

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій
(повне найменування факультету)

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

до дипломного проекту (роботи)
бакалавр
(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ
БЕЗПЕКИ НА ОСНОВІ БЕЗДРОТОВИХ ТЕХНОЛОГІЙ
DEVELOPMENT OF SECURITY SYSTEM SOFTWARE
BASED ON WIRELESS TECHNOLOGIES

Виконав: студент 4 курсу, групи КНТ-211сп
Спеціальності 122 Комп'ютерні науки
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Комп'ютерні науки

ЩЕРБАКОВ К.Ю.
(ПРИЗВИЩЕ та ініціали)

Керівник ПАРХОМЕНКО А.В.
(ПРИЗВИЩЕ та ініціали)

Рецензент ЗЕЛЕНЬОВА І.Я.
(ПРИЗВИЩЕ та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
Сергій СУББОТІН
“ ” 2024 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ЩЕРБАКОВА Кирила Юрійовича
(ПРИЗВИЩЕ, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмного забезпечення системи безпеки на основі бездротових технологій. Development of Security System Software Based on wireless Technologies

керівник проєкту (роботи) к.т.н., доцент, ПАРХОМЕНКО Анжеліка Володимирівна,
(науковий ступінь, вчене звання, ПРИЗВИЩЕ, ім'я, по батькові)

затверджені наказом закладу вищої освіти від “24” квітня 2024 року № 168

2. Строк подання студентом проєкту (роботи) 03 червня 2024 року

3. Вихідні дані до проєкту (роботи) рекомендована література, завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Проєктування програмного забезпечення 3. Конструювання програмно-апаратного забезпечення. 4. Керівництво користувача та приклад роботи.

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

Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	ПРИЗВИЩЕ, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-4 Основна частина	ПАРХОМЕНКО А.В., доцент		
Нормоконтроль	БЄЛОВА А.В., асистент		

7. Дата видачі завдання «24» квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	1 тиждень	Розділ 1
3	Вибір мови програмування та інших технологій розробки.	2 тиждень	Розділ 2
4	Розробка архітектури програми.	2 тиждень	Розділ 3
5	Розробка програми.	3-4 тижні	Розділ 3, 4
6	Тестування та експериментальне дослідження програмного забезпечення.	5 тиждень	Розділ 4
7	Оформлення пояснювальної записки та документів до неї.	6 тиждень	Додатки
8	Нормоконтроль та рецензування.	7 тиждень	
9	Захист роботи.	8 тиждень	

Студент(ка)

_____ Кирило ЩЕРБАКОВ
(підпис) (Ім'я ПРИЗВИЩЕ)

Керівник проєкту (роботи)

_____ Анжеліка ПАРХОМЕНКО
(підпис) (Ім'я ПРИЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 72 с., 10 табл., 23 рис., 2 дод., 22 джерела.

СИСТЕМА БЕЗПЕКИ, БЕЗДРОТОВІ ТЕХНОЛОГІЇ, РОЗПІЗНОВАННЯ ОБЛИЧЧЯ, ESP32, СИСТЕМА РОЗУМНИЙ БУДИНОК.

Об'єкт дослідження – розпізнавання облич для систем безпеки.

Предмет дослідження – методи та програмно-апаратні засоби розпізнавання облич для систем безпеки будинку.

Мета роботи – розробка програмно-апаратного рішення на основі бездротових технологій для системи безпеки з розпізнаванням облич.

Матеріали, методи та технічні засоби: Windows 11, Pentium G5400, 4 ГБ ОЗУ, середовище розробки Arduino IDE, мова програмування Arduino C, модуль ESP32-CAM, датчик руху, камера, бездротовий маршрутизатор.

Результати. Створено програмне забезпечення для контролера ESP32-CAM для системи безпеки будинку, що дозволяє додавати та редагувати список облич дозволених осіб, та контролювати появу руху або невідомих осіб біля будинку з подальшим оповіщенням власника у Telegram.

Висновки. Система безпеки “розумного” будинку може працювати більш ефективно завдяки розробленому рішенню на основі недорогих, доступних програмних та апаратних компонентів.

Галузь використання – системи безпеки квартир та приватних будинків чи інших території, які потребують контролю проходу.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 72 pages, 10 tables, 23 figures, 2 appendixes, 22 sources.

WIRELESS TECHNOLOGIES, SECURITY SYSTEM, FACE RECOGNITION, ESP32, SMART HOUSE SYSTEM.

Object of study is the security system of the house.

Subject study are methods and hardware/software for face recognition for home security systems.

The purpose of the work is to creation of a software and hardware solution based on wireless technology for home security using face recognition.

Materials, methods, and tools: Windows 11, Pentium G5400, 4GB RAM, Arduino IDE Development Environment, Arduino C Programming Language, ESP32-CAM, Motion Sensor, Camera, Wireless Router.

Results. Software for the ESP32-CAM controller for the home security system has been created, which allows you to add and edit the list of faces of permitted persons, and control the appearance of movement or unknown persons near the house with subsequent notification of the owner in Telegram.

Conclusions. A smart home security system can work more efficiently thanks to a developed solution based on inexpensive, accessible software and hardware components.

The field of use are security systems for apartments and private houses, or other areas that need to control the passage.

ЗМІСТ

	С.
Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Види систем безпеки.....	10
1.2 Підходи до створення програмного забезпечення систем безпеки	11
1.3 Огляд програм відеоспостереження для забезпечення безпеки.....	13
1.4 Дослідження методів розпізнавання обличчя	18
1.5 Бездротові технології.....	19
1.6 Постановка задач дипломної роботи	21
2 Проєктування програмного забезпечення	23
2.1 Аналіз вимог	23
2.2 Архітектура системи.....	23
2.3 Обґрунтування вибору апаратних засобів.....	24
2.4 Обґрунтування вибору програмних засобів	27
2.5 Висновки за розділом 2	31
3 Конструювання програмно-апаратного забезпечення	32
3.1 Схема алгоритму функціонування системи	32
3.2 Особливості програмної реалізації.....	34
3.3 Схема апаратної частини.....	36
3.4 Висновки за розділом 3	37
4 Керівництво користувача та приклад роботи.....	38
4.1 Створення Telegram бота.....	38
4.2 Налаштування змінних	41
4.3 Завантаження програми до контролера	41
4.4 Початок роботи.....	45
4.5 Висновки за розділом 4	49
Висновки	50
Перелік джерел посилань	51

Додаток А Текст програми.....	53
Додаток Б Слайди презентації	65

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
IDE	– Integrated Development Environment;
IoT	– Internet of Things;
LBP	– Local Binary Patterns;
LBPH	– Local Binary Patterns Histogram;
SMS	– Short Message Service;
SVM	– Support vector machine;
SMS	– Short Message Service;
ПЗСБ	– програмне забезпечення для систем безпеки.

ВСТУП

Сучасні системи безпеки “розумного” будинку дозволяють виявляти потенційні загрози, такі як рух на території будинку під час відсутності мешканців, відкриті двері або вікна, дим або вогонь. Такі системи зазвичай використовують засоби відеоспостереження, датчики руху тощо. Завдяки інтеграції різних датчиків та використанню програмних засобів, системи безпеки “розумного” будинку можуть ефективно реагувати на потенційні загрози та надавати швидку реакцію, включаючи автоматичне повідомлення власника або відповідних служб безпеки.

Тому при розробці програмного забезпечення для систем безпеки “розумного” будинку розробник повинен вирішити кілька ключових питань:

- забезпечення надійності та ефективності: програмне забезпечення для систем безпеки (ПЗСБ) повинне працювати надійно у всіх умовах і забезпечувати високий рівень захисту;
- керування доступом: ПЗСБ повинне забезпечувати контроль доступу до об'єктів і інформації;
- моніторинг та аналіз: ПЗСБ повинне забезпечувати можливість моніторингу подій та аналізу даних для виявлення потенційних загроз.

Використання ефективних систем безпеки “розумного” будинку може значно підвищити рівень захисту майна та безпеки мешканців. Це також може сприяти зменшенню кількості злочинів у житлових районах та підвищити загальний рівень комфорту від проживання в будинку.

У дипломній роботі бакалавра розглядається актуальне завдання подальшого розвитку методів забезпечення системи безпеки “розумного” будинку та розробки недорогих програмно-апаратних рішень.

Метою роботи є розробка програмно-апаратного рішення на основі бездротових технологій для системи безпеки з розпізнаванням облич.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Види систем безпеки

У сучасному світі існує багато загроз, які стосуються різних сфер життя. Для деяких галузей захист від одних загроз може бути більш важливим, ніж від інших. Тому розроблено різні види систем безпеки, щоб ефективно вирішувати найбільш поширені проблеми, що виникають, далі наведено список найпоширеніших систем безпеки на рисунку 1.1 [1]-[2].

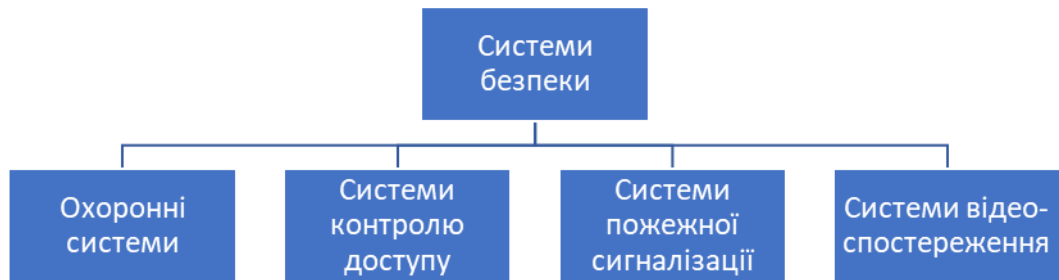


Рисунок 1.1 – Види систем безпеки [1]-[2]

Охоронні системи – це комплексні рішення, які включають різноманітні датчики, сирени і системи сповіщення. Датчики виявляють рух, вторгнення та інші неправомірні події на об'єкті, а системи сповіщення інформують охорону та персонал про інциденти. Такі системи можуть бути встановлені як у відкритих просторах, так і у приміщеннях. Крім цього, сучасні системи безпеки можуть бути інтегровані з хмарними сервісами, що дозволяють отримувати віддалений доступ до відеоспостереження та інших функцій забезпечення безпеки [1]-[2].

Система контролю доступу – це система, яка управляє доступом до об'єкту. Це може бути здійснено різними методами: від звичайних замків до більш складних систем, які використовують картки доступу або біометричні

дані. Система контролю доступу забезпечує безпеку, запобігаючи несанкціонованому входу на об'єкт [1]-[2].

Система пожежної сигналізації – це система, яка виявляє пожежу та повідомляє присутніх на об'єкті та рятувальні служби. Вона включає в себе датчики диму, тепла і газу, які реагують на ознаки виникнення пожежі. При виявленні пожежі датчиками, система автоматично активує сирени та сповіщає присутніх. Додатково, система може надсилати повідомлення на диспетчерський пульт або на мобільні телефони відповідальних осіб [1]-[2].

Система відеоспостереження – це ефективний засіб забезпечення безпеки на об'єкті. Вона складається з камер, розміщених у різних точках об'єкту, та системи запису, яка зберігає отримані зображення. Відеоспостереження корисне як для запобігання злочинів, так і для розслідування подій на об'єкті. Сучасні системи відеоспостереження можуть бути підключені до хмарних сервісів, що надають можливість віддаленого доступу до збережених даних [1]-[2].

1.2 Підходи до створення програмного забезпечення систем безпеки

Безпека є життєво важливим аспектом розробки програмного забезпечення, який ніколи не можна ігнорувати.

Забезпечення безпеки у розробці вимагає системного підходу, що враховує усі етапи життєвого циклу програмного забезпечення. Використання цього підходу дозволяє гарантувати, що на кожному етапі розробки буде забезпечено високий рівень безпеки.

Починати треба з визначення потреб і обмежень безпеки системи. Задokumentувати ці вимоги, щоб вони служили орієнтиром протягом усього процесу розробки.

Налагодити процес проведення регулярних перевірок коду, щоб визначити та виправити потенційні слабкі місця.

Включити тестування безпеки в процес розробки. Це може включати тестування на проникнення, сканування вразливостей і методи тестування, орієнтовані на безпеку. Регулярно тестувати програмне забезпечення, щоб виявити будь-які наявні слабкі місця або вразливості [3].

Проводити моделювання, щоб виявити потенційні загрози безпеці та активно реагувати на них. Це передбачає аналіз архітектури системи, виявлення потенційних векторів атак і налаштування відповідних засобів контролю безпеки [3].

Впроваджувати методи безпечного розгортання, такі як безпечне керування конфігурацією та безпечні процеси оновлення. Регулярно перевіряйте програмне забезпечення на наявність будь-яких дірок у безпеці та своєчасно застосовуйте виправлення та оновлення [3]-[4].

Також треба слідкувати за самим процесом розробки та не допускати поширених помилок.

Одним із найпоширеніших ризиків є небезпечна практика кодування. Недосвідчені або поспішні розробники можуть ненавмисно впровадити вразливості у свій код, створивши відкрите запрошення для експлуатації. Наприклад, розробник може не перевірити належним чином введені користувачем дані, що дозволяє зловмисникам впроваджувати шкідливий код у систему [3]- [4].

Слабкі протоколи автентифікації та авторизації, такі як використання паролів, які легко вгадати, або відсутність належного контролю доступу на основі ролей користувачів, можуть полегшити зловмисникам отримання несанкціонованого доступу до конфіденційної інформації. Це може призвести до витоку даних, крадіжки особистих даних та інших серйозних наслідків [4].

Команди розробників програмного забезпечення часто стикаються з ризиком вразливостей третіх сторін. Інтегруючи сторонні бібліотеки або компоненти у своє програмне забезпечення, розробники покладаються на заходи безпеки, запроваджені цими постачальниками, однак якщо ці сторонні

компоненти мають слабкі місця або не оновлюються регулярно, вони можуть стати слабкою ланкою в загальній системі безпеки програмної системи [4].

