакуації та небезпечними зонами.

7. Оновлення маршруту в реальному часі

У разі зміни умов (наприклад, поширення пожежі) система автоматично перебудовує маршрут.

8. Ручне управління системою

Оператор має можливість:

- запускати сигнал тривоги;
- переглядати стан системи;
- контролювати процес евакуації.

9. Збереження даних та подій

Система фіксує всі події (тривоги, маршрути, дії користувачів) у базі даних для подальшого аналізу.

Опис взаємодії акторів із системою

Користувач взаємодіє із системою переважно пасивно — отримує повідомлення та інструкції щодо евакуації. Його основна задача — слідувати запропонованому маршруту.

Оператор має активну роль і здійснює контроль за системою. Він може втручатися у роботу системи, змінювати сценарії або запускати оповіщення вручну.

Система датчиків працює в автоматичному режимі та є джерелом первинної інформації. Вона забезпечує своєчасне виявлення небезпечних ситуацій.

Сервер системи виступає центральним елементом, який об’єднує всі компоненти, обробляє дані та реалізує логіку роботи.

Узагальнена структура функціональної моделі

Функціональну модель системи можна описати як послідовність дій:

Отримання даних → Аналіз ситуації → Виявлення небезпеки →

Оповіщення → Побудова маршруту → Відображення маршруту → Контроль евакуації

Таким чином, функціональна модель системи охоплює всі ключові процеси, необхідні для забезпечення ефективного оповіщення та управління евакуацією. Використання підходу Use Case дозволяє чітко визначити ролі акторів та функціональні можливості системи, що є основою для подальшого проєктування алгоритмів та програмної реалізації.

2.3 Моделювання процесів (діаграми)

Моделювання процесів є важливим етапом проєктування інформаційної системи, оскільки дозволяє формалізувати логіку її роботи, визначити послідовність дій та взаємодію між компонентами. Для опису процесів у даній роботі використовуються **UML-діаграми**, зокрема діаграма активності та діаграма послідовності.

Діаграма активності системи евакуації

Діаграма активності відображає загальний алгоритм функціонування системи в умовах надзвичайної ситуації. Вона демонструє послідовність дій від моменту виявлення небезпеки до завершення процесу евакуації.

Процес починається зі стану очікування, у якому система постійно отримує дані від датчиків. Після цього виконується перевірка наявності небезпечної ситуації.

У разі відсутності загрози система продовжує моніторинг. Якщо ж небезпека виявлена, відбувається перехід до наступних дій:

- активація сигналу тривоги;
- аналіз поточної ситуації;
- визначення небезпечних зон;
- побудова маршруту евакуації;
- передача повідомлень користувачам.

Далі система відображає маршрут евакуації на інтерфейсі користувача та

супроводжує процес евакуації. У випадку зміни умов (наприклад, блокування проходу або поширення пожежі) виконується повторний аналіз і перебудова маршруту.

Процес завершується після того, як користувач залишає небезпечну зону або система повертається в режим очікування.

Діаграма послідовності взаємодії компонентів

Діаграма послідовності відображає обмін повідомленнями між основними компонентами системи у часі. Вона дозволяє зрозуміти, як саме відбувається взаємодія між датчиками, сервером та користувачем.

Основними елементами взаємодії є:

- датчики;
- сервер;
- база даних;
- клієнтський інтерфейс;
- користувач.

Процес взаємодії відбувається наступним чином:

1. Датчики передають дані про стан середовища на сервер.
2. Сервер аналізує отриману інформацію.
3. У разі виявлення небезпеки сервер звертається до бази даних для отримання інформації про будівлю та маршрути.
4. Виконується обчислення оптимального маршруту евакуації.
5. Сервер передає результати на клієнтський інтерфейс.
6. Користувач отримує повідомлення та бачить маршрут евакуації.
7. У разі зміни умов сервер повторює цикл обробки.

Такий підхід забезпечує чітку послідовність дій та дозволяє мінімізувати затримки у передачі інформації.

Значення моделювання процесів

Використання UML-діаграм дозволяє:

- формалізувати логіку роботи системи;

- виявити можливі помилки ще на етапі проєктування;
- забезпечити зрозумілість структури системи;
- спростити подальшу реалізацію програмного забезпечення.

Таким чином, моделювання процесів дозволяє чітко визначити алгоритм функціонування інформаційної системи та взаємодію її компонентів. Діаграми активності та послідовності є важливими інструментами проєктування, які забезпечують логічну цілісність системи та слугують основою для реалізації програмної частини.

2.4 Проєктування бази даних

База даних є одним із ключових компонентів інформаційної системи автоматизованого оповіщення та управління евакуацією, оскільки вона забезпечує зберігання, обробку та швидкий доступ до інформації, необхідної для функціонування системи. Правильне проєктування структури бази даних дозволяє підвищити ефективність роботи системи, забезпечити її масштабованість та надійність.

Для реалізації системи обрано **реляційну модель бази даних**, яка забезпечує логічну структурованість даних та підтримує зв’язки між сутностями. У якості системи управління базами даних може бути використано PostgreSQL або SQLite.

Основні сутності бази даних

У процесі аналізу функціональних вимог було визначено наступні основні сутності:

1. Users (Користувачі)

Зберігає інформацію про користувачів системи.

Основні поля:

- id (первинний ключ);
- name (ім’я користувача);
- role (роль: користувач, оператор);
- location (поточне місцезнаходження у будівлі).

2. Buildings (Будівлі)

Містить інформацію про будівлі, у яких функціонує система.

Основні поля:

- id;
- name;
- address;
- floors (кількість поверхів).

3. Rooms (Приміщення)

Описує структуру будівлі на рівні кімнат.

Основні поля:

- id;
- building_id (зовнішній ключ);
- name;
- floor;
- type (офіс, коридор, вихід тощо).

4. Sensors (Датчики)

Зберігає інформацію про датчики, встановлені у приміщеннях.

Основні поля:

- id;
- room_id (зовнішній ключ);
- type (дим, температура, рух);
- status (нормальний, тривога).

5. Alerts (Події / Тривоги)

Фіксує всі випадки виникнення небезпечних ситуацій.

Основні поля:

- id;
- sensor_id (зовнішній ключ);
- timestamp;

- alert_type;
- status.

6. EvacuationRoutes (Маршрути евакуації)

Містить інформацію про можливі маршрути евакуації.

Основні поля:

- id;
- start_room_id;
- end_room_id;
- path (опис маршруту);
- is_safe (позначка безпеки).

Зв’язки між сутностями

У базі даних реалізовано наступні зв’язки:

- **Buildings → Rooms** (один до багатьох);
- **Rooms → Sensors** (один до багатьох);
- **Sensors → Alerts** (один до багатьох);
- **Rooms → EvacuationRoutes** (зв’язок через маршрути);

Таке проєктування дозволяє ефективно організувати структуру даних і забезпечити швидкий доступ до інформації під час обробки запитів.

ER-діаграма бази даних

Для візуалізації структури бази даних використовується ER-діаграма, яка відображає сутності, їх атрибути та зв’язки між ними.

Нормалізація бази даних

Проєктована база даних відповідає основним вимогам нормалізації:

- відсутність дублювання даних;
- залежність неключових атрибутів від первинного ключа;
- розділення сутностей за логічними групами.

Це дозволяє забезпечити цілісність даних та уникнути аномалій при їх оновленні.

Приклад структури таблиці (SQL)

Для прикладу наведемо створення таблиці користувачів:

```
CREATE TABLE Users (
   id SERIAL PRIMARY KEY,
   name VARCHAR(100),
   role VARCHAR(50),
   location VARCHAR(100)
);
```

Таким чином, у процесі проєктування бази даних було визначено основні сутності, їх атрибути та зв’язки, що дозволяє забезпечити ефективне зберігання та обробку інформації. Реляційна модель бази даних створює надійну основу для реалізації функціональних можливостей системи та її подальшого розвитку.

2.5 Алгоритм евакуації

Одним із найважливіших елементів інформаційної системи автоматизованого оповіщення та управління евакуацією є алгоритм визначення оптимального маршруту виходу з будівлі. Саме від ефективності цього алгоритму залежить швидкість та безпечність евакуації людей у надзвичайних ситуаціях.

Загальна постановка задачі

Задача евакуації може бути сформульована як задача пошуку найкращого шляху у графі, де:

- вершини (nodes) — це приміщення або точки будівлі;
- ребра (edges) — можливі переходи між приміщеннями;
- ваги ребер — відстань або рівень небезпеки.

Основною метою є знаходження маршруту від поточного місцезнаходження користувача до найближчого безпечного виходу з урахуванням:

- довжини маршруту;
- наявності небезпечних зон;

- доступності проходів.

Вибір алгоритму

Для розв’язання задачі використовується алгоритм пошуку найкоротшого шляху. Найбільш доцільними є:

- алгоритм Дейкстри;
- алгоритм A* (A-star).

У межах даної роботи обрано **алгоритм Дейкстри**, оскільки він:

- простий у реалізації;
- гарантує знаходження оптимального маршруту;
- добре підходить для статичних графів.

Модифікація алгоритму для евакуації

Для врахування небезпеки стандартний алгоритм Дейкстри модифікується. Вага ребра визначається не лише довжиною, а й рівнем ризику:

вага = відстань + коефіцієнт небезпеки

Де:

- відстань — довжина шляху;
- коефіцієнт небезпеки — значення, що залежить від:
  - задимлення;
  - температури;
  - наявності перешкод.

Небезпечні ділянки можуть отримувати дуже високі ваги або повністю виключатися з розрахунку.

Етапи роботи алгоритму

Алгоритм евакуації працює за наступною схемою:

1. Отримання даних про будівлю (граф приміщень).
2. Визначення поточного місцезнаходження користувача.
3. Виявлення небезпечних зон.
4. Формування ваг графа з урахуванням ризиків.

5. Запуск алгоритму пошуку шляху.
6. Визначення оптимального маршруту до виходу.
7. Передача маршруту користувачу.
8. Перерахунок маршруту у разі зміни умов.

