Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Підвищення захищеності об'єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв)»

Виконав: здобувач 2-го курсу, групи КІТС-23мз спеціальності 125– Кібербезпека та захист інформації
Освітня програма – Кібербезпека інформаційних технологій та систем
(шифр і назва напряму підготовки, спеціальності)

Герасимчук Андрій Андрійович
(прізвище та ініціали)

Керівник: д.т.н., проф., голова секції "Управління інформаційної безпекою" каф. МБІС
Яремчук Ю.Є.
(прізвище та ініціали)
«____» _____ 2025 р.

Опонент: к.т.н., доц., доцент каф. ОТ
(прізвище та ініціали)
«____» _____ 2025 р.

Допущено до захисту
Голова секції УБ кафедри МБІС
Юрій ЯРЕМЧУК
"____" _____2025 р.

Вінниця ВНТУ - 2025 рік

Вінницький національний технічний університет
Факультет менеджменту та інформаційної безпеки
Кафедра менеджменту та безпеки інформаційних систем

Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 – Інформаційні технології
Спеціальність 125 – Кібербезпека та захист інформації
Освітньо-професійна програма - Кібербезпека інформаційних технологій та систем

ЗАТВЕРДЖУЮ
Голова секції УБ, кафедра МБІС
_____**Юрій ЯРЕМЧУК**
“_____”_____2025 р.

З А В Д А Н Н Я
на магістерську кваліфікаційну роботу здобувачу
Герасимчуку Андрію Андрійовичу
(прізвище, ім`я, по-батькові)

1. Тема роботи Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв)
Керівник роботи д.т.н., професор, Яремчук Ю.Є.
(прізвище, ім`я, по-батькові, науковий ступінь, вчене звання)
затверджені наказом вищого навчального закладу від “20” березня 2025 року № 96
2. Строк подання студентом роботи за тиждень до захисту
3. Вихідні дані до роботи: Стандарти, електронні джерела, підручники та наукові статті по темі, які стосуються теми магістерської кваліфікаційної роботи.
4. Зміст текстової частини:
Робота складається з трьох розділів. У першому розділі розглянуто теоретичні аспекти захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв), включаючи аналіз сучасних систем захисту. У другому розділі описано процес проектування системи підвищення безпеки, враховуючи особливості використання технологій відстеження обличчя, розробку алгоритмів і моделі автоматизованого прийняття рішень. У третьому розділі представлено програмну реалізацію системи, обгрунтовано вибір технологій, реалізовано основні модулі системи, включаючи графічний інтерфейс, проведено тестування для перевірки її ефективності.

5. Перелік ілюстративного матеріалу (з точним зазначенням обов`язкових креслень)
У першому розділі наведено 1 рисунок, у другому розділі магістерської кваліфікаційної роботи наведено 2 рисунки, у третьому розділі – 10 рисунків.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Основна частина			
I	д.т.н., професор, Яремчук Ю.Є.		
II	д.т.н., професор, Яремчук Ю.Є.		
III	д.т.н., професор, Яремчук Ю.Є.		
Економічна частина			
IV	зав. кафедри ЕПВМ, к.е.н., проф., Лесько О.Й.	20.03.25	15.06.25

7.Дата видачі завдання 20 березня 2025 р.

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи		Примітка
1	Визначення напрямку магістерської роботи, формулювання теми	05.03.2025	20.03.2025	
2	Аналіз предметної області обраної теми	21.03.2025	05.04.2025	
3	Розробка алгоритму роботи	06.04.2025	06.05.2025	
4	Написання магістерської роботи на основі розробленої теми			
5	Розробка економічної частини	06.05.2025	23.05.2025	
6	Передзахист магістерської кваліфікаційної роботи	24.05.2025	30.05.2025	
7	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	31.05.2025	10.06.2025	
8	Захист магістерської кваліфікаційної роботи	11.06.2025	14.06.2025	

Здобувач _____ Герасимчук А. А.
(підпис)

Керівник роботи _____ Яремчук Ю.Є.
(підпис)

АНОТАЦІЯ

УДК 621.374.415

Герасимчук А.А. Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв). Магістерська кваліфікаційна робота зі спеціальності 125 – «Кібербезпека та захист інформації», освітня програма «Кібербезпека інформаційних технологій та систем». Вінниця: ВНТУ, 2025. – 113 с.
На укр. мові. Бібліогр.: 46 назв; рис.: 14; табл. 10.

У роботі представлено розробку системи підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв), шляхом автоматичного розпізнавання обличь з використанням мови програмування Python та низки спеціалізованих бібліотек, зокрема OpenCV, face_recognition і Flask. Система призначена для відеомоніторингу в режимі реального часу, фіксації подій появи облич у кадрі, визначення їх належності до відомих чи невідомих осіб, а також сповіщення адміністратора через месенджер Telegram та веб-браузер.

Однією з ключових особливостей реалізованого рішення є локальне збереження бази даних обличь, підтримка режиму охорони, інтегрований веб інтерфейс керування, ведення журналу подій та авторизований доступ користувача до адміністративної панелі. Інтерфейс виконано в сучасному стилі з акцентом на простоту, зручність та інтеграцію із зовнішніми сервісами. Реалізовано двосторонню взаємодію з Telegram-ботом: користувач може отримати повідомлення про події, поточне фото з камери або увімкнути/вимкнути систему охорони безпосередньо через чат.

Розроблене рішення є повністю автономним, не вимагає хмарних сервісів або зовнішніх серверів, що забезпечує високий рівень конфіденційності та контроль над даними. Система може бути розгорнута як на персональному комп’ютері, так і на міні-комп’ютерах типу Raspberry P. З урахуванням відкритої

архітектури, вона легко модифікується та масштабується під конкретні потреби користувача.

Практична цінність полягає у створенні недорогого, ефективного та простого у впровадженні засобу локального відеоконтролю, що може бути застосований у побуті, малому бізнесі, освітніх та офісних установах.

Ключові слова: FaceRecognition, штучний інтелект, Python, захищеність

ABSTRACT

Gerasimchuk A.A. Increasing the security of smart home system objects based on artificial intelligence for incident analysis. Master's qualification work in specialty 125 - "Cybersecurity and information protection", educational program "Cybersecurity of information technologies and systems". Vinnytsia: VNTU, 2025. - 113 p.
In Ukrainian. Bibliography: 46 titles; fig.: 14; table. 10.

The work presents the development of a software complex for automatic face recognition using the Python programming language and a number of specialized libraries, in particular OpenCV, face_recognition and Flask. The system is designed for real-time video monitoring, recording events that appear in the frame, determining whether they belong to known or unknown persons, as well as administrator communication via the Telegram messenger and a web browser.

One of the key features of the implemented solution is local storage of face databases, support for security mode, an integrated web management interface, event logging, and authorized user access to the administrative panel. The interface is designed in a modern style with an emphasis on simplicity, convenience, and integration with external services. Two-way interaction with the Telegram bot is implemented: the user can receive notifications about events, current photos from the camera, or turn the security system on/off via chat.

The developed solution is completely autonomous, does not require cloud services or external servers, which provides a high level of confidentiality and control over data. The system can be deployed both on a personal computer and on mini-computers such as Raspberry Pi. As a result of the open architecture, it is easily modified and scaled to meet specific user needs.

The practical value arises in creating an inexpensive, effective and easy-to-implement alternative to local video surveillance that can be used in everyday life, small businesses, educational and office institutions.

Keywords: FaceRecognition, artificial intelligence, Python, security

ЗМІСТ

ВСТУП

Багато робіт в літературі вводять визначення «розумного будинку». Розумні будинки можна визначити з соціальної або технічної точки зору. Перший описує вплив «розумного дому» на людські та соціальні потреби, тоді як другий описує системи, процеси, послуги та інтелектуальні пристрої, які підключені для полегшення контролю над екосистемою будинку.

Головним чинником існування розумного будинку є його безпека. Безпека як всередині будинку, так і зовнішня його охорона. В сучасних розумних будинках для забезпечення безпеки використовуються датчики руху, інфрачервоні датчики, камери, датчики відчинення, вібрації, розбиття скла. Весь спектр засобів безпеки можна вимкнути за допомогою смартфона або смарт-брелка.

Рано чи пізно настає момент коли чи того чи іншого не має поряд, або ж банально розрядився акумулятор чи батарейка. Дані події значно впливають на сприйняття «розумного будинку», оскільки власникові доводиться вигадувати різні способи щоб потрапити в власний будинок, тоді він вже не сприймається таким «розумним». Крім того часто під час спроб відчинити власний будинок відбуваються «хибні» спрацювання цих самих датчиків, а інколи і зовсім без ніяких подій: пробігла тварина, підвищилась волога чи температура, повз будинок проїхав об’ємний вантажний автомобіль.

Актуальність. Зростання кількості інцидентів (хибних спрцювань), пов’язаних з несанкціонованим проникненням, крадіжками та загрозами безпеці, вимагає впровадження інтелектуальних технологій, які дозволяють не лише фіксувати події, а й оперативно на них реагувати без втрати часу на хибні спрацювання.

Особливу актуальність має використання систем підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту шляхом розпізнавання обличь — як однієї з найефективніших технологій ідентифікації особи. Інтеграція таких рішень у локальні охоронні комплекси дозволяє суттєво

підвищити ефективність моніторингу, зменшити вплив людського фактору та автоматизувати сповіщення у випадку появи невідомих осіб.

Мета і задачі дослідження. Розробка та впровадження системи підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв), що забезпечується шляхом автоматичного виявлення осіб, їх ідентифікацією та надсиланням сповіщень користувачу для підвищення рівня безпеки контрольованого об’єкта.

Для досягнення цієї мети було поставлено наступні задачі:

1. Аналіз існуючих систем розумного будинку та технологій штучного інтелекту.
2. Розробка архітектури системи, що включає мережу камер, алгоритми обробки та інтерфейс користувача.
3. Реалізація механізму збору та обробки даних від камер у реальному часі.
4. Використання алгоритмів машинного навчання для виявлення незнайомих осіб, аналізу інцидентів то сповіщення про них користувача.
5. Тестування системи у різних сценаріях та її оптимізація.

Об'єктом дослідження є розумні будинки та їхні системи автоматизації і охорони.

Предметом дослідження є використання технологій штучного інтелекту для підвищення захищеності об’єктів системи розумного будинку для аналізу інцидентів(сценаріїв)» .

Наукова новизна. Запропоновано новий підхід до використання алгоритмів штучного інтелекту для аналізу інцидентів у розумному будинку. Розроблено систему, що здатна аналізувати та навчатися на основі історичних, а також введених механічно. Використання методів машинного навчання для виявлення дійсних порушень периметру розумного будинку, водночас запровадження системи доступу до розумного будинку.

Практична цінність. Покращення захищеності об’єктів системи розумного будинку на основі штучного інтелекту шляхом автоматизованого моніторингу та реагування на потенційні загрози. Зниження кількості помилкових тривог та підвищення точності виявлення людей, та визначення їх статусу згідно БД.

1. АНАЛІЗ МОЖЛИВИХ ЗАГРОЗ ДЛЯ РОЗУМНОГО БУДИНКУ

Нові ІКТ-технології в середовищі розумного дому спрямовані на полегшення повсякденних завдань, таких як дистанційне керування функціями будинку та керування споживанням енергії. Еталонну архітектуру можна використовувати як шаблон для розробки конкретної архітектурної топології такого середовища, оскільки вона забезпечує спільну основу, навколо якої можна розробити більш детальну архітектуру.[4] Така модель здатна краще представляти такі аспекти, як люди-користувачі, реалізації пристроїв і серверні структури для більш детального уявлення про досліджуване середовище. [3] Для екосистеми розумного будинку було запропоновано кілька еталонних архітектур. Ghirardello та ін. [1] запропонував еталонну архітектуру розумного будинку, проаналізувавши три точки зору екосистеми:

- функціональний;
- фізичний;
- комунікаційний (зв’язковий).

Зокрема, функціональна точка зору складається з необхідних функцій, які повинні підтримуватися для нормальної роботи розумного будинку. Фізична точка зору описує всі фізичні компоненти, необхідні для виконання функцій розумного будинку. Точка зору зв'язку містить протоколи, необхідні для передачі потоків керування та інформації між компонентами. Ми вирішили базувати наш підхід до аналізу загроз на цій еталонній архітектурі, оскільки вона пропонує відповідний рівень деталізації для цієї мети. Зокрема, він забезпечує хороший баланс між рівнями абстракції та інформацією, необхідною для проведення оцінки ризику. Крім того, ключовою характеристикою цієї еталонної архітектури є класифікація точок зору Smart Home, яка полегшує ідентифікацію інформаційних, фізичних і комунікаційних активів. У цій роботі ми використовуємо цю еталонну модель для проведення аналізу загроз, як першого кроку до комплексної оцінки ризиків для розумних будинків.

1.1. Методи аналізу загроз у динамічних середовищах

Аналіз загроз – це твердження про загрози, які пов’язані з уразливістю активів і агентів загроз [5]. Таким чином, аналіз загроз є частиною процесу оцінки ризиків [6]. У динамічних середовищах важливо використовувати метод аналізу загроз, який дозволяє розглядати потенційні зміни в цільовому середовищі. У літературі можна виділити два різних підходи до аналізу загрози, а саме перспективу зловмисника та перспективу захисника. Перший є більш складним, у той час як другий ретельно перевіряє цільові системи, беручи до уваги також захисні прийоми. Методологія, яка буде використовуватися, важлива для виявлення всіх вразливостей, загроз і атак в архітектурі розумного будинку. Зосереджуючись на методах, які можуть автоматично ідентифікувати загрози за допомогою допоміжного інструменту, ми вирішили застосувати метод STRIDE [2] (підробка, фальсифікація, відмова, розкриття інформації, відмова в обслуговуванні та підвищення привілеїв) з англ. (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privilege), який підтримується інструментом моделювання загроз Microsoft. Метод був розроблений Конфельдером і Гаргом і використовувався як академічним середовищем, так і промисловістю; це дозволяє отримувати точні результати щодо ризиків, з якими стикаються цільові системи, і може бути застосовано ще на етапі проектування. Загрози STRIDE описав Шостак.[2]

Інструмент моделювання загроз від Microsoft дозволяє ідентифікувати потенційні загрози, спрямовані на потоки даних і серверні служби еталонної моделі Ghirardello [1]. Цей інструмент дозволяє ідентифікувати проблеми безпеки в процесах, сховищах даних і потоках даних, оскільки аналіз проводиться за допомогою DFD. Отже, необхідно створити DFD для екосистеми розумного будинку, кожен з яких відповідає різній топології. Ідентифікація загроз досягається за допомогою таксономії загроз STRIDE. Аналіз відбувається наступним чином:

Крок 1 — Опис сценарію: Опис сценарію має включати всі відповідні елементи в межах досліджуваного середовища. У цьому документі увага

зосереджена на екосистемі розумного будинку, зокрема на потоках даних і серверних службах.

Крок 2 — Ідентифікація активів: активи цільової системи мають бути ідентифіковані. Такі активи включають інформаційні активи та фізичні активи.

Крок 3. Створення DFD. Використовуючи простоту таких діаграм, аналітик може представити пристрої, служби та потоки даних між зазначеними вище активами.

Крок 4. Визначення обмеження для кожної вразливості: кожен із ідентифікованих активів має різні вразливості безпеки, які вже проаналізовано та можна знайти в існуючих базах даних вразливостей.

Крок 5. Визначення загрози: аналітик розробляє різні сценарії атак, враховуючи ідентифіковані активи та їхні взаємозв’язки. Інструмент автоматично визначає загрози, також враховуючи попередньо встановлені обмеження.

Крім того, враховуючи технічну точку зору, яка визначає середовище розумного будинку, ми розробляємо генератор топології мережі розумного будинку на основі існуючої еталонної архітектури. Згодом, щоб отримати знання про вплив динамічних залежностей мережі на розповсюдження зловмисного програмного забезпечення, ми використовуємо результати, надані генератором топології, для створення моделі атаки на основі графів. Модель атаки базується на початковій гіпотезі про те, що в будь-який момент часу кожен вузол можна спостерігати в дискретному стані. потім, через однорідні проміжки часу, він може або переходити в інший стан, або залишатися в поточному стані. Модель передбачає типи зловмисних програм, які використовують або випадкові, або локалізовані методи сканування. Зловмисне програмне забезпечення, яке використовує випадкове сканування, вибирає цільові IP-адреси випадковим чином. Зловмисне програмне забезпечення для локалізованого сканування переважно шукає хости в «локальному» адресному просторі замість випадкового вибору цілей.

1.2. Можливості використання ШІ для захисту розумного будинку

Штучний інтелект значно вплинув на безпеку вдома, допомагаючи створювати розумніші та надзвичайно проактивні системи, які забезпечують безпеку будинків.

- Розпізнавання обличчя для покращеної безпеки

Популярні захисні камери зі штучним інтелектом можуть розрізняти знайомі та незнайомі обличчя. Це можна застосувати для попередження вас, якщо буде виявлено невідомий персонаж, допомагаючи уникнути несанкціонованого доступу.

- Виявлення руху та сповіщення про аномалії

Повністю засновані на штучному інтелекті структури безпеки використовують чудові алгоритми для визначення незвичайних дій або звуків. Наприклад, якщо рух виявлено в незвичайний час або в несподіваному місці, машина може надіслати сповіщення власнику будинку або подати сигнал тривоги.

- Інтеграція зі службами екстреної допомоги

Деякі вищі системи можуть механічно зв’язуватися з надзвичайними пропозиціями, якщо вони натрапляють на небезпеку здібностей, разом із каміном або вторгненням. Ця функція миттєвої реакції може значно покращити захист будинку.

Охоронні структури на основі штучного інтелекту можуть спрощувати захист будинку звідусіль, забезпечуючи активний захист.

- Енергоефективність та сталість

Управління енергією є важливою темою для розумних будинків, і штучний інтелект зможе відігравати важливу роль у тому, щоб зробити будинки надзвичайно міцними, екологічними та екологічними.

- Планове технічне обслуговування приладів та оновлення прошивок

Штучний інтелект може аналізувати інформацію з обладнання розумного будинку, щоб передбачити проблеми з можливостями раніше, ніж вони стануть

фундаментальними проблемами. Наприклад, холодильник на основі штучного інтелекту може сповістити власника будинку, якщо він виявить дивне споживання електроенергії, вказуючи, що йому потрібно технічне обслуговування. В сучасному світі щодня з’являються нові вразливості того чи іншого програмного забезпечення систем розумного будинку, через що виробники регулярно випускають оновлення для підтримання безпеки на належному рівні. Проте, людина не завжди встигає їх проводити, в цьому і може допомогти ШІ.

ШІ революціонізує розумні будинки, перетворюючи їх на помітно персоналізовані, зелені та стабільні житлові зони. Від голосових помічників і автоматизованих процедур до прогнозованого оновлення та керування живленням, штучний інтелект покращує майже всі проблеми сучасного домашнього життя. Незважаючи на те, що виклики, які включають конфіденційність, захист і цінність, залишаються актуальними, безперервні вдосконалення штучного інтелекту та створення розумних домівок обіцяють долю, у якій домівки будуть органічно включені в наше життя, спостерігаючи за нашими бажаннями та сприяючи здоровішому та екологічнішому життю [7].

Оскільки штучний інтелект продовжує розширюватися, розумні будинки стають основою для більшого підключення, комфорту та ефективності.

1.3. Автоматичне виявлення загроз, як ключовий напрям у впровадженні ШІ

Автоматичне виявлення злочинців, навчання та адаптація ШІ в системах безпеки розумного будинку – це ключові напрями вдосконалення даної роботи. ШІ зможе аналізувати осіб що наближаються до будинку та виявляти загрозу в режимі реального часу, якщо така є.

Піднімається рівень фізичної безпеки за допомогою аналізу відеоспостереження для виявлення незнайомих обличь, підозрілих рухів або вторгнень; виявлення підозрілої поведінки біля входу (наприклад, якщо людина кілька разів підходить до дверей, але не заходить); інтеграція з датчиками для миттєвого реагування (закриття дверей, активація сирени).

Щоб система безпеки ставала розумнішою, вона повинна навчатися на нових даних, тому буде впроваджено машинне навчання для розпізнавання загроз, а саме постійне оновлення алгоритмів на основі поведінкових шаблонів мешканців; використання технології «аутентифікація за поведінкою» (якщо користувач зазвичай заходить о 18:00, а одного разу система фіксує спробу входу о 3:00 ночі – ШІ може активувати додатковий рівень перевірки).

Крім того обов’язковою умовою в впровадженні ШІ є адаптація до змін середовища це і вивчення типових рухів біля дому, щоб не спрацьовувати на домашніх тварин, і автоматичне регулювання чутливості камер залежно від часу доби чи пори року, і аналіз умов навколо будинку (наприклад, якщо виявлено багато руху вночі, система може збільшити частоту перевірки камер) [11].

Особливу роль у безпеці системи відіграє регулярне оновлення паролів доступу, а також програмного забезпечення датчиків, камер і тд., що включає в себе в використання глобальної бази даних атак (наприклад, оновлення інформації про нові методи злому) та автоматичне оновлення ПЗ безпеки та патчів для усунення вразливостей - дані обов’язки також будуть покладені на ШІ.

1.4. Аналіз існуючих рішень розумного будинку з інтегрованим ШІ

У процесі вивчення та аналізу предметної області було проведено аналіз існуючих робіт, які пов’язані з вирішенням питання моніторингу безпеки розумного будинку. Нижче наведено результати, отримані під час ознайомлення з різноманітними наявними рішеннями.

На сьогодні на ринку представлені різні системи розумного будинку, які використовують технології штучного інтелекту для автоматизації та безпеки. Серед найпоширеніших рішень можна виокремити:

1. **Google Nest** – система розумного будинку, що використовує машинне навчання для аналізу даних з сенсорів і адаптації до поведінки

користувачів. Nest Cam і Nest Protect забезпечують моніторинг безпеки та оповіщення у разі виявлення загроз.

2. **Amazon Alexa та Ring** – екосистема пристроїв, що дозволяє інтегрувати різні розумні пристрої у єдину систему. Використовує голосові команди та аналіз даних для автоматичного керування освітленням, температурою та безпекою.
3. **Apple HomeKit** – платформа, що дозволяє підключати розумні пристрої та контролювати їх за допомогою Siri. Використовує алгоритми штучного інтелекту для автоматизації сценаріїв.
4. **ADT Pulse та Vivint Smart Home** – комерційні системи безпеки, що використовують розширений аналіз загроз і автоматичне реагування, інтегруючи відеоспостереження та сенсори.
5. **Xiaomi Mi Home** – бюджетне рішення, яке включає інтелектуальні сенсори, камери та пристрої для автоматизації, що аналізують умови довкілля та активність користувача.

Порівняно з існуючими аналогами засіб що розробляється в даній роботі буде позбавлений мінусів [12], які присутні в існуючих рішеннях, а саме :

- Доступно не у всіх регіонах, деякі функції потребують підписки (Наприклад Google Nest Secure);

- Загроза приватності (Amazon збирає дані), пристуня вилика кількість хибних спрацьовувань (Наприклад Ring Alarm (Amazon));

- Дороге обслуговування, потребує підписки(Наприклад Deep Sentinel);

- Висока вартість, потребує технічної підтримки.

На сьогоднішній день є доступними для вільного придбання та використання багато «розумних» побутових пристроїв та систем, але у більшості вони характеризуються низьким рівнем комплексної інтеграції у системи «розумного» будинку, так як призначені для вирішення однієї конкретної задачі (визначення одного параметра середовища, керування окремим приладом, тощо) без можливості передачі інформації і взаємодії з іншими приладами. Однак, було реалізовано невелику кількість спроб розробити системи «розумного» будинку,

рівень автоматизації процесів яких відповідав би рівню виконання аналогічних завдань людиною [20, 21].