1.3 Огляд програм відеоспостереження для забезпечення безпеки

1.2.1 Blue Iris

Blue Iris є однією з програм для відеоспостереження, яка надає розширений набір функцій і високий рівень налаштувань. Роздивимось її можливості:

- Blue Iris підтримує велику кількість моделей IP-камер, включаючи камери від різних виробників, таких як Axis, Dahua, Hikvision та багатьох інших. Вона також підтримує USB-камери та інші відеозахоплюючі пристрої;
- програма має вбудований детектор руху з різними налаштуваннями чутливості, областей детекції та дій, які викликаються при виявленні руху. це дозволяє зменшити кількість хибних спрацювань;
- Blue Iris забезпечує можливість віддаленого доступу через веб-інтерфейс або мобільний застосунок. Це дозволяє переглядати відеопотік з камер у реальному часі і переглядати записи з будь-якого місця з доступом до Інтернету;
- програма пропонує деякі розширені функції відеоаналітики, такі як розпізнавання автомобільних номерів, відстеження об'єктів і аналіз дорожнього руху;
- Blue Iris надає велику кількість параметрів налаштувань, що дозволяє користувачам вибирати оптимальні налаштування для їх потреб;
- програма має вбудовану можливість резервного копіювання записів на зовнішні носії [5].

Хоча Blue Iris має багато переваг, включаючи розширений функціонал і високий рівень налаштувань, також присутні і недоліки:

- Blue Iris може бути досить вимогливою до ресурсів системи, особливо при використанні багатьох камер або високих роздільних здатностей відеопотоку. Це може вимагати потужного комп'ютера або сервера для оптимальної роботи;
- хоча Blue Iris має широкий набір налаштувань, це також може бути викликом для користувачів, які не мають досвіду у встановленні і налаштуванні систем відеоспостереження;
- Blue Iris є комерційним програмним забезпеченням, що означає, що вам потрібно придбати ліцензію для користування ним. Хоча є пробний період, після його закінчення ви повинні буде придбати ліцензію для продовження використання;
- Blue Iris доступна лише для операційних систем Windows, тому користувачам, які використовують інші операційні системи, може бути необхідно шукати альтернативні рішення [5].

1.2.2 Xeoma

Хеома - це програмне забезпечення для відеоспостереження, яке надає широкі можливості для налаштування систем відео нагляду, а саме:

- Хеома відомий своїм інтуїтивно зрозумілим інтерфейсом, який робить його легким у використанні для новачків. Він має дружній графічний інтерфейс, що дозволяє швидко налаштувати систему відеоспостереження без великих зусиль;
- Хеома пропонує широкий спектр функцій, таких як детектор руху, розпізнавання облич, звуку і тепловізії, а також можливість встановлення різних правил і сповіщень;
- Хеома має мобільний застосунок для iOS та Android, що дозволяє зручно переглядати відео потік з камер у реальному часі і переглядати архів записів з будь-якого місця з доступом до Інтернету;

- програма доступна для встановлення на різних платформах, включаючи Windows, Linux, Mac OS і деякі відеореєстратори NAS;
- Хеомта дозволяє легко додавати і налаштовувати нові камери та функції, що робить його досить гнучким і розширюваним рішенням;
- програма має вбудовані засоби захисту, такі як шифрування даних і автентифікація, що забезпечує високий рівень безпеки вашої системи відеоспостереження [6].

Хеомта має багато переваг, та слід також враховувати, що вона може не мати деяких розширених функцій, які доступні в інших програмах для відеоспостереження та інші недоліки:

- Хеомта має безкоштовну версію з обмеженими функціями, більш розширені можливості доступні лише за плату. Це може зробити Хеомта менш доступним для користувачів з обмеженим бюджетом або домашнього використання;
- порівняно з іншими популярними програмами для відеоспостереження, такими як Blue Iris або iSpy, Хеомта може мати меншу кількість активних користувачів і меншу спільноту. Це може ускладнити отримання допомоги або рішення проблем, які можуть виникнути;
- Хеомта підтримує багато моделей IP-камер, але усе одно можуть виникнути проблеми з сумісністю з деякими пристроями або виробниками;
- хоча Хеомта має широкий набір функцій, в порівнянні з деякими іншими програмами відеоспостереження вона може не мати деяких розширених можливостей, таких як розпізнавання номерів автомобілів або аналіз відео за допомогою штучного інтелекту;
- можливість виникнення проблем з продуктивністю Хеомта при використанні великої кількості камер або при високих навантаженнях системи [6].

1.2.3 iSpy

iSpy - це безкоштовна відкрита програма для відеоспостереження з широким набором функцій і можливостей. Ось огляд деяких основних аспектів iSpy:

- однією з основних переваг iSpy є те, що вона є безкоштовною для використання. Це робить її привабливим варіантом для користувачів з обмеженим бюджетом або для домашнього використання;
- iSpy має широкий набір функцій, включаючи детектор руху, звуковий детектор, сповіщення через електронну пошту або SMS, а також підтримку різних типів камер і пристроїв;
- програма дозволяє зручний віддалений доступ через веб-інтерфейс або мобільний застосунок, що дозволяє переглядати відео потік з камер у реальному часі і переглядати архів записів з будь-якого місця з доступом до Інтернету;
- iSpy має відкрите програмне забезпечення і активну спільноту користувачів, що дозволяє легко розширювати її функціонал за допомогою різних плагінів і застосунків;
- існує можливість перегляду вихідних кодів iSpy і внесення власних внесків або виправлень до програми;
- можливість інтегрувати майже будь-який пристрій відеозахоплення якщо його можливо під'єднати до комп'ютера [7].

Хоча iSpy має багато переваг, варто також враховувати деякі можливі недоліки:

- налаштування і налагодження iSpy може бути складним завданням, особливо для новачків або користувачів без досвіду з програмами відеоспостереження;
- iSpy може мати обмежену підтримку деяких моделей камер і пристроїв;

- можливість виникнення проблем з продуктивністю і стабільністю програми, особливо при використанні багатьох камер або при високих навантаженнях системи;
- порівняно з деякими комерційними програмами для відеоспостереження, iSpy може не мати деяких розширених функцій або аналітики, таких як розпізнавання номерів автомобілів або відстеження об'єктів [7].

Порівняльну характеристику програмного забезпечення для відеоспостереження наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння програмного забезпечення для відеоспостереження [5]-[7]

Критерій порівняння	Назва програми		
	Blue Iris	Хеoma	Ispy
Велика вартість	+	+-	-
Підтримка різних платформ	-	+	+-
Сумісність з різним обладнанням	+	+-	-
Великі вимоги до системи	+	-	+-
Велика спільнота	+	+-	+
Відкритий код	-	-	+
Віддалений доступ	+	+	+
Великий функціонал	+	+-	-
Сповіщення	-	-+	-+

1.4 Дослідження методів розпізнавання обличчя

Проаналізуємо методи, які застосовують для розпізнавання облич.

Метод EigenFaces базується на аналізі головних компонентів, що застосовується для зменшення розмірності зображення обличчя до вектора з невеликою кількістю головних компонентів. Цей вектор потім порівнюється з еталонними векторами, які зберігаються в базі даних. Одержаний навчальною вибіркою, набір власних векторів використовується для кодування зображень обличчя представлених у вигляді комбінації цих власних векторів [8].

FisherFaces є покращеною версією методу EigenFace. Основна відмінність полягає у використанні кластеризації навчальних векторів. Для цього проводяться розрахунки внутрішньокласових та між класових коваріаційних матриць. У порівнянні з EigenFaces, цей метод є більш стійким до змін умов освітленості [8].

Метод LBPH (Local Binary Patterns Histograms) використовується для розпізнавання облич та об'єктів на зображеннях. Він базується на аналізі локальних бінарних шаблонів (LBP) і гістограм. Основна мета полягає в описі текстурних особливостей зображення [9]-[10].

Для кожного пікселя на зображенні порівнюється його інтенсивність з інтенсивностями сусідніх пікселів. Результат порівняння формує бінарний шаблон, де біт встановлюється в 1, якщо інтенсивність сусіднього пікселя вища, і 0, якщо нижча [9]-[10].

Для кожного окремого пікселя формується LBP-шаблон, і відбувається підрахунок кількості входжень кожного можливого шаблону на зображенні. Ці значення групуються в гістограму, яка відображає розподіл LBP-шаблонів на зображенні [9]-[10].

Отримані гістограми можна використовувати для навчання класифікатора, такого як машини опорних векторів (SVM) або нейронні

мережі. Після навчання класифікатор може розпізнавати обличчя або об'єкти на нових зображеннях, порівнюючи їхні гістограми з вже відомими [9]-[10].

Таким чином, метод LBPH використовується для створення опису текстурних особливостей зображення і подальшої класифікації зображень за цими описами. Це ефективний метод для розпізнавання обличч і обробки зображень, зокрема в системах відеоспостереження та системах безпеки [9]-[10].

Порівняльну характеристику методів розпізнавання обличчя наведено у таблиці 1.2.

Таблиця 1.2 – Порівняння методів розпізнавання обличчя [8]-[10]

Критерій порівняння	Назва методу		
	EigenFaces	FisherFaces	LBPH
Ефективність розпізнавання	+/-	+	+
Стійкість до змін	-	+/-	+
Великі вимоги до навчальних даних	+	+	+/-
Швидкість розпізнавання	-	-	+

1.5 Бездротові технології

У сучасних пристроях головну роль відіграють мікроконтролери, які виступають посередниками між користувачем та пристроями. Застосунки, які знаходяться на користувацьких пристроях, відповідають за взаємодію з оточуючим середовищем, включаючи керування сповіщеннями через радіоканали. Універсальні чіпи у користувацьких пристроях забезпечують бездротовий зв'язок з іншими пристроями, через різні стандарти бездротового зв'язку [11]-[12].

Протоколи віддаленого зв'язку представляють собою набір промислових стандартів для бездротових персональних мереж, які функціонують як інтерфейс для взаємодії між сумісними пристроями. Вони дозволяють пристроям обмінюватися інформацією та забезпечують можливість дистанційного керування, наприклад, за допомогою спеціального пульта або мобільного застосунка на телефоні [11]-[12].

Розглянемо основні методи обміну даними на відстані, які доступні наразі для використання. А саме методи які не вимагають спеціальних дозволів, є безкоштовними та відкритими для створення власних продуктів, вони наведені у таблиці 1.3.

В ході дослідження було виявлено що най розповсюджений стандарт бездротових мереж є Wi-Fi [11]-[12].

Wi-Fi — це технологія бездротових локальних мереж, що базується на стандартах IEEE 802.11. Основні діапазони частот Wi-Fi — 2,4 ГГц (2412 МГц - 2472 МГц) та 5 ГГц (5160 МГц - 5825 МГц). Сигнал Wi-Fi може передаватися на кілька кілометрів навіть при низькій потужності передачі, проте для прийому сигналу від звичайного Wi-Fi маршрутизатора на великих відстанях потрібна антена з високим коефіцієнтом підсилення [11]-[12].

Основні переваги Wi-Fi полягають у можливості створення бездротових мереж без необхідності у прокладці кабелів, що спрощує процес розгортання та розширення інфраструктури. Це особливо важливо у випадках, коли кабелі неможливо прокласти, наприклад, на відкритих майданчиках чи в історичних спорудах. Також Wi-Fi забезпечує доступ до мережі для мобільних пристроїв, що широко поширені на ринку. Ви не обмежені місцем та можете користуватися Інтернетом у будь-якому зручному для вас місці. Також в межах Wi-Fi зони можуть підключатися до Інтернету одночасно кілька користувачів з різних пристроїв. [11]-[12].

Таблиця 1.3 – Стандарти бездротової передачі даних [11]-[12].

Характеристика	Назва стандарту		
	Wi-Fi	Bluetooth	ZigBee
Частотний діапазон	2.4 ГГц, 5 ГГц	2.4 ГГц, 2.1 ГГц	2.4 ГГц
Швидкість передачі даних	До 9.6 Гбіт/с	До 3 Мбіт/с	До 250 Кбіт/с
Радіус дії	До 100 м (в приміщенні), до 1 км (на відкритому повітрі)	До 100 м	До 100 м
Масштабованість	Висока	Обмежена	Висока
Призначення	Високошвидкісна передача даних	Передача даних між пристроями на короткій відстані	Сенсорні, мережі, автоматизація

1.6 Постановка задач дипломної роботи

За результатами проведеного аналізу можна зробити висновок що у сучасний час існує розмаїття програм для відеоспостереження. Проте деякі з них мають значні вартісні витрати, тоді як інші можуть бути складними у використанні або не мати можливості сповіщати власнику про небезпеку.

Тому розробка програмно-апаратного рішення для системи безпеки “розумного” будинку за допомогою розпізнавання облич є актуальною задачею.

Метою роботи є створення програмно-апаратного рішення на основі бездротових технологій для безпеки будинку з використанням розпізнавання облич.

Для досягнення цієї мети у роботі бакалавра необхідно виконати такі задачі:

- провести аналіз програмно-апаратних засобів для забезпечення системи безпеки “розумного” будинку;
- здійснити проєктування програмного забезпечення;
- створити програмне забезпечення для системи безпеки “розумного” будинку;
- виконати тестування розробленого програмного забезпечення для перевірки його ефективності.

2 ПРОЄКТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

2.1 Аналіз вимог

За проведеним аналізом предметної області було встановлено яким вимогам має відповідати програмно-апаратне рішення.

Функціональні вимоги:

- програмно-апаратного рішення повинно мати можливість працювати автономно;
- можливість підключитись до мережі інтернет;
- розпізнавати обличчя при наближенні людини;
- можливість додавати обличчя дозволених осіб;
- можливість видаляти обличчя дозволених осіб;
- надсилання повідомлення власнику у разі відсутності обличчя людини в списку дозволених;
- надсилання повідомлення власнику у разі неможливості розпізнати обличчя.

Нефункціональні вимоги:

- доступна елементна база;
- невелике споживання електроенергії.

2.2 Архітектура системи

Архітектура системи була розроблена на основі аналізу вимог і представлена на рисунку 2.1.