Псевдокод алгоритму

```
function findSafePath(graph, start, exits):
    initialize distances to all nodes as infinity
    distances[start] = 0
    create priority queue
    add start to queue

    while queue is not empty:
        current = node with smallest distance

        if current is in exits:
            return path to current

        for each neighbor of current:
            weight = distance + danger_factor
            new_distance = distances[current] + weight

            if new_distance < distances[neighbor]:
                distances[neighbor] = new_distance
                update queue

    return no path
```

Динамічна адаптація маршруту

Важливою особливістю алгоритму є можливість **динамічного оновлення**

маршруту. У разі зміни ситуації (наприклад, поширення пожежі або блокування проходу):

- оновлюються дані про небезпеку;
- перераховуються ваги графа;
- алгоритм запускається повторно;
- користувач отримує новий маршрут.

Переваги запропонованого алгоритму

- забезпечує знаходження оптимального маршруту;
- враховує небезпечні фактори;
- адаптується до змін у реальному часі;
- може бути масштабований для великих будівель;
- легко інтегрується у програмну систему.

Таким чином, запропонований алгоритм евакуації базується на використанні графових методів пошуку оптимального шляху та враховує особливості надзвичайних ситуацій. Його застосування дозволяє підвищити ефективність евакуації, зменшити ризики для людей та забезпечити адаптивність системи до змін у середовищі.

2.6 Інтерфейс користувача

Інтерфейс користувача є важливою складовою інформаційної системи автоматизованого оповіщення та управління евакуацією, оскільки саме через нього здійснюється взаємодія між системою та користувачами. Ефективність інтерфейсу безпосередньо впливає на швидкість реагування людей у надзвичайних ситуаціях, тому він повинен бути максимально простим, зрозумілим та інтуїтивно доступним.

Основні вимоги до інтерфейсу

При проєктуванні інтерфейсу користувача були враховані наступні вимоги:

- **простота та зрозумілість** — мінімум складних елементів;
- **швидкість сприйняття інформації** — важлива інформація повинна бути доступною одразу;

- **наочність** — використання графічних елементів (карти, кольори, позначки);
- **адаптивність** — можливість роботи на різних пристроях;
- **надійність** — коректне відображення даних навіть у критичних умовах.

Типи користувачів інтерфейсу

У системі передбачено два основних типи користувачів:

1. Користувач (людина в будівлі)

Отримує інформацію про небезпеку та маршрут евакуації. Його інтерфейс максимально спрощений.

Основні функції:

- отримання сигналу тривоги;
- перегляд маршруту евакуації;
- орієнтація у просторі;
- отримання оновлень маршруту.

2. Оператор (диспетчер)

Має розширений доступ до системи та виконує контроль і управління.

Основні функції:

- перегляд стану системи;
- моніторинг датчиків;
- запуск сигналу тривоги;
- контроль евакуації;
- аналіз подій.

Структура інтерфейсу користувача

Інтерфейс системи може бути реалізований у вигляді веб-додатку та включає такі основні екрани:

1. Головний екран

Містить загальну інформацію про стан системи:

- статус (нормальний / тривога);
- коротке повідомлення;

- кнопки управління (для оператора).

2. Екран тривоги

Активується у разі виникнення небезпечної ситуації та відображає:

- повідомлення про небезпеку;
- тип загрози (пожежа, задимлення тощо);
- рекомендації щодо дій.

3. Карта будівлі

Є центральним елементом інтерфейсу та містить:

- план приміщень;
- розташування користувача;
- позначення небезпечних зон (червоний колір);
- безпечні маршрути (зелений колір);
- евакуаційні виходи.

4. Маршрут евакуації

Відображається у вигляді:

- графічної лінії на карті;
- покрокових інструкцій (за потреби).

5. Панель оператора

Містить розширений функціонал:

- список датчиків;
- журнал подій;
- управління системою;
- статистичні дані.

Візуальне оформлення інтерфейсу

Для забезпечення швидкого сприйняття інформації використовуються стандартні кольорові індикатори:

- **зелений** — безпечна зона або маршрут;
- **жовтий** — потенційна небезпека;

- **червоний** — зона високого ризику;

Також використовуються піктограми та символи, що полегшують орієнтацію користувача.

Особливості реалізації інтерфейсу

Інтерфейс може бути реалізований з використанням веб-технологій:

- HTML, CSS — для структури та стилізації;
- JavaScript — для динамічного оновлення даних;
- інтеграція з сервером через API.

Оновлення інформації відбувається в реальному часі, що дозволяє користувачам отримувати актуальні дані про ситуацію.

Таким чином, інтерфейс користувача розроблюваної системи орієнтований на забезпечення швидкого та ефективного сприйняття інформації у надзвичайних ситуаціях. Простота, наочність та адаптивність інтерфейсу дозволяють підвищити ефективність евакуації та зменшити ризик виникнення паніки серед користувачів.

РОЗДІЛ 3. ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Вибір технологій

Програмна реалізація інформаційної системи автоматизованого оповіщення та управління евакуацією повинна забезпечувати стабільну роботу ядра обробки подій, можливість зберігання оперативних даних, побудову маршруту евакуації у реальному часі та наочне відображення результатів для користувача й оператора. Саме тому на етапі реалізації було важливо обрати такий стек технологій, який поєднує відносну простоту, хорошу документаційну підтримку, відкритість, гнучкість до змін і достатню продуктивність для розв’язання задачі курсової роботи.

У ролі базової мови програмування доцільно застосувати Python. Ця мова широко використовується для створення інформаційних систем, вебсервісів, прототипів аналітичних модулів і програмних засобів підтримки прийняття рішень. Її перевагами є зрозумілий синтаксис, велика кількість бібліотек, висока швидкість розробки та зручність інтеграції з алгоритмами пошуку шляху, засобами обробки даних і реляційними базами даних. Для навчального проєкту Python є особливо доречним, оскільки дозволяє концентрувати увагу не на низькорівневих деталях реалізації, а на логіці системи, моделюванні сценаріїв і якості архітектурних рішень.

У якості серверного фреймворку обрано Flask. Він дозволяє реалізувати структуру застосунку у вигляді набору URL-маршрутів, модулів логіки, шаблонів представлення і REST-подібних API-методів. Flask не нав’язує складної структури, тому підходить для курсового проєкту, де потрібно прозоро показати логіку реалізації. Крім того, документація Flask серії 3.1.x підкреслює його придатність для локальної розробки, тестування та побудови розширюваних вебзастосунків, а механізм створення застосунку через фабрику дає змогу зручно масштабувати проєкт.

Для зберігання даних використано SQLite. На відміну від великих серверних СУБД, SQLite працює як вбудована бібліотека та не потребує окремого сервера, що

спрощує розгортання прототипу. Водночас вона підтримує транзакції, індексацію, зовнішні ключі, тригери, обмеження цілісності та SQL-запити, достатні для реалізації інформаційної системи евакуації. Офіційна документація SQLite описує її як малу, швидку, самодостатню та надійну СУБД, що робить її доцільним вибором для демонстраційних і навчальних систем, а також для вбудованих рішень.

Клієнтський рівень реалізується за допомогою HTML, CSS та JavaScript. HTML формує структуру сторінок, CSS забезпечує стилізацію карти будівлі, панелі датчиків та блоку оповіщення, а JavaScript відповідає за інтерактивну поведінку, асинхронні запити до серверного API та оновлення результатів без повного перезавантаження сторінки. Такий стек добре підходить для побудови простого, але наочного вебінтерфейсу, який може використовуватися як оператором, так і кінцевим користувачем у межах локальної мережі будівлі.

Для реалізації алгоритму евакуації використовується вбудований алгоритмічний модуль на Python. Це рішення дає змогу безпосередньо інтегрувати модифікований алгоритм Дейкстри в серверну логіку, оперативно отримувати стан датчиків, враховувати коефіцієнти ризику, блокування ділянок шляху та на льоту формувати новий маршрут. На відміну від винесення алгоритму в окремий сервіс, такий підхід спрощує синхронізацію бізнес-логіки, зменшує кількість точок відмови та робить структуру демонстраційного проєкту компактнішою.

Окремо важливо відзначити інструменти тестування та налагодження. Для базової перевірки працездатності системи застосовуються модульні перевірки Python, ручне сценарне тестування вебінтерфейсу, аналіз журналу подій та контроль результатів API-запитів. Це дозволяє перевірити правильність маршрутизації, поведінку системи у разі спрацювання датчиків, коректність запису подій у базу даних і відображення станів на інтерфейсі користувача.

Таким чином, сформований технологічний стек Python - Flask - SQLite - HTML/CSS/JavaScript є достатньо гнучким, прозорим і методично виправданим для реалізації програмного прототипу інформаційної системи автоматизованого

оповіщення та управління евакуацією. Його переваги полягають у доступності, простоті впровадження, зрозумілій архітектурі та можливості подальшого розвитку системи до більш складної багаторівневої платформи.

3.2 Реалізація системи

Програмна реалізація системи побудована як інтегрований вебзастосунок, у якому серверна частина відповідає за обробку сигналів, прийняття рішень, роботу з даними та побудову маршрутів, а клієнтська - за відображення стану системи та взаємодію з користувачем. Такий підхід дозволяє моделювати реальний режим роботи інформаційної системи оповіщення, коли диспетчер або оператор отримує консолідовану картину ситуації, а користувач бачить лише ту інформацію, яка необхідна для швидкої евакуації.

Структура реалізованого проєкту включає декілька функціональних модулів. Перший модуль - серверний застосунок app.py, що містить логіку ініціалізації, обробки HTTP-запитів, побудови графа приміщень, запуску алгоритму маршрутизації та формування JSON-відповідей для клієнта. Другий модуль - база даних SQLite, у якій зберігаються таблиці users, rooms, sensors, routes та alerts. Третій модуль - шаблони вебінтерфейсу, що формують сторінки користувача та оператора. Четвертий модуль - статичні файли, які забезпечують оформлення інтерфейсу та виконання асинхронних запитів до API. П’ятий модуль - сценарії тестування та допоміжна документація для запуску проєкту.

З погляду життєвого циклу даних робота системи організована у такій послідовності. Після запуску застосунку виконується перевірка наявності локальної бази даних. Якщо база відсутня, відпрацьовує функція ініціалізації, яка створює таблиці, заповнює довідники приміщень, датчиків і маршрутів та формує стартове середовище. Далі користувач або оператор відкриває вебінтерфейс. На головній сторінці завантажується карта приміщень та поточний стан датчиків. У разі імітації або реального спрацювання датчика формується подія тривоги, що записується до

журналу alerts та одночасно змінює параметри відповідного вузла у графі будівлі.