Як ми бачимо з Рисунка 1, який відображає інтерес впродовж проміжку часу на основі сервісу Google Trends, на якому можна побачити, що розробка систем «розумного» будинку - це галузь, яка демонструє стабільне зростання.

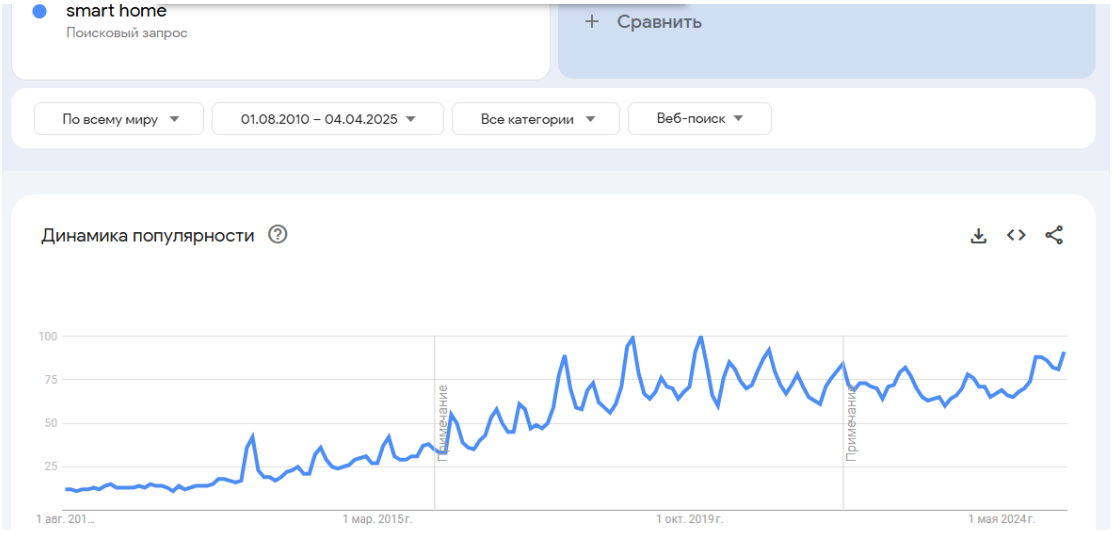


Рис.1 Графік популярності пошукових запитів з 2010 року по т.ч.

Технології AI, системи «розумних» будинків та користувачі мають різні моделі взаємодії, які поділяються на три. Перша модель, коли користувачі безпосередньо дають команди для кожного «розумного» побутового пристрою, а штучний інтелект вбудований у кожен окремий пристрій. Такий підхід приносить користь тільки окремому конкретному пристрою. «Розумне» управління будинком, охорона здоров'я та питання безпеки надають перевагу цьому підходу. Друга модель, коли користувачі задають інструкції AI, а AI самостійно контролює кожен окремий пристрій. Управління побутовими приладами у системах «розумного» будинку переважно працює за цією схемою. В ідеалі, система повинна забезпечувати постійний процес навчання з урахуванням можливостей зміни поведінки користувачів. В рамках аналізу були досліджені методи та моделі, що використовуються у системах «розумного» будинку для розпізнавання діяльності та аналізу поведінки користувачів системи.

Було досліджено розроблену мультиагентну систему для відстеження поведінки користувачів для динамічної зміни логіки роботи, результат роботи

якої продемонстрував успішну ідентифікацію мешканців на протязі різних тривалостей моніторингу. Логіка функціонування «розумного» будинку була розроблена на основі адаптації до змін поведінки користувачів. Система здатна адаптуватися до змін людини у виявлених ситуаціях, і може автоматично оновити свою модель відповідно до нових змін поведінки користувачів. Метод нечіткої логіки використовується для вирішення проблем невизначеності і працює подібним до міркування людини чином, де використовуються неповні або неточні дані. Нечітка логіка керується механізмом логічного висновку, що слідує умові IF-THEN, які визначають залежності між нечіткими входами і виходами системи.

Приклад правил IF-THEN наведено у проаналізованій роботі, де розглянуто розроблену систему адаптивного нечіткого управління для «розумних» будинків. У сценарії в якості тесту була використана система освітлення, де прикладом умови є наступне правило: «ЯКЩО людина присутня і РІВЕНЬ зовнішнього освітлення ТЕМНИЙ, ТОДІ потужність освітлення МАКСИМУМ». Також, використання нечіткої логіки було використано в мультисенсорному середовищі для системи «розумного» будинку у модулі моніторингу охорони здоров’я. Система складається з мультимодульної платформи з камерами встановленими по всьому будинку для збору даних. Далі дані обробляються та аналізуються за допомогою нечіткої логіки, що дозволяє гнучко змінювати модульність або додавати більше камер та датчиків.

Перевага використання нечіткої логіки полягає в тому, що вона ефективна для приблизних міркувань, проте їй бракує здібності до навчання та адаптаційних можливостей. Великою перевагою машинного навчання є те, що воно усуває необхідність у зусиллях з програмування, дозволяючи комп’ютеру вчитися на досвіді. Машинне навчання за допомогою методу опорних векторів основане на навчанні для класифікації об’єктів на основі прикладів з навчальної вибірки. Метод опорних векторів у системі «розумного» будинку використаний для моніторингу (за допомогою камер) активності людей похилого віку в будинку, з метою раннього виявлення втрати ними дієздатності. Дані, зібрані камерами, пізніше були проаналізовані за допомогою методу опорних векторів

для класифікації семи видів активності у повсякденному житті, тобто гігієни, прийому їжі, сну тощо. Експеримент був протестований з 13 людьми молодого віку для того, щоб знайти різні моделі активності, а потім перевірити правильність класифікації методом опорних векторів, використовуючи реальні дані.[14]

1.5. Висновки та постановка задачі

В рамках даного розділу було проведено дослідження існуючих рішень розумного будинку з інтегрованим штучним інтелектом, а також виявлено їхні основні недоліки.

На основі отриманих даних було сформульовано ключові задачі, які необхідно вирішити в рамках даного проєкту:

1. Розробити систему, що дозволить ефективно збирати, обробляти та аналізувати дані в режимі реального часу.
2. Запровадити алгоритми машинного навчання, здатні виявляти незнайомців, власників та відповідно діяти.
3. Інтегрувати систему в існуючі пристрої розумного будинку для підвищення її функціональності та зручності використання.
4. Оптимізувати механізми взаємодії користувача із системою для забезпечення інтуїтивного керування.
5. Забезпечити резервний метод доступу до системи.
6. Провести тестування та валідацію запропонованої системи в умовах реального використання.

Запропоновані рішення сприятимуть створенню ефективної системи розумного будинку, що забезпечить високий рівень безпеки, адаптивність до змін та зручність використання для кінцевого користувача.

2. ПРОЕКТУВАННЯ СИСТЕМИ РОЗУМНОГО БУДИНКУ НА ОСНОВІ ШТУЧНОГО ІНТЕЛЕКТУ ДЛЯ АНАЛІЗУ ІНЦИДЕНТІВ

В даному розділі буде спроектовано систему розумного будинку на основі штучного інтелекту для аналізу інцидентів, що базується на інтеграції технологій відстежування рухів і погляду користувача із моделями контролювання дій та впровадження сценаріїв за допомогою ШІ. Розглянуто особливості вдосконалення існуючих систем розумного будинку на основі штучного інтелекту для аналізу інцидентів.

Описано особливості застосування технологій відслідковування рухів і обличчя в контексті моніторингу дій користувача, розроблено алгоритм роботи вдосконаленої системи розумного будинку, а також спроектовано модель автоматизованого прийняття рішень на основі аналізу поведінкових характеристик. У висновках сформульовано результати досліджень та визначено основні підходи до реалізації системи.

2.1. Розробка архітектури системи для обробки та аналізу даних у режимі реального часу

Щоб спроектувати ефективну та загальну архітектуру розумного будинку, великі дані, отримані з існуючого розумного будинку, повинні бути ретельно досліджені та проаналізовані. Процес збору даних можна здійснити, розмістивши камери та датчики в різних місцях у розумному домі. Офлайн-обробка великих даних може допомогти у проектуванні та плануванні середовища будинку.

Вся архітектура складається з різних частин, таких як блок збору даних (брокер Інтернету речей) і агент Інтернету речей (репозиторій для зберігання даних), модуль обробки великих даних і сервер зв’язку моделі будинку, який забезпечує засоби зв’язку із зовнішнім об’єктом. Дані з різних програм збираються та передаються на сервер моделі будинку. Сервер моделі будинку обробляє дані та передає їх брокеру IoT. Посередник IoT розділяє дані на основі ідентифікаторів датчиків і призначає даним номер індексу. Нарешті, IoT-брокер

надсилає дані агенту IoT для подальшої обробки. Запропонована схема забезпечує більш високу пропускну здатність обробки даних.

Підсумовуючи вище згадане розроблена система складатиметься з таких основних компонентів:

1. **Мережа камер**:

    - Камери відеоспостереження з функціями розпізнавання обличь та об'єктів.
    - Розумні замки та дверні датчики.

2. **Обробка та аналіз даних**:

    - Локальний шлюз для обробки даних.
    - Алгоритми машинного навчання для ідентифікації особистостей та потенційних загроз.
    - Використання моделей глибокого навчання для аналізу відео потоків у реальному часі.

3. **Керування та взаємодія**:

    - Телеграм бот та веб-браузер для керування системою та отримання повідомлень.
    - Автоматизовані сценарії реакції на інциденти (увімкнення сигналізації, виклик екстрених служб).

4. **Безпека та захист даних**:

    - Шифрування даних при передачі та зберіганні.
    - Аутентифікація користувачів та контроль доступу.
    - Виявлення несанкціонованого доступу та аномальних дій.

Ефективність роботи алгоритму штучної нейронної мережі залежить від її типу та внутрішньої структури, відповідно до задач різної специфіки. Наприклад,

для розв'язання задач розпізнавання образів на зображеннях та відео потоці ефективно використовувати згорткові нейронні мережі, а для перекладу текстів, або програмування ботів, що ведуть діалоги з людиною оптимальним вибором є рекурентні нейронні мережі. Тому, постає питання вибору оптимальної структури штучної нейронної мережі, яка буде використана для автоматизації керування температурним режимом, системою безпеки та освітленням у будинку. Початковим етапом вибору структури штучної нейронної мережі є створення списку відомих типів штучних нейронних мереж. До такого списку у розглянутому прикладі відносяться: Нейронна мережа прямого поширення; Рекурентна нейронна мережа; Довга короткочасна пам'ять; Вентильний рекурентний вузол. Наступним кроком методу є генерування навчальної вибірки, що використовуватиметься для навчання штучних нейронних мереж розглянутих типів. Навчальна вибірка генерується шляхом відкидання п'ятьох записів, що утворюють тестову вибірку, яка буде використовуватись в подальшому для розрахунку помилки роботи алгоритму штучної нейронної мережі в порівнянні з очікуваними даними.

Наступним етапом методу є послідовне навчання та використання кожного з типів штучної нейронної мережі та аналізу отриманих результатів. Вибравши черговий тип штучної нейронної мережі проводиться процес навчання за допомогою створеної навчальної вибірки. Далі відбувається запуск навченої штучної нейронної мережі з подачею на вхід параметрів тестової вибірки та збереженням отриманих вихідних результатів. Для кожного вихідного параметру розраховується значення похибки за допомогою порівняння отриманого значення відповідного параметра тестової вибірки. Серед отриманих значень похибок кожного вихідного параметра розраховується середнє значення, що є загальним значенням похибки використання конкретного типу штучної нейронної мережі для розв’язання поставлених задач. Після розрахунку середнього значення похибки, це значення порівнюється з найменшим отриманим середнім значенням похибки попередньо розглянутих типів штучної нейронної мережі. У разі якщо дане значення є меншим за попереднє - поточна

штучна нейронна мережа зберігається як найоптимальніша для розв'язання поставленої задачі, а середнє значення похибки зберігається як найоптимальніше на конкретний момент дослідження. Процес навчання, запуску та аналізу вихідних результатів роботи штучної нейронної мережі послідовно проводиться для кожного типу штучної нейронної мережі зі списку, поки не буде проаналізовано всі типи.

Традиційні алгоритми навчання нейронної мережі (наприклад, алгоритм зворотного поширення помилки) основані на знаходженні найкращих коефіцієнтів зв'язків цієї мережі при її фіксованій структурі. Проблема вибору відповідної структури нейронної мережі при такому підході лежить на плечах розробника і багато в чому 83 визначає наперед успішність побудови моделі. Крім цього, сама по собі проблема підбору структури мережі є вкрай складною. Для спрощення побудови нейронних мереж на даний час існує багато методів і алгоритмів.

Всі вони можуть бути розділені на дві групи: методи та алгоритми нарощування структури мережі і методи та алгоритми спрощення структури мережі. Методи спрощення структури мережі є більш формалізованими, проте вони пов'язані зі свідомо великими витратами ресурсів на навчання початкових структур нейронних мереж. Алгоритми нарощування структури мережі, як правило, спираються на емпіричні дані про поліпшення ступеня навчання нейронної мережі при нарощуванні її структури. Існують алгоритми нарощування структури нейронної мережі, засновані на одночасному обмеженні числа прихованих шарів мережі та послідовному заповненні прихованих шарів мережі нейронами до стану насичення, тобто такого, коли при додаванні нового нейрона в даний шар, помилка роботи навченої нейронної мережі практично перестає зменшуватися. У таких алгоритмах відсутня жорстка фіксація числа прихованих шарів, однак їх кількість можна обмежити двома внутрішніми шарами з огляду на те, що теоретично доведена достатність одного внутрішнього шару, а також існує твердження про оптимальну кількість прихованих шарів, що дорівнює двом.

2.2. Розробка алгоритму роботи системи розумного будинку на основі штучного інтелекту для аналізу інцидентів.

Алгоритми машинного навчання відіграють вирішальну роль у функціональності пристроїв розумного будинку. Ці алгоритми аналізують дані, зібрані з різних камер і взаємодії користувача, щоб приймати обґрунтовані рішення. Наприклад, розумна камера може виявити незнайому особу на території будинку та відправити екстрене сповіщення, забезпечуючи комфорт та безпеку.

Нижче описано орієнтовний алгоритм роботи майбутньої системи з урахуванням всіх мінусів існуючих аналогів:

1. **Активація системи**. Користувач налаштовує сценарії: вдома, поза домом, режим сну.
2. **Збір даних**. На цьому етапі налаштовано камери та сенсори для збору даних у реальному часі. Це основа для подальшого аналізу інцидентів.
3. **Пристрої та сенсори:**

- **Камери спостереження**: з можливістю розпізнавання осіб, руху, аномалій.
- **Датчики руху**: для виявлення переміщень у приміщенні, визначення незвичайної активності.
- **Датчики відкриття дверей та вікон**: для виявлення несанкціонованого доступу.

Дані з усіх сенсорів постійно оновлюються, що дозволяє отримувати повну картину стану будинку. Важливо забезпечити швидку передачу даних до центральної системи для їх аналізу.

4. **Ідентифікація та класифікація інцидентів**

Алгоритм ШІ повинен аналізувати зібрані дані та класифікувати інциденти на основі виявлених аномалій.

Обробка сигналів:

- **Фільтрація шуму**: Існує необхідність в обробці даних для виключення фальшивих спрацьовувань. Наприклад, сенсори можуть інколи

спрацьовувати через нормальні зміни (наприклад, рухи тварин або зміна температури, що не є критичною).

- **Аналіз аномалій**: Алгоритм використовує методи машинного навчання для виявлення аномальних подій. Наприклад, якщо двері відкриваються вночі, це може бути сигналом вторгнення.

Головним завданням системи є виявлення вторгнень та ідентифікація осіб що наближаються. Для даного інциденту потрібно визначити порогові значення та сценарії їх розвитку. Це також вимагає наявності бази історичних даних, на основі яких система може навчатися.

5. **Аналіз та прогнозування сценаріїв**

Система повинна бути здатною не лише реагувати на інцидент, а й передбачати розвиток подій.

Прогнозування розвитку подій:

- **Оцінка ймовірності**: На основі аналізу даних ШІ може розрахувати ймовірність того, що поточна ситуація призведе до серйозного інциденту. Наприклад, якщо температура в приміщенні поступово підвищується, ймовірність того, що це призведе до пожежі, зростає.
- **Моделювання розвитку інцидентів**: За допомогою алгоритмів прогнозування (наприклад, моделі часових рядів) можна оцінювати, як буде змінюватися ситуація протягом часу.

6. **Реакція системи**

Після ідентифікації інциденту ШІ має визначити, які дії слід виконати.

Автоматичні дії:

- **Управління пристроями**: На основі сценарію система може автоматично впливати на інші пристрої. Наприклад, увімкнути сигналізацію, зачинити всі двері, викликати екстренні служби.

- **Інтеграція з іншими системами**: Система може бути інтегрована з іншими сервісами та датчиками, це обмежується лише фантазією власника.

Сповіщення користувачів:

- **Повідомлення через телеграм**: Алгоритм має передбачити сповіщення користувача через мобільний додаток «Телеграм» з детальною інформацією про інцидент.
- **Інтерактивні відслідковування в реальному часі у веб браузері**: У деяких випадках користувач може бути запитаний про те, чи хоче він, щоб система здійснила певну дію (наприклад, викликати службу безпеки).

7. **Навчання та покращення**

Для того щоб система працювала ефективно в довгостроковій перспективі, важливо постійно покращувати її на основі нових даних.

Постійне навчання:

- **Безнаглядне навчання**: Використання методів безнаглядного навчання дозволяє системі постійно навчатися на основі нових даних і виявляти нові патерни.
- **Адаптація до змін**: Система повинна адаптуватися до змін у навколишньому середовищі, наприклад, зміни в умовах освітлення або звуків.

Оцінка результатів:

- **Моніторинг ефективності**: Для оцінки точності роботи алгоритмів можна використовувати різні метрики, такі як точність, чутливість, специфічність, час реакції.

8. **Безпека та конфіденційність**

Не менш важливим аспектом є безпека системи.

Захист даних:

- **Шифрування**: Всі зібрані дані, особливо персональні та чутливі, повинні бути зашифровані для захисту від несанкціонованого доступу.
- **Аутентифікація**: Важливо забезпечити безпечну аутентифікацію користувачів через надійні методи.

Захист від хакерських атак:

- Система повинна бути захищена від хакерських атак, тому важливо регулярно оновлювати програмне забезпечення і проводити аудит безпеки.

У більш детальному вигляді алгоритм роботи системи виглядає наступним чином. Після зміни стану камери у приміщеннях помешкання відбувається надсилання даного значення від камери до клієнтської частини системи. Модуль клієнтської частини, що відповідає за аналого-цифрове перетворення значень датчиків перетворює значення згенерованих сенсором у цифровий вигляд. На основі отриманого значення запускається процес прийняття рішення зміни стану налаштування системи за допомогою використання моделі штучної нейронної мережі або внутрішніх алгоритмів “якщо-тоді”. Потреба у застосуванні алгоритмів штучного інтелекту визначається складністю задачі. Для легких завдань, таких як встановлення чи зняття з охорони, оновлення ПЗ можуть застосовуватися внутрішні звичні алгоритми “якщо-тоді”, але для складних задач таких як аналіз обличчя, що вимагає опрацювання багатьох масивів даних є потреба у застосуванні алгоритмів штучного інтелекту. Після калькулювання змін стану за допомогою моделей штучної нейронної мережі, відбувається зміна налаштувань систем будинку відповідно до нового стану системи. Для покращення методів застосування алгоритмів штучної нейронної мережі, виникнена подія та стан системи надсилаються на сервер провайдера. Якщо виникнена подія є унікальною, дана подія записується у базу даних навчальної вибірки. Після запису події відбувається процес перенавчання моделі штучної нейронної мережі на основі оновленої навчальної вибірки. Після завершення процесу навчання нова актуальна модель штучної нейронної мережі

надсилається клієнтській частині для збереження та подальшого використання під час автоматизованого процесу визначення змін стану налаштувань системи.

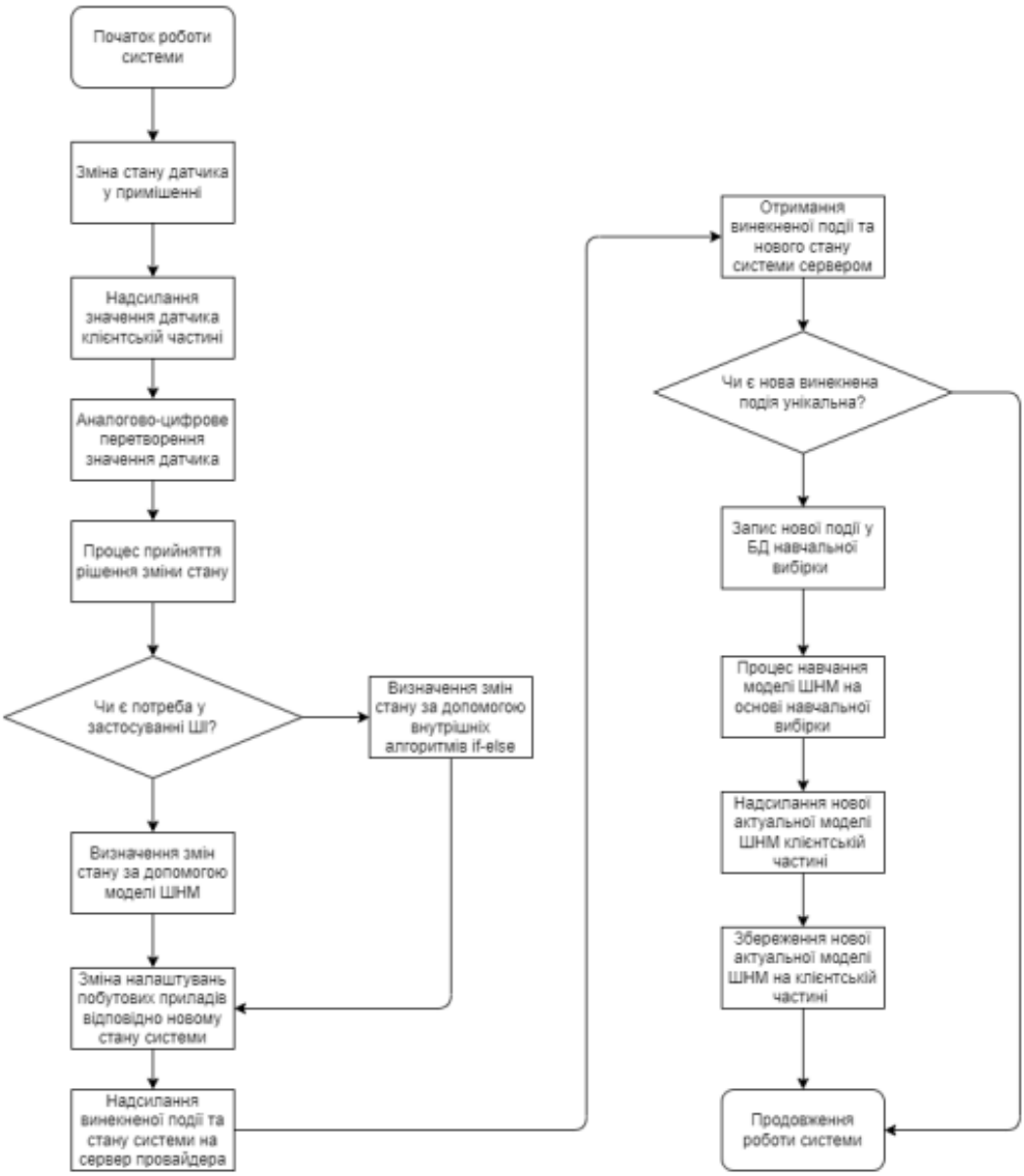


Рис. 2. Алгоритм роботи системи

2.3. Особливості використання ШІ в системах розумного будинку

ШІ перетворює розумний дім за допомогою передових інтелектуальних технологій. ШІ не тільки розуміє команди та реагує належним чином, але також враховує людські вподобання та смаки, приймаючи зручні для людей рішення.