Архітектура системи складається з трьох основних частин:

- бездротового маршрутизатора;
- месенджера;
- контролера.

Бездротовий маршрутизатор (роутер), має бути під'єднаний до мережі інтернет, він використовується щоб надати доступ до інтернету контролеру щоб тої міг надсилати повідомлення користувачу.

Контролер основний модуль до якого під'єднано детектор руху та модуль камери, на ньому зберігаються обличчя та виконується їх порівняння, надсилається повідомлення у разі можливої небезпеки користувачу через API месенджера.

Месенджер програмне забезпечення на яке користувачу приходитимуть повідомлення у разі можливої небезпеки.

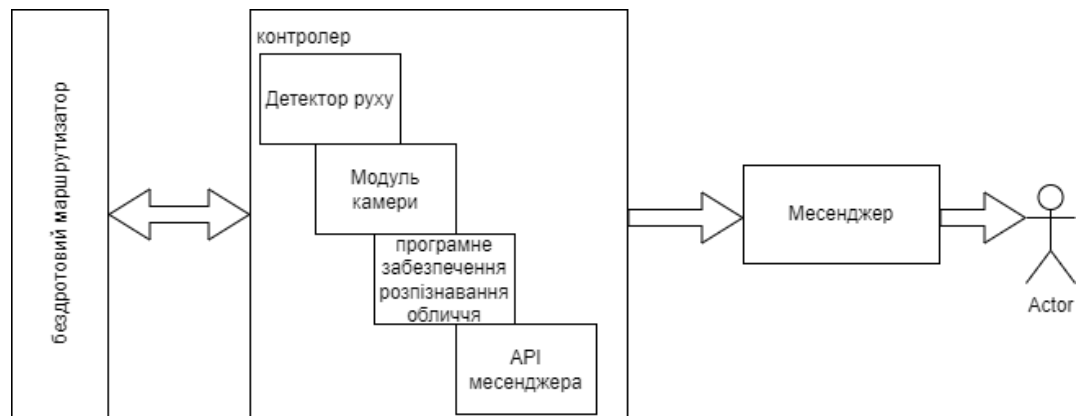


Рисунок 2.1 – Узагальнена архітектура системи

2.3 Обґрунтування вибору апаратних засобів

Для розробки проєкту апаратною базою було обрано контролер сімейства ESP32 а саме ESP32-CAM.

ESP32 – це платформа для розробки вбудованих систем, яка використовує мікроконтролери ESP32 від Espressif Systems. Платформа ESP32 дозволяє розробникам створювати програмне забезпечення для різноманітних застосувань, включаючи Інтернет речей (IoT), автоматизацію, робототехніку та інше [10]-[12].

ESP32 працює на мові програмування C та C++, що робить його доступним для широкого кола розробників з різним досвідом. Мови C та C++

відомі своєю ефективністю та швидкодією, що робить їх ідеальними для вбудованих систем з обмеженими ресурсами [13]-[15].

ESP32 підтримує різноманітні функції, включаючи бездротове з'єднання Wi-Fi та Bluetooth, а також можливості для роботи з різними сенсорами та периферійними пристроями. Його гнучкість і потужність дозволяють розробникам створювати різноманітні проєкти від простих IoT пристроїв до складних систем автоматизації [13]-[15].

ESP32 також підтримується активною спільнотою розробників, що сприяє розвитку та вдосконаленню платформи. Це дозволяє швидко знаходити відповіді на питання та розв'язувати проблеми, що можуть виникнути під час розробки програмного забезпечення для ESP32 [13]-[15].

Отже, ESP32 є потужною та гнучкою платформою для розробки вбудованих систем, яка використовує мови програмування C та C++, що робить його привабливим вибором для різноманітних проєктів у сфері IoT та вбудованих систем.

Для обґрунтування вибору у якості апаратної бази ESP32-CAM роздивимось його характеристики та характеристики близьких аналогів, що наведено у таблиці 2.1.

Дивлячись на характеристики вже можливо побачити що ESP32 дає більше за меншу ціну, має велику потужність та при цьому невелике споживання енергії, на відміно до близьких аналогів. Крім потужності у ESP32-CAM ще є переваги які можливо побачити у порівняльній характеристиці цих контролерів у таблиці 2.2.

Отже, для реалізації проєкту для система безпеки “розумного” будинку було обрано модуль ESP32-CAM, який задовольняє критеріям проєкту завдяки вбудованим модулям бездротового з'єднання, наявності бібліотек для розпізнавання об'єктів на камері, легкого підключення та наявності самих модулів камери, достатньої потужності, невеликого споживання енергії та низької вартості.

Таблиця 2.1 – Характеристики мікроконтролерів [13]-[15]

Характеристика	Назва модуля		
	ESP32-CAM	Arduino Nano	Raspberry Pi Pico
Тактова частота процесору	240MHz	16MHz	133MHz
Кількість флеш-пам'яті	4MB	32KB	133MHz
Кількість оперативної пам'яті	520KB	2KB	264KB
Комунікаційні інтерфейси	I2C, SPI, UART, Wi-Fi, Bluetooth	I2C, SPI, UART	I2C, SPI, UART, Wi-Fi, Bluetooth
Робоча напруга	2.2 - 3.6 V	5 V	1.8 - 5.5 V
Ціна \$	3-10	5-20	4-10

Таблиця 2.2 – Порівняльна характеристика мікроконтролерів [13]-[15]

Характеристика	Назва модуля		
	ESP32-CAM	Arduino Nano	Raspberry Pi Pico
Можливість бездротового з'єднання	+	-	+
Легке підключення модулів камери	+	-	-
Легка робота з бібліотеками для роботи з розпізнаванням на камері	+	-	-
Достатня потужність для обробки зображень	+	-	+
Невелике споживання енергії	+-	+	+-

2.4 Обґрунтування вибору програмних засобів

2.4.1 Обґрунтування вибору мови програмування

Для розробки програмної частини проєкту було обрано мову програмування Arduino program language або Arduino C.

Мова програмування Arduino була розроблена спеціально як діалект C++, спрощений і адаптований для використання з мікроконтролером Arduino. Вона базується на стандартній мові C++, але має деякі корисні особливості для роботи з мікроконтролерами [16]-[17].

Однією з головних особливостей мови програмування Arduino є простота використання вбудованих бібліотек для зручної взаємодії з різними пристроями та датчиками.

Базова структура програми Arduino складається з двох основних функцій: `setup()` та `loop()`. Функція `setup()` виконується лише один раз на початку програми і використовується для встановлення початкових параметрів, таких як ініціалізація входів/виходів. Функція `loop()` виконується після функції `setup()` постійно [16]-[17].

Базова синтаксична структура мови програмування Arduino подібна до стандартної мови C++, але є деякі спеціальні функції та синтаксис для роботи з мікроконтролерами. Наприклад, функції `digitalRead()` і `digitalWrite()` використовуються для роботи з цифровими входами і виходами, а функції `analogRead()` і `analogWrite()` використовуються для роботи з аналоговими входами і виходами [16]-[17].

Основна перевага мови програмування Arduino полягає в тому, що вона проста у вивченні і пропонує широкий спектр можливостей для розробки проєктів, починаючи від простих світлодіодних вивісок і закінчуючи складними системами автоматизації.

Обґрунтування вибору мови програмування для розробки програмної частини наведено у таблиці у таблиці 2.3.

Таблиця 2.3 – Обґрунтування вибору мови програмування [16]-[17]

Критерій порівняння мов програмування	Назва мови		
	C	Rust	Arduino C
Висока складність при програмуванні контролерів	+/-	+/-	-
Доступність бібліотек для програмування контролерів	+/-	-	+
Швидкодія	+	+	+/-
Вбудовані бібліотеки для роботи з контролерами	-	-	+

Таким чином, мова Arduino C була обрана за свою адаптованість для програмування мікроконтролерів, простоті у використанні, обширному набору бібліотек, при тому що деякі вже вбудовані у мову, прості адоптації до мови завдяки ідентичного синтаксису з мовами C. За все це поступаючись лише невеликою різницею у швидкодії порівняно з іншими мовами, що є не значною ціною, яка не як не вплине на розробку програмно забезпечення для системи безпеки “розумного” будинку за допомогою відеоспостереження з розпізнаванням облич.

2.4.2 Обґрунтування вибору середовища розробки

В якості середовища розробки для реалізації проєкту було обрано Arduino IDE. Arduino IDE - це інтегроване середовище розробки, спеціально розроблене для програмістів, що працюють з платформою Arduino. Створена спільнотою Arduino, ця IDE надає зручний інтерфейс для написання, відладки та завантаження програмного забезпечення на платформу Arduino [18]-[20].

Що стосується функціональності, Arduino IDE містить редактор коду, компілятор та інструменти для завантаження програм на платформу Arduino. Вона підтримує різні мови програмування, зокрема мову Arduino, яка

базується на мові C++, та дозволяє розробникам працювати з різними моделями плат [18]-[20].

Крім того, Arduino IDE має вбудовану бібліотеку функцій та прикладів коду, які допомагають розробникам почати роботу з платформою швидше. Вона також підтримує різні розширення та плагіни, що дозволяють розширити її можливості та пристосувати під конкретні потреби користувача [18]-[20].

Узагальнюючи, Arduino IDE – це потужний інструмент, який дозволяє розробникам легко створювати програмне забезпечення для платформи Arduino, надаючи їм необхідні інструменти та ресурси для цього [18]-[20].

Обґрунтування вибору середовища розробки наведено у таблиці у таблиці 2.4. Таким чином, для реалізації проєкту було обрано середовище розробки Arduino IDE, за його підтримку обраної у попередньому розділі мови програмування, можливість легко завантажити програму на платформу, доступність та простоту у використанні.

Таблиця 2.4 – Обґрунтування вибору середовища розробки [18]-[20]

Критерій порівняння середовищ розробки	Назва середовища розробки		
	Arduino IDE	Visual Studio	Atmel studio
Вбудована підтримка Arduino C	+	-	-
Доступність бібліотек для програмування контролерів	+	+-	+
Відкритий код	+	+	-
Можливість завантажувати програму на платформу	+	+-	+
Вбудовані бібліотеки для роботи з контролерами	+	-	-

2.4.3 Telegram API

Telegram – це потужний месенджер, що пропонує широкі можливості для спілкування та організації робочих процесів. Він поєднує в собі швидкість, безпеку, конфіденційність та багатофункціональність, гарантуючи користувачам високоякісний досвід [21]-[22].

Розглянемо його основні переваги:

- мультиплатформеність;
- швидкість та надійність;
- конфіденційність та безпека;
- боти для автоматизації процесів.

Для проєкту більш всього важливий останній пункт Telegram надає безкоштовний арі з яким легко можна створювати ботів, даний функціонал або відсутній у інших месенджерах або обмежений, що можна побачити у порівняльній характеристиці месенджерів у таблиці 2.5.

Бот в Telegram – це обліковий запис, керований програмою (сервером), призначений для взаємодії з користувачами [21]-[22].

Вони можуть брати участь у приватних розмовах з користувачами, бути учасниками груп і каналів, надсилати повідомлення, відповідати на команди користувачів, надсилати зображення та виконувати інші завдання [21]-[22].

Однак боти мають обмежені можливості порівняно зі звичайними користувачами для забезпечення їх приватності: наприклад, вони не можуть самостійно приєднуватися до груп.

Тобто у проєкті бот надасть можливість повідомляти власника про небезпеку а також дистанційно керувати модулем.

Таблиця 2.5 – Обґрунтування вибору месенджера [21]-[22]

Критерій порівняння	Назва месенджера		
	Telegram	WhatsApp	Viber
Шифрування	End-to-end, MTProto	End-to-end	End-to-end
Хмара зберігання	Необмежено	Немає	Немає
Платформи	iOS, Android, Windows, macOS, Linux, Web	iOS, Android, Windows, macOS, Web	iOS, Android, Windows, macOS
Боти	Є	Немає	Є, але обмежено
Швидкість роботи	Висока	Середня	Середня
Обмеження на розмір файлу	2 ГБ	100 МБ	200 МБ

2.5 Висновки за розділом 2

Для реалізації проекту системи безпеки “розумного” будинку за допомогою розпізнавання облич було обрано мову програмування Arduino C, яка використовується для програмування мікроконтролерів.

В якості апаратної бази було обрано платформу ESP32 а сам модуль ESP32-CAM за його вигідні характеристики які переважають найближчих конкурентів.

Для створення програмного забезпечення та його відладки обрано середовище розробки Arduino IDE.

Також для надсилання повідомлень власнику було обрано TelegramAPI за його легкість використання та безкоштовність.

3 КОНСТРУЮВАННЯ ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

3.1 Схеми алгоритму функціонування системи

На рисунку 3.1 наведено узагальнену схему алгоритму функціонування системи.