API рівень реалізований через набір окремих маршрутів. Метод /api/state повертає поточний стан системи у зручному для фронтенду форматі, включаючи список кімнат, параметри датчиків та журнал останніх подій. Метод /api/trigger дозволяє створити подію тривоги, змінюючи статус датчика й рівень ризику. Метод /api/reset повертає систему до штатного стану. Метод /api/route/<start_room_id> викликає алгоритм маршрутизації та повертає оптимальний шлях від вибраного приміщення до найближчого безпечного виходу. Це дозволяє забезпечити чітке розмежування між логікою представлення та логікою предметної області.

Особливу роль у реалізації відіграє механізм побудови графа будівлі. На основі записів таблиці routes формується орієнтований або неорієнтований граф, у якому вузлами є приміщення, а ребрами - дозволені переходи між ними. Перед запуском алгоритму система обчислює вагу кожного переходу. У найпростішому випадку вона дорівнює фізичній відстані, але в реальному режимі до неї додається коефіцієнт ризику, який залежить від стану датчиків у приміщенні або суміжних зонах. Це забезпечує перехід від звичайного пошуку найкоротшого маршруту до пошуку найменш ризикованого маршруту.

Реалізація також враховує вимогу журналювання подій. Кожне спрацювання датчика, зміна статусу системи, активація режиму тривоги та побудова маршруту можуть бути відображені у журналі для подальшого аналізу. Такий підхід важливий не лише для демонстрації роботи системи, а й для післяаварійного аналізу, аудиту інцидентів, виявлення вузьких місць у плануванні евакуації та покращення правил реагування.

Застосунок має два основні режими використання. Перший - користувацький, де обирається поточне приміщення і відображається рекомендований маршрут до виходу. Другий - операторський, де відображається повний журнал подій, перелік датчиків, рівні ризику, повідомлення про тривоги та загальна картина стану об’єкта. Такий поділ ролей відповідає логіці реальних систем безпеки, у яких кінцевий

користувач не повинен отримувати надмірну технічну інформацію, а оператор, навпаки, має бачити повну аналітичну картину.

З погляду подальшого розвитку запропонована реалізація може бути легко масштабована. До системи можна додати автентифікацію користувачів, інтеграцію з MQTT або іншими брокерами повідомлень для реального обміну даними з IoT-пристроями, підтримку багатоповерхових планів з графічним перемиканням, інтеграцію з картографічним відображенням на SVG або Canvas, а також розширений модуль аналітики для розрахунку часу евакуації, щільності потоків та критичних вузлів у графі будівлі.

Узагальнюючи, реалізація системи побудована за модульним принципом і забезпечує взаємодію між усіма ключовими компонентами предметної області: датчиками, серверною логікою, базою даних, алгоритмом маршрутизації та користувацьким інтерфейсом. Саме така структура робить створений прототип не лише навчальним прикладом, а й основою для більш серйозної прикладної системи автоматизованого оповіщення та управління евакуацією.

Таблиця 3.1 - Основні компоненти реалізованої системи

Компонент	Призначення	Реалізація
Серверний модуль	Обробка запитів, маршрутизація, бізнес-логіка	Python + Flask
База даних	Зберігання приміщень, датчиків, подій і маршрутів	SQLite
Клієнтський інтерфейс	Відображення карти, маршрутів і повідомлень	HTML/CSS/JavaScript
API-рівень	Обмін даними між клієнтом і сервером	JSON over HTTP
Алгоритмічний модуль	Пошук безпечного шляху	Модифікований алгоритм Дейкстри

3.3 Реалізація алгоритму евакуації

Практична реалізація алгоритму евакуації ґрунтується на представленні будівлі у вигляді зваженого графа. Кожне приміщення у системі має унікальний ідентифікатор і виступає вузлом графа. Переходи між приміщеннями, сходами, коридорами та виходами подаються як ребра з певною довжиною. На відміну від класичної навігації у приміщеннях, у задачі евакуації недостатньо враховувати лише геометричну довжину маршруту, оскільки критичним фактором стає рівень небезпеки кожної ділянки шляху. Саме тому алгоритм пошуку був модифікований за рахунок включення коефіцієнта ризику до ваги ребра.

Базова ідея алгоритму полягає у мінімізації інтегральної ваги шляху. Якщо позначити фізичну довжину переходу як d, а рівень ризику суміжної зони як r, то інтегральна вага може бути подана у вигляді функції $W = d + \alpha r$, де α - коефіцієнт чутливості системи до небезпеки. У тестовій реалізації використовується лінійна модель, у якій коефіцієнт ризику множиться на значення 2,5. Це дозволяє системі уникати небезпечних ділянок навіть тоді, коли вони є коротшими за довжиною, але суттєво гіршими за умовами безпеки.

Алгоритм запускається щоразу, коли користувач обирає своє поточне приміщення або коли стан системи змінюється після спрацювання датчика. Спочатку формується словник ризиків по кімнатах. Для цього аналізується максимальний рівень ризику, пов’язаний із датчиками конкретного приміщення. Далі на основі таблиці routes створюється граф суміжності, у якому для кожного переходу обчислюється підсумкова вага. Якщо перехід заблокований або зона оголошена недоступною, відповідне ребро не включається до графа. Після цього система запускає алгоритм Дейкстри від стартового вузла до множини допустимих виходів.

У класичній реалізації алгоритм Дейкстри зберігає для кожної вершини поточну найменшу відстань від старту та використовує пріоритетну чергу для вибору вершини з мінімальною накопиченою вагою. У реалізованій моделі цей підхід збережено, однак оптимізовано під сценарій вибору одного з кількох виходів. Щойно з пріоритетної

черги вилучається вершина, що належить до множини безпечних виходів, побудова маршруту завершується, а відновлення шляху здійснюється через словник попередників. Це зменшує час пошуку у порівнянні з необхідністю повного проходження всього графа.

Практична цінність реалізованого підходу полягає в тому, що він добре пояснюється, легко модифікується та забезпечує стабільну роботу на невеликих і середніх схемах будівель, що характерно для більшості навчальних або локальних сценаріїв. Хоча у великих інфраструктурах доцільно розглядати складніші методи, наприклад A*, генетичні алгоритми або евристичні багатокритеріальні підходи, для курсового проєкту саме Дейкстра є оптимальним компромісом між зрозумілістю, доведеністю коректності та практичною реалізованістю.

Окремої уваги потребує механізм динамічного перерахунку маршруту. Якщо після побудови шляху спрацьовує новий датчик або оператор позначає ділянку як заблоковану, то граф автоматично перебудовується: ваги окремих ребер збільшуються або переходи повністю виключаються. Після цього алгоритм запускається повторно, а користувач отримує новий маршрут. Таким чином, система переходить від моделі одноразового розрахунку до моделі подієвої навігації, що краще відповідає реальним умовам розвитку надзвичайної ситуації.

У реалізації алгоритму враховано й питання відмовостійкості. Якщо безпечний шлях до виходу відсутній, система повертає початковий вузол і нескінченну вагу, що інтерпретується клієнтським рівнем як відсутність безпечного маршруту. У такому разі оператор повинен або коригувати сценарій руху, або переводити користувача до альтернативної зони безпеки. Наявність такої перевірки важлива, оскільки в реальних умовах не завжди існує гарантовано вільний шлях до виходу.

Крім розрахунку маршруту, алгоритмічний модуль може бути розширений блоком оцінювання часу евакуації. Для цього до ваги ребра може додаватися не лише показник ризику, а й коефіцієнт щільності потоку, який залежить від кількості осіб у приміщеннях і на суміжних переходах. Такий розвиток системи дозволив би перейти

від мінімізації інтегрального ризику до багатокритеріальної оптимізації, де враховується і довжина, і небезпека, і потенційні затори. У межах цієї курсової роботи така модель окреслюється як перспектива подальшого розвитку.

Отже, реалізація алгоритму евакуації є центральним функціональним елементом усієї системи. Вона пов’язує дані про стан середовища, графову модель будівлі, логіку прийняття рішень і користувацький інтерфейс. Саме завдяки їй інформаційна система переходить від пасивного оповіщення до активного підтримання безпечної евакуації людей із будівлі.

Фрагмент реалізації алгоритму пошуку безпечного шляху наведено нижче:

```
def dijkstra_safe_path(graph, start, exits):
    queue = [(0.0, start)]
    distances = {start: 0.0}
    previous = {start: None}
    exits_set = set(exits)

    while queue:
        current_distance, current_node = heapq.heappop(queue)
        if current_node in exits_set:
            path = []
            node = current_node
            while node is not None:
                path.append(node)
                node = previous[node]
            return list(reversed(path)), current_distance

        for neighbor, weight in graph.get(current_node, []):
            next_distance = current_distance + weight
            if next_distance < distances.get(neighbor, float("inf")):
```

```
                distances[neighbor] = next_distance
                previous[neighbor] = current_node
                heapq.heappush(queue, (next_distance, neighbor))

    return [start], float("inf")
```

Формула інтегральної ваги маршруту має вигляд: $W = d + \alpha r$, де d - довжина переходу, r - оцінка небезпечності суміжної зони, α - коефіцієнт чутливості до ризику.

3.4 База даних

Практична реалізація бази даних орієнтована на підтримку ключових інформаційних процесів системи: зберігання плану будівлі, опису приміщень, датчиків, переходів між вузлами графа, поточних користувачів, а також журналу тривожних подій. Вибір реляційної моделі для такого завдання є виправданим, оскільки предметна область природно описується через набір сутностей та відношень між ними. Реляційний підхід забезпечує цілісність даних, можливість накладання обмежень і виконання запитів, які добре відповідають потребам оперативного моніторингу.

У структурі реалізованої бази даних центральне місце займає таблиця rooms. Вона зберігає перелік приміщень, їх назви, номер поверху та тип приміщення. Саме ця таблиця визначає вузли графа будівлі. Тип приміщення використовується не лише для візуального відображення, а й для логіки системи, оскільки виходи, сходи, коридори та службові кімнати мають різну роль у побудові маршруту. Таблиця routes описує переходи між приміщеннями, їх довжину та стан доступності. Це дозволяє у будь-який момент змінити топологію руху без переписування алгоритму.