Розумні будинки з підтримкою ШІ дуже корисні для розумних будинків. Такі розумні помічники, як Google Assistant, Siri від Apple і Alexa від Amazon, революціонізують побутову техніку та системи безпеки. Вони розуміють голосові команди та виконують завдання, вони можуть автоматизувати операції, аналізуючи переваги користувача.

ШІ може думати та приймати рішення відповідно до команд і ситуацій. Одним із основних способів використання ШІ в системах розумного дому є використання віртуальних помічників. Ці помічники, такі як Alexa від Amazon і Google Assistant, здатні розуміти голосові команди та відповідати на них, дозволяючи користувачам керувати своїми розумними домашніми пристроями, просто розмовляючи з ними.

AI також використовується для підвищення продуктивності пристроїв розумного дому. Наприклад, розумні термостати можуть використовувати алгоритми машинного навчання, щоб з часом дізнаватися про вподобання користувача щодо опалення та охолодження, що дозволяє їм автоматично регулювати температуру в домі для максимального комфорту. Подібним чином розумні камери безпеки можуть використовувати штучний інтелект для ідентифікації та попередження власників будинків про потенційні загрози, такі як зловмисники або підозрілі дії.

Окрім покращення функціональності пристроїв розумного дому, AI також використовується, щоб зробити ці системи більш зручними для користувачів. Наприклад, деякі системи розумного дому тепер включають функції, які дозволяють користувачам створювати власну автоматизацію та процедури на основі їхніх конкретних потреб і вподобань. Ці автоматизації можуть бути викликані різними факторами, такими як час доби, погода або присутність певних людей у домі. Загалом інтеграція штучного інтелекту в системи

розумного дому допомагає зробити ці системи більш розумними, ефективними та зручними.

Через велику кількість зібраних даних про життя користувачів, систему потрібно захищати від несанкціонованого доступу. Шифрування даних, двофакторна аутентифікація та інші заходи безпеки — необхідні елементи. Розумний будинок працює з особистими даними користувачів (наприклад, інформацією про їхні звички, розклад тощо), тому важливо забезпечити їх конфіденційність і відповідність вимогам законодавства (GDPR у Європі, CCPA у Каліфорнії).

Голосові асистенти (Siri, Alexa, Google Assistant) є важливим елементом взаємодії з користувачем. Вони мають бути інтегровані з іншими пристроями та мати високу точність розпізнавання мови, щоб забезпечити коректне виконання команд. Розумний будинок повинен мати зручний мобільний додаток для керування системою. Це дає користувачеві можливість віддалено моніторити та керувати різними аспектами будинку, такими як освітлення, температура, безпека.

Система повинна бути масштабованою, щоб згодом можна було додавати нові пристрої, сенсори чи функціональності без необхідності в переробці всього проекту. Наприклад, ви можете почати з кількох пристроїв і поступово додавати нові функції або інтегрувати нові сервіси.

Кожен користувач має свої вподобання та вимоги до системи. Розробка має забезпечити можливість гнучкого налаштування системи відповідно до індивідуальних потреб. Розумний будинок зазвичай складається з пристроїв від різних виробників. Важливо, щоб система могла інтегруватися з різними протоколами (Wi-Fi, Zigbee, Z-Wave, Bluetooth, та ін.) та працювати з різними брендами. Система повинна бути здатною взаємодіяти з іншими пристроями IoT, такими як розумні годинники, освітлювальні прилади, термостати, побутова техніка тощо, забезпечуючи їх інтеграцію в єдину систему.

Система повинна бути протестована в реальних умовах для виявлення можливих недоліків або помилок у роботі. Це може включати перевірку всіх функцій у різних сценаріях (наприклад, виявлення пожежі, порушення безпеки). Оскільки система розумного будинку обробляє великий обсяг даних в реальному часі, важливо оптимізувати продуктивність, щоб уникнути затримок або перебоїв у роботі. Технології швидко змінюються, тому важливо розробляти систему з можливістю для легких оновлень, щоб вона могла інтегрувати нові технології без необхідності повної перебудови.

Оскільки більшість користувачів не є технічними експертами, інтерфейс системи повинен бути простим у використанні. Важливо, щоб користувач міг налаштовувати й керувати системою без складних інструкцій. Розробка сповіщень для користувачів має бути зручною й зрозумілою, щоб користувач міг швидко реагувати на події (наприклад, аварії чи спроби вторгнення).

2.4.Проектування та розробка моделей для аналізу інцидентів

У цьому проєкті використовується алгоритм індивідуальних облич з використанням бібліотеки opencv для розпізнавання облич. Скрипт може захопити зображення, яке перетворюється на зображення у шкалі сірого, потім застосувати підхід знайомих облич та порівняти обличчя, а потім обрізати зображення в N^2 вимірах. Щоб зменшити кількість зображень, зображення облич перетворюються у двовимірний масив з восьмибітними значеннями інтенсивності, з точним вилученням ознак алгоритм обчислює загальне зображення обличчя (у режимі шкали сірого для додаткового зменшення даних) та віднімає результуючий вектор з кожного вектора індивідуальних облич, щоб остаточно отримати основний вектор, щоб когось помітити, ця стандартизація ожного вектора називається аналізом головних компонентів і допомагає знайти відповідний вектор з найменшим інтервалом часу, мета аналізу головних компонент полягає у формуванні лінійної комбінації вектора, яка оптимально представляє вхідне зображення з меншим обсягом пам'яті.

База даних для змінного освітлення, розрізнення та умов вираження зберігається і, отже, зберігається. На цьому етапі система оцінюється з урахуванням багатьох обмежень, таких як зміни освітлення, обертання обличчя та зміна масштабу, навіть з цими змінами алгоритм вилучення ознак видобуває середнє зображення та власну грань.

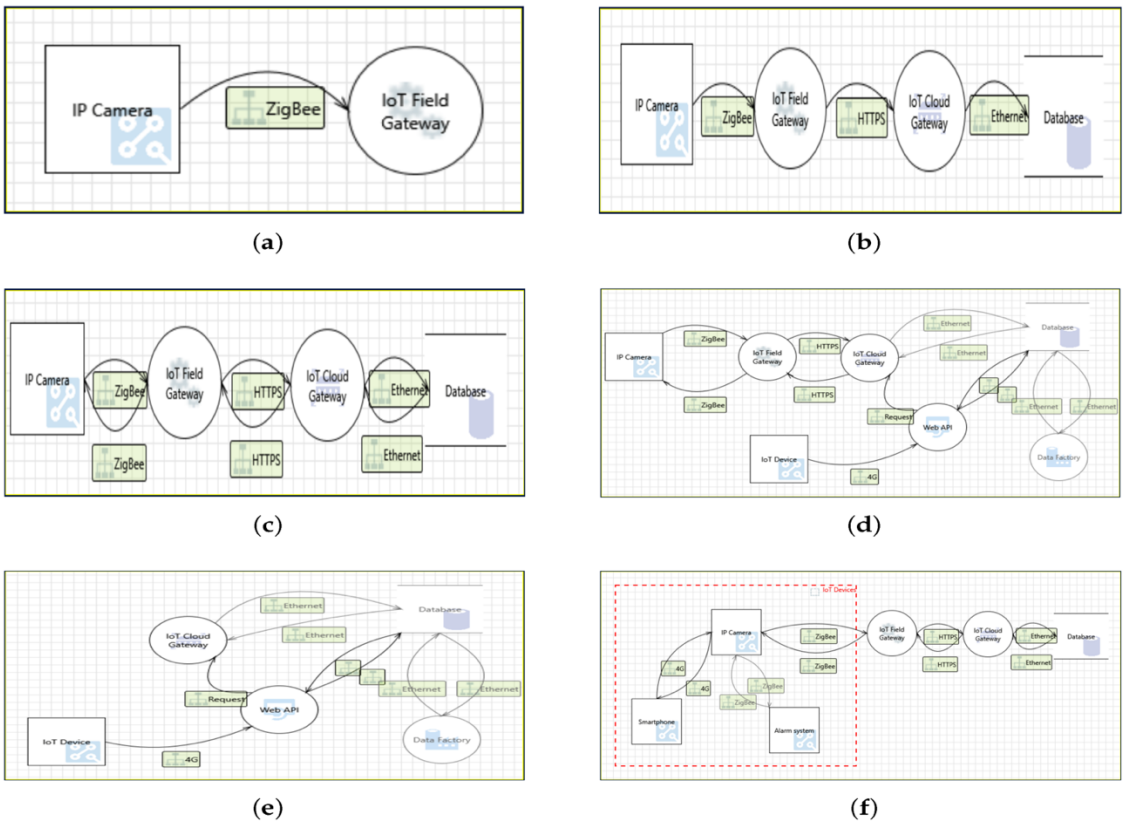
Для розробки моделі аналізу інцидентів необхідно розібрати принцип роботи самої екосистеми розумного будинку, один з таких випадків буде описано нижче.

Екосистема розумного будинку включає в себе кілька активів, залежно від точки зору. У цій роботі увага зосереджується на фізичних і комунікаційних точках зору, таким чином ми прагнемо ідентифікувати інформацію та фізичні активи. Вони зображені в таблиці 1.

Інформаційні активи	Фізичні активи
Облікові дані користувача	Розумні пристрої Інтернету речей
Інформація, зібрана розумними пристроями	IoT-центр
Інформація про стан розумного будинку	Шлюзи IoT
Інформація про встановлені активи	Датчики/Актуатори
Інформація журналу	Хмарний сервер
Відео, зображення, голосова інформація	
Інформація про відстеження місцезнаходження	
Особиста інформація (наприклад, дані про здоров’я)	

Табл. 1 Ідентифікація активів.

На основі ідентифікованих активів і різних сценаріїв зв’язку між пристроями та серверними службами розроблено шість окремих сценаріїв «Розумного дому», описаних у відповідних DFD, що представляють шість топологій різної складності, щоб наблизитися до динамічної природи цільового середовища. Вони описані нижче, а відповідні DFD показані на малюнку 1:



Мал. 3 Розроблені сценарії [13]. (a) IP-камера та шлюз IoT. (b) Односпрямований зв’язок між IP-камерою та хмарою. (c) Двонаправлений зв’язок між IP-камерою та хмарою. (d) IP-камера, керована смартфоном. (e) Зв’язок смартфона з хмарою. (f) Посилання між розумними пристроями.

Сценарій 1. IP-камера та шлюз IoT: перший сценарій представляє з’єднання між IP-камерою та шлюзом, як показано на малюнку 1a. Для зв’язку використовується протокол ZigBee, і наш аналіз зосереджується на загрозах, які можуть завдати шкоди фізичним активам або інформації, що передається між пристроями. Припускаємо, що IP-камера підключена безпосередньо до шлюзу.

Сценарій 2 — односпрямований зв’язок між IP-камерою та хмарою: ця топологія представляє з’єднання між IP-камерою та хмарним сервером через два шлюзи. Зв'язок встановлюється за допомогою трьох різних протоколів, як показано на малюнку 1b. У цьому випадку IP-камера надсилає лише запит до бази даних. Перший підключається безпосередньо до шлюзу через протокол ZigBee, а другий є простою базою даних, яка використовує MySQL 2016.

Сценарій 3. Двонаправлений зв’язок між IP-камерою та хмарою: третя топологія схожа на другу, але тепер пристрій IoT спілкується з хмарою двонаправлено. Використовувані протоколи зв'язку зображені на малюнку 1c.

Сценарій 4 — IP-камера, керована смартфоном: складніша топологія представлена в наступному сценарії, зображеному на малюнку 1d. Ця топологія описує зв’язок між керованою зі смартфона IP-камерою та хмарою. Пристрій IoT (смартфон) надсилає запити до хмарного API, щоб контролювати IP-камеру через стільниковий зв’язок.

Сценарій 5. Зв’язок смартфона з хмарою: наступна топологія представляє зв’язок між смартфоном і хмарою, як показано на малюнку 1e. У цій топології ми прагнемо визначити потенційні загрози, які можуть спровокувати пошкодження запитів на керування.

Сценарій 6. Зв’язки між інтелектуальними пристроями: остання топологія націлена лише на інтелектуальні пристрої (IP-камера, система сигналізації та смартфон) і спрямована на виявлення потенційних загроз, які походять від паралельних зв’язків між цими пристроями. Зокрема, IP-камера здатна спілкуватися за допомогою 4G і протоколу ZigBee. Топологія на малюнку 1f зображує взаємодію IP-камери зі смартфоном і системою сигналізації.

Застосовуючи метод STRIDE до цих сценаріїв, детально визначено відповідні загрози. Кількість загроз, виявлених у кожному з шести сценаріїв, зображено в таблиці 2. Можна зробити наступні висновки про те, як динамічний характер середовища розумного будинку впливає на виявлені загрози:

Сценарії	Ідентифіковані загрози
Сценарій 1	13
Сценарій 2	27
Сценарій 3	46
Сценарій 4	48
Сценарій 5	23
Сценарій 6	16

Табл. 2 Ідентифіковані загрози по кожному сценарію

Зі збільшенням складності топології виявляється більше загроз безпеці. У двох останніх сценаріях було виявлено менше загроз, оскільки аналіз зосереджувався на зв’язку між смартфоном і хмарою та на ідентифікації загроз, які походять від паралельних з’єднань між певними пристроями.

Складніші топології успадковують загрози, з якими стикаються простіші.

Пристрої IoT, такі як IP-камери та інтелектуальні пристрої, збільшують поверхню атаки розумного будинку. Зокрема, зловмисник може ефективніше запускати атаки на підвищення привілеїв, використовуючи вразливості IP-камери та її протоколів зв’язку, зокрема ZigBee.

Пристрої з транзитивними або паралельними з’єднаннями, такі як IP-камера, більш уразливі до кібератак, оскільки вони успадковують уразливості безпеки кожного без винятку протоколу.

Для покращення системи та запобігання вказаним вразловостям буде використано штучний інтелект, який крім іншого зможе запускати та аналізувати сценарії у розмному будинку.

2.5. Висновок до розділу

У другому розділі було здійснено проектування системи розумного будинку з інтегрованим ШІ, що інтегрує модуль контролювання дій користувача з використанням технологій відстежування обличчя. Основна мета розробки полягала у створенні механізму автоматизованого прийняття рішень, здатного забезпечити безпеку та зручність у користуванні будинком, особливо в умовах дистанційного тестування.

Проаналізовані результати продемонстрували важливість адаптації системи розумного будинку до індивідуальних особливостей користувачів, що дозволяє мінімізувати хибні спрацьовування та враховувати природні поведінкові особливості кожного користувача. Інтеграція технологій штучного інтелекту дозволяє фіксувати навіть незначні відхилення від нормальної поведінки, такі як аномальні сценарії ввімкнення світла, відчинення дверей, тощо, що значно підвищує можливості виявлення потенційних спроб взлому, проникнення, нещасних випадків у режимі реального часу.

Розробка системи розумного будинку з використанням штучного інтелекту — це складний процес, що вимагає інтеграції різних технологій, оптимізації роботи системи та забезпечення високого рівня безпеки й конфіденційності. Необхідно враховувати всі аспекти, від зручності використання до можливості самонавчання та адаптації до змін у поведінці користувачів і навколишньому середовищі.

Штучний інтелект дозволяє суттєво підвищити безпеку розумного будинку, мінімізуючи ризики вторгнення. Використання алгоритмів машинного навчання та глибокого аналізу інцидентів сприяє оперативному виявленню загроз та адаптивному реагуванню на нові виклики в сфері безпеки. Запропоновані методи дозволяють створити ефективну систему захисту, здатну адаптуватися до нових загроз і забезпечувати безпеку користувачів та їхніх пристроїв.

Наукова новизна цього підходу полягає у впровадженні адаптивних алгоритмів, які дозволяють системі не лише уникати «хибних» спрацювань систем безпеки, а й за допомогою аналізу обличь ідентифікувати власника та запускати відповідні системи. Гібридне використання у проекті методів машинного та глибокого навчання допомагає поєднувати різні моделі аналізу загроз для більш точної детекції проникнень, а система автоматизованого прийняття рішень у реальному часі не лише виявляє загрозу, але й приймає відповідні заходи без участі людини. Алгоритми самонавчання аналізують попередні інциденти та коригують свої моделі для підвищення ефективності розпізнавання загроз. Використання децентралізованих методів аналізу зменшує залежності від центрального сервера для підвищення стійкості системи до атак. ШІ може значно покращити аналіз та прогнозування загроз завдяки таким підходам, які не використовувались раніше в жодній доступній на сьогоднішній день системі. Сукупність цих характеристик робить проект унікальним в цій сфері.

Очікуваним результатом роботи є підвищення рівня захищеності об'єктів системи розумного будинку за рахунок своєчасного виявлення та ефективного реагування на інциденти на основі інтелектуальних систем.

3 ПРОГРАМНА РЕАЛІЗАЦІЯ ПІДВИЩЕННЯ ЗАХИЩЕННОСТІ ОБ'ЄКТІВ СИСТЕМИ РОЗУМНОГО БУДИНКУ

Цей розділ описує програмну реалізацію Підвищення захищеності об'єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв). Основна увага приділяється вибору бібліотек, інструментів розробки, вибору мови програмування, реалізації ключових модулів системи та їх об'єднання в єдиний проект.

Розділ складається з обґрунтування вибору мови програмування та середовища розробки, детальної реалізації принципу дії підвищення захищеності розумного будинку, використання відслідковування та розпізнавання обличчя у проекті, а також створення графічного інтерфейсу, що забезпечує зручність роботи з системою. Крім того, проведено тестування реалізованої системи для перевірки її коректності та ефективності, а у висновках підведено підсумки щодо програмної реалізації та її відповідності поставленим задачам.

3.1 Обґрунтування вибору мови програмування

Для реалізації підвищення захищеності об'єктів системи розумного будинку на основі штучного інтелекту було обрано мову програмування Python. Він є найкращим вибором для побудови моделей машинного навчання завдяки простоті використання, великій бібліотеці фреймворків, гнучкості та що особливо важливо в даному проекті це інтеграція з технологіями комп'ютерного зору та машинного навчання.

Він надає виняткову потужність та універсальність середовищам машинного навчання . Простий синтаксис мови спрощує перевірку даних та оптимізує процеси парсингу, обробки, уточнення, очищення, упорядкування та аналізу, тим самим зменшуючи перешкоди для співпраці з іншими програмістами. Python також пропонує величезну екосистему бібліотек, які позбавляють розробників більшої частини монотонних рутинних завдань з

написання функцій, дозволяючи їм зосередитися на коді та зменшуючи ймовірність помилок під час програмування.

Машинне навчання охоплює машини, що імітують дії людини в різних сферах, таких як медична діагностика, фінансові послуги, прогнозна аналітика, розпізнавання зображень, розпізнавання мовлення та навіть статистичний арбітраж. Це стало можливим завдяки бібліотекам та фреймворкам машинного навчання PyTorch у Python, які включають два найвідоміші інструменти для кластеризації та вибору моделей — TensorFlow та scikit-learn.
TensorFlow зосереджений на спрощенні створення моделей машинного навчання для настільних комп'ютерів, мобільних пристроїв, веб-сайтів та хмарних платформ як для початківців, так і для досвідчених розробників. Sci-kit-learn, також відома як sklearn, — це бібліотека машинного навчання на Python, розроблена для роботи з науковими та числовими бібліотеками Python, SciPy та NumPy.

Python є найближчим кандидатом на звання найкращої мови програмування для штучного інтелекту[15].

Для реалізації функцій комп’ютерного зору та розпізнавання обличчя в роботі ключовими є бібліотеки FaceRecognition - OpenCV, Mediapipe та Dlib.

FaceRecognition – це відмінна бібліотека для розпізнавання обличчя. Експерименти, описані в [16], показали, що навіть немовлята віком від одного до трьох днів здатні розрізняти відомі обличчя. Девід Хубель і Торстен Візель показали , що наш мозок має спеціалізовані нервові клітини, що реагують на певні локальні ознаки сцени, такі як лінії, краї, кути або рух. Оскільки ми не бачимо світ як розрізнені фрагменти, наша зорова кора повинна якимось чином поєднувати різні джерела інформації в корисні шаблони. Автоматичне розпізнавання облич полягає у вилученні цих значущих ознак із зображення, поєднанні їх у корисне представлення та виконанні певної класифікації. Розпізнавання обличчя на основі геометричних ознак обличчя, ймовірно, є найбільш інтуїтивним підходом до розпізнавання обличчя. Саме через ці особливості було обрано бібліотеку FaceRecognition.

OpenCV (Open Source Computer Vision Library) — це бібліотека програмних функцій, головним чином для комп'ютерного зору в реальному часі . Спочатку розроблена Intel , пізніше підтримувалася Willow Garage , а потім Itseez (яку пізніше придбала Intel [17]). Бібліотека є крос платформною та ліцензована як безкоштовне програмне забезпечення з відкритим вихідним кодом за ліцензією Apache License 2, що дозволяю працювати з нею абсолютно безкоштовно без обмеження її функцій. Починаючи з 2011 року, OpenCV пропонує прискорення на графічному процесорі для операцій у реальному часі.

Наразі OpenCV надає доступ до таких алгоритмів:

- Власні обличчя (див. EigenFaceRecognizer::create)
- Fisherfaces (див. FisherFaceRecognizer::create)
- Гістограми локальних бінарних шаблонів (див. LBPHFaceRecognizer::create)

Dlib в свою чергу це сучасний інструментарій на C++, що містить алгоритми машинного навчання та інструменти для створення складного програмного забезпечення на C++ (а завдяки гнучкості мови Python і на ній теж) для вирішення реальних проблем. Він використовується як у промисловості, так і в академічних колах у широкому спектрі галузей, включаючи робототехніку, вбудовані пристрої, мобільні телефони та великі високопродуктивні обчислювальні середовища. Ліцензія Dlib з відкритим кодом дозволяє використовувати його в будь-якій програмі безкоштовно, через що його було обрано.

MediaPipe надає набір бібліотек та інструментів для швидкого застосування методів штучного інтелекту (ШІ) та машинного навчання (МН) у додатках. Ці рішення одразу інтегруються в додаток, налаштуваються відповідно до своїх потреб та використовуються на різних платформах розробки.

Для пришвидшення роботи проекту необхідно було використати бібліотеку, яка змогла б працювати з великою кількості масивів даних, такою стала NumPy. Прив'язки Python широко використовуваної бібліотеки комп'ютерного зору OpenCV використовують масиви NumPy для зберігання та обробки даних. Оскільки зображення з кількома каналами просто представлені як тривимірні масиви, індексація, нарізання або маскування за допомогою інших масивів є дуже ефективними способами доступу до певних пікселів зображення. Масив NumPy як універсальна структура даних в OpenCV для зображень, вилучених точок ознак , ядер фільтрів та багато іншого значно спрощує робочий процес програмування та налагодження.