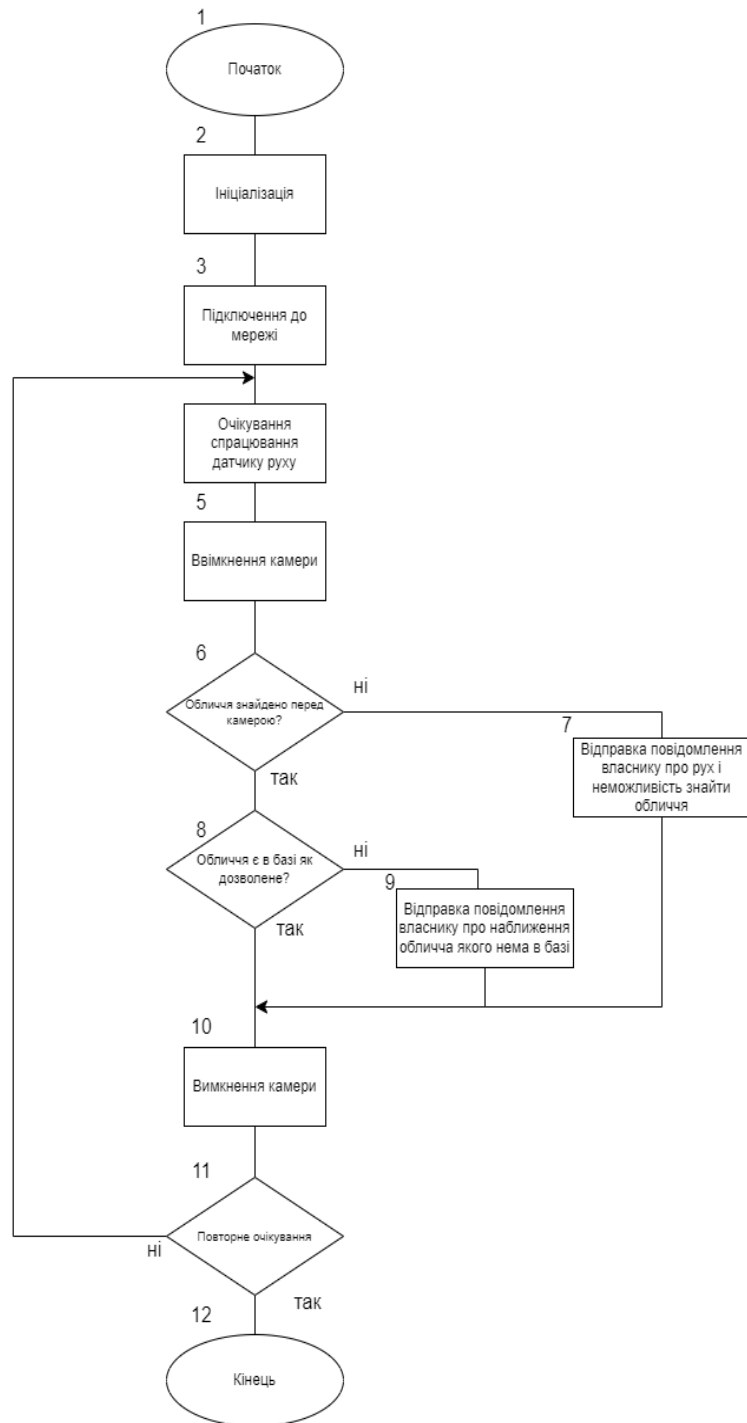


Рисунок 3.1 – Узагальнений алгоритм функціонування системи

Блок 1. Початок.

Блок 2. Встановлення значення змінних підключення бібліотек.

Блок 3. Підключення до бездротової мережі.

Блок 4. Очікування спрацювання датчику руху.

Блок 5. Вихід контролера з сплячого режиму ввімкнення модуля камери.

Блок 6. Пошук обличчя людини за допомогою алгоритму MTMN у випадку коли не буде передано інформацію про знайдені обличчя перехід до 7 блоку інакше до 8.

Блок 7. Створення повідомлення з попередження про наявність руху перед модулем та не можливість знайти обличчя перед камерою , збереження поточного зображення, після чого формується json об'єкт з фото та повідомлення який відправляється POST запитом власнику у телеграм бота.

Блок 8. Після отримання координат обличчя з блоку 6 виконується ідентифікація обличчя за допомогою моделі MobileFace, після результат ідентифікації порівнюється з дозволеними обличчями, у випадку якщо обличчя не буду у списку дозволених відбувається перехід до 9 блоку інакше до 10.

Блок 9. Створення повідомлення з попередження про приближення особи якої нема у списку дозволених, збереження поточного зображення, після чого формується json об'єкт з фото та повідомлення який відправляється POST запитом власнику у телеграм бота.

Блок 10. Повернення контролеру до сплячого режиму вимикання камери.

Блок 11. Якщо контролер продовжує працювати перехід до 3 блоку інакше до 12.

Блок 12. Кінець.

3.2 Особливості програмної реалізації

Текст програми наведено в Додатку А. В програмі оголошено декілька змінних: `ssid`, `pas`, які відповідають за під'єднання до бездротового маршрутизатора, зберігаючи ідентифікатор та пароль для під'єднання, а також `token` і `id` зберігають ідентифікатор бота та чату власника для надсилання повідомлень про небезпеку.

Основною функцією програми є `check()` вона викликається після спрацювання датчику руху, і у ній відбуваються всі перевірки на наявність обличчя та його співпадіння, та надсилання повідомлення у випадку потенційної небезпеки.

Пошук та розпізнавання обличчя на зображенні виконується за допомогою бібліотеки `esp-who` яка додана в бібліотеку `esp_camera`, пошук виконується за допомогою моделі `MTMN` до якої включено алгоритм `MTCNN`, та архітектуру `MobilNets` яка замінює модуль `CNN` щоб зменшити розмір моделі полегшати та прискорити швидкість дії на мобільних пристроях. Пошук обличчя виконується у три етапи спочатку мережа шукає потенційне місце де може знаходитися обличчя поділяючи зображення на сектора та за допомогою класифікатора та регресора визначає чи є потенційно в цих секторах обличчя. Після чого в потенційних точка підтверджується наявність обличчя та задаються рамки обличчя. На останньому етапі на виділених рамках де знаходяться обличчя виробляються ключові точки такі як очі, ніс, рот. Отримані точки передаються у модель `MobileFace`, навчену розпізнавати та порівнювати обличчя, де вони порівнюються зі збереженим списком дозволених облич.

Надсилання повідомлення у телеграм виконується з допомогою бібліотек `UniversalTelegramBot.h` та `ArduinoJson.h`, перша надає функції з якими легко під'єднати бота та надсилати і отримувати повідомлення, друга допомагає формувати JSON об'єкти які потім надсилаються як повідомлення.

Опис використаних бібліотек які використовувалися для проєкту наведено у таблиці 3.1, а опис базових функцій у таблиці 3.2.

Таблиця 3.1 – Опис використаних бібліотек

Бібліотека	Опис бібліотеки
esp_system.h	Надає функції для взаємодії з системою, такі як скидання налаштувань пристрою або отримання ідентифікатора чіпа пристрою
esp_camera.h	Містить функції для взаємодії з модулем камери ESP32-CAM(включаючи функцію розпізнавання облич). Використовується для прямої трансляції через веб-сервер
esp_http_server.h	Надає функції для налаштування веб-сервера HTTP на пристрої
esp_timer.h	Містить функції для взаємодії з апаратним таймером пристрою
camera_pins.h	Визначає константи для призначень контактів для модуля ESP32-CAM.
connect_wifi.h	Містить функції для підключення пристрою до мережі Wi-Fi
UniversalTelegramBot.h	Містить функції для роботи з телеграм ботом
ArduinoJson.h	Містить функції для створення, парсингу та серіалізації JSON структур

Таблиця 3.2 – Опис використаних базових функцій

Функція	Опис функції
1	2
setup()	Виконується лише один раз при запуску програми і дає можливість встановити початкові налаштування.
loop()	Виконується регулярно до вимкнення плати.
pinMode(pin, mode)	Функція pinMode() визначає режим роботи портів: pin - це вивід, з яким ми працюємо, і mode - це спосіб, яким цей вивід має працювати: як вихід (OUTPUT), вхід (INPUT) або вхід з підтяжкою (INPUT_PULLUP).
digitalWrite()	Функція digitalWrite() використовується для управління цифровими портами. Вона приймає два параметри: pin - це номер цифрового порту, на який ми надсилаємо сигнал, і value - це значення, яке ми відправляємо на цей порт. Для цифрових пінів можливі два значення: HIGH (високий рівень, 1) або LOW (низький рівень, 0). Ця функція не повертає значення.
delay(ms)	Ця функція призупиняє виконання програми на певний проміжок часу, вказаний у мілісекундах.
Serial.begin(speed)	Ця функція встановлює послідовне з'єднання і швидкість передачі даних, яка вимірюється в бітах на секунду.
Serial.println(val, format)	Ця функція обробляє дані, які надходять з послідовного порту у форматі ASCII-тексту. Вона приймає значення val для друку та параметр format для визначення формату виводу даних.

3.3 Схема апаратної частини

На рисунку 3.2 наведено схему підключення детектору руху та опціонального програматора у якості елемента живлення.

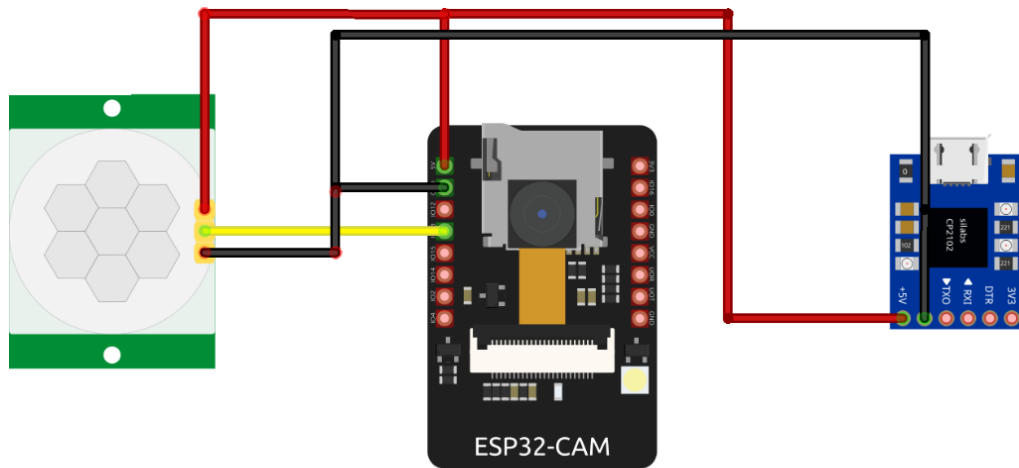


Рисунок 3.2 – Принципова схема пристрою з живленням по USB

Як можна побачити з схеми для роботи контролера та модуля датчику руху потрібно живлення в 5В, контакт даних датчику руху підключається до контакту GPIO 15 на контролері ESP32.

На рисунку 3.3 наведено схему підключення програматора до контролера для завантаження програми.

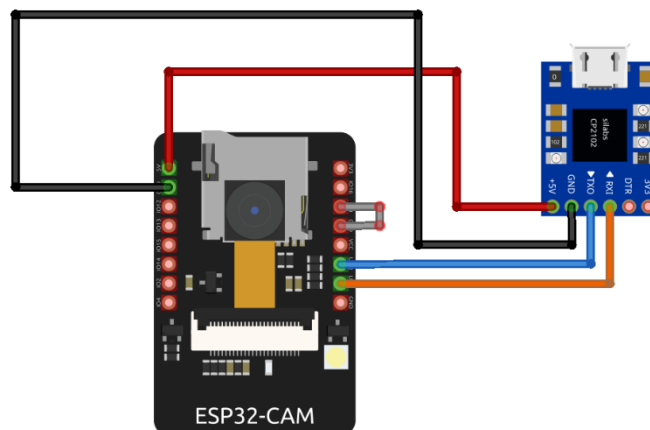


Рисунок 3.3 – Схема підключення програматора для запису програми

Щоб програмувати плату ESP32-CAM через Arduino IDE, спочатку треба встановити саме Arduino IDE та розширення для ESP32.

При підключенні до комп'ютера, використовують модуль послідовного перетворювача, наприклад, FTDI або USB TO TTL. На якому є роз'єм USB та 6 виводів на одному боці: +3.3V, GND, +5V, TXD (TX), RXD (RX), DTR. З іншого боку плати є монтажні отвори з функціями DCD, D3R, RTS, CTS, SUS, SUS, R1, RST, а також три світлодіоди: червоний (POWER) та два (RX і TX), які миготять під час передачі даних.

Для підключення плати до програматора або комп'ютера використовують GPIO 1 (TX) і GPIO 3 (RX), які є послідовними контактами. GPIO 0 повинен бути підключений до GND для режиму завантаження коду. Також варто зауважити, що GPIO 1 і GPIO 3 можуть використовуватися для інших пристроїв, тому важливо враховувати це при плануванні підключень.

3.4 Висновки за розділом 3

Було розглянуто та описано алгоритм функціонування програми, зазначені та описанні бібліотеки які використовувались, а також основні функції та зміни які є важливими для проєкту. Виконано побудову та описано схему підключення апаратної частини.

4 КЕРІВНИЦТВО КОРИСТУВАЧА ТА ПРИКЛАД РОБОТИ

4.1 Створення Telegram бота

Оскільки нотифікація власника буде відбуватись у телеграмі, для отримання повідомлень та відправки команд на контролер треба створити бота, яким контролер буде керувати.

Крок 1. Відкрити застосунок телеграм.

Крок 2. У рядку пошуку написати BotFather.

Крок 3. Відкрити сторінку повідомлень з BotFather клацнувши на нього у результаті пошуку (рис. 4.1.).



Рисунок 4.1 – BotFather у результаті пошуку

Крок 4. Натиснути кнопку START.

Крок 5. Написати у повідомлення /new bot, або натиснути на цей варіант у повідомленні від бота (рис. 4.2).

Крок 6. Написати будь-яке ім'я яким буде називатися бот у системі.

Крок 7. Надати ідентифікатор боту, який обов'язково повинен закінчуватись на bot.

Крок 8. Після цього буде повідомлення, у якому буде токен бота, який потрібно зберегти (рис. 4.3).