Таблиця sensors виконує роль оперативного джерела даних про небезпечні фактори. У ній для кожного датчика зберігається тип, пов’язане приміщення, поточний стан і рівень ризику. Наявність поля risk_level дозволяє не лише фіксувати двійковий стан норма/тривога, а й моделювати градуйовану небезпеку, що має

принципове значення для адаптивної маршрутизації. Наприклад, задимлення низького рівня і критична температура можуть по-різному впливати на вагу переходу, а значить і на побудову безпечного маршруту.

Таблиця alerts використовується для журналювання подій. У ній зберігається інформація про тип тривоги, текст повідомлення, час створення та, за потреби, приміщення, з яким подія пов’язана. Журнал подій виконує одразу декілька функцій: оперативну, аналітичну та аудиторську. На оперативному рівні він дозволяє оператору бачити хронологію інцидентів. На аналітичному - здійснювати післяаварійний розбір. На аудиторському - підтверджувати коректність роботи системи у тестових сценаріях.

Крім того, у структурі наявна таблиця users. Вона є спрощеним представленням суб’єктів взаємодії із системою і містить ім’я, роль та поточне розташування. Для демонстраційного прототипу цього достатньо, однак у більш розвиненій версії структура може бути доповнена атрибутами авторизації, правами доступу, історією переміщення, контактними каналами для персоналізованого оповіщення та атрибутами належності до окремих категорій користувачів.

Практична реалізація схеми виконується через файл schema.sql. Його використання є доцільним з погляду керованості проєкту: структура бази описується окремо від коду, а отже, може незалежно розвиватися, версіонуватися та перевірятися. У скрипті визначаються первинні та зовнішні ключі, унікальні обмеження для переходів між приміщеннями, базові значення за замовчуванням і режим примусової підтримки зовнішніх ключів. Саме такий підхід відповідає вимогам до якісної розробки інформаційних систем.

З точки зору нормалізації структура відповідає базовим вимогам третьої нормальної форми. Дані про приміщення, датчики, переходи, події та користувачів зберігаються в окремих таблицях, що зменшує дублювання і полегшує оновлення. Наприклад, зміна назви або типу приміщення не потребує модифікації таблиці тривог чи датчиків. Подібним чином один і той самий датчик може породжувати багато

подій, але його характеристики зберігаються лише в одному місці.

Важливою перевагою бази даних є можливість швидкої побудови аналітичних вибірок. Оператор може отримати список активних датчиків тривоги, перелік приміщень із найбільшим накопиченим рівнем ризику, журнал подій за проміжок часу, поточні маршрути до виходів або статистику інцидентів. Отже, база даних виконує не лише функцію сховища, а й функцію оперативного аналітичного центру для системи прийняття рішень.

У перспективі база даних може бути розширена таблицями plans, devices, floors, users_sessions, evacuation_logs, maintenance_tasks та іншими сутностями, що забезпечать глибшу інтеграцію з реальними підсистемами об’єкта. Проте навіть у поточній реалізації вона забезпечує достатній рівень функціональності, логічну цілісність та підтримку основних сценаріїв автоматизованого оповіщення й управління евакуацією.

Таблиця 3.2 - Структура ключових таблиць бази даних

Таблиця	Ключові поля	Призначення
rooms	id, name, floor, room_type	Опис вузлів графа будівлі
routes	from_room_id, to_room_id, distance, blocked	Опис переходів між приміщеннями
sensors	room_id, sensor_type, status, risk_level	Зберігання станів датчиків
alerts	room_id, alert_type, message, created_at	Журнал подій та тривог
users	name, role, current_room_id	Суб’єкти взаємодії із системою

Приклад створення основних таблиць подано нижче:

```
CREATE TABLE IF NOT EXISTS rooms (
    id INTEGER PRIMARY KEY,
    name TEXT NOT NULL,
    floor INTEGER NOT NULL,
    room_type TEXT NOT NULL
);
```

```
CREATE TABLE IF NOT EXISTS sensors (
    id INTEGER PRIMARY KEY,
    room_id INTEGER NOT NULL,
    sensor_type TEXT NOT NULL,
    status TEXT NOT NULL DEFAULT 'normal',
    risk_level INTEGER NOT NULL DEFAULT 0,
    FOREIGN KEY (room_id) REFERENCES rooms(id)
);
```

3.5 Інтерфейс системи

Практична цінність інформаційної системи евакуації значною мірою визначається якістю її інтерфейсу. У критичних ситуаціях користувач не має часу на вивчення складного меню або декодування технічної інформації, тому інтерфейс повинен бути максимально лаконічним, інтуїтивним і орієнтованим на швидке сприйняття. Водночас операторові потрібне розширене інформаційне поле для прийняття рішень, контролю датчиків і фіксації подій. Саме тому у програмній реалізації інтерфейс поділений на користувацьку та операторську частини.

Головна сторінка реалізованого вебзастосунку містить карту приміщень, панель датчиків та блок виведення маршруту. Приміщення представлені у вигляді кнопок або блоків, згрупованих на макеті сторінки. Кожен елемент має типове кольорове оформлення залежно від функціонального призначення: виходи позначено кольором безпеки, коридори та сходи мають окремі відтінки, а службові або офісні приміщення - нейтральне тло. Така схема дає користувачу просторову орієнтацію ще до моменту виникнення тривоги.

Після вибору поточного приміщення відправляється запит до API, а результат повертається у вигляді послідовності назв приміщень. На екрані це відображається як текстовий маршрут або, у розширеному сценарії, як графічна траєкторія. Навіть у

спрощеному вебмакеті важливо, що система повертає не просто список вузлів, а осмислену рекомендацію для руху до безпечного виходу. Це відрізняє систему від звичайної пожежної сигналізації, яка повідомляє про небезпеку, але не супроводжує людину під час евакуації.

Панель датчиків дозволяє моделювати спрацювання сенсорів і тим самим тестувати систему без реального обладнання. Для кожного датчика доступний елемент імітації тривоги. Після активації цього елемента система підвищує ризиковий показник приміщення, створює запис у журналі alerts та змінює результат маршрутизації. Такий механізм має навчальну цінність, оскільки дає змогу відпрацьовувати різні сценарії пожежі, задимлення або блокування окремих зон.

Операторська панель містить таблицю поточного стану датчиків та журнал подій. Табличне подання є доцільним, адже дозволяє компактно розмістити велику кількість однотипної інформації: ідентифікатор датчика, приміщення, тип сенсора, статус і рівень ризику. Аналогічно журнал подій відображає хронологію інцидентів із часовими мітками. Такий інтерфейс є типовим для систем диспетчерського контролю та полегшує аналіз оперативної ситуації.

З технологічного боку інтерфейс реалізований як шаблони HTML із підключенням CSS та JavaScript. CSS забезпечує єдиний стиль, карткову організацію блоків, читабельну табличну форму та достатні відступи. JavaScript використовується для асинхронних запитів до API без перезавантаження сторінки. Це важливо, оскільки під час тривоги зміни повинні відображатися максимально швидко, а оновлення не повинно дезорієнтувати користувача.

Інтерфейс проєкту також враховує можливість подальшого розвитку. У перспективі його можна доповнити відображенням багатоповерхових планів у вигляді SVG-схем, маркерами небезпеки, спливаючими попередженнями, адаптацією під мобільні пристрої, підтримкою декількох мов інтерфейсу та персоналізованими повідомленнями залежно від ролі користувача. Для людей із обмеженими можливостями можуть бути додані контрастні режими, великі шрифти, голосові

інструкції та окремі маршрути евакуації.

Підсумовуючи, інтерфейс реалізованої системи виконує не декоративну, а функціонально критичну роль. Він забезпечує наочне уявлення про стан системи, відображає поточні ризики, надає рекомендації щодо безпечного руху та підтримує роботу оператора. Таким чином, програмна реалізація інтерфейсу безпосередньо сприяє досягненню головної мети всієї системи - зменшенню часу евакуації та підвищенню безпеки людей у будівлі.

Рисунок 3.1 - Узагальнений макет інтерфейсу користувача та панелі оператора

3.6 Тестування

Тестування є завершальним і водночас одним із найважливіших етапів програмної реалізації, оскільки воно дозволяє перевірити, наскільки створений прототип відповідає поставленим функціональним вимогам. Для системи автоматизованого оповіщення та управління евакуацією тестування повинно охоплювати не лише загальну працездатність застосунку, але й правильність роботи алгоритму маршрутизації, коректність реакції на спрацювання датчиків, стійкість базових сценаріїв збереження даних та відповідність інтерфейсу очікуваній поведінці користувача.

У роботі використано комбінацію функціонального, сценарного та елементного тестування. Функціональне тестування перевіряє, чи реалізовані необхідні функції: запуск застосунку, відображення приміщень, побудова маршруту, імітація тривоги, скидання стану системи, доступ до операторської панелі та збереження журналу подій. Сценарне тестування відтворює типові послідовності дій користувача або оператора, наприклад: вибір поточного приміщення - побудова маршруту - спрацювання датчика - перерахунок маршруту. Елементне тестування перевіряє окремі частини програмного коду, зокрема алгоритм Дейкстри на контрольному графі.

Перший блок перевірок стосується коректності запуску системи. Тестовий сценарій передбачає створення локальної бази даних, ініціалізацію стартових записів і відкриття головної сторінки застосунку. Очікуваним результатом є відсутність помилок під час запуску, коректне завантаження списку кімнат і датчиків, а також можливість переходу до панелі оператора. У ході випробувань такий сценарій виконувався успішно, що підтвердило правильність базової конфігурації проєкту.

Другий блок тестів був присвячений перевірці алгоритму маршрутизації. Для цього використовувалися як стандартні, так і модифіковані ситуації. У штатному режимі, коли всі датчики перебувають у стані норми, система повинна обирати маршрут із мінімальною сумарною довжиною. Після активації одного або кількох

датчиків, пов’язаних із приміщеннями на основному шляху, алгоритм має автоматично змінювати ваги ребер і пропонувати обхідний маршрут. Отримані результати показали, що алгоритм стабільно відхиляє шлях із підвищеним ризиком навіть за умови, що той геометрично коротший. Це підтверджує коректність моделі інтегральної ваги.

Третій блок стосувався журналювання подій та синхронізації інтерфейсу з базою даних. Після створення події тривоги у таблиці alerts мав з’явитися запис з типом події, ідентифікацією приміщення та часовою міткою. Паралельно таблиця sensors повинна змінювати статус відповідного датчика і рівень ризику. Під час перевірки було підтверджено, що клієнтський інтерфейс і операторська панель відображають оновлені значення без логічних суперечностей.