Python широко визнаний як найпрестижніша, а також найадаптивніша мова програмування штучного інтелекту, яка сягає 70% її використання в проектах штучного інтелекту та машинного навчання . Нещодавнє опитування[18] показало, що Python продовжує домінувати в програмуванні штучного інтелекту завдяки своїм великим бібліотекам. Однак такі мови, як R, Java та C++, також роблять успіхи в певних галузях, таких як статистичний аналіз та програми, критично важливі для продуктивності.

Проте порівняно з іншими мовами у штучному інтелекті Python забезпечує найкращий функціонал, тому що в нього висока читабельність та просте використання - код на Python часто вважається дуже читабельним через його схожість з природною мовою, яка є приємнішою для ока під час кодування та дає відносно швидші результати; багата екосистема бібліотек та фреймворків ШІ - Python має нескінченну кількість дещо спеціалізованих бібліотек та фреймворків для конкретних завдань ШІ. Відомі приклади включають вище згадані TensorFlow, PyTorch, а також scikit-learn як набори інструментів із попередньо визначеними функціями для машинного навчання, глибокого навчання та аналізу даних.

Ці бібліотеки допомагають розробникам уникнути необхідності створювати або проектувати складні повторювані концепції, коли вони можуть

покладатися на попередньо встановлені системи; швидке прототипування: Python легко реалізувати, що дійсно може зробити його перевагою для прототипування ШІ, оскільки його використання дозволяє розробникам швидко переходити від одного підходу до іншого. Це важливо в сучасному застосуванні рішень ШІ, оскільки воно вимагає постійних змін через високу конкуренцію.

Коли йдеться про необхідність швидкого створення моделі, простоту коду та широкий вибір бібліотек штучного інтелекту, Python неперевершений. Завдяки простоті використання та мовному потоку, він ідеально підходить як для новачків, так і для досвідчених розробників.

Хоча, у випадках, коли виконується багато обчислень, можна виявити, що його швидкість не найкраща порівняно з іншими мовами програмування, такими як R.

3.2 Обґрунтування вибору середовища розробки

При розробці системи Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв) було використано середовище розробки Visual Studio (рис 3.2), що було розроблено компанією Microsoft. Вона поєднує простоту редактора вихідного коду з потужними інструментами для розробників, такими як автодоповнення та налагодження коду IntelliSense.

Перш за все, це редактор, який не заважає. Дозволяє безперешкодно обробляти цикл редагування-збірки-налагодження, що в свою чергу означає менше часу на маніпуляції з середовищем та більше часу на реалізацію ідеї.

Рисунок 3.2 – Visual Studio

Оскільки бібліотеки Dlib та OpenCV є досить вибагливими в їх використанні під час розробки застосовувались платформи з Windows, MacOS та Linux. Завдяки Visual Studio перехід між відбуватись швидко та без помилок, оскільки дане середовище є доступним на цих OS.

В основі Visual Studio лежить швидкий редактор вихідного коду, який є ідеальним для щоденного використання. Завдяки підтримці сотень мов, VS допомагає миттєво досягти продуктивності завдяки підсвічуванню синтаксису, зіставленню дужок, автоматичному відступу, вибору блоків, фрагментам коду тощо. Інтуїтивно зрозумілі комбінації клавіш, просте налаштування та зіставлення комбінацій клавіш, розроблені спільнотою, дозволяють легко орієнтуватися в коді.

Для серйозного кодування часто застосовувались інструменти з кращим розумінням коду, ніж просто блоки тексту. Visual Studio включає вбудовану підтримку автодоповнення коду IntelliSense, багатого семантичного розуміння та навігації кодом, а також ре факторингу коду, що значно пришвидшує роботи з ним.

Під час складного кодування, найбільш потрібною та незамінною функцією є Debugging. Він часто є тією функцією, якої розробникам найбільше не вистачає в більш спрощеному кодуванні. Visual Studio включає інтерактивний Debugging, завдяки якому є можливість крок за кроком переглядати вихідний код, перевіряти змінні, переглядати стеки викликів та виконувати команди в консолі.

Також він є досить гнучким до користувача, кожну функцію можна під лаштувати та встановити будь-яку кількість сторонніх розширень. Хоча

більшість сценаріїв працюють «з коробки» без налаштування, VS також розвивається і є можливість додавати власні функції відповідно до унікальних потреб. VS– це проект з відкритим кодом, тому можна зробити свій внесок у зростаючу та динамічну спільноту на GitHub.

Так як основний інтерфейс розроблявся за допомогою HTML, CSS, а в VS включено розширену вбудовану підтримку розробки для веб-технологій, таких як JSX/React, HTML, CSS, SCSS, Less та JSON, це значно спростило розробку за допомогою однієї платформи.

3.3 Реалізація модуля доступу до системи (автентифікація)

При створенні системи розумного будинку вона повинна бути захищена не тільки фізично, як от система розпізнавання лиця, датчики рухів, камери і т .д., а і програмно, тобто щоб доступ мали лише авторизовані користувачі. Тому було розроблено систему автентифікації при вході в програму.

```
import os
import sys
from flask import Flask, request, session, redirect, url_for, render_template
from functools import wraps
from werkzeug.security import generate_password_hash, check_password_hash


def login_required(f):
    @wraps(f)
    def decorated_function(*args, **kwargs):
        if not session.get('logged_in'):
            return redirect(url_for('login'))
        return f(*args, **kwargs)
    return decorated_function

VALID_USERNAME = os.getenv("VALID_USERNAME", "admin")
VALID_PASSWORD = os.getenv("VALID_PASSWORD", "1234")
hashed_password = generate_password_hash("1234")
if username == VALID_USERNAME and check_password_hash(hashed_password, password):
    session['logged_in'] = True
@app.route('/login', methods=['GET', 'POST'])
def login():
    error = None
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        if username == VALID_USERNAME and password == VALID_PASSWORD:
            session['logged_in'] = True
            return redirect(url_for('index'))
        else:
            error = "❌ Невірний логін або пароль"
    return render_template('login.html', error=error)
```

За допомогою декоратора *login_required* обмежуємо доступ до певних маршрутів (сторінок) лише для авторизованих користувачів. Якщо користувач не ввійшов у систему (**'logged_in'** не встановлено в сесії), його переадресує на сторінку входу.

Вводимо змінні VALID_PASSWORD, VALID_USERNAME для вказання правильного паролю. Щоб підвищити безпеку системи паролі хешуємо за допомогою *werkzeug.security.* Якщо хешування не застосовувати це призведе до низької захищеності перед атаками хакерів, оскільки пароль в такому разі буде зберігатись в середині коду.

При вході запит POST — перевіряє логін і пароль, якщо вони правильні — зберігає *logged_in = True* у сесії та перенаправляє на головну сторінку, проте якщо ні — показує помилку.

3.4 Реалізація модуля генерації відео потоку з розпізнаванням обличь

Для початку роботи модуля за алгоритмом повідомляємо функції *generate_frames* в якому стані знаходиться система в змінній *is_armed,* яка присвоюється спочатку за допомогою інтерфейсу користувача, тобто якщо «під охороною» модуль працює, якщо ні, проходить далі по алгоритму.

```
def generate_frames():
  global is_armed
```

За допомогою безкінечного циклу функція постійно зчитує кадри з камери та аналізує їх.

```
  while True:
    if not initialize_camera():
```

initialize_camera(): - перевіряє чи камера готова до зчитування. Якщо ні — переходить до наступної ітерації. Система налагоджена таким чином щоб при тестування та використанні максимально спростити пошук несправностей або помилок які виникатимуть в процесі використання.

```
      continue
    with camera_lock:
```

При роботі з програмою було виявлено що при запиті на камеру з іншого потоку (від іншої програми) система переходила в статус помилки через що було використано функцію *camera_lock.*

```
    success, frame = camera.read()
if not success:
    logger.error("Failed to read frame from camera")
    continue
try:
    if is_armed:
        small_frame = cv2.resize(frame, (0, 0), fx=0.75, fy=0.75)
        rgb_small_frame = small_frame[:, :, ::-1]
        face_locations = face_recognition.face_locations(rgb_small_frame)
```

За допомогою цієї функції (*face_locations*) здійснюється основна мета проекту- тобто пошук обличь на відео в реальному часі. Вона шукає обличчя на кадрі. Якщо знайдені — переходить до кодування (*face_encodings*), тобто отримання числового опису обличчя.

```
        if face_locations:
            face_encodings = safe_face_encodings(rgb_small_frame)
            if not face_encodings:
                logger.warning("Обличчя виявлено, але кодування не отримано.")
```

Дана частина коду була введена під час тестування, оскільки при поганій якості камери функція не могла розпізнати обличчя, тобто вона його фіксувала отримувала *face_locations*, але кодування *face_encodings*: отримати не могла через погану якість камери. Отож для майбутнього користувача було введено дане повідмлення: «Обличчя виявлено, але кодування не отримано».

```
            else:
                for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
                    matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
                    name = "Unknown"
                    face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
                if len(face_distances) > 0:
```

Розпізнавання обличчя відбувається з отриманих кодувань за допомогою змінних, а саме *matches* — список булевих значень, чи співпадає обличчя з якимось у базі, або ж воно є незнайомим і потрібно подати сигнал тривоги; *face_distances* — числові відстані між обличчями (чим менше, тим схожіше); *np.argmin(...)* — знаходить найкраще співпадіння.

```
                    best_match_index = np.argmin(face_distances)
                    if matches[best_match_index]:
                        name = known_face_names[best_match_index]
```

```
            log_event(name)
            if name == "Unknown":
                trigger_alarm()
```

Якщо не вдалося впізнати обличчя — вмикає сигналізацію, повідомляє користувача (або виконує інші захисні дії які можна доповнювати).

```
            top *= 2
            right *= 2
            bottom *= 2
            left *= 2
            color = (0, 255, 0) if name != "Unknown" else (0, 0, 255)
            cv2.rectangle(frame, (left, top), (right, bottom), color, 2)
            cv2.putText(frame, name, (left, top - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
            cv2.putText(frame, name, (left, top - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)
        ret, buffer = cv2.imencode('.jpg', frame)
        frame = buffer.tobytes()
```

yield повертає закодований у JPEG кадр у форматі, який підтримує потокове відео.

```
            yield (b'--frame\r\nContent-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
        except Exception as e:
            logger.error(f"Error in video stream: {e}")
            continue
```

3.5 Реалізація та впровадження функції знайомих обличь

Для того щоб система розуміла кого пропускати, а кого ні, було створено папку з фото для порівняння.

```
def load_known_faces():
    if not os.path.exists("faces"):
        os.makedirs("faces")
```

Ця частина коду перевіряє чи папка faces існує і якщо ні, створює її.

```
    known_face_encodings.clear()
    known_face_names.clear()
```

Перед завантаженням видаляє старі данні для уникнення дублювання.

```
    for filename in os.listdir("faces"):
        if filename.endswith(".jpg") or filename.endswith(".png"):
```

Шукає всі зображення формату .jpg або .png. у папці.

```
            filepath = os.path.join("faces", filename)
            image = face_recognition.load_image_file(filepath)
            encodings = face_recognition.face_encodings(image)
```

Завантажує зображення та отримує кодування облич.

```
            if encodings:
```

```
        known_face_encodings.append(encodings[0])
        known_face_names.append(os.path.splitext(filename)[0])
```

Зберігає перше знайдене обличчя і його ім’я (назву файлу), тобто якщо файл має назву olena.jpg, то в список імен потрапить "olena".

```
load_known_faces()
```

3.6 Реалізація модуля оповіщення про тривогу

Для повноцінної роботи системи обов’язково потрібно розробити систему оповіщення про тривогу, тобто про порушення периметру будинку незнайомою особою. Для цього в рамках виконання дипломної роботи було використано стандартний звук Windows та оповіщення за допомогою телеграм. При виконання в комерційних масштабах замість winsound буде використано сирену для відлякування злодіїв та привертання уваги. Ця дія була зроблена навмисно для зменшення вартості розробки проекту.

```
import winsound

def send_telegram_alert(message):
  try:
    logger.info(f"Sending Telegram alert: {message}")
    response = requests.post(f"{TELEGRAM_API}/sendMessage", data={
      "chat_id": TELEGRAM_CHAT_ID,
      "text": message
    })
    if response.status_code != 200:
      logger.error(f"Telegram API error: {response.text}")
  except Exception as e:
    logger.error(f"Telegram send failed: {e}")
```

Функція send_telegram_alert(message) відправляє текстове повідомлення у Telegram чат через бот API. TELEGRAM_API — це URL типу: https://api.telegram.org/bot<TOKEN>, який є індивідуально створеним під цей проект. Щоб сповіщення надходило від бота саме потрібній особі було застосовано змінну TELEGRAM_CHAT_ID, при застосуванні якої сповіщення надходить потрібній особі. Якщо щось піде не так, логує помилку, для подальшого її виправлення і спрощення її пошуків.

```
def trigger_alarm():
  global last_alarm_time
  now = time.time()
  if now - last_alarm_time < 10:
```

Не дозволяє тривозі спрацьовувати частіше ніж один раз на 10 секунд аби уникнути спаму в повідомленнях телеграм та прибрати зайву навантаження на компоненти системи.

```
    return
  last_alarm_time = now
  def beep():
    try:
      winsound.Beep(1000, 1000)
```

Видає писк частотою 1000 Гц, тривалістю 1000 мс (1 секунда) та запускається в окремому потоці, щоб не блокувати відеопотік, оскільки при запуску в одному потоці при поданні сигналу під час тестувань відео зникало. Важливо зауважити що winsound працює тільки на **Windows** для Linux або кросплатформенно можна використовувати інші бібліотеки, як playsound або pygame.

```
    except:
      logger.error("Beep failed")
  threading.Thread(target=beep).start()
  send_telegram_alert("🚨 Виявлено невідоме обличчя!")
```

Функція *trigger_alarm():* є основною функцією, яка: перевіряє, коли востаннє була тривога**;** видає звуковий сигнал та надсилає сповіщення у Telegram.

3.7 Реалізація модуля реєстрації подій.

Для підвищення аудиту дій та безпеки було додану систему логування входу та виходу в HTML-інтерфейс, а також подій та помилок що були зафіксовані при роботі програми, що пришвидшує та полегшує їх виявлення та виправлення за необхідності.

```
def logout():
  session.pop('logged_in', None)
  return redirect(url_for('login'))
```

```
logging.basicConfig(
```

```
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    handlers=[
        logging.FileHandler('security_app.log', encoding='utf-8'),
        logging.StreamHandler(sys.stdout)
    ]
)
logger = logging.getLogger(__name__)
```

Код записує логи у два місця - файл security_app.log (з підтримкою кирилиці завдяки utf-8) та у **консоль** (термінал, через sys.stdout), де можна переглядати журнал подій або логів.

Для підвищення безпеки видаляє ключ **'logged_in'** з сесії**,** тобто користувач більше не авторизований, якщо сесію було завершено та знову переадресовує на сторінку входу (/login). Формат логів має такий вигляд: дата, рівень, повідомлення.

```
def log_event(name):
    with open(event_log_path, "a") as f:
        f.write(f"{datetime.datetime.now()} - {name}\n")
```

Функція log_event(name) реалізує запис подій у текстовий файл, який виступає у ролі історії подій. Відкриває файл, шлях до якого вказано в event_log_path ("events.log"), у режимі додавання ("a"). Кожен рядок — це нова подія: дата + ім'я особи (або Unknown). Викликається в коді після кожного розпізнавання обличчя з міркувань безпеки та аналізу післядії.

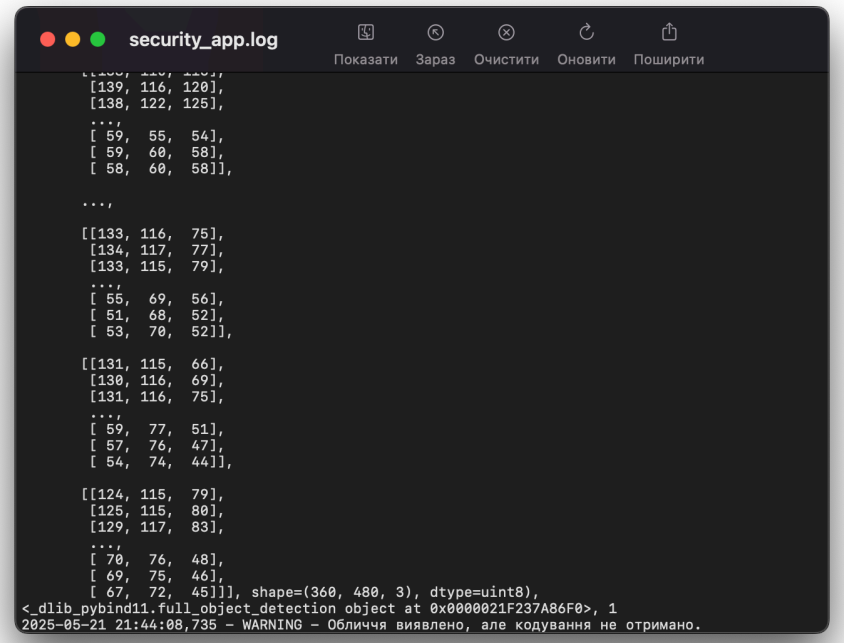


Рисунок 3.1 – Приклад логування під час роботи додатку

3.8 Реалізація основного графічного інтерфейсу користувача

Знайомство користувача з системою починається із сторінки входу що була описана у пункті 3.3, графічний інтерфейс якої реалізовано за допомогою HTML в наступному вигляді:

```
<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <title>Вхід до системи</title>
  <style>
    body {
      font-family: 'Segoe UI', sans-serif;
      background-color: #ecf0f1;
      text-align: center;
      padding-top: 60px;
    }
    form {
      display: inline-block;
      background-color: white;
      padding: 30px;
      border-radius: 10px;
      box-shadow: 0 0 10px #bbb;
```

```
            }
            input[type="text"], input[type="password"] {
                display: block;
                margin: 15px auto;
                padding: 10px;
                width: 250px;
                font-size: 16px;
            }
            button {
                background-color: #2980b9;
                color: white;
                border: none;
                padding: 10px 25px;
                font-size: 16px;
                border-radius: 5px;
                cursor: pointer;
            }
            button:hover {
                background-color: #1c5980;
            }
            .error {
                color: red;
                margin-top: 15px;
            }
        </style>
    </head>
    <body>

    <h2>🔐 Вхід до системи</h2>

    <form method="POST">
        <input type="text" name="username" placeholder="Ім’я користувача" required>
        <input type="password" name="password" placeholder="Пароль" required>
        <button type="submit">Увійти</button>
```

В цій частині вводиться два текстових поля «Ім’я користувача» та «Пароль», а також кнопка «Увійти» за допомогою якої відбувається перехід на основну сторінку index.html, якщо пароль введено вірно.

```
        {% if error %}
            <div class="error">{{ error }}</div>
        {% endif %}
    </form>
```

Якщо в шаблон передано змінну `error` і вона не пуста → виводиться текст помилки, вміст `{{ error }}` — це сам текст який виводиться з додатку Flask.

```
</body>
</html>
```

Далі користувач переходить на основну сторінку це index.html, яка дозволяє користувачеві легко взаємодіяти із системою та керувати нею. Код сторінки виглядає наступним чином:

```
<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <title>Система Безпеки</title>
   <style>
      body {
         font-family: 'Segoe UI', sans-serif;
         background-color: #f3f4f6;
         text-align: center;
         margin: 0;
         padding: 0;
      }
      header {
         background-color: #2c3e50;
         color: white;
         padding: 20px;
         font-size: 24px;
      }
      .status {
         margin-top: 20px;
         font-size: 20px;
      }
      .armed {
         color: green;
         font-weight: bold;
      }
      .disarmed {
         color: red;
         font-weight: bold;
      }
      .video-container {
         margin-top: 30px;
```

```
        }
        .video-container img {
            border: 4px solid #ccc;
            border-radius: 10px;
            box-shadow: 0 0 10px #999;
        }
        .button-group {
            margin-top: 40px;
        }
        .button-group a button {
            background-color: #3498db;
            color: white;
            border: none;
            border-radius: 5px;
            padding: 12px 25px;
            font-size: 16px;
            cursor: pointer;
            margin: 10px;
            transition: background-color 0.3s ease;
        }
        .button-group a button:hover {
            background-color: #2980b9;
        }
    </style>
```

Задаємо фірмовий стиль, який буде схожим у всіх сторінках HTML, це зроблено для впізнаваності додатку.

```
</head>
<body>

<header>🔐 Система Розпізнавання Облич</header>

<div class="status">
    Стан системи:
    {% if is_armed %}
        <span class="armed">ОХОРОНА УВІМКНЕНА</span>
    {% else %}
        <span class="disarmed">ОХОРОНА ВИМКНЕНА</span>
    {% endif %}
</div>

<div class="video-container">
    <img src="{{ url_for('video') }}" width="720" height="540">
</div>
```

Виведення відео з камери в режимі реального часу для можливості спостерігати за тим що відбувається на вулиці власноруч, для перевірки реакції штучного інтелекту.

```
<div class="button-group">
  <a href="{{ url_for('toggle_security') }}"><button>🔁 Увімк./Вимк. охорону</button></a>
  <a href="{{ url_for('add_face') }}"><button>➕ Додати обличчя</button></a>
  <a href="{{ url_for('show_log') }}"><button>📜 Журнал подій</button></a>
</div>
```

Додаємо відповідні кнопки для керування системою ці ж самі кнопку будуть продубльованими через команди у телеграм боті.

```
</body>
</html>
```

Взаємодія Flask додатку з html відбувається за допомогою наступного коду який є повноцінним модулем охоронної системи із відеопотоком, додаванням нових облич, журналом подій:

```
@app.route('/video')
def video():
  return Response(generate_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')
```

На цьому етапі створюється MJPEG-потік для трансляції відео з камери в браузері. Дана функція працює з *generate_frames(),* яка обробляє розпізнавання облич.

```
@app.route('/add_face', methods=['GET', 'POST'])
def add_face():
  if request.method == 'POST':
    name = request.form['name']
    file = request.files['image']
    filepath = os.path.join('faces', f"{name}.jpg")
    file.save(filepath)
    load_known_faces()
    return redirect(url_for('index'))
  return render_template('add_face.html')
```

За допомогою браузера є можливість додавати лиця знайомих, осіб що потрібно пропустити без тривоги, вона реалізується за допомогою цієї частини коду. В якій через HTML-форму приймає зображення й ім’я, після чого зберігає в *faces/,* наприклад: *faces/ivan.jpg*. Після додавання викликає функцію *load_known_faces()* для оновлення бази.