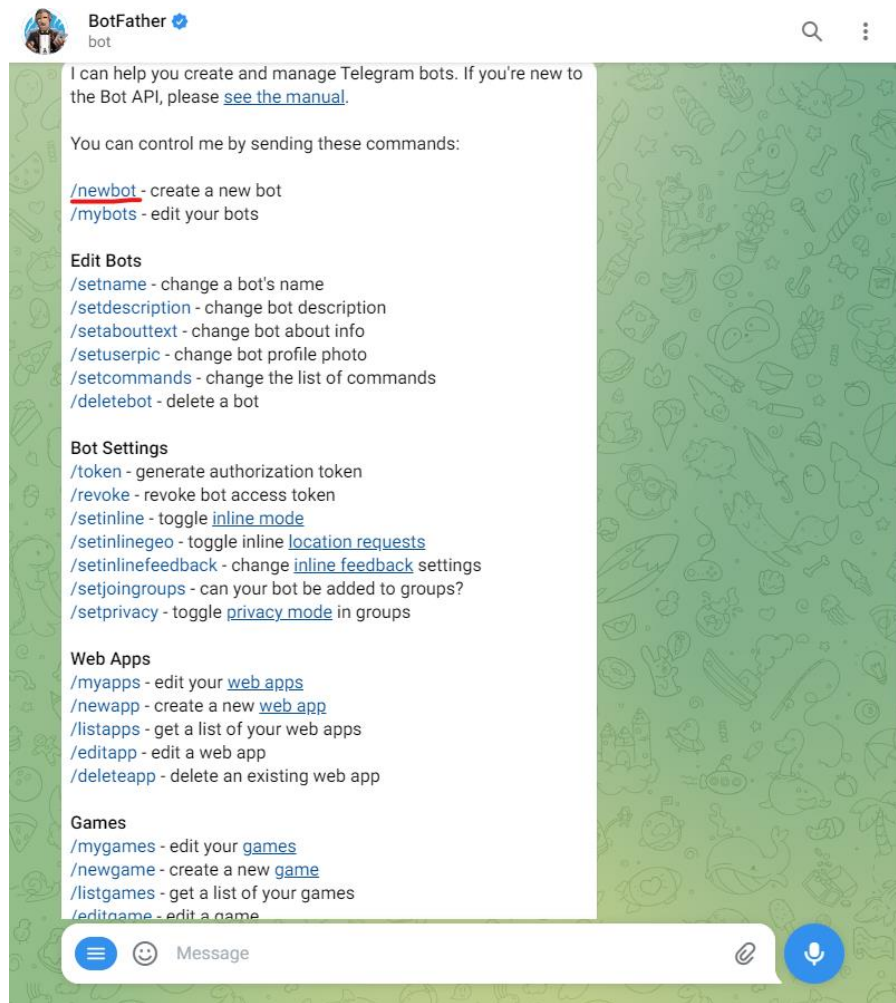


Рисунок 4.2 – Повідомлення з варіантами дії

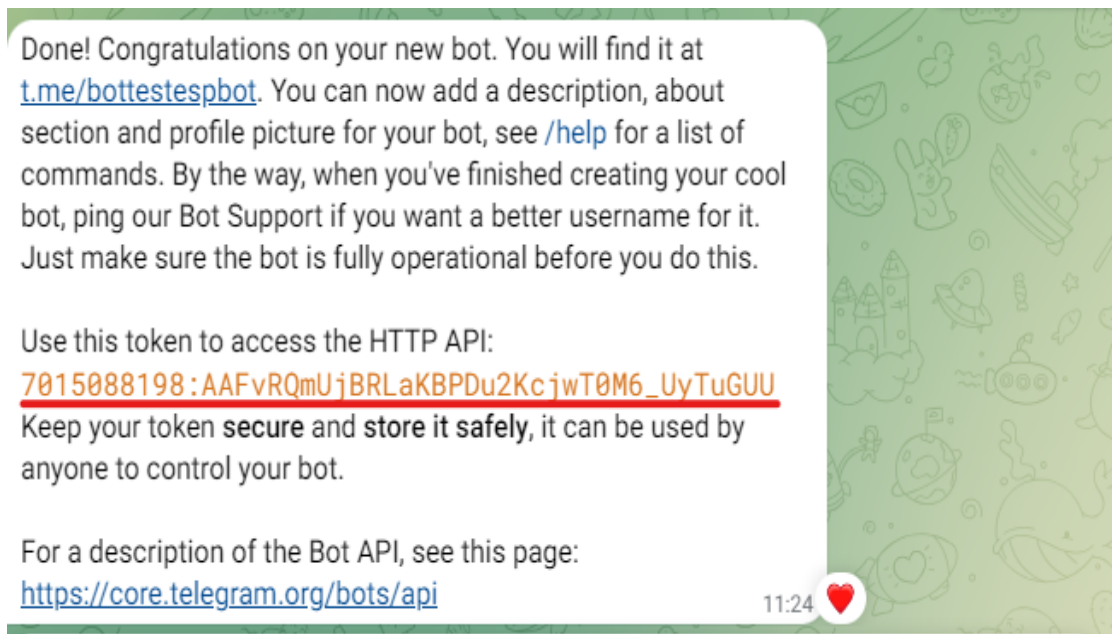


Рисунок 4.3 – Токен боту

Також для безпеки щоб не хто сторонній не міг користуватися ботом та також отримувати повідомлення та відсилати команди контролеру треба отримати ідентифікатор користувача, за допомогою якого бот буде перевіряти чи обробляти команди від даного користувача.

Крок 1. Відкрити застосунок телеграм.

Крок 2. У рядку пошуку написати idbot.

Крок 3. Відкрити сторінку повідомлень з idbot, клацнувши на нього у результаті пошуку(рис. 4.4.).

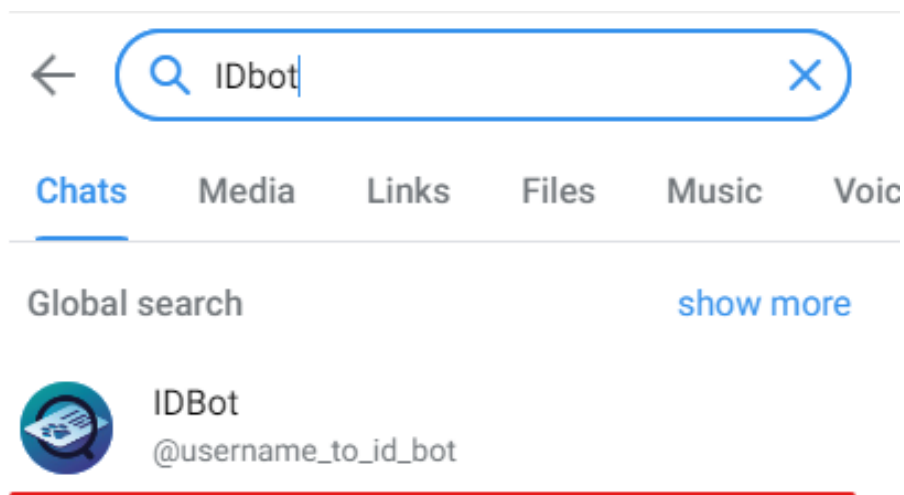


Рисунок 4.4 – IDbot у результаті пошуку

Крок 4. Натиснути кнопку “START”.

Вам відправиться повідомлення у я кому буде ваш id який також потрібно зберегти (рис. 4.5).

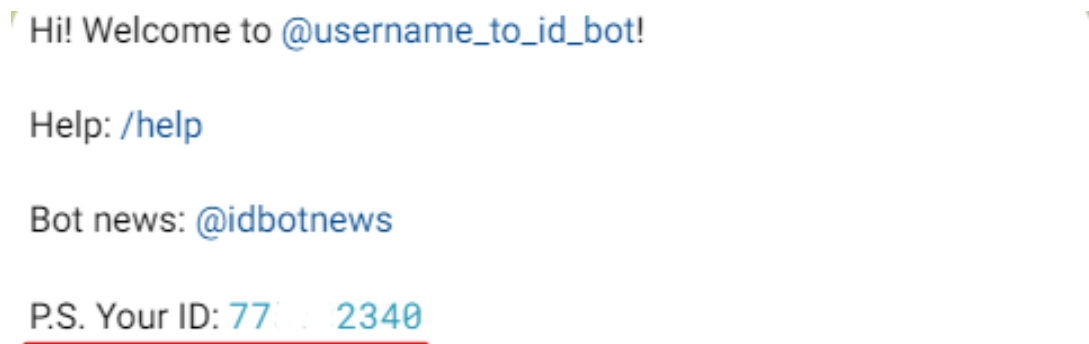


Рисунок 4.4 – Id користувача

4.2 Налаштування змінних

Перед тим як завантажувати програму до контролера треба встановити змінні для під'єднання до бездротового маршрутизатора (роутера), це змінні Ssid у якій визначається назва бездротової мережі до якої буде підключатись контролер, а також passwd який визначає пароль від цієї мережі. Інші дві змінні відповідають за бота, bottoken у якій вставляється токен бота, який був здобутий у минулому підпункті, щоб визначити яким ботом керуватиме контролер, та userid який також був отриманий у минулому підпункті і використовуються для перевірки чи обробляти запити від даного користувача.

Приклад, як це повинно виглядати наведено нижче.

```
const char* ssid = "name";  
const char* passwd = "123456";  
String userid = "468157343";  
String bottoken="7015088198:AAFvRQmUjBRLaKBPDu2KcjwUyTuU";
```

4.3 Завантаження програми до контролера

Завантаження програми до контролера буди виконуватись у Arduino IDE, тож воно повинно бути встановлено, а також деякі бібліотеки з якими працює програма, ось кроки підготовки для завантаження програми до контролера.

Крок 1. Відкрити Arduino IDE.

Крок 2. Натиснути на випадаючий список Tools , обрати пункт Board, Board Manager (рис. 4.5).

Крок 3. У відкритому боковому вікні у пошуки задати esp32.

Крок 4. У результаті пошуку обрати esp32 by Espressif натиснути install (рис. 4.6).

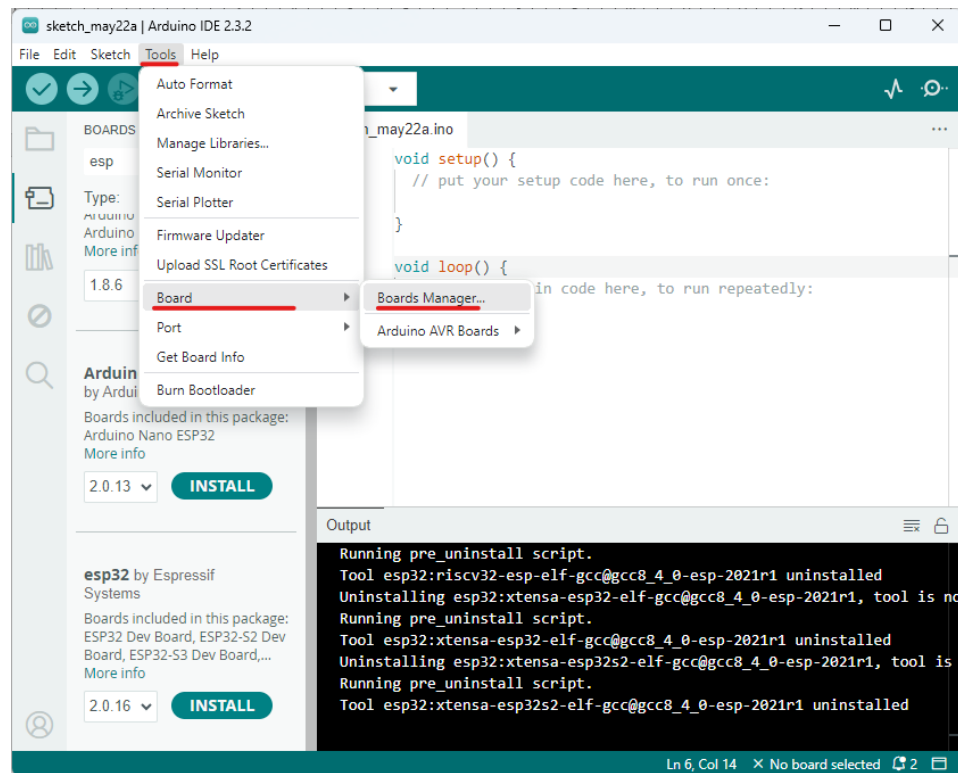


Рисунок 4.5 – Board Manager

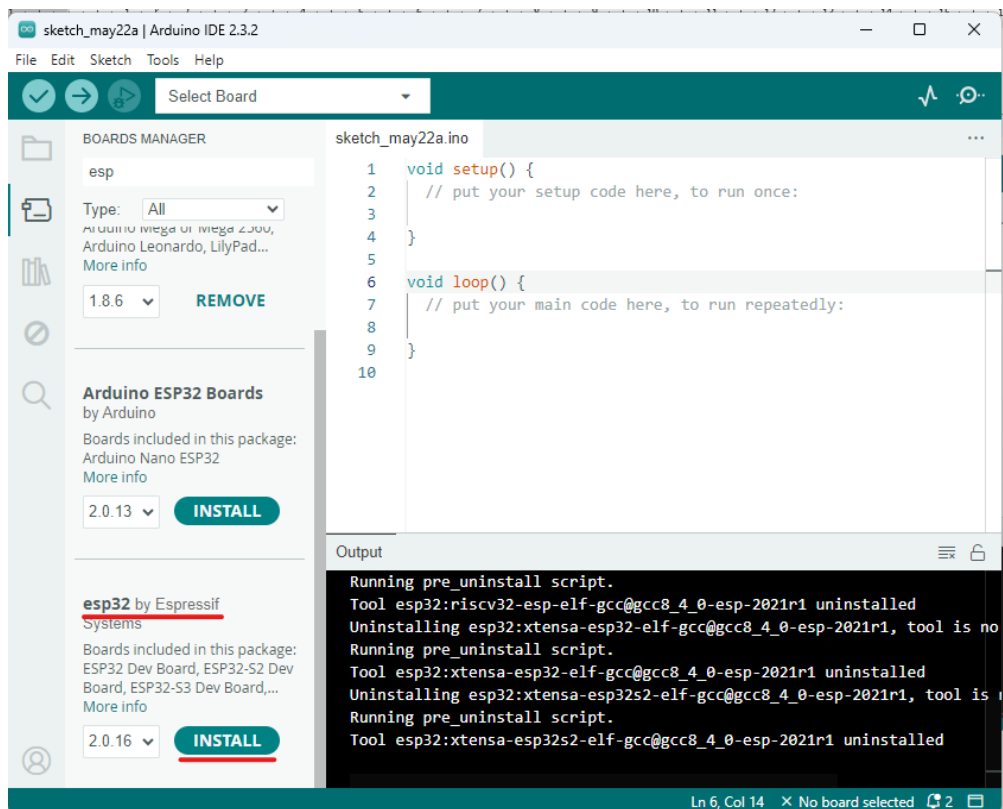


Рисунок 4.6 – Board Manager esp32

Крок 5. Далі відкрити управління бібліотеками, натиснувши на випадаючий список Sketch, далі Include Library, Manage libraries (рис 4.7).