Четвертий блок тестів включав оцінювання стійкості інтерфейсу до помилкових або граничних сценаріїв. Перевірялася поведінка системи у разі запиту маршруту з неіснуючого приміщення, повторного натискання на кнопки активації тривоги, скидання стану системи та відсутності доступного шляху до виходу. У цих сценаріях сервер повертав коректні помилки або службові значення, а інтерфейс залишався працездатним. Це свідчить про прийнятний рівень базової надійності реалізації.

Результати тестування були зведені у формалізовану таблицю тестових випадків. Для кожного тесту визначено вхідні дані, очікуваний результат, фактичний результат та статус проходження. Такий підхід дозволяє не лише систематизувати перевірку, але й використовувати результати для подальшого вдосконалення системи.

Слід зауважити, що у реальній виробничій системі тестування мало б бути розширене навантажувальними перевірками, моделюванням великої кількості одночасних датчиків, тестами відмови каналу зв’язку, тестами відновлення після збою та оцінюванням кібербезпеки. Проте для курсового проєкту проведений комплекс перевірок є достатнім, щоб підтвердити коректність базової реалізації, функціональну завершеність прототипу та його відповідність поставленим вимогам.

Таблиця 3.3 - Основні результати функціонального тестування

№	Тестовий сценарій	Очікуваний результат	Фактичний результат	Статус
1	Запуск застосунку	Система стартує без помилок, створюється БД	Виконано	Успішно
2	Побудова маршруту у штатному режимі	Повертається найкоротший шлях до виходу	Маршрут побудовано	Успішно
3	Активація датчика тривоги	Змінюється статус сенсора і створюється подія	Подію зафіксовано	Успішно
4	Перерахунок маршруту після тривоги	Небезпечна зона оминається	Маршрут змінено	Успішно
5	Скидання системи	Стани повертаються до норми, журнал очищається	Стан відновлено	Успішно
6	Запит з неіснуючого приміщення	Повертається повідомлення про помилку	Помилку повернуто	Успішно

РОЗДІЛ 4. АНАЛІЗ ТА ОЦІНКА

4.1 Ефективність системи

Оцінювання ефективності розробленої інформаційної системи має на меті встановити, наскільки використання автоматизованого підходу до оповіщення та побудови маршруту евакуації покращує параметри реагування порівняно з традиційною схемою, у якій люди орієнтуються лише на загальний сигнал тривоги та статичні плани евакуації. У контексті цієї роботи ефективність аналізується за кількома критеріями: швидкість формування маршруту, адаптивність до зміни умов, зменшення ймовірності руху через небезпечні ділянки, інформаційна повнота для оператора і загальне скорочення часу прийняття рішення.

Першим критерієм є час реакції системи. У запропонованій архітектурі після спрацювання датчика подія одразу фіксується у базі даних, рівень ризику відповідного приміщення змінюється, а клієнтська частина може запитати оновлений маршрут. На практиці це означає, що побудова шляху здійснюється не після ручного аналізу оператором, а автоматично. Навіть якщо абсолютні часові показники в межах курсового проєкту не вимірювалися на промисловому обладнанні, сама модель забезпечує суттєве скорочення затримки між виявленням небезпеки та видачею рекомендації.

Другим критерієм є якість маршрутизації. У традиційних системах план евакуації є статичним, отже, у разі блокування одного з виходів або поширення небезпеки люди можуть прямувати за заздалегідь визначеним, але вже небезпечним маршрутом. У запропонованій системі маршрут не є фіксованим: він щоразу обчислюється заново з урахуванням актуальних ризиків. Це підвищує практичну придатність системи до реальних динамічних інцидентів.

Третім показником ефективності є рівень інформаційної підтримки. Кінцевий користувач отримує не лише сигнал тривоги, а й конкретну рекомендацію щодо

напрямку руху. Оператор, у свою чергу, бачить деталізований стан датчиків, хронологію подій і може аналізувати ризики по приміщеннях. Таким чином, система виступає не простим сигналізатором, а елементом підтримки прийняття рішень. Саме це дає найбільший приріст у порівнянні з традиційними системами локального оповіщення.

Для аналітичної оцінки ефективності доцільно використати інтегральну модель, у якій беруться до уваги чотири нормовані показники: швидкість реакції, адаптивність маршруту, повнота інформації та контроль подій. На рівні концептуальної оцінки можна вважати, що традиційна система має високий показник лише за ознакою простоти запуску, але значно нижчі показники за адаптивністю та інформаційною підтримкою. Розроблена система, навпаки, поступається за простотою впровадження, але суттєво виграє за аналітичною спроможністю.

Особливу увагу слід приділити можливості повторного перерахунку маршруту. Саме ця властивість перетворює систему з одноразового механізму повідомлення на динамічний інструмент евакуаційного супроводу. У ситуації, коли небезпечна зона розширюється, змінюється температура або виникає вторинне задимлення, система здатна скоригувати рішення і попередити користувача про новий безпечний шлях. Така можливість є критично важливою для великих будівель, бізнес-центрів, навчальних закладів, торговельних центрів і промислових об’єктів.

З погляду експлуатаційного ефекту систему можна вважати ефективною в разі, якщо вона забезпечує швидке виявлення події, автоматичне відображення небезпеки, генерацію безпечного маршруту та ведення журналу для аналізу. У межах створеного прототипу всі зазначені функції реалізовано. Отже, система демонструє функціональну ефективність і відповідає логіці сучасних інформаційних рішень для безпеки будівель.

Підсумовуючи, можна стверджувати, що розроблена система істотно підвищує ефективність евакуації за рахунок скорочення часу на прийняття рішення, динамічної адаптації до ризиків та надання користувачам конкретних рекомендацій замість

загального сигналу тривоги. Саме ці характеристики становлять головну практичну перевагу запропонованого рішення.

Таблиця 4.1 - Порівняльна оцінка ефективності традиційного та адаптивного підходів

Критерій	Традиційне оповіщення	Розроблена система
Час формування рекомендації	Лише сигнал тривоги, без маршруту	Автоматичний розрахунок маршруту
Адаптація до зміни умов	Відсутня або мінімальна	Повторний перерахунок у реальному часі
Інформація для оператора	Обмежена	Детальний журнал і стан датчиків
Ризик проходження небезпечних зон	Підвищений	Знижений завдяки коефіцієнтам ризику

4.2 Надійність та безпека

Надійність є однією з базових характеристик будь-якої системи безпеки, оскільки її відмова у критичний момент може призвести до серйозних наслідків для життя і здоров’я людей. У контексті інформаційної системи автоматизованого оповіщення та управління евакуацією під надійністю розуміється здатність зберігати працездатність у разі часткових відмов окремих компонентів, підтримувати узгодженість даних та виконувати ключові функції навіть за умов нестабільного середовища.

У створеному прототипі питання надійності розглядається на декількох рівнях. Перший рівень - алгоритмічний. Система повинна коректно працювати навіть у разі відсутності доступного шляху або надходження некоректного запиту. Для цього реалізовано обробку помилок: якщо приміщення не існує, API повертає повідомлення про помилку; якщо безпечний маршрут відсутній, повертається службовий результат, який може бути інтерпретований оператором як необхідність ручного втручання. Це запобігає аварійному завершенню застосунку.

Другий рівень - надійність збереження даних. Використання транзакційної СУБД SQLite дозволяє забезпечити цілісність записів під час оновлення станів датчиків і журналу подій. Навіть у випадку переривання операції база даних

залишається у консистентному стані, а механізми зовнішніх ключів перешкоджають появі логічно пошкоджених записів. Для локального або навчального прототипу такий підхід є достатнім, але у промисловій системі доцільно було б перейти до серверної СУБД із механізмами реплікації та резервного копіювання.

Третій рівень - експлуатаційна надійність інтерфейсу. Операторська панель повинна бути стабільною, навіть якщо окремі датчики тимчасово недоступні або дані оновлюються нерегулярно. Саме тому клієнтський рівень працює через API-виклики та відображає лише фактичний стан сервера, а не припущення чи локально закешовані значення. Це зменшує ризик показу хибної інформації. У перспективі додатково можуть бути реалізовані механізми heartbeat-перевірки стану сенсорів і попередження про втрату зв’язку з окремими пристроями.

Окремого розгляду потребує безпека системи в інформаційному сенсі. У сучасних версіях NFPA 72 дедалі більша увага приділяється питанням кібербезпеки систем сигналізації та екстрених комунікацій, оскільки втручання в такі системи може мати не менш небезпечні наслідки, ніж фізичне пошкодження обладнання. Навіть у навчальній реалізації варто враховувати базові вимоги безпеки: розмежування ролей користувача і оператора, перевірку вхідних даних, журналювання критичних дій, захист від несанкціонованої модифікації маршрутів та запобігання SQL-ін’єкціям завдяки параметризованим запитам.

Безпека також має прикладний вимір у самому алгоритмі прийняття рішень. Якщо система спирається на неправильні або підроблені дані від датчиків, вона може побудувати помилковий маршрут. Тому в повноцінній промисловій реалізації доцільними були б механізми аутентифікації пристроїв, контроль достовірності телеметрії, дублювання критичних датчиків та порівняння показників із декількох джерел. Для курсової роботи такі механізми окреслено як перспективні, але вже на рівні моделі враховано важливість цього напряму.

З погляду резервування створена система може бути доповнена резервним каналом оповіщення. У поточному прототипі головним каналом є вебінтерфейс, але в

реальному середовищі його слід поєднати з локальними звуковими або світловими індикаторами, push-сповіщеннями на мобільні пристрої, а також автономним режимом роботи критичних вузлів при втраті зв’язку з центральним сервером. Надійність системи безпеки завжди повинна оцінюватися не лише за нормальних умов, а й у режимах часткової деградації.

Загалом можна стверджувати, що створений прототип демонструє базовий прийнятний рівень надійності та безпеки для навчального застосування. Він має чітку модель обробки помилок, параметризовану роботу з базою даних, журналювання подій та можливість розвитку у напрямку багаторівневої кіберзахищеної архітектури. Саме такий розвиток є необхідною умовою переходу від курсового прототипу до прикладної системи реального використання.