Код HTML форми для додавання обличь з якою контактує користувач виглядає наступним чином:

```
<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <title>Додати обличчя</title>
   <style>
      body {
         font-family: 'Segoe UI', sans-serif;
         background-color: #f9f9f9;
         margin: 0;
         padding: 0;
         text-align: center;
      }
      header {
         background-color: #2c3e50;
         color: white;
         padding: 20px;
         font-size: 24px;
      }
      form {
         margin-top: 50px;
         background: white;
         display: inline-block;
         padding: 30px;
         border-radius: 10px;
         box-shadow: 0 0 10px #ccc;
      }
      input[type="text"],
      input[type="file"] {
         display: block;
         margin: 15px auto;
         padding: 10px;
         font-size: 16px;
         width: 250px;
         border: 1px solid #ccc;
```

```
            border-radius: 5px;
        }
        button {
            background-color: #27ae60;
            color: white;
            padding: 10px 20px;
            font-size: 16px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
        button:hover {
            background-color: #219150;
        }
        a {
            display: block;
            margin-top: 20px;
            color: #2980b9;
            text-decoration: none;
        }
    </style>

</head>
<body>

<header>➕ Додати нове обличчя</header>

<form method="POST" enctype="multipart/form-data">
    <label for="name">Ім’я:</label>
    <input type="text" name="name" required>

    <label for="image">Фото обличчя (.jpg або .png):</label>
    <input type="file" name="image" accept="image/*" required>
```

Вводимо поля форми: це ім’я: — текстове поле (name="name") та фото: — файл (name="image", приймає .jpg, .png тощо).

```
    <button type="submit">Зберегти</button>
</form>
```

При натисканні кнопки «Зберегти» файл автоматично зберігається в папку Faces та завантажується в систему.

```
<div style="margin-top: 30px;">
    <a href="{{ url_for('index') }}">
        <button style="background-color: #34495e; color: white; padding: 10px 20px; font-size: 16px; border: none; border-radius: 5px; cursor: pointer;">
            ← Назад на головну
        </button>
    </a>
</div>


</body>
</html>
```

```
@app.route('/toggle_security')
def toggle_security():
    global is_armed
    is_armed = not is_armed
    return redirect(url_for('index'))
```

toggle_security перемикає змінну is_armed і повертає на головну сторінку, тобто index.html.

```
@app.route('/log')
def show_log():
    if os.path.exists(event_log_path):
        with open(event_log_path, 'r') as f:
            log_lines = f.readlines()
    else:
        log_lines = ["Журнал подій порожній."]
    return render_template('log.html', log_lines=log_lines)
```

Відображає файл журналу подій *(events.log)*, якщо він порожній видає повідомлення "Журнал подій порожній." у шаблоні log.html, який в свою чергу виглядає наступним чином:

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Журнал подій</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
```

```
            background-color: #f4f6f8;
            margin: 0;
            padding: 0;
            text-align: center;
        }
        header {
            background-color: #2c3e50;
            color: white;
            padding: 20px;
            font-size: 24px;
        }
        .log-container {
            background-color: white;
            margin: 30px auto;
            padding: 20px;
            width: 80%;
            max-width: 800px;
            border-radius: 10px;
            box-shadow: 0 0 10px #ccc;
            text-align: left;
        }
        .log-entry {
            font-size: 15px;
            padding: 5px 0;
            border-bottom: 1px solid #eee;
        }
        .log-entry:last-child {
            border-bottom: none;
        }
        .back-button {
            margin-top: 20px;
        }
        .back-button a button {
            background-color: #34495e;
            color: white;
            padding: 10px 25px;
            font-size: 16px;
            border: none;
            border-radius: 5px;
            cursor: pointer;
        }
        .back-button a button:hover {
            background-color: #2c3e50;
        }
    </style>
</head>
```

```
<body>

<header>📜 Журнал подій</header>

<div class="log-container">
    {% for line in log_lines %}
        <div class="log-entry">{{ line }}</div>
    {% endfor %}
</div>
```

Проходить по списку рядків із логу, що був переданий з Flask.

```
<div class="back-button">
    <a href="{{ url_for('index') }}">
        <button>← Назад на головну</button>
    </a>
</div>
</body>
</html>
```

3.9 Реалізація резервного інтерфейсу користувача на базі Telegram

За допомогою *@BotFather* було створено телеграм бот та отримано його токен, що був вказаний в коді нижче для синхронізації. Крім того вказано id чату в який будуть надходити повідомлення та команди. А також було створено webhook який приймає повідомлення від бота через */bot*, він дозволяє Telegram-боту автоматично отримувати повідомлення в режимі реального часу — без потреби опитувати сервер (getUpdates). Його було створено за допомогою ngrok та автоматизовано на допомогою .bat файлу аби не робити це власноруч кожного разу.

Скрипт .bat у виглядає наступним чином:

```
@echo off

setlocal

set TOKEN="7955466062:AAF7GbkMbcW_VD5Xq6_epjokKdjWIKIf3Bk"

set FLASK_PATH=C:\Users\andre\OneDrive\Рабочий стол\diploma\main.py
```

```
set PYTHON=C:\Users\andre\AppData\Local\Programs\Python\Python312\python.exe

set NGROK_PATH=C:\Users\andre\Downloads\ngrok-v3-stable-windows-amd64\ngrok.exe
```

Налаштовуємо змінні для середовища та проводимо шлях до них на власному ПК.

```
start "" "%PYTHON%" "%FLASK_PATH%"

timeout /t 3 > nul
```

Відкриває Flask у фоновому режимі (на порту 5000) та робимо проброс порту 5000 через HTTPS на публічну адресу.

```
start "" "%NGROK_PATH%" http 5000

timeout /t 5 > nul

for /f "tokens=2 delims=\": \"" %%a in ('curl -s http://localhost:4040/api/tunnels ^| findstr
public_url') do (

   set URL=%%a

   goto CONTINUE

)
```

API ngrok повертає JSON з полем `public_url`, далі скрипт витягує цю адресу і записує в `%URL%`.

```
:CONTINUE

:: === Встановлення Webhook ===

set FULL_URL=https://api.telegram.org/bot%TOKEN%/setWebhook?url=%URL%/bot

echo Встановлення Webhook: %FULL_URL%

curl %FULL_URL%
```

Встановлює webhook Telegram на маршрут `/bot`. Цей .bat файл автоматично запускає Flask-сервер, ngrok і встановлює Webhook для Telegram-бота. Це робиться для того щоб автоматизувати процес та пришвидшити і полегшити запуск додатку.

Нижче наведено частину коду який є повноцінним модулем охоронної системи на Flask із відеопотоком, де Telegram-бот використовується для керування та налагодження роботи.

```
TELEGRAM_BOT_TOKEN = "7777031361:AAFvHtQP9eh2w9VjDzFlzQQDpLbamMoFFSM"
TELEGRAM_CHAT_ID = "511403810"
TELEGRAM_API = f"https://api.telegram.org/bot{TELEGRAM_BOT_TOKEN}"

@app.route('/bot', methods=['POST'])
```

Ініціалізація отриманого раніше токену та чат ІД в відповідну змінну та увімкнення прийому команд ботом у застосунку.

```
def bot():

    global is_armed
    data = request.get_json(force=True)
    message = data.get("message", {})
    text = message.get("text", "")
    chat_id = message.get("chat", {}).get("id")
    if chat_id and str(chat_id) != TELEGRAM_CHAT_ID:
        return "Unauthorized", 403
    if text == "/status":
        state = "🔒 ОХОРОНА УВІМКНЕНА" if is_armed else "🔓 ОХОРОНА ВИМКНЕНА"
        send_telegram_alert(state)
```

Команда "/status" надсилає поточний стан охоронної системи (is_armed).

```
    elif text == "/toggle":
        is_armed = not is_armed
        new_state = "✅ ОХОРОНА УВІМКНЕНА" if is_armed else "⚠️ ОХОРОНА ВИМКНЕНА"
        send_telegram_alert(new_state)
```

Команда /toggle" вмикає або вимикає охорону й сповіщає про новий стан системи.

```
    elif text == "/log":
        if os.path.exists(event_log_path):
            with open(event_log_path, 'r') as f:
                log = f.read()[-4000:]
        else:
            log = "Журнал порожній."
        send_telegram_alert(f"📄 Журнал подій:\n{log}")
```

Команда "/log" витягує останні 4000 символів з журналу подій (`events.log`). Якщо файл не існує або він пустий — надсилає `"Журнал порожній."`

```
    elif text == "/snapshot":
        if initialize_camera():
            with camera_lock:
                success, frame = camera.read()
```

```
        if success:
            temp_file = tempfile.NamedTemporaryFile(delete=False, suffix=".jpg")
            cv2.imwrite(temp_file.name, frame)
            send_telegram_photo(temp_file.name, caption="📸 Поточне зображення з камери")
```

"/snapshot" - Ініціалізує камеру та зчитує кадр після чого тимчасово зберігає у `.jpg` та надсилає зображення у Telegram.