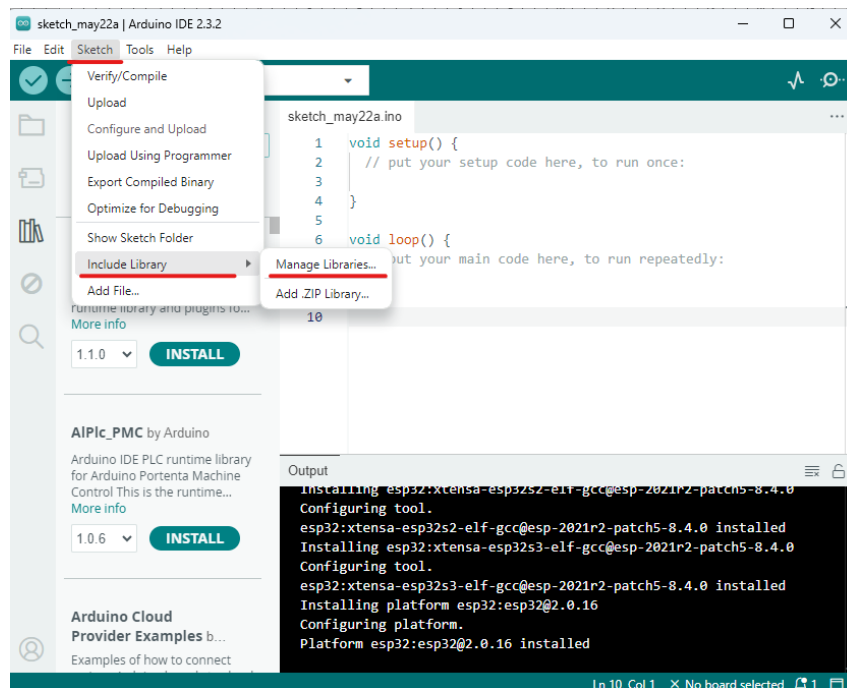


Рисунок 4.7 – Manage Libraries

Крок 6. У відкритому боковому вікні у пошуки задати json.

Крок 7. У результаті пошуку обрати Arduinoj_Json натиснути install.

Крок 8. У відкритому боковому вікні у пошуки задати telegram .

Крок 9. У результаті пошуку обрати AsyncTelegram натиснути install.

Після цих кроків можна переходити до завантаження програми до контролеру за допомогою програматора, схема підключення програматора до контролера наведено у минулому розділі на рисунку 3.3, а також описані особливості на які треба звернути увагу.

Крок 1. Підключити програматор.

Крок 2. Обрати випадаючий список з пристроями та обрати Select board and port (рис. 4.8).

Крок 3. У вікні, що відкрилося, у Boards обрати Ai Thinker ESP32-CAM, а у Ports обрати порт до якого підключено програматор, підтвердити натиснувши ok (рис. 4.9).

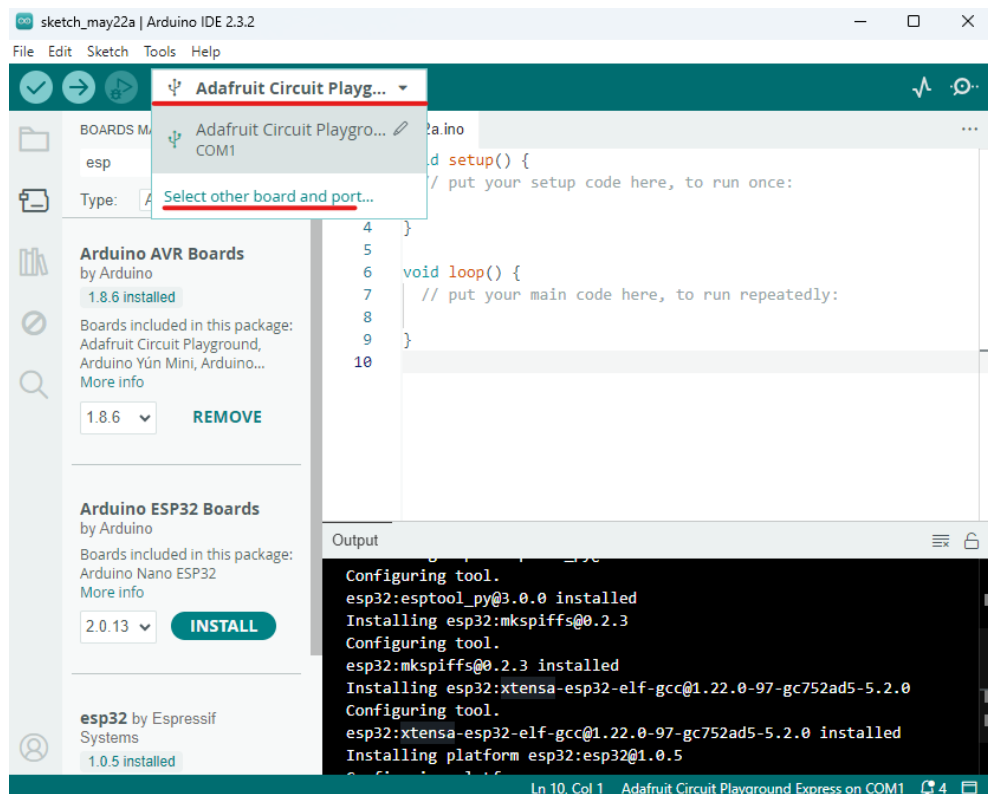


Рисунок 4.8 – Обрання пристрою

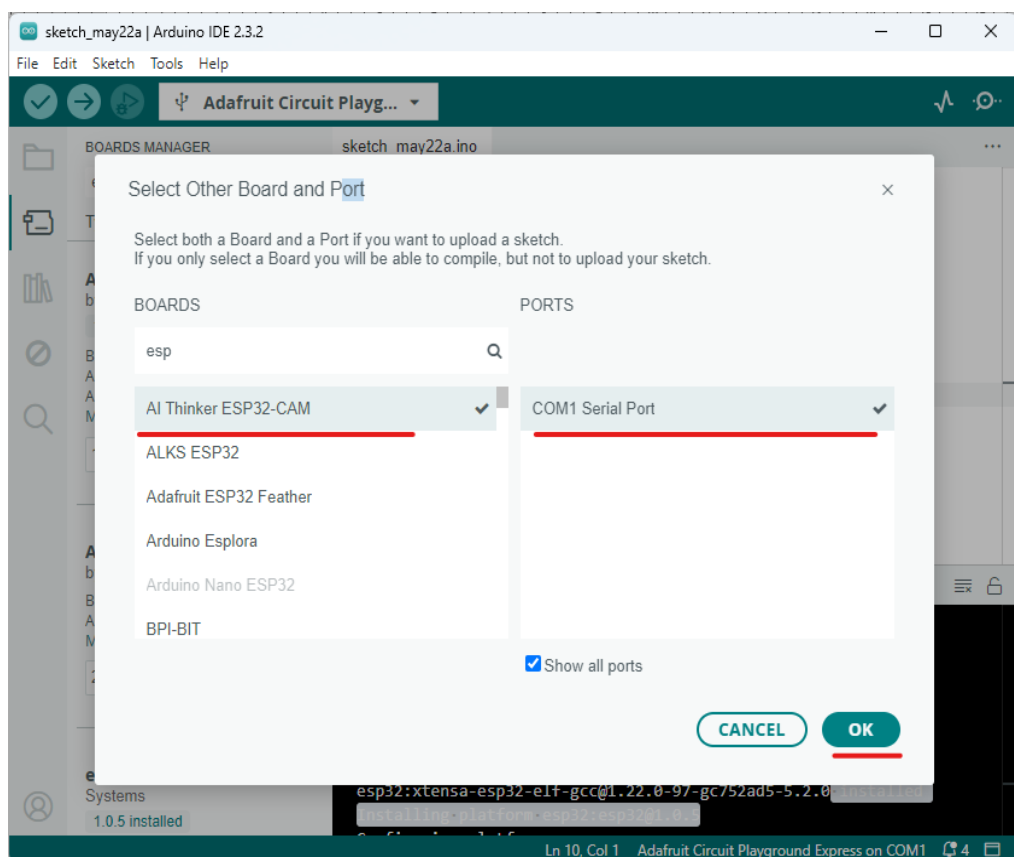


Рисунок 4.9 – Обрання дошки та порту

Крок 4. Натиснути кнопку upload (стрілка праворуч).

Крок 5. Коли з'явиться напис Connecting натиснути кнопку RST на контролері.

Крок 6. Після завершення завантаження від'єднати з'єднувач GPIO і GND.

Крок 7. Відкрити serial monitor, натиснути кнопку RST на контролері та побачити напис про успішний запуск.

4.4 Початок роботи

Після того як було завершено створення бота та завантаження програми до контролера, залишається лише додати обличчя до списку дозволених, для цього після завантаження контролер не знаходиться в режимі очікування, для відкриття інтерфейсу роботи з обличчями треба будь-який пристрій з можливістю перегляду веб сторінок, який буде знаходитись в одній локальній мережі з контролером.

Крок 1. Відкрити чат з ботом написати /settings.

Крок 2. Бот дасть ір-посилання за яким треба перейти (рис. 4.10).

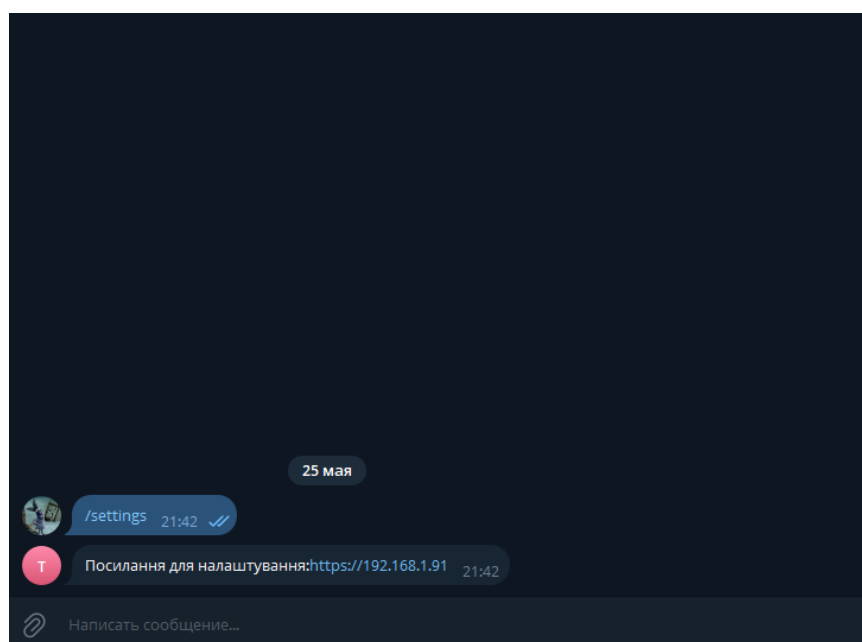


Рисунок 4.10 – Отримання посилання для налаштування.

Крок 3. Після відкриття інтерфейсу можна включити перегляд камери натиснувши “Перегляд камери”(рис. 4.11).

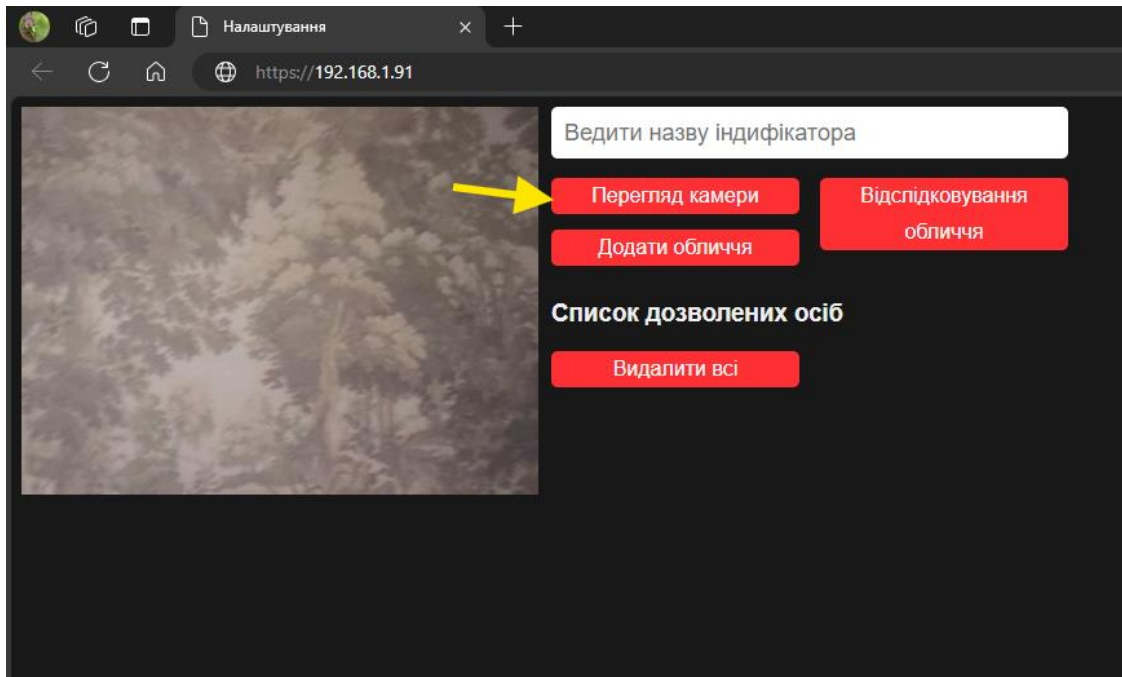


Рисунок 4.11 – Перегляд камери

Крок 4. Придумати та ввести ідентифікатор для власника обличчя у рядок для цього (рис. 4.12).