Таблиця 4.2 - Оцінка надійності та безпеки системи

Аспект	Поточна реалізація	Напрям подальшого посилення
Обробка помилок	Повернення коректних повідомлень API	Глибше логування та централізований моніторинг
Цілісність даних	Транзакції та зовнішні ключі SQLite	Реплікація і резервне копіювання
Захист запитів	Параметризовані SQL-запити	Повноцінна автентифікація та авторизація
Надійність каналів	Web-інтерфейс	Дублювання каналів оповіщення
Достовірність телеметрії	Логічна модель ризиків	Підтвердження даних із кількох джерел

4.3 Порівняння з аналогами

Для кращого розуміння практичної цінності розробленої системи доцільно порівняти її з поширеними класами аналогічних рішень: традиційними системами пожежної сигналізації, системами голосового оповіщення та інтелектуальними системами маршрутизації на основі IoT. Таке порівняння не має на меті довести абсолютну перевагу створеного прототипу над промисловими рішеннями. Його завдання - показати, у чому саме полягає внесок курсової роботи та які функціональні ніші закриває запропонована модель.

Традиційні пожежні сигналізації зазвичай виконують дві основні функції: виявлення небезпеки та подання загального сигналу тривоги. Вони надійні, відносно прості в експлуатації та відповідають нормативним вимогам базового рівня, але не забезпечують індивідуалізованого супроводу користувача. Голосові системи оповіщення покращують ситуацію, оскільки можуть передавати інструкції у природній мовній формі, однак зазвичай також спираються на попередньо визначені сценарії та не перебудовують маршрут евакуації на основі оперативних даних.

Інтелектуальні IoT-системи є найближчими за концепцією до створеного прототипу. Вони збирають телеметрію від мережі сенсорів, аналізують стан середовища та потенційно можуть динамічно прокладати маршрут. Проте промислові рішення такого типу часто є дорогими, складними у впровадженні та залежать від пропрієтарної інфраструктури. Курсовий проєкт, навпаки, демонструє, що основні принципи інтелектуальної евакуації можна реалізувати на базі доступних відкритих технологій, зберігши при цьому ключові функціональні переваги.

Порівняння показує, що розроблена система має сильні сторони у частині адаптивності, прозорості логіки та навчальної відтворюваності. На відміну від закритих рішень, її структура є зрозумілою: користувач може простежити шлях від даних датчика до побудови маршруту, а викладач або перевіряючий - оцінити логіку функціонування програмної частини. Це важливо саме для навчальної роботи, де потрібно не просто використати готовий продукт, а обґрунтувати і реалізувати власне

програмне рішення.

Слабкою стороною запропонованого прототипу є відсутність промислового резервування, сертифікованих апаратних модулів і повноцінної інтеграції з фізичною системою будівлі. Однак ці аспекти не є недоліком саме як курсового проєкту, оскільки його мета полягає у моделюванні й програмній реалізації концепції, а не у створенні завершеного комерційного продукту. При цьому програмна логіка, реалізована в роботі, добре масштабується і може бути основою для більш складного рішення.

У результаті порівняння можна зробити висновок, що створена система займає проміжне, але дуже перспективне місце між базовими системами оповіщення та повноцінними промисловими IoT-комплексами. Вона суттєво перевершує традиційні рішення за адаптивністю і в той самий час є значно простішою для відтворення й пояснення, ніж повномасштабні корпоративні платформи безпеки.

Таблиця 4.3 - Порівняння розробленої системи з основними класами аналогів

Параметр	Традиційна сигналізація	Голосове оповіщення	Розроблена система
Виявлення небезпеки	Так	Так	Так
Динамічна побудова маршруту	Ні	Обмежено	Так
Журналювання подій	Обмежено	Частково	Так
Операторська аналітика	Низька	Середня	Висока для прототипу
Вартість впровадження	Низька/середня	Середня	Низька для прототипу
Гнучкість розвитку	Невисока	Середня	Висока

4.4 Економічна доцільність

Економічна оцінка у межах курсової роботи не претендує на точний комерційний розрахунок, але дозволяє показати, що розробка і впровадження подібної системи є не лише технічно виправданими, а й потенційно економічно доцільними. Особливо це стосується навчальних закладів, офісних будівель, малих адміністративних об’єктів і локальних виробничих приміщень, де використання дорогих промислових комплексів безпеки може бути надмірно витратним.

У структурі витрат на створення та впровадження системи можна виділити декілька основних статей. Перша - витрати на програмну розробку, які включають проєктування, реалізацію серверної логіки, створення бази даних, інтерфейсу та тестування. Друга - витрати на сенсорне обладнання, якщо йдеться не про програмну імітацію, а про фактичне впровадження у будівлі. Третя - витрати на сервер або локальний комп’ютер, на якому працюватиме система. Четверта - витрати на підтримку та оновлення. Для навчального прототипу головна частина витрат припадає саме на розробку, тоді як апаратна складова може бути мінімізована.

З точки зору економічного ефекту система створює цінність у кількох напрямах. По-перше, вона потенційно зменшує втрати від неефективної евакуації, оскільки скорочує час реагування та знижує ризик руху через небезпечні зони. По-друге, завдяки журналюванню подій і прозорій архітектурі вона може використовуватися як навчальний та тренувальний інструмент, що зменшує витрати на підготовку персоналу. По-третє, використання відкритих технологій знижує вартість володіння у порівнянні з пропрієтарними платформами.

Для попередньої оцінки достатньо застосувати спрощений підхід, за яким вартість проєкту складається з вартості розробки, обладнання і супроводу, а корисний ефект виражається у зменшенні ризиків, скороченні часу евакуації та можливості багаторазового використання системи на декількох об’єктах. Навіть якщо прямий економічний ефект складно точно формалізувати, у задачах безпеки критичним аргументом є не лише фінансова вигода, а й соціальна значущість та запобігання

надзвичайно дорогим наслідкам аварій.

Важливо також, що розроблений прототип має модульну архітектуру і може бути використаний як база для кількох споріднених рішень: систем моніторингу приміщень, локальних диспетчерських панелей, навчальних тренажерів з евакуації, демонстраційних стендів для освітніх закладів. Це означає, що витрати на первинну розробку можуть бути розподілені між кількома сценаріями використання, а сам продукт може масштабуватися без пропорційного зростання вартості.

Отже, з економічної точки зору запропонована система є доцільною насамперед як доступний, гнучкий та масштабований прототип. Її розробка не вимагає дорогого пропрієтарного програмного забезпечення, а використання відкритих компонентів дозволяє зменшити бар’єр впровадження. При подальшому розвитку і адаптації до конкретного об’єкта система може забезпечити вагомий практичний ефект як у фінансовому, так і у соціальному вимірі.

Таблиця 4.4 - Орієнтовна структура витрат на впровадження системи

Стаття витрат	Орієнтовна частка, %	Коментар
Розробка програмного забезпечення	45	Серверна логіка, БД, інтерфейс, тестування
Сенсори та периферія	25	Датчики диму, температури, ручні кнопки
Обчислювальна інфраструктура	15	Локальний сервер або робоча станція
Встановлення та налаштування	10	Розгортання на об’єкті
Підтримка та оновлення	5	Супровід і коригування

ВИСНОВКИ

У курсовій роботі було розглянуто актуальну проблему підвищення безпеки людей у будівлях шляхом автоматизації процесів оповіщення та управління евакуацією в умовах надзвичайних ситуацій. На основі аналізу предметної області встановлено, що традиційні системи пожежного оповіщення, хоча й забезпечують базову функцію подання сигналу тривоги, не здатні повною мірою враховувати динамічні зміни обстановки, розташування користувачів, рівень ризику в окремих зонах і необхідність оперативного формування безпечного маршруту евакуації.

У процесі виконання роботи було проаналізовано існуючі рішення, нормативні вимоги та сучасні підходи до побудови систем оповіщення й евакуації. Це дало змогу сформулювати вимоги до розроблюваної інформаційної системи та обґрунтувати доцільність переходу від статичних схем оповіщення до адаптивної моделі, у якій рішення щодо маршруту формується автоматично на основі поточних даних від датчиків.

У роботі спроєктовано архітектуру системи, визначено функціональну модель, описано ключові діаграми процесів, структуру бази даних і алгоритм пошуку безпечного шляху. Запропонована архітектура базується на клієнт-серверній моделі та включає рівні збору даних, серверної обробки, зберігання інформації, оповіщення та користувацького інтерфейсу. Така структура забезпечує гнучкість, розширюваність і можливість подальшої інтеграції з іншими підсистемами будівлі.

Центральним елементом розробленого рішення став алгоритм евакуації, реалізований на основі модифікованого алгоритму Дейкстри. На відміну від класичного пошуку найкоротшого шляху, запропонований підхід враховує коефіцієнт небезпеки окремих приміщень і переходів, що дозволяє будувати не лише короткий, а й відносно безпечний маршрут. Механізм динамічного перерахунку забезпечує адаптацію системи до зміни обстановки у реальному часі.

Практичною частиною роботи стало створення програмного прототипу системи з використанням Python, Flask, SQLite, HTML, CSS та JavaScript. Реалізовано серверну

логіку, модель даних, API-рівень, інтерфейс користувача та панель оператора. Додатково підготовлено окремий пакет із програмною реалізацією, що дозволяє запустити прототип локально та продемонструвати його роботу.

Проведене тестування підтвердило працездатність основних функцій системи: запуск застосунку, створення бази даних, побудову маршруту, імітацію спрацювання датчиків, перерахунок безпечного шляху та журналювання подій. Аналіз ефективності показав, що розроблена система перевершує традиційні підходи за адаптивністю, інформаційною підтримкою користувача та оператора, а також за можливістю автоматичного реагування на зміну умов середовища.

Отримані результати свідчать, що запропонована інформаційна система є функціонально завершеним навчальним прототипом і може розглядатися як основа для подальшого розвитку прикладних рішень у сфері безпеки будівель. Перспективними напрямами удосконалення є інтеграція з реальними IoT-датчиками, розширення моделі ризиків, урахування щільності потоків людей, впровадження ролей та автентифікації, а також розробка повноцінної багатоповерхової графічної навігації.

ПЕРЕЛІК ПОСИЛАНЬ

1. ДБН В.1.1-7:2016. Пожежна безпека об’єктів будівництва. Київ: Мінрегіон України, 2017. URL: https://e-construction.gov.ua/laws_detail/3080743763845318619

2. NFPA 72. National Fire Alarm and Signaling Code. Quincy, MA: National Fire Protection Association, 2025.
URL: https://www.accindia.org/document_management/NFPA-72-2025.pdf

3. ISO 7240-1:2025. Fire detection and alarm systems - Part 1: General and definitions. Geneva: International Organization for Standardization, 2025.