```
    else:
        send_telegram_alert("🤖 Доступні команди:\n/status — статус охорони\n/toggle — увімк./вимк.\n/log — журнал\n/snapshot — фото")
    return "ok"
```

У відповідь на все інше буде надсилати довідку про доступні команди та опис їх застосування.

3.10 Тестування реалізованої системи

Тестування реалізованої системи проходить в декілька етапів, перший з яких це тестування та перегляд графічного інтерфейсу та телеграм боту з якими буде взаємодіяти користувач.

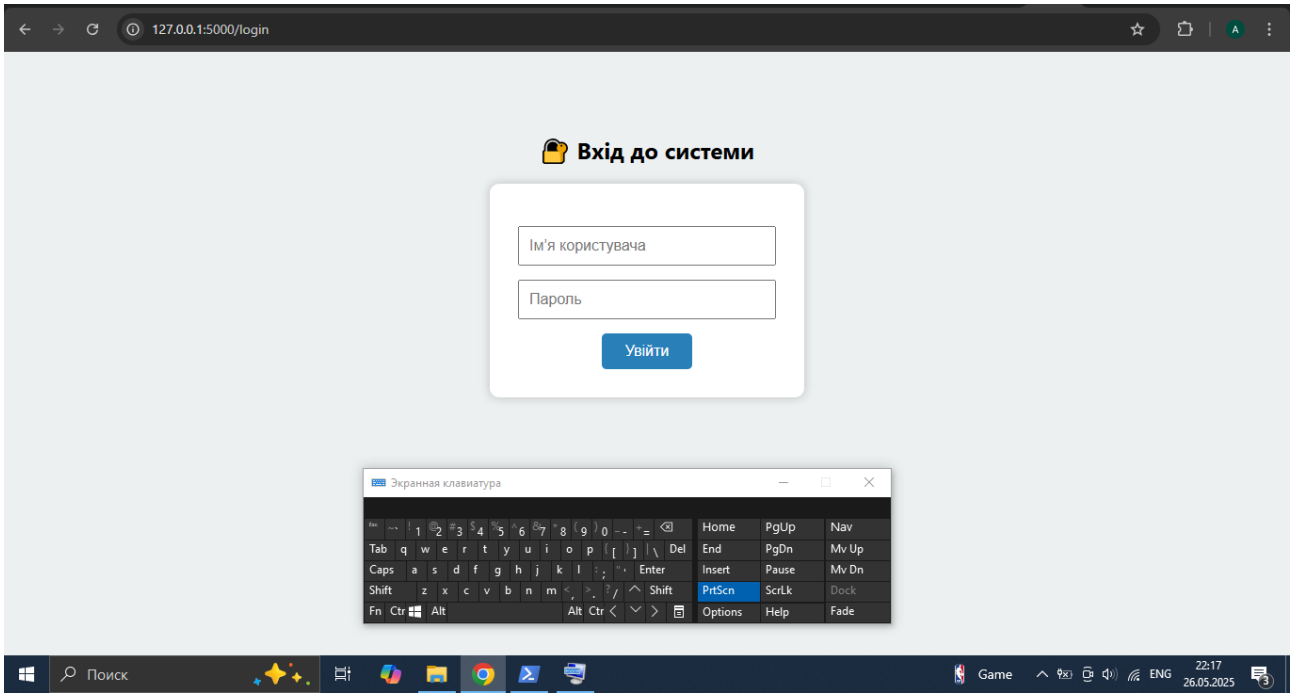


Рисунок 3.2 – Тестування графічного інтерфейсу (login.html)

На рисунку 3.2 можна спостерігати реалізацію графічного інтерфейсу у фірмовому стилі на сторінці входу де реалізовано два поля форми для заповнення, це Ім’я користувача та Пароль, а також кнопку Увійти, яка при

введені вірних значень перенаправляє на головну сторінку index.html (Рисунок 3.3).

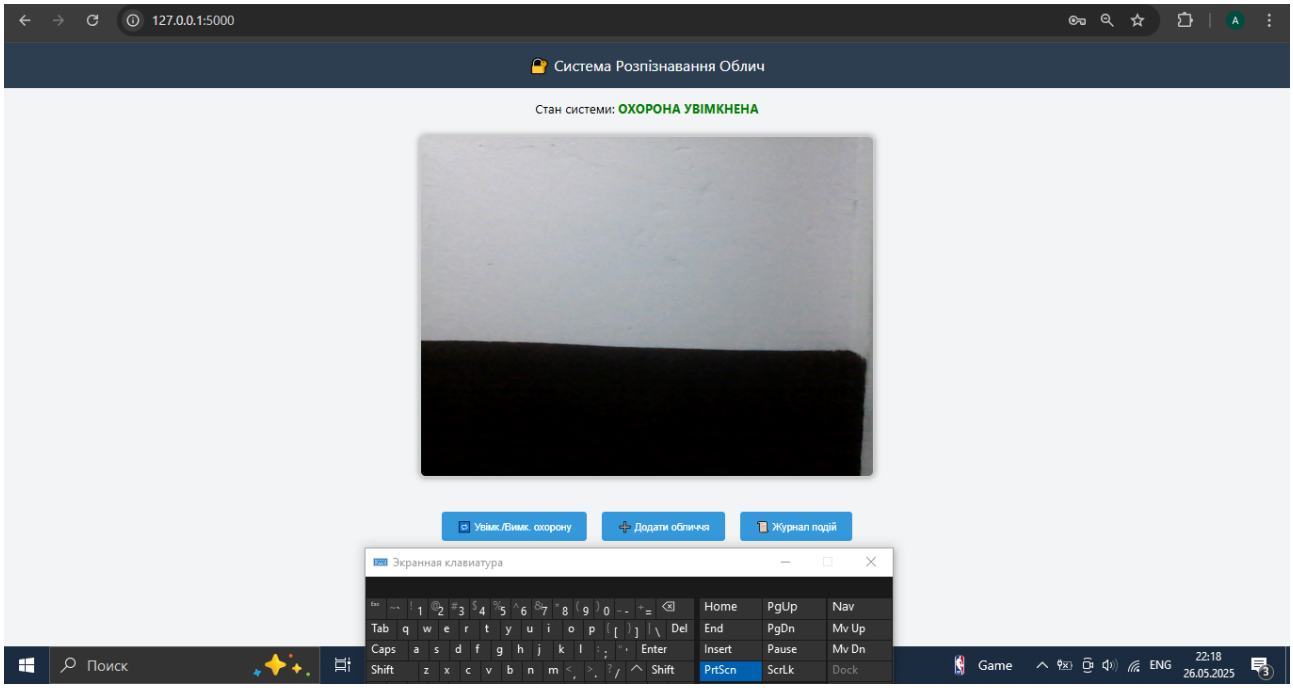


Рисунок 3.3 - Тестування графічного інтерфейсу (index.html)

На рисунку 3.3 зображена основна сторінка у web браузері із виведеним відео з камери в реальному часі та кнопками для взаємодії з системою. В верхній часині рисунку в адресному полі можна помітити що система працює за адресою **`127.0.0.1:5000`** — це **адреса** для відкриття сторінки входу у Flask-додатку **локально із** портом (:5000), який використовується за замовчуванням.

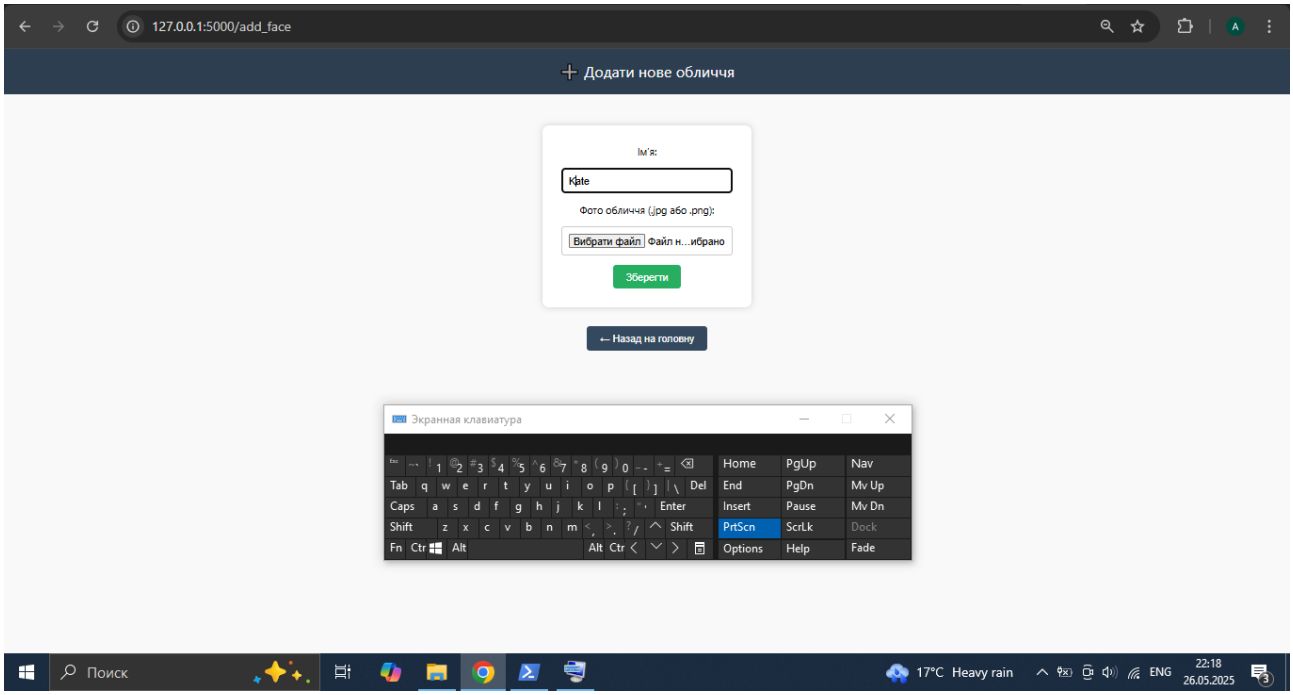


Рисунок 3.4 - Тестування графічного інтерфейсу (add_face.html)

Рисунок 3.4 відображає можливість додання обличь на які система не буде реагувати тривогою, а буде дозволяти допуск до будинку, за побажанням функції що буде виконувати система при виявленні того чи іншого обличчя є обмеженими лиши фантазією, це може бути від банального відчинення замку дверей до запуску відповідного сценарію (ввімкнення світла, телевізора, кавоварки тощо).

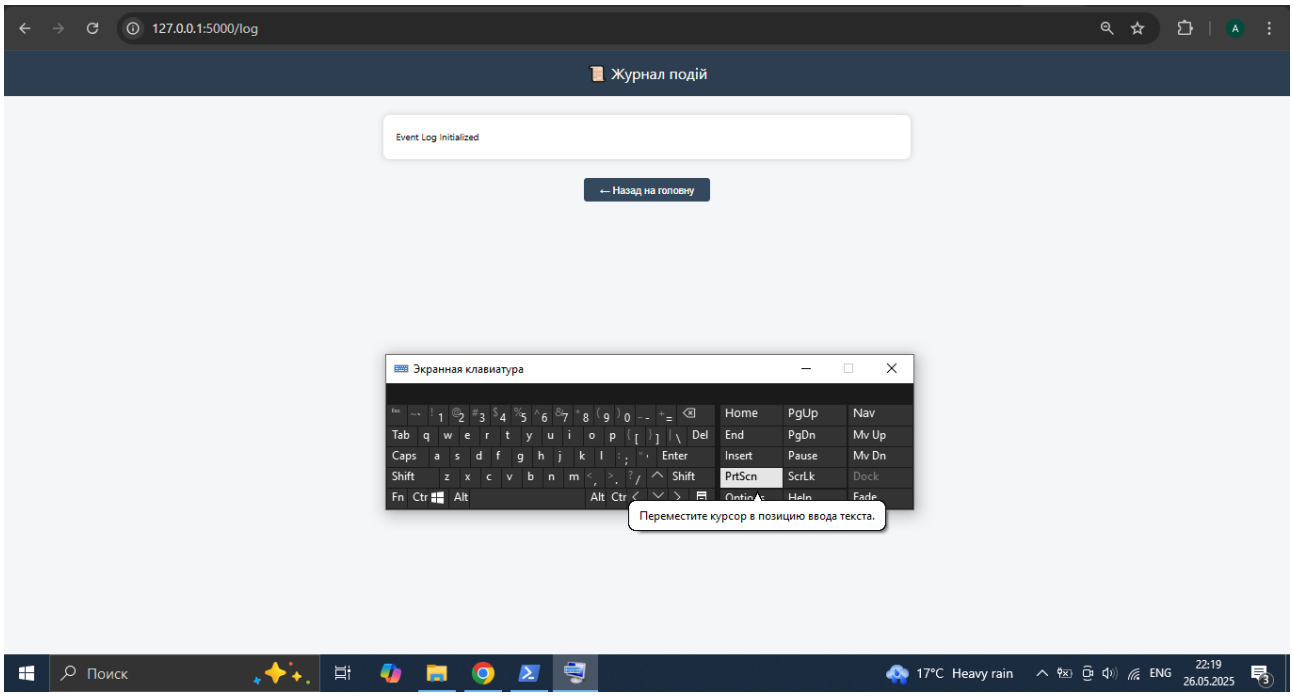


Рисунок 3.5 - Тестування графічного інтерфейсу (log.html)

У журналі подій (Рисунок 3.5) виводяться зареєстровані події, якщо таких в файлі не виявлено виводиться Event Log Initializated. Також на сторінці є функціональна кнопка для повернення на головну сторінку.

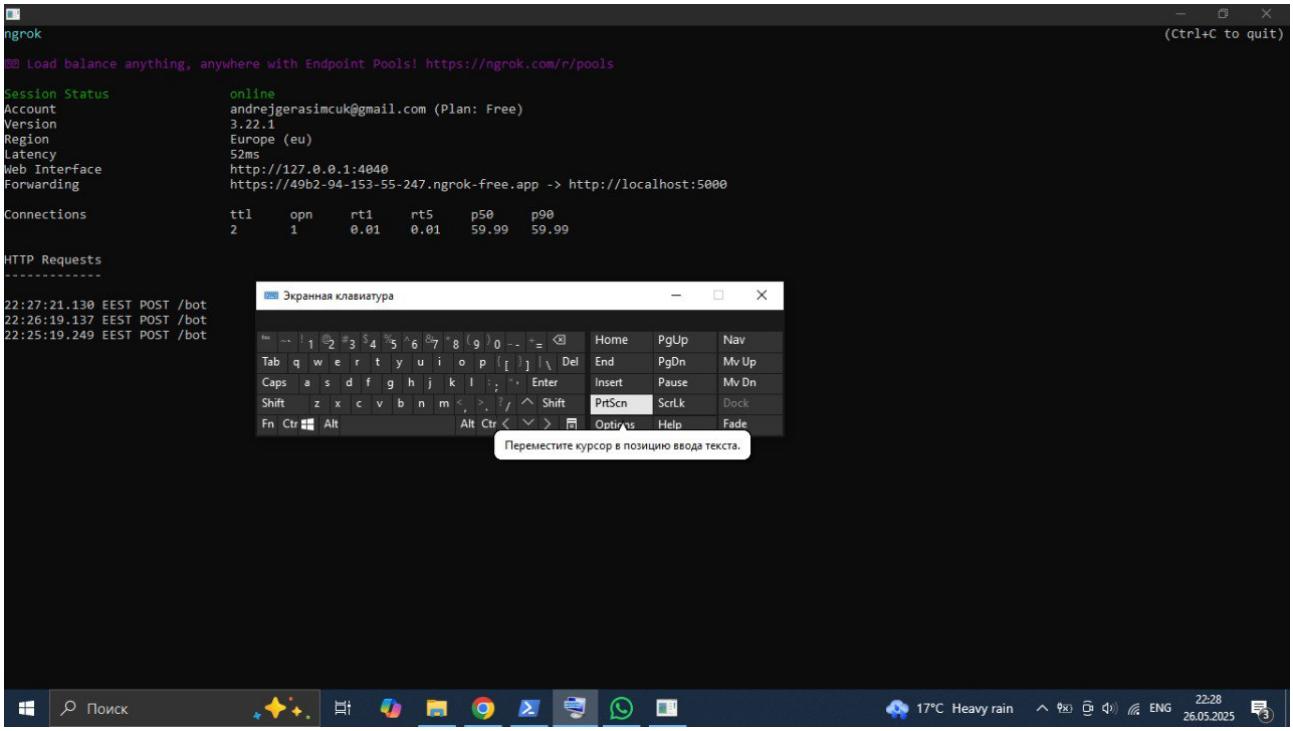


Рисунок 3.6 – Запуск webhook для роботи телеграм боту

На рисунку 3.7 зафіксовано запуск та успішну роботу webhook за допомогою ngrok. Зображено електронну адресу власника, тарифний план, версію та адресу веб інтерфейсу.

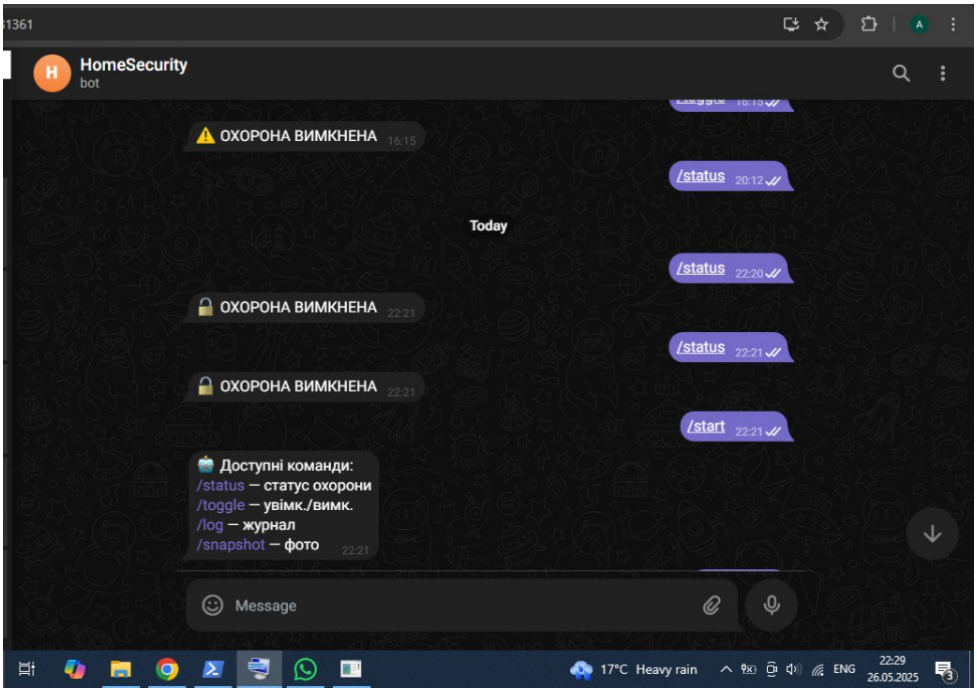


Рисунок 3.8 - Тестування роботи телеграм боту

Згідно з рисунком 3.8 та 3.9 телеграм бот запущено коректно він успішно виконує введені команди та обробляє їх. Функціонал телеграм боту та веб

інтерфейсу дублюється, відповідно охорону можна ввімкнути з телеграм боту, а вимкнути через веб інтерфейс.

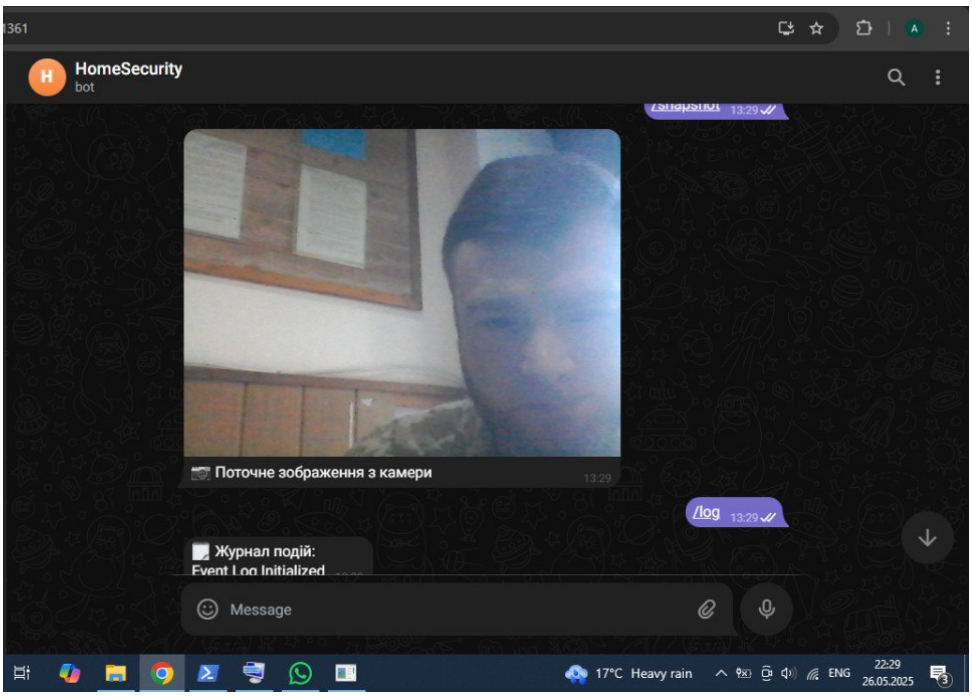


Рисунок 3.9 - Тестування виконання команд телеграм ботом

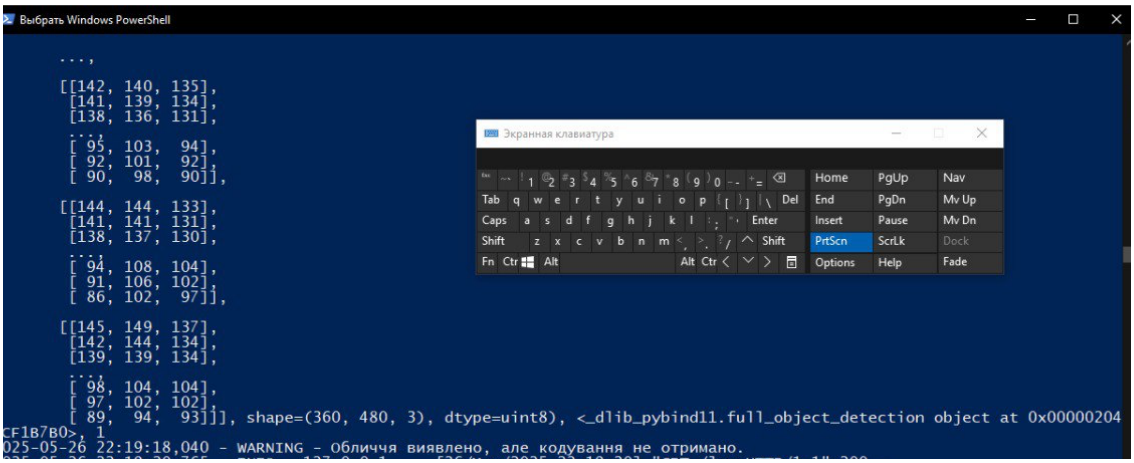


Рисунок 3.10 - Тестування роботи застосунку за допомогою PowerShell

Для фіксації логіки роботи програми та відображення логів в реальному часі було застосовано інструмент Windows Powershell, що зображено на рисунку 3.10.

Згідно з ним можна зафіксувати трьох вимірні масиви які надходять від OpenCV та те що об'єкт `dlib.full_object_detection` виводиться з форматом зображення `shape=(360, 480, 3), dtype=uint8`. Отож на вхід надходить зображення з веб-камери. Насамперед спрацьовує OpenCV, який

визначає, чи є на малюнку обличчя (особи). Якщо особи знайдені, ми отримуємо їх координати (що бачимо на рисунку 3.10) та умовно вирізаємо із загального кадру лише цю частину, передаємо її на розпізнавання та порівнюємо з базою – шукаємо збіг. У результаті якщо обличчя є у базі, охороно пройдено, якщо ні вмикається тривога. Також для низько якісних камер було розроблено окрему помилку, яку ми і басимо на нижній частині рисунка 3.10 – «Обличчя виявлено, але кодування не отримано», що означає необхідність замінити камеру на якіснішу.

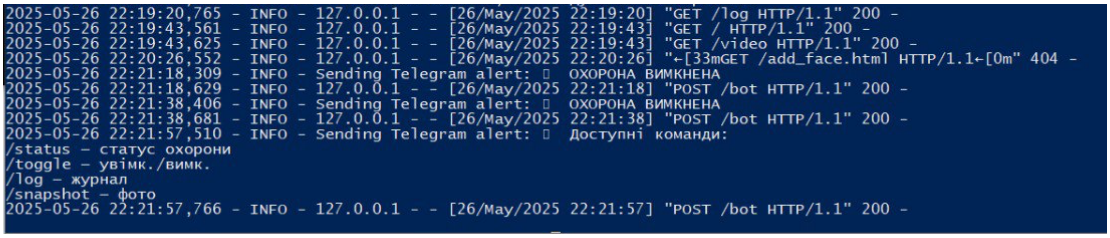


Рисунок 3.11 Тестування роботи інтерфейсу та телеграм боту

На рисунку 3.11 зображено взаємодію Flask додатку з HTML інтерфейсом та телеграм ботом, а саме їхню активність: ввімкнення та вимкнення охорони, а також відповіді на команди що надходять.

3.11 Висновки до розділу

У цьому розділі розроблено систему підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв) для вивчення потенційного застосування в домашній автоматизації, забезпеченні безпеки дверей з реагуванням у режимі реального часу та кращою швидкістю розпізнавання. Серед інших біометричних методів, підхід розпізнавання обличь пропонує одну велику перевагу – зручність використання. Метою цієї роботи є створення повноцінної системи розпізнавання обличь: простої у створенні, дешевої та ефективної. На основі сучасних технологій і засобів програмування були створені модулі для реалізації розпізнавання обличь користувачів та відповідною реакцією на них, а також графічний інтерфейс, який дозволяє зручно взаємодіяти із системою. Крім того для зручності роботи з системою було додатково розроблено резервний спосіб керування у вигляді телеграм боту, хоча він може використовуватись не тільки

як резервний а як альтернативний варіант, оскільки реалізовує всі ті ж самі функції. Особливу увагу було приділено інтеграції цих модулів для забезпечення безперервної і надійної роботи системи.

Було використано алгоритми OpenCV серед багатьох алгоритмів для розпізнавання обличь. OpenCV – виявився дуже універсальним статистичним методом, в якому спостережувані випадкові дані лінійно перетворюються на компоненти, які максимально незалежні один від одного та одночасно мають «цікаві» розподіли.

Отже, згідно з тестуваннями, можна зробити висновок, що OpenCV є ефективнішим, ніж інші, з точки зору кращого розпізнавання обличь, підвищеної безпеки та чіткості зображень.

Графічний інтерфейс користувача (GUI), реалізовано на основі Flask, HTML, CSS та JavaScript, забезпечує інтерактивну і зручну взаємодію з системою. Інтерфейс підтримує перегляд відео безпосередньо з камери користувачем, що дає змогу перевіряти реакцію ШІ. Додавання обличь, ввімкнення та вимкнення системи, перегляд подій – все це є влаштованим у графічний інтерфейс. Таким чином, у цьому розділі була досягнута головна мета – створення повнофункціональної системи розпізнавання обличь, яка поєднує в собі дві функції одночасно, функцію охоронної системи та функцію доступу до будинку без сторонніх пристроїв.

Запропонована система буде корисною для тих, хто не перебуває вдома більшу частину часу та потребує відстеження відвідувачів. ЇЇ корисність полягає в тому, щоб налаштувати його як сповіщення для відвідувачів дому та надати інформацію про відвідувачів на динамічному веб-сайті та в телеграм боті, система може використовуватися в інших сферах, таких як промисловість, офіси та навіть аеропорти для ідентифікації розшукуваних осіб.

4 ЕКОНОМІЧНА ЧАСТИНА

У цьому розділі проведено дослідження щодо економічного потенціалу розробки за темою « Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв) ». Аналіз включає в себе проведення оцінки комерційних можливостей та прогнозування витрат на виконання науково-дослідної роботи, впровадження результатів, в тому числі проведення оцінки очікуваних економічних вигод від впровадження розробленого продукту.

Додатково проведено розрахунки ефективності вкладених інвестицій і терміну їх окупності, що є ключовими показниками для залучення потенційних інвесторів.

За допомогою отриманих даних буде зроблено висновок щодо економічної доцільності розробки системи для захисту розумних будинків, що базується на сучасних алгоритмах і методах штучного інтелекту, та її перспективності для впровадження у практичну діяльність.

4.1 Оцінювання комерційного потенціалу розробки програмного забезпечення

Основною метою проведення технологічного аудиту є оцінка комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

У межах магістерської роботи було розроблено систему Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв). Дана система реалізована у вигляді програмного забезпечення, яке забезпечує високий рівень захисту будь яких об’єктів на які її буде встановлено, включно з майном що знаходиться в середині.

Для проведення технологічного аудиту залучено трьох незалежних експертів. У рамках цієї роботи експертами виступають викладачі кафедри МБІС, зокрема:

- Яремчук Ю. Є. (д.т.н., професор МБІС ВНТУ);

– Грицак А. В. (доцент, викладач кафедри МБІС ВНТУ);

– Карпінець В. В. (к.т.н., доцент кафедри МБІС ВНТУ).

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в реальних умовах
Ринкові переваги (недоліки)					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в	Технічні та споживчі влас-тивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в	Технічні та споживчі властивості продукту значно кращі, ніж в
5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї

Продовження таблиці 4.1

9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінки науково-технічного рівня та комерційного потенціалу науково-технічної розробки будуть узагальнені за допомогою таблиця для більш зручного сприйняття.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	5	4	4
2. Ринкові переваги (наявність аналогів)	4	5	4
3. Ринкові переваги (ціна продукту)	4	4	4
4. Ринкові переваги (технічні властивості)	5	4	3
5. Ринкові переваги (експлуатаційні витрати)	4	4	4
6. Ринкові перспективи (розмір ринку)	3	4	5
7. Ринкові перспективи (конкуренція)	5	4	4

8. Практична здійсненність (наявність фахівців)	3	3	3
9. Практична здійсненність (наявність фінансів)	3	3	4
10. Практична здійсненність (необхідність нових матеріалів)	4	4	3
11. Практична здійсненність (термін реалізації)	4	4	4
12. Практична здійсненність (розробка документів)	3	3	3
Сума балів	47	46	45
Середньоарифметична сума балів $СБ_c$	**46**		

На основі даних, наведених у таблиці 4.2, можна здійснити аналіз комерційного потенціалу розробки. Далі порівняємо ці результати з рівнями комерційного потенціалу, представленими в таблиці 4.3.

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Результати досліджень показали, що рівень комерційного потенціалу розробки нової системи захисту розумного будинку на основі штучного інтелекту становить 46 балів. Це підтверджує високу важливість та потенційну комерційну успішність проведених досліджень, як вказано у таблиці 4.3.

4.2 Прогнозування витрат на виконання наукової роботи та впровадження її результатів

Прогнозування витрат на виконання науково-дослідних, дослідно-конструкторських та конструкторсько-технічних робіт складається з трьох основних етапів, які детально аналізують різні аспекти витрат і мають вплив на всі етапи реалізації проекту.

На першому етапі визначаються витрати, безпосередньо пов'язані з ресурсами та зусиллями, витраченими виконавцями на виконання цієї частини роботи. Це включає витрати на оплату праці, навчання та інші витрати, що напряму пов'язані з реалізацією конкретних завдань.

Другий етап полягає у розрахунку загальних витрат на виконання всієї роботи, охоплюючи витрати на матеріали, обладнання, послуги та інші витрати, що стосуються всього проекту.

Третій етап передбачає прогнозування витрат, пов'язаних з впровадженням результатів роботи, включаючи витрати на реалізацію розробок, рекламні кампанії, підготовку персоналу та інші витрати, пов'язані з реалізацією отриманих результатів.

Витрати на основну заробітну плату дослідників (Зо) розраховують відповідно до посадових окладів працівників, за формулою:

$$З_о = \frac{М}{Т_р} \times t(грн), \qquad (4.1)$$

де М – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$Т_р$ – число робочих днів в місяці; приблизно $Т_р \approx$ 21...23 дні;

t – число робочих днів роботи дослідника.

$$З_о = \frac{20000}{22} \times 40 = 45 = 36\ 363 \text{ грн.}$$

Для розробки програмні засоби необхідно залучити програміста з посадовим окладом 11000 грн. Кількість робочих днів у місяці складає 40, а кількість робочих днів програміста складає 22. Зведемо сумарні розрахунки до таблиця 4.4.

Таблиця 4.4 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату
Керівник	25000	1 136,4	5	5 682
Програмний інженер	20000	909,1	40	36 363
Всього				42 045

Додаткова заробітна плата Зд для всіх розробників та працівників, залучених до створення нового технічного рішення, визначається як 10–12% від їхньої основної заробітної плати. У цьому підприємстві розмір додаткової заробітної плати становить 10% від основної заробітної плати.

$$З_д = (З_о + З_р) \times \frac{Н_{дод.}}{100\%} \qquad (4.2)$$

,де $Н_{дод}$ – норма нарахування додаткової заробітної плати.

$$Зд = 0{,}1 * 42\ 045 = 4204{,}5 \text{ (грн)}.$$

Нарахування на заробітну плату дослідників та робітників розраховується як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$Н_{зп} = (З_о + З_{дод}) \times \frac{\beta}{100\%} \qquad (4.3)$$

Де β – норма нарахування на заробітну плату.

$$Н_{зп} = (42\ 045 + 4\ 204{,}5) \times \frac{22}{100} = 10\ 175 \text{ (грн)}.$$

Витрати на комплектуючі вироби, які використовують при виготовленні одиниці продукції, розраховуються, згідно їх номенклатури, за формулою:

$$К = \sum_{i=1}^{n} H_i \times Ц_i \times К_{i,} \qquad (4.4)$$

де Ні – кількість комплектуючих і-го виду, шт.;

Ці – покупна ціна комплектуючих і-го найменування, грн.;

Кі – коефіцієнт транспортних витрат (1,1…1,15).

Таблиця 4.5 – Витрати на матеріали

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Набір канцелярський	220	2	440

Папір	215	1	215
USB накопичувач	300	1	300
Всього			955
З врахуванням коефіцієнта транспортування			1050

Таблиця 4.6 – Комплектуючі, що використані на розробку

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Веб-камера Verbatim AWC-03 Ultra HD 4K	1 шт.	2 500 грн.	2 500 грн.
Всього			2 500

Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження.

Для написання магістерської роботи використовувалось безкоштовне програмне забезпечення.

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо можуть бути розраховані з використанням прямолінійного методу амортизації за формулою:

$$А_{обл} = \frac{Ц_б}{Т_в} \times \frac{t_{вик}}{12} \qquad (4.5)$$

де Цб – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

tвик – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

Тв – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$$А_{обл} = \frac{40\,000 \times 2}{3 \times 12} = 2\,222{,}2 \text{ грн.}$$

Таблиця 4.7 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Ноутбук Macbook Pro 2015 р.в.	40 000	3	2	2 222,2
Ноутбук Asus L3050	12 000	3	2	666
Офісне приміщення	12 000	5	2	400
Організаційне обладнання	4000	4	2	166,6
Всього				3 454,8

До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на придбання палива у сторонніх підприємств, установ та організацій, яке використовується з технологічною метою для проведення досліджень. Ця стаття формується у разі проведення енергоємних наукових досліджень за методом прямого віднесення витрат і може становити значну частку у собівартості досліджень.

Витрати на силову електроенергію (Ве) розраховуються за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \qquad (4.6)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{вп}$ – коефіцієнт, що враховує використання потужності, $K_{вп} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$$B_e = \frac{0{,}3 \times 200 \times 7{,}50 \times 0{,}95}{0{,}97} = 440{,}7 \text{ грн.}$$

Таблиця 4.8 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Ноутбук Macbook Pro 2015 р.в.	0,3	200	440,7
Ноутбук Asus L3050	0,3	400	881,4
Офісне приміщення	0,1	350	257,08
Організаційне обладнання	0,2	30	44,07
Всього			1 623,3

Стаття «Службові відрядження» охоплює витрати, пов'язані з відрядженням штатних працівників, працівників за цивільно-правовими договорами, аспірантів, що зайняті науково-дослідницькою діяльністю, які пов’язані з тестуванням машин та приладів, а також витрати на відрядження на наукові заходи, конференції, наради, що мають прямий зв’язок з виконанням конкретних досліджень.

Витрати за цією статтею розраховуються у розмірі 20–25% від суми основної заробітної плати дослідників та робітників за допомогою формули:

$$B_{св} = (З_о + З_р) \cdot \frac{H_{св}}{100\%}, \qquad (4.7)$$

де $H_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $H_{св}$ = 20%.,

$$B_{св} = (42\ 045) \times \frac{20}{100\%} = 8\ 409 \text{ грн.}$$

Стаття «Витрати на роботи, виконані сторонніми підприємствами, установами і організаціями» охоплює витрати на проведення досліджень, які не

можуть бути здійснені штатними працівниками або наявним обладнанням організації, і виконуються на умовах договору іншими підприємствами, установами і організаціями незалежно від форми власності та за допомогою позаштатних працівників.

Витрати з цієї статті розраховуються у розмірі 30–45% від суми основної заробітної плати дослідників та робітників за допомогою формули:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \qquad (4.8)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$$В_{сп} = (42\ 045) \times \frac{30}{100\%} = 12\ 613{,}5 \text{ грн.}$$

Стаття «Інші витрати» включає витрати, які не були охарактеризовані у попередніх статтях витрат і можуть бути прямо віднесені до собівартості досліджень за безпосередніми показниками. Витрати за цією статтею обчислюються у розмірі 50–100% від суми основної заробітної плати дослідників та робітників за допомогою такої формули:

$$I_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \qquad (4.9)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$$І_в = (42\ 045) \times \frac{50}{100\%} = 21\ 022{,}5 \text{ грн.}$$

Сталими (загальновиробничими) витратами охоплюються різноманітні витрати, пов'язані з управлінням організацією, зусиллями в інноваціях та раціоналізації, а також з набором та підготовкою персоналу, банківськими послугами, освоєнням виробництва, а також науково-технічною інформацією та рекламою.

Витрати за цією статтею розраховуються у розмірі 100–150% від суми основної заробітної плати дослідників та працівників з використанням такої формули:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \qquad (4.10)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$$В_{нзв} = (42\ 045) \times \frac{100}{100\%} = 42\ 045 \text{ грн.}$$

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

$$В_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + В_{спец} + В_{прг} + А_{обл} + В_е + В_{св} + В_{сп} + І_в + В_{нзв} \quad (4.11)$$

$$В_{заг} = 42\ 045 + 4\ 204{,}5 + 10\ 175 + 1\ 050 + 3\ 454{,}8 + 1\ 623{,}3 + 8\ 409 + 12\ 613{,}5 + 21\ 022{,}5 + 42\ 045 = 146\ 642{,}6$$

Вартість завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів обчислюється відповідно до наступної формули:

$$ЗВ = \frac{В_{заг}}{\eta}, \quad (4.12)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,7.

$$ЗВ = \frac{146\ 642{,}6}{0{,}7} = 209\ 489{,}4 \text{ грн.}$$

Отже, прогноз загальних витрат ЗВ на виконання та впровадження результатів виконаної роботи складає 209 489,4 грн.

4.3 Прогнозування комерційних ефектів від реалізації результатів розробки

В даному розділі буде розглянуто майбутній економічний ефект від реалізації результатів розробки з підвищення захищеності об’єктів системи розумного будинку за допомогою штучного інтелекту. Для проведення аналізу було використано часовий проміжок у два роки з розрахунком на розширення бази користувачів.

У зазначеному випадку, майбутній економічний ефект базується на зростанні кількості користувачів продукту протягом аналізованого періоду часу:
у перший рік – 80 користувачів;
у другий – 200 користувачів;

N – кількість споживачів які використовували аналогічний продукт у році до впровадження результатів нової науково-технічної розробки, приймемо 1100 користувачів;

$Ц_б$ – вартість програмного продукту у році до впровадження результатів розробки, приймемо 10500,00 грн;

$\pm\Delta Ц_о$ – зміна вартості програмного продукту від впровадження результатів науково-технічної розробки, приймемо 5000,00 грн.

Для кожного з випадків потенційне збільшення чистого прибутку у потенційного інвестора ΔПі в роки очікуваного позитивного результату від можливого впровадження та комерціалізації науково-технічної розробки розраховується за відповідною формулою:

$$\Delta П_i = (\pm\Delta Ц_о \cdot N + Ц_о \cdot \Delta N)_i \cdot \lambda \cdot \rho \cdot (1 - \frac{\vartheta}{100}), \qquad (4.13)$$

де λ – коефіцієнт, який враховує сплату потенційним інвестором податку на додану вартість. У 2024 році ставка податку на додану вартість складає 20%, а коефіцієнт $\lambda = 0{,}8333$;

ρ – коефіцієнт, який враховує рентабельність інноваційного продукту. Приймемо $\rho = 20\%$;

ϑ – ставка податку на прибуток, який має сплачувати потенційний інвестор, у 2025 році $\vartheta = 18\%$;

Збільшення чистого прибутку 1-го року:

$$\Delta П_1 = (5000 \times 1100 + 10500 \times 80) \times 0{,}83 \times 0{,}2 \times \left(1 - \frac{0{,}18}{100}\right)$$

$$= 1\,050\,545{,}6 \text{ грн.}$$

Збільшення чистого прибутку 2-го року:

$$\Delta П_2 = \left(5000 \times 1100 + 10500 \times (80 + 200)\right) \times 0{,}83 \times 0{,}2 \times \left(1 - \frac{0{,}18}{100}\right)$$

$$= 1\,398\,518{,}1 \text{ грн.}$$

Для кожного з випадків потенційне збільшення чистого прибутку у потенційного інвестора $\Delta П_i$ в роки очікуваного позитивного результату від

можливого впровадження та комерціалізації науково-технічної розробки розраховується за відповідною формулою:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \quad (4.14)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,2;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання потенційним інвестором додаткових чистих прибутків у цьому році.

$$\text{ПП} = \frac{1\,050\,545{,}6}{(1+0{,}2)^1} + \frac{1\,398\,518{,}1}{(1+0{,}2)^2} = 1\,846\,647{,}7 \text{ грн.}$$

4.4 Розрахунок ефективності вкладених інвестицій та періоду їх окупності

Для забезпечення наукової розробки інвесторами вона повинна бути обґрунтованою для інвестування ними, а основними показниками цього є абсолютна та відносна ефективність інвестицій, а також термін їх повернення.

Перш за все для їх визначення потрібно розпочати з розрахунку сучасної вартості інвестицій (PV), вкладених у науково-технічну розробку.

Для цього можна використати формулу:

$$PV = k_{інв} \cdot ЗВ, \quad (4.15)$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=3;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 209 489,4 грн.

$$\text{PV} = 3 * 209\,489{,}4 = 628\,468{,}2 \text{ грн.}$$

Таким чином, чистий приведений дохід (NPV) або абсолютний економічний ефект ($Е_{абс}$) для потенційного інвестора від можливого впровадження та комерціалізації науково-технічної розробки буде таким:

$$E_{абс} = ПП - PV \qquad (4.16)$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки,1 846 647,7 грн;

PV – теперішня вартість початкових інвестицій, 628 468,2 грн.

$$\text{Е}_{\text{абс}} = 1\,846\,647{,}7 - 628\,468{,}2 = 1\,218\,179{,}5 \text{ грн.}$$

Внутрішня економічна дохідність (Ев) інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, обчислюється за допомогою такої формули:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \qquad (4.17)$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 1 218 179,5 грн;

PV – теперішня вартість початкових інвестицій, 628 468,2 грн;

$T_ж$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 2 роки.

$$\text{Е}_\text{в} = \sqrt[2]{1 + \frac{1\,218\,179{,}5}{628\,468{,}2}} - 1 = 0{,}7$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій (мін τ) визначається згідно такою формулою:

$$\tau_{мін} = d + f, \qquad (4.18)$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2024 році в Україні d =0,15;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,1.

$$\tau_{min} = 0{,}1 + 0{,}15 = 0{,}25$$

Оскільки $E_в$ = 70% > τ_{min} = 25%, це свідчить про те, що внутрішня економічна дохідність інвестицій, які можуть бути вкладені потенційним інвестором у впровадження та комерціалізацію науково-технічної розробки, перевищує мінімальну внутрішню дохідність. Таким чином, інвестування у науково-дослідну роботу за темою « Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв) » є економічно обґрунтованим і доцільним.

Далі обчислюємо період окупності інвестицій ($T_{ок}$ або DPP, Discounted Payback Period), які потенційний інвестор може вкласти у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_в}, \qquad (4.19)$$

$$\mathrm{T_{ок}} = \frac{1}{0{,}7} = 1{,}4 \text{ року.}$$

З огляду на те, що період окупності інвестицій у реалізацію наукового проєкту становить менше трьох років, можна дійти висновку, що фінансування цієї нової розробки є виправданим.

4.5 Висновки до розділу

Згідно з проведеними дослідженнями, рівень комерційного потенціалу розробки за темою «Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв)» становить 46 балів. Це свідчить про високу комерційну значущість проведених досліджень.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 146 642,6 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 209 489,4 грн.

Термін окупності розробки складає 1,4 року, що значно менше стандартного трирічного періоду. Це підтверджує її комерційну привабливість і обґрунтованість інвестицій для фінансування впровадження та виходу продукту на ринок.

Підсумовуючи, можна зробити висновок про доцільність проведення науково-дослідної роботи за цією темою.

ВИСНОВКИ

У ході виконання магістерсько-кваліфікаційної роботи було успішно створено систему підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв). Реалізація була виконана за допомогою мови програмування Python, бібліотек OpenCV та face_recognition, а також фреймворку Flask для створення веб інтерфейсу. Додатково було впроваджено Telegram-інтеграцію для віддаленого сповіщення користувача у випадку виявлення невідомих обличь.

В рамках роботи було проведено детальний аналіз сучасних методів та моделей захисту розумних будинків, з якими стикаються люди у сучасному світі. Особливу увагу приділено аналізу сценаріїв поведінки, а також розробці архітектури системи для обробки та аналізу даних у режимі реального часу та особливості при роботі з ШІ. Це дозволило сформувати концепцію системи, яка поєднує аналіз відео та поведінки для підвищення точності виявлення проникнень до об’єкту що захищається системою.

Система дозволяє додавати нові обличчя до бази, вести журнал подій, здійснювати перемикання охоронного режиму, а також керувати всіма функціями через авторизований веб інтерфейс. Завдяки реалізованій авторизації доступ до панелі керування є захищеним.

Отримані результати підтверджують працездатність та ефективність розробленого рішення, яке може бути використане для локальних систем безпеки в офісах, навчальних закладах, приватних будинках або інших об’єктах. Система є автономною, не залежить від хмарних сервісів, та може бути легко масштабована або адаптована під конкретні вимоги користувача.

Таким чином, поставлені в роботі цілі були досягнуті. Розроблена система має практичну цінність і може бути використана в реальних умовах для забезпечення охорони приміщень, контролю доступу, або як елемент "розумного

дому". Вона поєднує у собі точність, зручність використання, функціональність та масштабованість.

Економічна частина роботи підтверджує доцільність розробки та впровадження її у життя, особливо зважаючи на відсутність аналогів що працюють за схожим принципом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Ghirardello, K.; Maple, C.; Ng, D.; Kearney, P. Cyber Security of Smart Homes: Development of a Reference Architecture for Attack Surface Analysis; IET: Beijing, China, 2018. (дата звернення 10.03.2025)

2. Shostack, A. *Threat Modeling: Designing for Security*; Wiley: Hoboken, NJ, USA, 2014. (дата звернення 12.03.2025)

3. Lin, S.; Miller, B.; Durand, J.; Bleakley, G.; Chigani, A.; Martin, R.; Crawford, M. The Industrial Internet of Things, Volume G1: Reference Architecture. Ind. Internet Consort. 2017, G1, 10–46. (дата звернення 12.03.2025)

4. Weyrich, M.; Ebert, C. Reference architectures for the internet of things. IEEE Softw. 2016, 33, 112–116 (дата звернення 12.03.2025)
5. Vidalis, S. *A Critical Discussion of Risk and Threat Analysis Methods and Methodologies*; Technical Report CS-04-03; School of Computing, University of Glamorgan: Pontypridd, UK, 2004. (дата звернення 12.03.2025)

6. *Information Technology—Security Techniques—Information Security Risk Management*; Standard; International Organization for Standardization: Geneva, Switzerland, 2018. (дата звернення 12.03.2025)

7. Штучний інтелект в будинку: не фантастика, а реальність. URL: https://worldvision.com.ua/iskusstvennyy-intellekt-v-dome-ne-fantastika-a-realnost/?srsltid=AfmBOoo0FGZNfbV3K6gWEx_v_BAETGCUUSc_bVkbF0WeRwnhAxxKwZ8p (дата звернення 9.03.2025)

8. Як змінюється світ зі штучним інтелектом в робототехніці та інтернеті речей. URL: https://startit.ua/jak-zminuetsa-svit-zi-shtichnim-intelektom-v-robototehnici (дата звернення 14.03.2025)

9. Пошук та виявлення вразливостей у мережах. URL: https://cip.gov.ua/ua/news/poshuk-ta-viyavlennya-vrazlivostei-u-merezhakh-ta-sistemakh-zovnishnimi-fakhivcyami-derzhspeczv-yazku-zatverdila-neobkhidni-dlya-bug-bounty-dokumenti (дата звернення 9.03.2025)

10. Ресурси та бази даних для пошуку вразливостей. URL: https://hackyourmom.com/pryvatnist/resursy-ta-bazy-danyh-dlya-poshuku-vrazlyvostej-2/ (дата звернення 10.03.2025)

11. Artificial Intelligence (AI) Cybersecurity | IBM. IBM in Deutschland, Österreich und der Schweiz. URL: https://www.ibm.com/ai-cybersecurity?utm_medium=OSocial&utm_source=Youtube& utm_content=RSRWW&utm_id=YT-101-AI-and-Cybersecurity (дата звернення: 11.03.2025).

12. How to Integrate AI Solutions for Smart Home Automation? URL:https://sudosuai.medium.com/how-to-integrate-ai-solutions-for-smart-home-automation-016bebbf13d8 (дата звернення: 11.03.2025).

13. Kavallieratos, G.; Gkioulos, V.; Katsikas, S.K. Threat Analysis in Dynamic Environments: The Case of the Smart Home. In Proceedings of the 2019 15th International Conference on Distributed Computing in Sensor Systems (DCOSS), Santorini, Greece, 29–31 May 2019; c. 234–240. (дата звернення: 11.03.2025).

14. Brdiczka, O.; Langet, M.; Maisonnasse, J.; Crowley, J.L. Detecting human behavior models from multimodal observation in a smart home. IEEE Trans. Autom. Sci. Eng. 2009, 6, 588–597. (дата звернення: 11.04.2025).

15. Which AI Programming Languages are Best for Startups and Tech Companies? URL: https://turnkeystaffing.com/tech-trends/ai-programming-languages/ (дата звернення: 11.04.2025).

16. Chiara Turati, Viola Macchi Cassia, Francesca Simion, and Irene Leo. Newborns' face recognition: Role of inner and outer facial features. *Child development*, 77(2):297–311, 2006. (дата звернення: 11.04.2025).
17. Intel acquires Itseez URL: https://opencv.org/blog/intel-acquires-itseez/ (дата звернення: 11.04.2025).
18. Опитування щодо популярності Python серед розробників URL: https://www.jetbrains.com/lp/devecosystem-2024/ (дата звернення: 11.04.2025).

ДОДАТКИ

Додаток А. Технічне завдання

Вінницький національний технічний університет

Факультет менеджменту та інформаційної безпеки

Кафедра менеджменту та безпеки інформаційних систем

ЗАТВЕРДЖУЮ

Голова секції "Управління інформаційною безпекою" кафедри МБІС

д.т.н., професор

_____ Юрій ЯРЕМЧУК

"____" _____ 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ

до магістерської кваліфікаційної роботи на тему:

Підвищення захищеності об'єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв)

08-72.МКР.001.01.103.ТЗ

Керівник магістерської кваліфікаційної роботи

д.т.н., професор, Яремчук Ю.Є.

_____

Вінниця – 2025 р.

1. Найменування та область застосування

Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв). Область застосування: захист об’єктів системи розумного дому від несанкціонованого доступу.

2.Підстава для розробки

Розробка виконується на основі наказу ректора ВНТУ №96 від 20. 03. 2025 р.

3. Мета та призначення розробки

3.1 Мета розробки: створення автономної системи відеоспостереження з функцією розпізнавання обличь у режимі реального часу.

3.2 Призначення: підвищенні рівня безпеки приміщень шляхом автоматизованого контролю за вхідною групою або зоною доступу.

4. Джерела розробки

4.1. Гайфуліна Д. А., Котенко І. В. Аналіз моделей глибокого навчання задач виявлення мережевих аномалій інтернету речей // Інформаційно-керуючі системи, 2021 № 1, с. 28–37.

4.2. Berkovsky, V. V.; Bessonov, A. S. Аналіз та класифікація методів виявлення вторгнень в інформаційну систему. Системи управління, навігації та зв’язку. Збірник наукових праць, 2017

4.3. Бабкін А. А., and О. В. Кудін. "Огляд нейромережевих моделей систем виявлення вторгнень." Вчені записки Таврійського національного університету імені ВІ Вернадського Серія: Технічні науки 31.70 (2020): 77-82

5. Вимоги до програми

5.1 Вимоги до функціональних характеристик:

5.1.1 Програмний засіб повинен мати зручний, легкий у використанні інтерфейс користувача;

5.1.2 Реалізація методу не повинна вимагати спеціальних ліцензійних програмних додатків;

5.1.3 Програмний засіб повинен виконувати процес автентифікації користувачів у системі.

5.2 Вимоги до надійності:

5.2.1 Програмний засіб повинен працювати без помилок, у випадку виникнення критичних ситуацій необхідно передбачити виведення відповідних повідомлень;

5.2.2 Бази даних повинні бути налаштовані на автоматичне створення резервних копій;

5.2.3 Програмний засіб повинен виконувати свої функції.

5.3 Вимоги до складу і параметрів технічних засобів:

– процесор – Pentium 1500 МГц і подібні до них;

– оперативна пам’ять – не менше 512 Mb;

– середовище функціонування – операційна система сімейство Windows;

– вимоги до техніки безпеки при роботі з програмою повинні відповідати існуючим вимогам та стандартам з техніки безпеки при користуванні комп’ютерною технікою.

6. Вимоги до програмної документації

6.1 Обов’язкова поетапна інструкція для майбутніх користувачів, наведена у пункті 3.4

7. Вимоги до технічного захисту інформації

7.1 Необхідно забезпечити захист розроблюваного програмного засобу від несанкціонованого використання.

7.2 Неможливість отримання доступу незареєстрованих користувачів до інформаційних ресурсів.

8. Техніко-економічні показники

8.1 Цінність результатів використання даного проекту повинна перевищувати витрати на його реалізацію.

8.2 Має бути реалізований таким чином, щоб підходити для використання широкого загалу.

9. Стадії та етапи розробки

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Початок	Закінчення
1	Визначення напрямку магістерської роботи, формулювання теми	05.03.2025	20.03.2025
2	Аналіз предметної області обраної теми	21.03.2025	05.04.2025
3	Розробка алгоритму роботи	06.04.2025	06.05.2025
4	Написання магістерської роботи на основі розробленої теми		
5	Розробка економічної частини	06.05.2025	23.05.2025
6	Передзахист магістерської кваліфікаційної роботи	24.05.2025	30.05.2025
7	Виправлення, уточнення, корегування магістерської кваліфікаційної роботи	31.05.2025	10.06.2025
8	Захист магістерської кваліфікаційної роботи	11.06.2025	14.06.2025

10. Порядок контролю та прийому

10.1 До приймання магістерської кваліфікаційної роботи надається:

– ПЗ до магістерської кваліфікаційної роботи;

– програмний додаток;

– презентація;

– відзив керівника роботи;

– відзив опонента

Технічне завдання до виконання прийняв _____Герасимчук А.А.

Додаток Б. Лістинг програми

Основний лістинг:

```
from flask import Flask, render_template, Response, request, redirect, url_for, send_file, session, url_for
from functools import wraps
import cv2
import os
import numpy as np
import datetime
import face_recognition
import winsound
import requests
import tempfile
import threading
import time
import logging
import sys

app = Flask(__name__)
app.secret_key = os.urandom(24).hex()

known_face_encodings = []
known_face_names = []
event_log_path = "event_log.txt"
is_armed = True
last_alarm_time = 0
camera = None
camera_lock = threading.Lock()

TELEGRAM_BOT_TOKEN = "7777031361:AAFvHtQP9eh2w9VjDzFlzQQDpLbamMoFFSM"
TELEGRAM_CHAT_ID = "511403810"
TELEGRAM_API = f"https://api.telegram.org/bot{TELEGRAM_BOT_TOKEN}"


def login_required(f):
    @wraps(f)
    def decorated_function(*args, **kwargs):
        if not session.get('logged_in'):
            return redirect(url_for('login'))
        return f(*args, **kwargs)
    return decorated_function

VALID_USERNAME = os.getenv("VALID_USERNAME", "admin")
VALID_PASSWORD = os.getenv("VALID_PASSWORD", "1234")

@app.route('/')
@login_required
def index():
    return render_template('index.html', is_armed=is_armed)

@app.route('/login', methods=['GET', 'POST'])
def login():
    error = None
    if request.method == 'POST':
        username = request.form['username']
        password = request.form['password']
        if username == VALID_USERNAME and password == VALID_PASSWORD:
            session['logged_in'] = True
            return redirect(url_for('index'))
        else:
            error = "❌ Невірний логін або пароль"
    return render_template('login.html', error=error)
```

```
@app.route('/logout')
def logout():
    session.pop('logged_in', None)
    return redirect(url_for('login'))


logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    handlers=[
        logging.FileHandler('security_app.log', encoding='utf-8'),
        logging.StreamHandler(sys.stdout)
    ]
)
logger = logging.getLogger(__name__)

def send_telegram_alert(message):
    try:
        logger.info(f"Sending Telegram alert: {message}")
        response = requests.post(f"{TELEGRAM_API}/sendMessage", data={
            "chat_id": TELEGRAM_CHAT_ID,
            "text": message
        })
        if response.status_code != 200:
            logger.error(f"Telegram API error: {response.text}")
    except Exception as e:
        logger.error(f"Telegram send failed: {e}")

# Use only the image as input for encodings to avoid dlib parameter issues
def safe_face_encodings(image):
    try:
        encodings = face_recognition.face_encodings(image)  # без других аргументів
        if not encodings:
            logger.warning("Face found but no encodings returned.")
        return encodings
    except Exception as e:
        logger.error(f"face_recognition.face_encodings failed: {e}")
        return []

def send_telegram_photo(image_path, caption=""):
    with open(image_path, 'rb') as photo:
        requests.post(f"{TELEGRAM_API}/sendPhoto", data={"chat_id": TELEGRAM_CHAT_ID, "caption": caption},
files={"photo": photo})

def load_known_faces():
    if not os.path.exists("faces"):
        os.makedirs("faces")
    known_face_encodings.clear()
    known_face_names.clear()
    for filename in os.listdir("faces"):
        if filename.endswith(".jpg") or filename.endswith(".png"):
            filepath = os.path.join("faces", filename)
            image = face_recognition.load_image_file(filepath)
            encodings = face_recognition.face_encodings(image)
            if encodings:
                known_face_encodings.append(encodings[0])
                known_face_names.append(os.path.splitext(filename)[0])

load_known_faces()

def initialize_camera():
    global camera
    if camera is None or not camera.isOpened():
```

```
        camera = cv2.VideoCapture(0)
        if not camera.isOpened():
            logger.error("Failed to initialize camera")
            return False
    return True

def log_event(name):
    with open(event_log_path, "a") as f:
        f.write(f"{datetime.datetime.now()} - {name}\n")

def trigger_alarm():
    global last_alarm_time
    now = time.time()
    if now - last_alarm_time < 10:
        return
    last_alarm_time = now
    def beep():
        try:
            winsound.Beep(1000, 1000)
        except:
            logger.error("Beep failed")
    threading.Thread(target=beep).start()
    send_telegram_alert("🚨 Виявлено невідоме обличчя!")

def generate_frames():
    global is_armed
    while True:
        if not initialize_camera():
            continue
        with camera_lock:
            success, frame = camera.read()
        if not success:
            logger.error("Failed to read frame from camera")
            continue
        try:
            if is_armed:
                small_frame = cv2.resize(frame, (0, 0), fx=0.75, fy=0.75)
                rgb_small_frame = small_frame[:, :, ::-1]
                face_locations = face_recognition.face_locations(rgb_small_frame)
                if face_locations:
                    face_encodings = safe_face_encodings(rgb_small_frame)
                    if not face_encodings:
                        logger.warning("Обличчя виявлено, але кодування не отримано.")
                    else:
                        for (top, right, bottom, left), face_encoding in zip(face_locations, face_encodings):
                            matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
                            name = "Unknown"
                            face_distances = face_recognition.face_distance(known_face_encodings, face_encoding)
                        if len(face_distances) > 0:
                            best_match_index = np.argmin(face_distances)
                            if matches[best_match_index]:
                                name = known_face_names[best_match_index]
                        log_event(name)
                        if name == "Unknown":
                            trigger_alarm()
                        top *= 2
                        right *= 2
                        bottom *= 2
                        left *= 2
                        color = (0, 255, 0) if name != "Unknown" else (0, 0, 255)
                        cv2.rectangle(frame, (left, top), (right, bottom), color, 2)
                        cv2.putText(frame, name, (left, top - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, color, 2)
                        cv2.putText(frame, name, (left, top - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 255, 0), 2)
            ret, buffer = cv2.imencode('.jpg', frame)
```

```
            frame = buffer.tobytes()
            yield (b'--frame\r\nContent-Type: image/jpeg\r\n\r\n' + frame + b'\r\n')
        except Exception as e:
            logger.error(f"Error in video stream: {e}")
            continue


@app.route('/video')
def video():
    return Response(generate_frames(), mimetype='multipart/x-mixed-replace; boundary=frame')