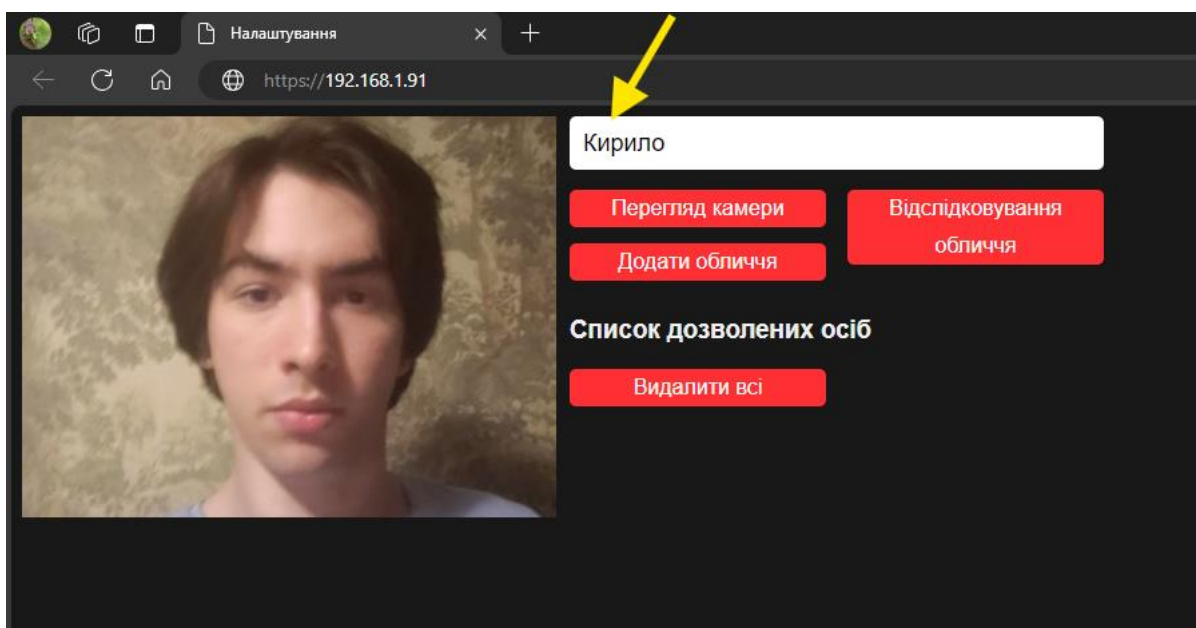


Рисунок 4.12 – Введення ідентифікатору

Крок 5. Натиснути додати обличчя після чого воно відобразиться у списку дозволених, повторювати крок 4-5 потрібну кількість разів в залежності від кількості людей яких ви хочете додати (рис. 4.13).

Крок 6. Закрити сторінку повернутись до бота.

Крок 7. Написати боту /secure.

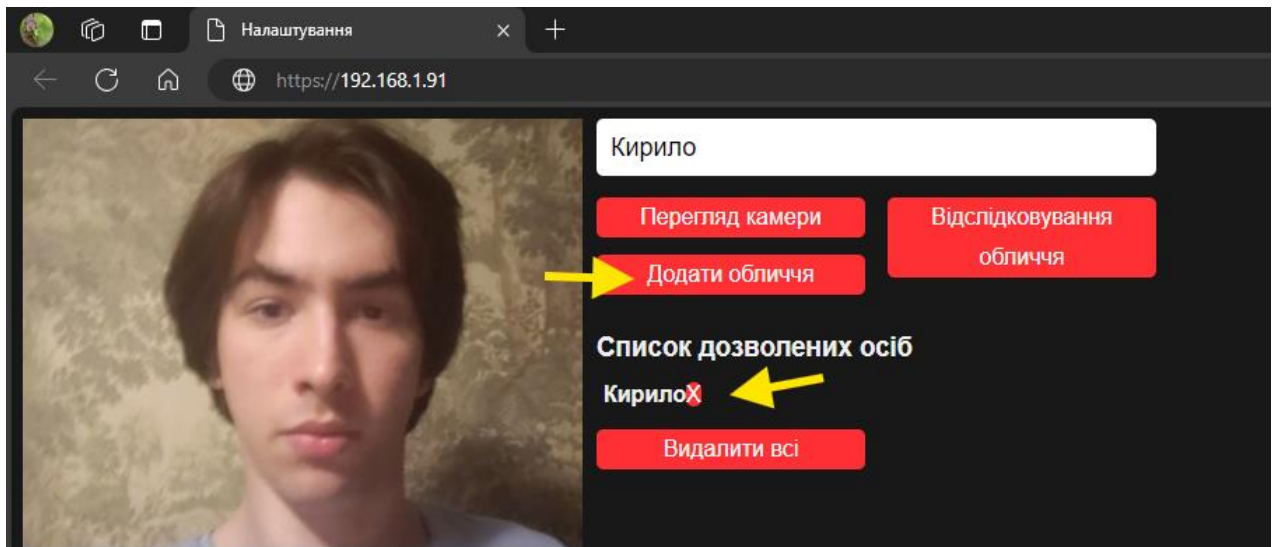


Рисунок 4.13 – Додавання обличчя та список дозволених осіб

Після цих кроків пристрій буде виконувати свої функції, за потреби можливо його перезавантажити натиснувши RST, наприклад для налаштування знову. Також в інтерфейсі були ще декілька кнопок перша з них це видалення всіх облич, вона видаляє всі дозваленні обличчя , також це можливо зробити натиснувши хрестик біля обличчя яке ви хочете видалити (рис. 4.14). Також є кнопка відслідковування обличчя. Вона більше потрібна для тестування та впевненості в роботі, при натисканні на неї буде відображено контур облич, які контролер знаходить перед собою. Приклад зображено на рисунку 4.15.

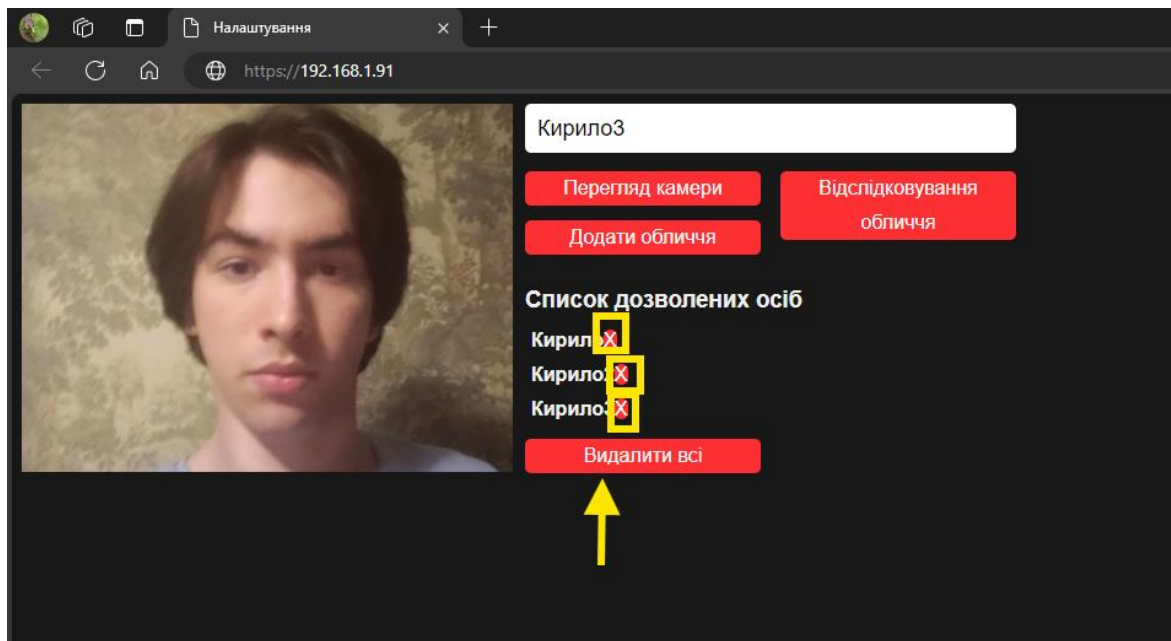


Рисунок 4.14 – Елементи для редагування списку облич

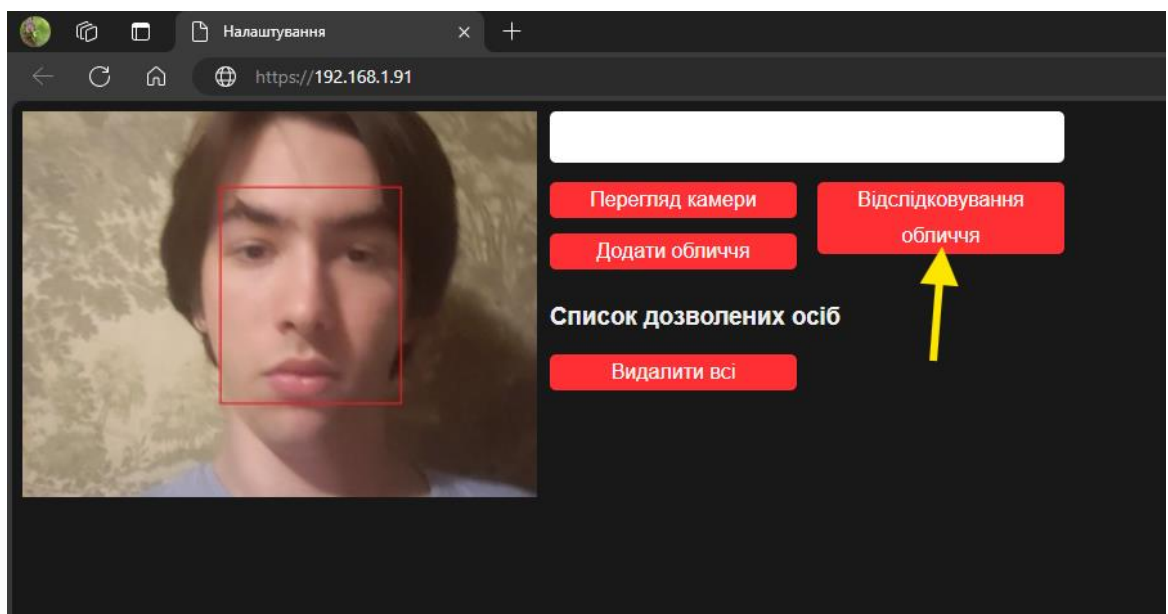


Рисунок 4.15 – Приклад знаходження обличчя

Далі буде наведено приклад спрацювання на знаходження руху та не можливості знайти обличчя (рис. 4.16), а також випадок коли приблизиться не зареєстрована особа (рис. 4.17). У випадку коли приближається зареєстрована особа нічого не відбувається.

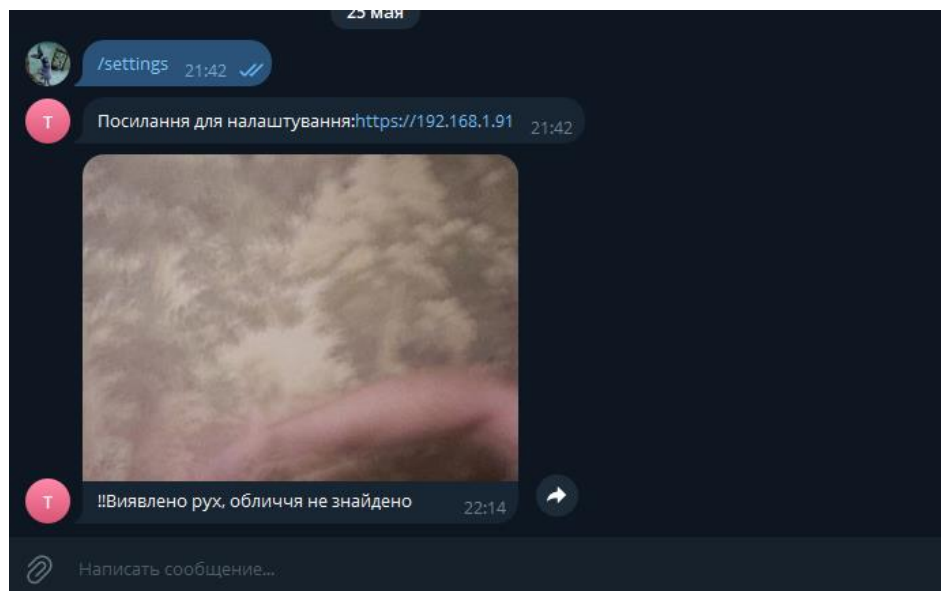


Рисунок 4.16– Сповіщення про наявність руху

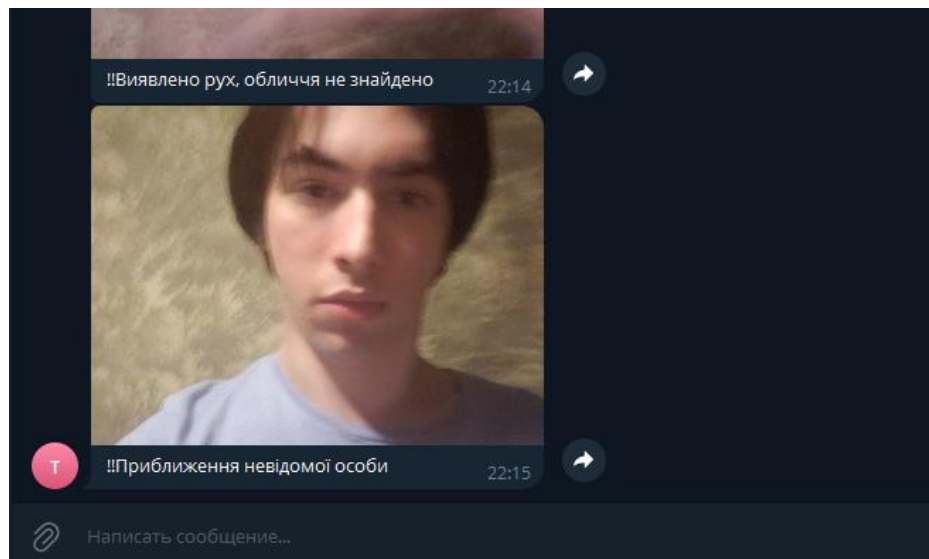


Рисунок 4.17 – Сповіщення про приближення невідомої особи

4.5 Висновки за розділом 4

У даному розділі було у подробицях описано як виконати усі кроки для початку роботи з апаратно-програмним рішенням, від створення бота у Telegram до завантаження програми до контролера та її налаштування для початку роботи. Також було протестовано та наведено приклад спрацювання програми при неможливості знайти обличчя, чи приближення невідомої особи.