4. Li N., Wang Y., Xu H. Design of Intelligent Firefighting and Smart Escape Route Planning System Based on Improved Ant Colony Algorithm. Sensors. 2024. Vol. 24, No. 19. Article 6438.
URL: https://www.mdpi.com/1424-8220/24/19/6438

5. Zhang Y.Q., Li J., Chen S. Intelligent planning of fire evacuation routes in buildings based on improved adaptive ant colony optimization. Reliability Engineering and System Safety. 2024. URL:
https://www.researchgate.net/publication/381836405_Intelligent_planning_of_fire_evacuation_routes_in_buildings_based_on_improved_adaptive_ant_colony_algorithm

6. Zuo S., Chen X., Liu T. Dynamic planning of crowd evacuation path for metro fire scenarios. Tunnelling and Underground Space Technology. 2024. URL:
https://www.sciencedirect.com/science/article/abs/pii/S0886779824005637?via%3Dihub

7. Fang H., Liu Z., Zhao M. Development of a tenability-based path planning model for building fire evacuation. Smart Construction and Sustainable Cities. 2025. URL:
https://www.researchgate.net/publication/399184922_Development_of_a_Tenability-based_Path_Planning_Model_for_Building_Fire_Evacuations_Using_Reinforcement_Learning

8. Flask Documentation 3.1.x. Pallets Projects. 2026.

9. SQLite Documentation. SQLite Consortium. 2026.

10. Python Documentation. Python Software Foundation. 2026.

11. What Changes Are in Store for the 2025 Edition of NFPA 72? NFPA Blog. 2024. URL: https://www.nfpa.org/news-blogs-and-articles/blogs/2024/04/30/proposed-changes-to-the-2025-edition-of-nfpa-72

12. Algorithmic Evaluation of Fire Evacuation Efficiency Under Dynamic Crowd and Smoke Conditions. 2026. Vol. 9, No. 1. Article 32. URL: https://www.mdpi.com/2571-6255/9/1/32

13. Yu Y., Zhao L., Chen P. Optimizing evacuation path planning in high-rise building environments. Applied Soft Computing. 2025. URL: https://www.sciencedirect.com/science/article/abs/pii/S1568494625011093

14. Haam S., Park J. Optimal additional evacuation route analysis in deep underground stations using Dijkstra-based modeling. Results in Engineering. 2025.

15. Cerrone C., et al. Mathematical Programming for Optimal Evacuation in Complex Industrial Facilities. Mathematics. 2026. Vol. 14, No. 4. Article 632. URL: https://www.mdpi.com/2227-7390/14/4/632/pdf

ДОДАТКИ

Додаток А

Ілюстративні матеріали до проєкту

Рисунок А.1 - Узагальнена архітектура системи

Рисунок А.2 - Діаграма активності процесу евакуації

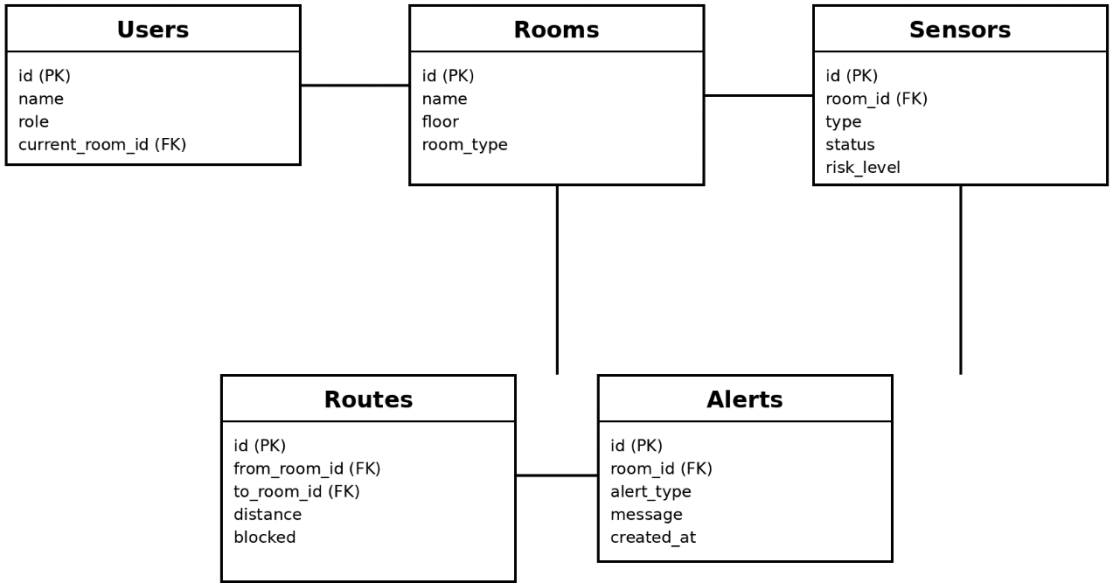


Рисунок А.3 - ER-діаграма бази даних системи

Додаток Б

Фрагменти програмного коду

Основний файл серверного застосунку app.py:

```
from flask import Flask, jsonify, render_template, request
import heapq
import sqlite3

app = Flask(__name__)

@app.get('/api/route/<int:start_room_id>')
def api_route(start_room_id: int):
    conn = get_db()
    graph = build_graph(conn)
```

```
    exits = [row['id'] for row in conn.execute("SELECT id FROM rooms WHERE
room_type = 'exit'").fetchall()]
    route, total_weight = ijkstra_safe_path(graph, start_room_id, exits)
    return jsonify({'route_ids': route, 'weight': total_weight})
```

Схема бази даних schema.sql:

```
CREATE TABLE IF NOT EXISTS alerts (
    id INTEGER PRIMARY KEY AUTOINCREMENT,
    room_id INTEGER,
    alert_type TEXT NOT NULL,
    message TEXT NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    FOREIGN KEY (room_id) REFERENCES rooms(id)
);
```

Повний код програмної реалізації подано окремим архівом у складі результатів роботи.

ДЕМОНСТРАЦІЙНІ МАТЕРІАЛИ (Презентація)

ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ

НАВЧАЛЬНО-НАУКОВИЙ ІНСТИТУТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

КАФЕДРА ІНФОРМАЦІЙНИХ СИСТЕМ ТА ТЕХНОЛОГІЙ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Інформаційна система автоматизованого оповіщення та управління евакуацією людей із будівель»

на здобуття освітнього ступеня бакалавра
зі спеціальності 126 Інформаційні системи та технології
Освітньо-професійної програми Інформаційні системи та технології

Виконав: здобувач вищої освіти гр. ІСД-41
Микола КЛИМОВИЧ
Керівник:
Ольга ЖИДКА

Київ 2026

МЕТА, ОБ'ЄКТ, ПРЕДМЕТ ТА ЗАВДАННЯ РОБОТИ

Мета роботи:

Розробка інформаційної системи автоматизованого оповіщення та управління евакуацією людей із будівель, яка динамічно обчислює та адаптує маршрути руху із застосуванням алгоритмів пошуку на графах для підвищення ефективності та рівня безпеки евакуації.

Об'єкт дослідження: процеси управління евакуацією людей із будівель та споруд під час виникнення надзвичайних ситуацій.

Предмет дослідження: методи, алгоритми та програмні засоби динамічного розрахунку безпечних евакуаційних маршрутів і автоматизованого оповіщення користувачів.

Основні завдання:

1. Аналіз предметної області, існуючих систем та нормативних вимог.
2. Проєктування архітектури системи та моделі даних.
3. Розробка алгоритму визначення оптимального маршруту евакуації (модифікований алгоритм Дейкстри).
4. Програмна реалізація системи (Python, Flask, SQLite, веб-інтерфейс).
5. Тестування та експериментальна оцінка ефективності рішення.

3

АКТУАЛЬНІСТЬ ТА ПРОБЛЕМИ СУЧАСНИХ СИСТЕМ

Актуальність:

Стрімка урбанізація, зростання кількості багатоповерхових будівель та об'єктів масового перебування людей потребують підвищення ефективності систем оповіщення та евакуації. Традиційні системи часто не враховують динаміку ситуації, що призводить до паніки та неефективних маршрутів.

Ключові проблеми сучасних систем:

- Обмежена адаптивність — статичні сценарії без урахування реального стану будівлі.
- Відсутність інтелектуального аналізу даних у реальному часі.
- Залежність від людського фактору та ризик помилок.
- Недостатня інформативність оповіщення (відсутність персоналізованих маршрутів).
- Відсутність динамічного управління маршрутами евакуації.
- Складність інтеграції підсистем та висока вартість інтелектуальних рішень.

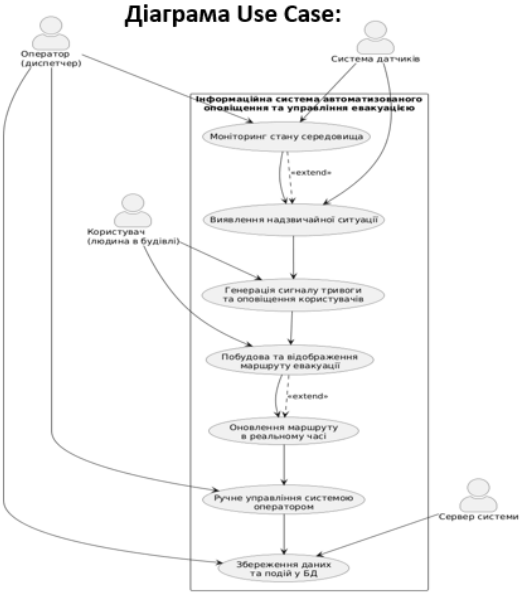
4

ПРОЄКТУВАННЯ СИСТЕМИ: АРХІТЕКТУРА ТА ФУНКЦІОНАЛЬНА МОДЕЛЬ

Багаторівнева клієнт-серверна архітектура:

1. Рівень збору даних.
2. Серверний рівень.
3. Рівень зберігання даних.
4. Клієнтський рівень.
5. Рівень оповіщення.