@app.route('/add_face', methods=['GET', 'POST'])
def add_face():
    if request.method == 'POST':
        name = request.form['name']
        file = request.files['image']
        filepath = os.path.join('faces', f"{name}.jpg")
        file.save(filepath)
        load_known_faces()
        return redirect(url_for('index'))
    return render_template('add_face.html')

@app.route('/toggle_security')
def toggle_security():
    global is_armed
    is_armed = not is_armed
    return redirect(url_for('index'))

@app.route('/log')
def show_log():
    if os.path.exists(event_log_path):
        with open(event_log_path, 'r') as f:
            log_lines = f.readlines()
    else:
        log_lines = ["Журнал подій порожній."]
    return render_template('log.html', log_lines=log_lines)

@app.route('/bot', methods=['POST'])
def bot():
    global is_armed
    data = request.get_json(force=True)
    message = data.get("message", {})
    text = message.get("text", "")
    chat_id = message.get("chat", {}).get("id")
    if chat_id and str(chat_id) != TELEGRAM_CHAT_ID:
        return "Unauthorized", 403
    if text == "/status":
        state = "🔒 ОХОРОНА УВІМКНЕНА" if is_armed else "🔓 ОХОРОНА ВИМКНЕНА"
        send_telegram_alert(state)
    elif text == "/toggle":
        is_armed = not is_armed
        new_state = "✅ ОХОРОНА УВІМКНЕНА" if is_armed else "⚠️ ОХОРОНА ВИМКНЕНА"
        send_telegram_alert(new_state)
    elif text == "/log":
        if os.path.exists(event_log_path):
            with open(event_log_path, 'r') as f:
                log = f.read()[-4000:]
        else:
            log = "Журнал порожній."
        send_telegram_alert(f"📝 Журнал подій:\n{log}")
    elif text == "/snapshot":
        if initialize_camera():
            with camera_lock:
                success, frame = camera.read()
```

```
            if success:
                temp_file = tempfile.NamedTemporaryFile(delete=False, suffix=".jpg")
                cv2.imwrite(temp_file.name, frame)
                send_telegram_photo(temp_file.name, caption="📸 Поточне зображення з камери")
    else:
        send_telegram_alert("🤖 Доступні команди:\n/status — статус охорони\n/toggle — увімк./вимк.\n/log — журнал\n/snapshot — фото")
    return "ok"

if __name__ == '__main__':
    if not os.path.exists(event_log_path):
        with open(event_log_path, 'w') as f:
            f.write("Event Log Initialized\n")
    app.run(host='0.0.0.0', port=5000)
```

Лістинг .bat файлу:

```
@echo off
setlocal

set TOKEN="7955466062:AAF7GbkMbcW_VD5Xq6_epjokKdjWIKIf3Bk"
set FLASK_PATH=C:\Users\andre\OneDrive\Рабочий стол\diploma\main.py
set PYTHON=C:\Users\andre\AppData\Local\Programs\Python\Python312\python.exe
set NGROK_PATH=C:\Users\andre\Downloads\ngrok-v3-stable-windows-amd64\ngrok.exe

start "" "%PYTHON%" "%FLASK_PATH%"
timeout /t 3 > nul

start "" "%NGROK_PATH%" http 5000
timeout /t 5 > nul

for /f "tokens=2 delims=\": \"" %%a in ('curl -s http://localhost:4040/api/tunnels ^| findstr public_url') do (
    set URL=%%a
    goto CONTINUE
)

:CONTINUE
:: === Встановлення Webhook ===
set FULL_URL=https://api.telegram.org/bot%TOKEN%/setWebhook?url=%URL%/bot
echo Встановлення Webhook: %FULL_URL%
curl %FULL_URL%

echo
pause
```

Лістинг index.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Система Безпеки</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
            background-color: #f3f4f6;
            text-align: center;
            margin: 0;
            padding: 0;
        }
        header {
            background-color: #2c3e50;
            color: white;
            padding: 20px;
            font-size: 24px;
        }
        .status {
            margin-top: 20px;
            font-size: 20px;
```

```
        }
        .armed {
            color: green;
            font-weight: bold;
        }
        .disarmed {
            color: red;
            font-weight: bold;
        }
        .video-container {
            margin-top: 30px;
        }
        .video-container img {
            border: 4px solid #ccc;
            border-radius: 10px;
            box-shadow: 0 0 10px #999;
        }
        .button-group {
            margin-top: 40px;
        }
        .button-group a button {
            background-color: #3498db;
            color: white;
            border: none;
            border-radius: 5px;
            padding: 12px 25px;
            font-size: 16px;
            cursor: pointer;
            margin: 10px;
            transition: background-color 0.3s ease;
        }
        .button-group a button:hover {
            background-color: #2980b9;
        }
    </style>
</head>
<body>

<header>🔐 Система Розпізнавання Облич</header>

<div class="status">
    Стан системи:
    {% if is_armed %}
        <span class="armed">ОХОРОНА УВІМКНЕНА</span>
    {% else %}
        <span class="disarmed">ОХОРОНА ВИМКНЕНА</span>
    {% endif %}
</div>

<div class="video-container">
    <img src="{{ url_for('video') }}" width="720" height="540">
</div>

<div class="button-group">
    <a href="{{ url_for('toggle_security') }}"><button>🔄 Увімк./Вимк. охорону</button></a>
    <a href="{{ url_for('add_face') }}"><button>➕ Додати обличчя</button></a>
    <a href="{{ url_for('show_log') }}"><button>📜 Журнал подій</button></a>
</div>

</body>
</html>
```

Лістинг login.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
    <meta charset="UTF-8">
    <title>Система Безпеки</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
```

```
            background-color: #f3f4f6;
            text-align: center;
            margin: 0;
            padding: 0;
        }
        header {
            background-color: #2c3e50;
            color: white;
            padding: 20px;
            font-size: 24px;
        }
        .status {
            margin-top: 20px;
            font-size: 20px;
        }
        .armed {
            color: green;
            font-weight: bold;
        }
        .disarmed {
            color: red;
            font-weight: bold;
        }
        .video-container {
            margin-top: 30px;
        }
        .video-container img {
            border: 4px solid #ccc;
            border-radius: 10px;
            box-shadow: 0 0 10px #999;
        }
        .button-group {
            margin-top: 40px;
        }
        .button-group a button {
            background-color: #3498db;
            color: white;
            border: none;
            border-radius: 5px;
            padding: 12px 25px;
            font-size: 16px;
            cursor: pointer;
            margin: 10px;
            transition: background-color 0.3s ease;
        }
        .button-group a button:hover {
            background-color: #2980b9;
        }
    </style>
</head>
<body>

<header>🔐 Система Розпізнавання Облич</header>

<div class="status">
    Стан системи:
    {% if is_armed %}
        <span class="armed">ОХОРОНА УВІМКНЕНА</span>
    {% else %}
        <span class="disarmed">ОХОРОНА ВИМКНЕНА</span>
    {% endif %}
</div>

<div class="video-container">
    <img src="{{ url_for('video') }}" width="720" height="540">
</div>

<div class="button-group">
    <a href="{{ url_for('toggle_security') }}"><button>🔄 Увімк./Вимк. охорону</button></a>
    <a href="{{ url_for('add_face') }}"><button>➕ Додати обличчя</button></a>
    <a href="{{ url_for('show_log') }}"><button>📜 Журнал подій</button></a>
```

```
</div>

</body>
</html>
```

Лістинг log.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <title>Система Безпеки</title>
   <style>
      body {
         font-family: 'Segoe UI', sans-serif;
         background-color: #f3f4f6;
         text-align: center;
         margin: 0;
         padding: 0;
      }
      header {
         background-color: #2c3e50;
         color: white;
         padding: 20px;
         font-size: 24px;
      }
      .status {
         margin-top: 20px;
         font-size: 20px;
      }
      .armed {
         color: green;
         font-weight: bold;
      }
      .disarmed {
         color: red;
         font-weight: bold;
      }
      .video-container {
         margin-top: 30px;
      }
      .video-container img {
         border: 4px solid #ccc;
         border-radius: 10px;
         box-shadow: 0 0 10px #999;
      }
      .button-group {
         margin-top: 40px;
      }
      .button-group a button {
         background-color: #3498db;
         color: white;
         border: none;
         border-radius: 5px;
         padding: 12px 25px;
         font-size: 16px;
         cursor: pointer;
         margin: 10px;
         transition: background-color 0.3s ease;
      }
      .button-group a button:hover {
         background-color: #2980b9;
      }
   </style>
</head>
<body>

<header>🔐 Система Розпізнавання Облич</header>
```

```
<div class="status">
   Стан системи:
   {% if is_armed %}
      <span class="armed">ОХОРОНА УВІМКНЕНА</span>
   {% else %}
      <span class="disarmed">ОХОРОНА ВИМКНЕНА</span>
   {% endif %}
</div>

<div class="video-container">
   <img src="{{ url_for('video') }}" width="720" height="540">
</div>

<div class="button-group">
   <a href="{{ url_for('toggle_security') }}"><button>🔁 Увімк./Вимк. охорону</button></a>
   <a href="{{ url_for('add_face') }}"><button>➕ Додати обличчя</button></a>
   <a href="{{ url_for('show_log') }}"><button>📜 Журнал подій</button></a>
</div>

</body>
</html>
```

Лістинг add_face.html

```
<!DOCTYPE html>
<html lang="uk">
<head>
   <meta charset="UTF-8">
   <title>Додати обличчя</title>
   <style>
      body {
         font-family: 'Segoe UI', sans-serif;
         background-color: #f9f9f9;
         margin: 0;
         padding: 0;
         text-align: center;
      }
      header {
         background-color: #2c3e50;
         color: white;
         padding: 20px;
         font-size: 24px;
      }
      form {
         margin-top: 50px;
         background: white;
         display: inline-block;
         padding: 30px;
         border-radius: 10px;
         box-shadow: 0 0 10px #ccc;
      }
      input[type="text"],
      input[type="file"] {
         display: block;
         margin: 15px auto;
         padding: 10px;
         font-size: 16px;
         width: 250px;
         border: 1px solid #ccc;
         border-radius: 5px;
      }
      button {
         background-color: #27ae60;
         color: white;
         padding: 10px 20px;
         font-size: 16px;
         border: none;
         border-radius: 5px;
```

```
            cursor: pointer;
        }
        button:hover {
            background-color: #219150;
        }
        a {
            display: block;
            margin-top: 20px;
            color: #2980b9;
            text-decoration: none;
        }
    </style>
</head>
<body>

<header>✚ Додати нове обличчя</header>

<form method="POST" enctype="multipart/form-data">
    <label for="name">Ім’я:</label>
    <input type="text" name="name" required>

    <label for="image">Фото обличчя (.jpg або .png):</label>
    <input type="file" name="image" accept="image/*" required>

    <button type="submit">Зберегти</button>
</form>


<div style="margin-top: 30px;">
    <a href="{{ url_for('index') }}">
        <button style="background-color: #34495e; color: white; padding: 10px 20px; font-size: 16px; border: none; border-radius:
5px; cursor: pointer;">
            ← Назад на головну
        </button>
    </a>
</div>


</body>
</html>
```

Додаток В. Ілюстративний матеріал

Вінницький національний технічний університет

Кваліфікаційна робота освітнього ступеня магістр
на тему:
«Підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв)»

Виконав: ст.гр. КІТС-23мз
спеціальності: 125 «Кібербезпека та захист інформації»
Освітня програма: «Кібербезпека інформаційних технологій та систем»
Герасимчук А.А.
Керівник: д.т.н., професор, Яремчук Ю.Є.

Вінниця ВНТУ - 2025 рік

Актуальність теми

У сучасному світі, де питання безпеки набувають дедалі більшого значення, системи відеоспостереження відіграють ключову роль у забезпеченні контролю за доступом до приватних і громадських об’єктів. Зростання кількості інцидентів, пов’язаних з несанкціонованим проникненням, крадіжками та загрозами безпеці, вимагає впровадження інтелектуальних технологій, які дозволяють не лише фіксувати події, а й оперативно на них реагувати без втрати часу на хибні спрацювання.

Особливу актуальність має використання систем підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту шляхом розпізнавання обличь — як однієї з найефективніших технологій ідентифікації особи. Інтеграція таких рішень у локальні охоронні комплекси дозволяє суттєво підвищити ефективність моніторингу, зменшити вплив людського фактору та автоматизувати сповіщення у випадку появи невідомих осіб.

У контексті воєнного стану, крадіжки набули нових форм, зокрема розкрадання майна із залишених будівель та житла під час евакуації . Це підкреслює необхідність впровадження інтелектуальних систем відеоспостереження з розпізнаванням обличь, які можуть забезпечити оперативне реагування на загрози та підвищити рівень безпеки громадян.

ЗАВДАННЯ ДОСЛІДЖЕННЯ

Для досягнення заданої мети було поставлено наступні завдання:

1. Аналіз існуючих систем безпеки розумного будинку та технологій штучного інтелекту в цій сфері.
2. Розробка архітектури системи, що включає мережу камер, алгоритми обробки та інтерфейс користувача.
3. Реалізація механізму збору та обробки даних від камер у реальному часі.
4. Використання алгоритмів машинного навчання для виявлення незнайомих осіб, аналізу інцидентів то сповіщення про них користувача.
5. Тестування системи у різних сценаріях та її оптимізація.

МЕТА ДОСЛІДЖЕННЯ

Розробка та впровадження системи підвищення захищеності об’єктів розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв), що забезпечується шляхом автоматичного виявлення осіб, їх ідентифікацією, веденням журналів подій та надсиланням сповіщень користувачу для підвищення рівня безпеки контрольованого об’єкта.

ОБ'ЄКТ ТА ПРЕДМЕТ ДОСЛІДЖЕННЯ. НАУКОВА НОВИЗНА

Об'єктом дослідження є розумні будинки та їхні системи автоматизації і охорони.

Предметом дослідження є використання технологій штучного інтелекту для підвищення захищеності об’єктів системи розумного будинку для аналізу інцидентів(сценаріїв)» .

Наукова новизна. Запропоновано новий підхід до використання алгоритмів штучного інтелекту для аналізу інцидентів (сценаріїв) у розумному будинку. Розроблено адаптивну систему, що здатна аналізувати та навчатися на основі історичних, а також введених механічно даних та вдосконалювати свою ефективність у реальному часі. Використання методів машинного навчання для виявлення дійсних порушень периметру розумного будинку та запровадження системи доступу до розумного будинку.

ВИКОРИСТАНІ ТЕХНОЛОГІЇ

Python - інтерпретована об'єктно-орієнтована мова програмування високого рівня із суворою динамічною типізацією.

Flask - мікрофреймворк для вебдодатків, створений з використанням Python.

OpenCV - бібліотека функцій та алгоритмів комп'ютерного зору, обробки зображень і чисельних алгоритмів загального призначення з відкритим кодом.

face_recognition - опенсорс-бібліотека, заснована на C++ бібліотеці dlib.

Telegram Bot API - це інтерфейс, який дозволяє розробникам створювати та керувати Telegram-ботами.

HTML/CSS - стандартизована мова розмітки документів для перегляду вебсторінок у браузері.

Алгоритм роботи системи

Ініціалізація системи - Запуск Flask-сервера, підключення вебкамери, завантаження бази відомих облич.

Обробка відеопотоку в реальному часі - Кожен кадр з камери зменшується для прискорення обробки, після чого виконується виявлення облич.

Розпізнавання облич - Для кожного виявленого обличчя обчислюється дескриптор (вектор ознак), який порівнюється з базою відомих облич.

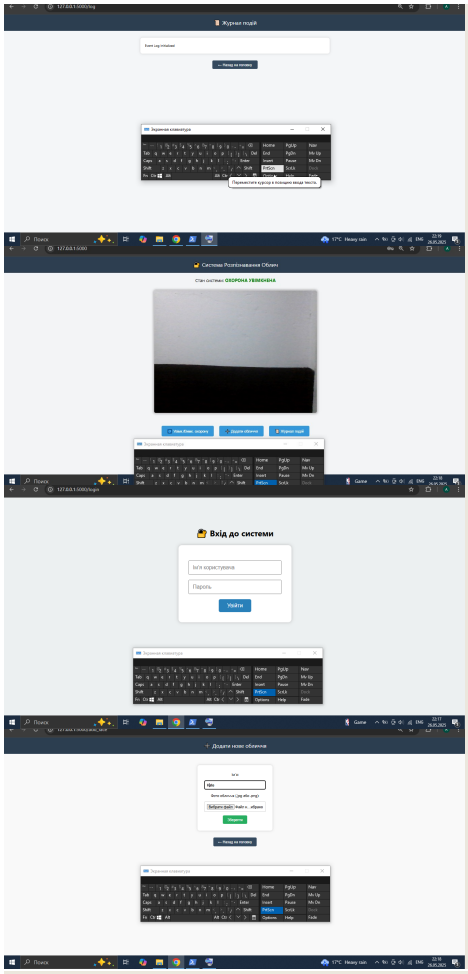
Класифікація особи
- Якщо обличчя невідоме або зареєстровано як злочинець — система переходить до наступного кроку.

- Якщо обличчя знайдено у базі — виводиться ім'я особи, за необхідності вмикається відповідний сценарій

Реакція системи

У режимі охорони при виявленні «невідомого» обличчя:
- запускається сигнал тривоги (звуковий сигнал),
- надсилається сповіщення через Telegram-бота, веб-інтерфейс
- записується подія у лог-файл.

Продовження роботи системи



Інтерфейс користувача

Вебінтерфейс дозволяє переглядати відео, додавати обличчя, переглядати лог та вмикати/вимикати охорону.

Режими охорони

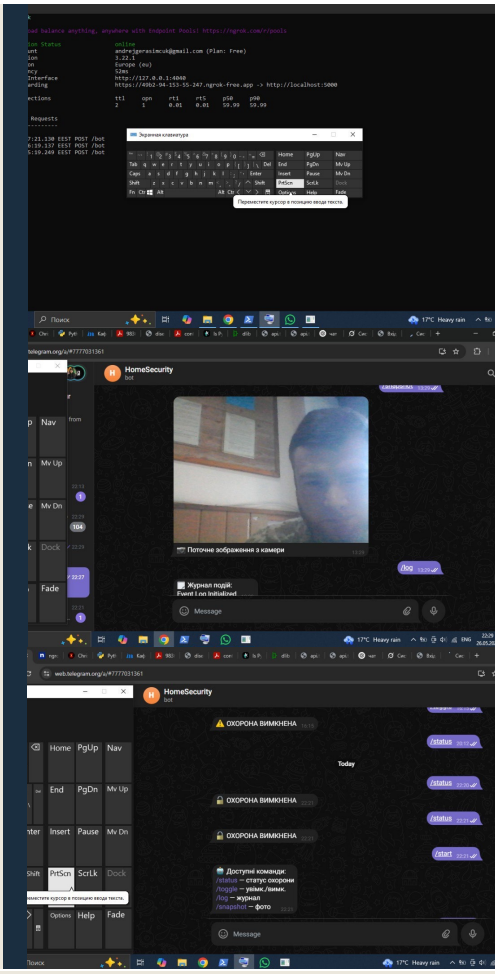
- Режим «На охороні» — система розпізнає та реагує на невідомі обличчя.
- Режим «Вимкнено» — обробка не виконується.
- Режим «Сон» — обробка виконується в заданий час.

Telegram-бот

Користувач отримує повідомлення, фотографії, може керувати охороною та переглядати журнал.

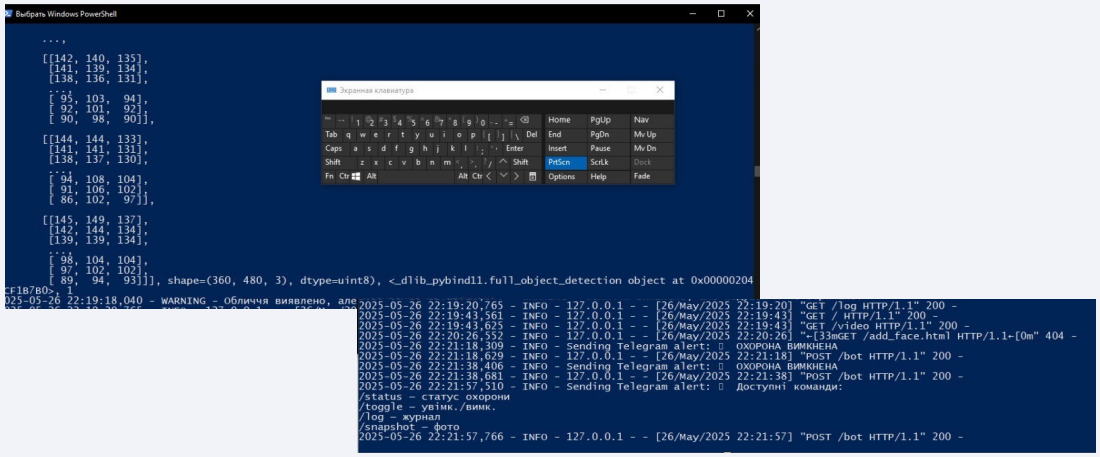
На рисунках зафіксовано запуск та успішну роботу webhook, який налаштовано за допомогою ngrok. Для того щоб не налаштовувати webhook кожного разу вручну, це було автоматизовано за допомогою .bat файлу.

Функціонал телеграм боту та веб інтерфейсу дублюється, відповідно охорону можна ввімкнути з телеграм боту, а вимкнути через веб інтерфейс.



Результати

Система працює стабільно, успішно розпізнає обличчя та зрозуміло для користувача коментує помилки в разі їх виявлення, реагує на події та зручно керується через інтерфейс.



Висновки

У процесі розробки було створено систему підвищення захищеності об’єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів(сценаріїв), яка успішно виконує завдання з підвищення рівня захисту будинків (об'єктів), шляхом ідентифікації осіб у режимі реального часу. Реалізація побудована на основі бібліотек face_recognition, OpenCV та фреймворку Flask, що дозволило досягти високої точності розпізнавання та забезпечити простий, інтуїтивно зрозумілий веб інтерфейс.

Система забезпечує стабільну роботу з відео потоком; точне виявлення невідомих осіб; ведення журналу подій; інтеграцію з Telegram-ботом для оперативного сповіщення користувача та авторизацію для безпечного доступу до функцій керування.

Таким чином, реалізоване програмне забезпечення повністю відповідає поставленій меті, має практичну цінність і може бути впроваджене як в особистих, так і в комерційних системах безпеки. Подальші покращення можуть включати додаткове навчання моделі на глибоких нейронних мережах, підтримку декількох камер та використання хмарних сховищ для архівації відео.

Дякую за увагу!

Додаток Г. Перевірка на антиплагіат

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: Підвищення захищеності об'єктів системи розумного будинку на основі штучного інтелекту для аналізу інцидентів (сценаріїв)

Тип роботи: магістерська кваліфікаційна робота

Підрозділ Кафедра менеджменту на безпеки інформаційних систем
Факультет менеджменту та інформаційної безпеки
Гр. КІТС-23мз

Керівник проф. Яремчук Ю.Є. _____

Показники звіту подібності

Strike Plagiarism	
Оригінальність	83,63%
Загальна схожість	16,37%

Аналіз звіту подібності (відмітити потрібне)

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

☐ Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

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

Заявляю, що ознайомлений (-на) з повним звітом подібності, який був згенерований Системою щодо роботи (додається)

Автор _____ Герасимчук А.А.
(підпис) (прізвище, ініціали)

Опис прийнятого рішення

☐ **Допустити до захисту.**

_____

_____

_____

_____

Особа, відповідальна за перевірку _____ Коваль Н.П.
(підпис) (прізвище, ініціали)

Експерт _____ _____
(за потреби) (підпис) (прізвище, ініціали, посада)