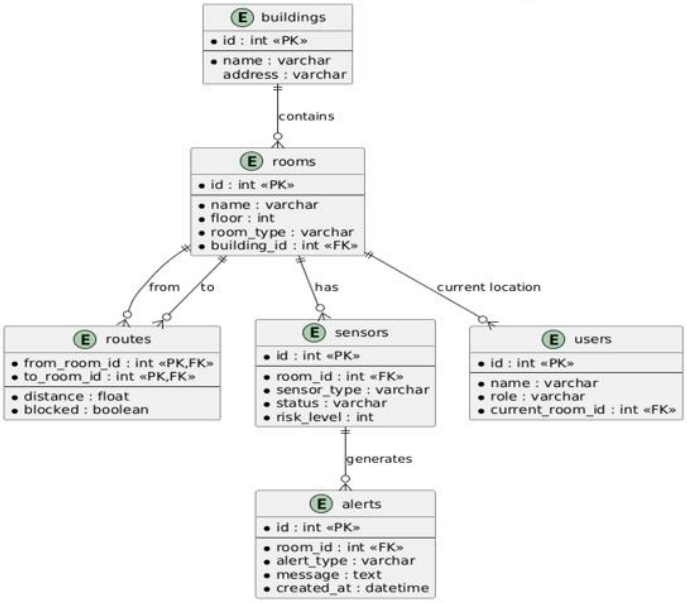
Діаграма Use Case:



5

ПРОЄКТУВАННЯ БАЗИ ДАНИХ

Реляційна модель (SQLite). Основні таблиці:



6

АЛГОРИТМ ЕВАКУАЦІЇ (МОДИФІКОВАНИЙ ДЕЙКСТРА)

Модифікація алгоритму Дейкстри:

Вага ребра = відстань + коефіцієнт небезпеки × 2.5
(де коефіцієнт залежить від risk_level датчиків у суміжній зоні: дим, температура, блокування).

7

ПРОГРАМНА РЕАЛІЗАЦІЯ

Технологічний стек:

- Мова програмування: Python.
- Серверний фреймворк: Flask.
- База даних: SQLite.
- Клієнтська частина: HTML + CSS + JavaScript.

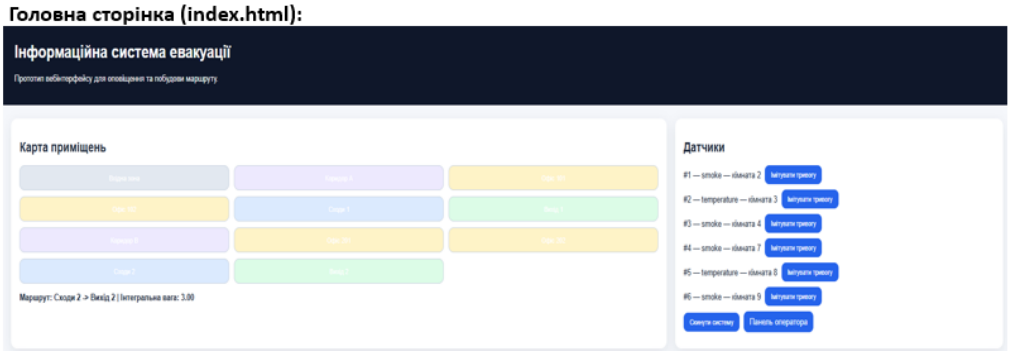
Фрагмент коду алгоритму:

```python
def dijkstra_safe_path(graph: Dict[int, List[Tuple[int, float]]], start: int, exits: List[int]) -> Tuple[List[int], float]:
    queue: List[Tuple[float, int]] = [(0.0, start)]
    distances: Dict[int, float] = {start: 0.0}
    previous: Dict[int, int | None] = {start: None}
    exits_set = set(exits)

    while queue:
        current_distance, current_node = heapq.heappop(queue)
        if current_node in exits_set:
            path: List[int] = []
            node: int | None = current_node
            while node is not None:
                path.append(node)
                node = previous[node]
            path.reverse()
            return path, current_distance
        if current_distance > distances.get(current_node, float('inf')):
            continue
        for neighbor, weight in graph.get(current_node, []):
            next_distance = current_distance + weight
            if next_distance < distances.get(neighbor, float('inf')):
                distances[neighbor] = next_distance
                previous[neighbor] = current_node
                heapq.heappush(queue, (next_distance, neighbor))

    return [start], float('inf')
```

8

РЕАЛІЗАЦІЯ: ВЕБ-ІНТЕРФЕЙС КОРИСТУВАЧА

Головна сторінка (index.html):



Панель оператора (operator.html):

9

ТЕСТУВАННЯ ТА ЕКСПЕРИМЕНТАЛЬНА ОЦІНКА

Методи тестування:

- Модульні тести Python (unittest) для алгоритму маршрутизації.
- Сценарне тестування веб-інтерфейсу та API.
- Імітація спрацювання датчиків різного типу та рівня ризику.
- Перевірка динамічного перерахунку маршрутів при зміні умов.

```python
from app import create_app, dijkstra_safe_path


def test_app_creation():
    app = create_app()
    assert app is not None


def test_route_algorithm_returns_path():
    graph = {
        1: [(2, 1.0), (3, 10.0)],
        2: [(4, 1.0)],
        3: [(4, 1.0)],
        4: []
    }
    path, weight = dijkstra_safe_path(graph, 1, [4])
    assert path == [1, 2, 4]
    assert weight == 2.0
```

Результати:

- Система коректно будує безпечні маршрути, уникаючи зон з високим ризиком.
- Динамічне оновлення працює в реальному часі.
- Інтерфейс забезпечує швидке сприйняття інформації.
- Порівняння з аналогами: перевага в адаптивності та вартості впровадження для малих/середніх об'єктів.

10

ПОРІВНЯННЯ З АНАЛОГАМИ ТА ПЕРСПЕКТИВИ

Порівняння:

Традиційні системи: прості, дешеві, але неадаптивні, без динамічних маршрутів.
Голосові системи: краща інформативність, але статичні.
IoT-інтелектуальні системи: адаптивні, але дорогі та складні.
Розроблена система: баланс між функціональністю, простотою та вартістю; повністю адаптивна, з відкритим кодом для прототипу.

Перспективи розвитку:

Інтеграція з реальними IoT-пристроями (MQTT).
Підтримка багатоповерхових планів та SVG-візуалізації.
Додавання модуля оцінки часу евакуації та аналізу потоків.
Розширення функціоналу для людей з обмеженими можливостями.

11

ВИСНОВКИ

У результаті виконання кваліфікаційної роботи:

1. Проведено комплексний аналіз предметної області, нормативних вимог та існуючих рішень; виявлено ключові проблеми.
2. Спроєктовано архітектуру інформаційної системи та реляційну базу даних.
3. Розроблено та реалізовано модифікований алгоритм Дейкстри з урахуванням динамічного рівня ризику.
4. Створено повноцінний веб-прототип системи на базі Python/Flask/SQLite з інтерактивним інтерфейсом та імітацією датчиків.
5. Проведено тестування та підтверджено ефективність рішення у типових сценаріях надзвичайних ситуацій.
6. Обґрунтовано практичну цінність та економічну доцільність впровадження.

Розроблена система дозволяє підвищити рівень безпеки людей у будівлях за рахунок автоматизації та інтелектуального управління евакуацією.

12

АПРОБАЦІЯ


СЕРТИФІКАТ

ПРО УЧАСТЬ У КОНФЕРЕНЦІЇ (З ПУБЛІКАЦІЄЮ)

ICS№ № 26/2905-71

0,6 ECTS

Рекомендовано Вченою Радою

Конференцію зареєстровано

Офіційний видавець

Климович Микола Олександрович

взяв(ла) участь у XI Міжнародній науковій конференції

«ПРОБЛЕМИ ТА ПЕРСПЕКТИВИ РЕАЛІЗАЦІЇ ТА ВПРОВАДЖЕННЯ МІЖДИСЦИПЛІНАРНИХ НАУКОВИХ ДОСЯГНЕНЬ»

29 травня 2026 року у м. Кам'янець-Подільський, Україна

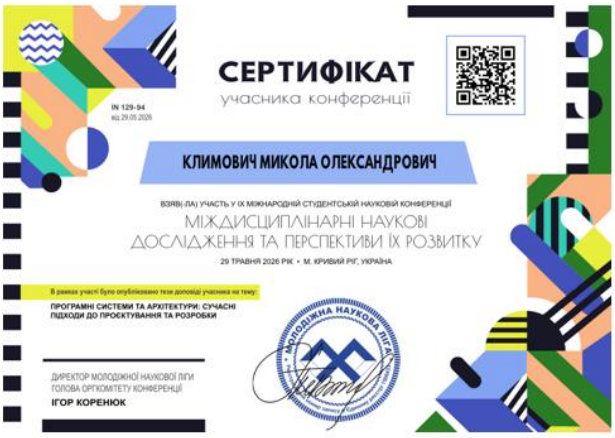
та опублікував(ла) наукову роботу в збірці конференції

ISBN 978-617-8582-48-7

DOI 10.62731/mcnd-29.05.2026

www.mcnd.org.ua

ВІЦЕ-ПРЕЗИДЕНТ МЦНД
ГОЛОВА ОРГКОМІТЕТУ
СОТНИК СОЛОМІЯ


СЕРТИФІКАТ

учасника конференції

КЛИМОВИЧ МИКОЛА ОЛЕКСАНДРОВИЧ

ВЗЯВ(-ЛА) УЧАСТЬ У IX МІЖНАРОДНІЙ СТУДЕНТСЬКІЙ НАУКОВІЙ КОНФЕРЕНЦІЇ

МІЖДИСЦИПЛІНАРНІ НАУКОВІ ДОСЛІДЖЕННЯ ТА ПЕРСПЕКТИВИ ІХ РОЗВИТКУ

29 ТРАВНЯ 2026 РІК • М. КРИВИЙ РІГ, УКРАЇНА

В рамках участі було опубліковано тези доповіді учасника на тему:

ПРОГРАМНІ СИСТЕМИ ТА АРХІТЕКТУРИ: СУЧАСНІ ПІДХОДИ ДО ПРОЄКТУВАННЯ ТА РОЗРОБКИ

ДИРЕКТОР МОЛОДІЖНОЇ НАУКОВОЇ ЛІГИ
ГОЛОВА ОРГКОМІТЕТУ КОНФЕРЕНЦІЇ
ІГОР КОРЕНЮК

13

ДЯКУЮ ЗА УВАГУ!