ВИСНОВКИ

Результатом виконання дипломної роботи бакалавра стало створення апаратно-програмного рішення для системи безпеки “розумного” будинку, за допомогою системи розпізнавання облич та використання бездротових технологій.

Було проведено аналіз предметної області, з визначенням підходу до створення програмного забезпечення безпеки, розглянуті представлені на даний момент рішення для забезпечення відеоспостереження, на основі аналізу яких було визначено вимоги до розроблюваної системи безпеки, а також розглянуто най розповсюдженні бездротові технології та методи розпізнавання облич.

Розроблено архітектуру системи яка задовольняє вимогам які були виведенні під час аналізу. Обґрунтовано вибір апаратної частини якою став контролер ESP32, а також мови програмування та середовища розробки яким став Arduino C та Arduino відповідно.

Створено програмне забезпечення яка виконує основні дві функції перша з яких пошук та порівняння облич перед модулем камери у разі наявності перед ним руху, а друга нотифікація власника у Telegram в разі потенційної небезпеки такої як приближення не авторизованої особи, чи не можливості знайти або опізнати обличчя.

Цінність розробки полягає в реалізації незатратного та ефективного способу забезпечення безпеки у “розумному” будинку, що підвищує рівень комфорту та безпеки мешканців будинку.

У створеного рішення є багато можливостей розширення функціоналу: додавання більшої кількості датчиків та пристроїв для ще більш ефективного забезпечення безпеки; додавання не пов’язаних з безпекою функцію, наприклад автоматизації певних дій за допомогою зв’язування з іншими пристроями у тій самій локальній мережі.

Результати роботи були представлені на НПК «Тиждень науки-2024».

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Бєляєва, М. Системи безпеки [Електрон. ресурс] / М. Бєляєва. – Режим доступу: <https://vikna.tv/dlia-tebe/bezpeka/kontrol-yua-systemy-bezpeky-dost-upni-kozhnomu/>.
2. Системи безпеки, сучасні технології для максимального захисту [Електрон. ресурс]. – Режим доступу: https://gazeta.ua/articles/promotion/_systemi-bezpeki-suchasni-tehnologiyi-dlya-maksimalnogo-zahistu/1139945.
3. Berdyshev, A. Software Development Security Standards [Electronic resource] / A. Berdyshev. – Access mode: <http://surl.li/tgzfa>.
4. Fundamental Practices for Secure Software Development [Electronic resource]. – Access mode: https://safecode.org/wp-content/uploads/2018/03/SAFECode_Fundamental_Practices_for_Secure_Software_Development_March_2018.pdf.
5. Blue iris Software [Electronic resource]. – Access mode: <https://blueirissoftware.com/>.
6. Xeoma [Electronic resource]. – Access mode: <https://felenasoft.com/xeoma/en/>.
7. Ispy [Electronic resource]. – Access mode: <https://www.ispyconnect.com/>.
8. Belhumeur, P. Eigenfaces versus Fisherfaces / P. Belhumeur // Recognition Using Class Specific Linear Projection. – 1997. – Vol. 19, Issue 7. – P. 711–720.
9. Ahonen, T. Face Description with Local Binary Patterns / T. Ahonen, A. Hadid, M. Pietikäinen. // Application to Face Recognition, IEEE, 2006. – P. 2037 - 2041.
10. Huang, D. Local Binary Patterns and its Application to Facial Image Analysis / D. Huang // Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE, 2011. – P. 1–17.
11. Kurniawan, A. Internet of Things Projects with ESP32 / A. Kurniawan

// Build exciting and powerful IoT projects using the all-new Espressif ESP32. – 2019. – Vol. 3. – P. 54–61.

12. Cameron, N. Electronics Projects with the ESP8266 and ESP32 / N. Cameron // Building Web Pages, Applications, and WiFi Enabled Devices. – 2020. – P. 145–190.

13. ESP32 Development Platforms [Electronic resource]. – Access mode: <https://webbylab.com/blog/esp32-development-platfos-acomprehensive-analysis-and-comparison/>.

14. ESP32 platform [Electronic resource]. – Access mode: <https://www.espressif.com/en/products/socs/esp32>.

15. Isaac. ESP32-CAM: що ви повинні знати про цей модуль [Електрон. ресурс] / Isaac. – Режим доступу: <https://www.hwlibre.com/uk/%D0%BA%D0%B0%D0%BC%D0%B5%D1%80%D0%B0-esp32/>.

16. What is Arduino? [Electronic resource]. – Access mode: <https://docs.arduino.cc/learn/starting-guide/whats-arduino/>.

17. Програмування Ардуіно [Електрон. ресурс]. – Режим доступу: <https://doc.arduino.ua/ru/prog/>.

18. Arduino IDE [Electronic resource]. – Access mode: <https://docs.arduino.cc/software/ide/#ide-v2>.

19. Using the Arduino Software (IDE) [Electronic resource]. – Access mode: <https://docs.arduino.cc/learn/starting-guide/the-arduino-software-ide/>.

20. Introduction to Arduino IDE [Electronic resource]. – Access mode: <https://www.theengineeringprojects.com/2018/10/introduction-to-arduino-ide.html>

21. Месенджер Telegram [Електрон. ресурс]. – Режим доступу: <https://bizmag.com.ua/telegram/>.

22. Introduction to Developing Telegram Bots [Electronic resource]. – Access mode: <https://handbook.tmat.me/en/dev/basics>.

ДОДАТОК А
Текст програми

```

#include <ArduinoWebsockets.h>
#include "esp_http_server.h"
#include "esp_timer.h"
#include "esp_camera.h"
#include "camera_index.h"
#include "Arduino.h"
#include "fd_forward.h"
#include "fr_forward.h"
#include "fr_flash.h"
#include <UniversalTelegramBot.h>
#include <ArduinoJson.h>

const char* ssid = "XXXXXXXXXXXX";
const char* password = "XXXXXXXXXXXX";
const char* bottoken = "XXXXXXXXXXXX";
const char* userid = "XXXXXXXXXXXX";
bool checking = false;

int botRequestDelay = 1000;
unsigned long lastTimeBotRan;

#define CAMERA_MODEL_AI_THINKER
#include "camera_pins.h"

using namespace websockets;
WebsocketsServer socket_server;

camera_fb_t * fb = NULL;

long current_millis;
long last_detected_millis = 0;

#define pir_pin 2
unsigned long delay_opened_millis = 0;
long interval = 30000;
bool face_recognised = false;

void app_facenet_main();
void app_httpserver_init();

typedef struct
{
    uint8_t *image;

```

```

    box_array_t *net_boxes;
    dl_matrix3d_t *face_id;
} http_img_process_result;

static inline mtmn_config_t app_mtmn_config()
{
    mtmn_config_t mtmn_config = {0};
    mtmn_config.type = FAST;
    mtmn_config.min_face = 80;
    mtmn_config.pyramid = 0.707;
    mtmn_config.pyramid_times = 4;
    mtmn_config.p_threshold.score = 0.6;
    mtmn_config.p_threshold.nms = 0.7;
    mtmn_config.p_threshold.candidate_number = 20;
    mtmn_config.r_threshold.score = 0.7;
    mtmn_config.r_threshold.nms = 0.7;
    mtmn_config.r_threshold.candidate_number = 10;
    mtmn_config.o_threshold.score = 0.7;
    mtmn_config.o_threshold.nms = 0.7;
    mtmn_config.o_threshold.candidate_number = 1;
    return mtmn_config;
}
mtmn_config_t mtmn_config = app_mtmn_config();

face_id_name_list st_face_list;
static dl_matrix3du_t *aligned_face = NULL;

httpd_handle_t camera_httpd = NULL;

typedef enum
{
    START_STREAM,
    START_DETECT,
    SHOW_FACES,
    START_RECOGNITION,
    START_ENROLL,
    ENROLL_COMPLETE,
    DELETE_ALL,
} en_fsm_state;
en_fsm_state g_state;

typedef struct
{
    char enroll_name[ENROLL_NAME_LEN];
} httpd_resp_value;

```

```

httpd_resp_value st_name;

void setup() {
  Serial.begin(115200);
  Serial.setDebugOutput(true);
  Serial.println();

  UniversalTelegramBot bot(bottoken, userid);

  camera_config_t config;
  config.ledc_channel = LEDC_CHANNEL_0;
  config.ledc_timer = LEDC_TIMER_0;
  config.pin_d0 = Y2_GPIO_NUM;
  config.pin_d1 = Y3_GPIO_NUM;
  config.pin_d2 = Y4_GPIO_NUM;
  config.pin_d3 = Y5_GPIO_NUM;
  config.pin_d4 = Y6_GPIO_NUM;
  config.pin_d5 = Y7_GPIO_NUM;
  config.pin_d6 = Y8_GPIO_NUM;
  config.pin_d7 = Y9_GPIO_NUM;
  config.pin_xclk = XCLK_GPIO_NUM;
  config.pin_pclk = PCLK_GPIO_NUM;
  config.pin_vsync = VSYNC_GPIO_NUM;
  config.pin_href = HREF_GPIO_NUM;
  config.pin_sscb_sda = SIOD_GPIO_NUM;
  config.pin_sscb_scl = SIOC_GPIO_NUM;
  config.pin_pwdn = PWDN_GPIO_NUM;
  config.pin_reset = RESET_GPIO_NUM;
  config.xclk_freq_hz = 20000000;
  config.pixel_format = PIXFORMAT_JPEG;

  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("WiFi connected");

  app_httpserver_init();

```

```

app_facenet_main();
socket_server.listen(82);

Serial.print("Camera Ready! Use 'http://'");
Serial.print(WiFi.localIP());
Serial.println(" to connect");
}

static esp_err_t index_handler(httpd_req_t *req) {
    httpd_resp_set_type(req, "text/html");
    httpd_resp_set_hdr(req, "Content-Encoding", "gzip");
    return httpd_resp_send(req, (const char *)index_ov2640_html_gz,
index_ov2640_html_gz_len);
}

httpd_uri_t index_uri = {
    .uri      = "/",
    .method   = HTTP_GET,
    .handler  = index_handler,
    .user_ctx = NULL
};

void app_httpserver_init ()
{
    httpd_config_t config = HTTPD_DEFAULT_CONFIG();
    if (httpd_start(&camera_httpd, &config) == ESP_OK)
        Serial.println("httpd_start");
    {
        httpd_register_uri_handler(camera_httpd, &index_uri);
    }
}

void handleNewMessages(int numNewMessages) {
    Serial.print("Handle New Messages: ");
    Serial.println(numNewMessages);

    for (int i = 0; i < numNewMessages; i++) {
        String chat_id = String(bot.messages[i].chat_id);
        if (chat_id != userid ){
            bot.sendMessage(chat_id, "Unauthorized user", "");
            continue;
        }

        String text = bot.messages[i].text;

```

```

Serial.println(text);

String from_name = bot.messages[i].from_name;
if (text == "/settings") {
    String msg = "Посилання для налаштування:" + WiFi.localIP();
    bot.sendMessage(userid, msg, "");
}

}
}

String sendPhotoTelegram() {
    const char* myDomain = "api.telegram.org";
    String getAll = "";
    String getBody = "";

    camera_fb_t * fb = NULL;
    fb = esp_camera_fb_get();
    esp_camera_fb_return(fb);

    fb = NULL;
    fb = esp_camera_fb_get();
    if(!fb) {
        Serial.println("Camera capture failed");
        delay(1000);
        ESP.restart();
        return "Camera capture failed";
    }

    Serial.println("Connect to " + String(myDomain));

    if (clientTCP.connect(myDomain, 443)) {
        Serial.println("Connection successful");

        String head = "--RandomNerdTutorials\r\nContent-Disposition: form-
data; name=\"chat_id\"; \r\n\r\n" + userid + "\r\n--
RandomNerdTutorials\r\nContent-Disposition: form-data; name=\"photo\";
filename=\"esp32-cam.jpg\"\r\nContent-Type: image/jpeg\r\n\r\n";
        String tail = "\r\n--RandomNerdTutorials--\r\n";

        size_t imageLen = fb->len;
        size_t extraLen = head.length() + tail.length();

```

```

size_t totalLen = imageLen + extraLen;

clientTCP.println("POST /bot"+bottoken+"/sendPhoto HTTP/1.1");
clientTCP.println("Host: " + String(myDomain));
clientTCP.println("Content-Length: " + String(totalLen));
                clientTCP.println("Content-Type:      multipart/form-data;
boundary=RandomNerdTutorials");
clientTCP.println();
clientTCP.print(head);

uint8_t *fbBuf = fb->buf;
size_t fbLen = fb->len;
for (size_t n=0;n<fbLen;n=n+1024) {
    if (n+1024<fbLen) {
        clientTCP.write(fbBuf, 1024);
        fbBuf += 1024;
    }
    else if (fbLen%1024>0) {
        size_t remainder = fbLen%1024;
        clientTCP.write(fbBuf, remainder);
    }
}

clientTCP.print(tail);

esp_camera_fb_return(fb);

int waitTime = 10000;
long startTimer = millis();
boolean state = false;

while ((startTimer + waitTime) > millis()){
    Serial.print(".");
    delay(100);
    while (clientTCP.available()) {
        char c = clientTCP.read();
        if (state==true) getBody += String(c);
        if (c == '\n') {
            if (getAll.length()==0) state=true;
            getAll = "";
        }
        else if (c != '\r')
            getAll += String(c);
        startTimer = millis();
    }
}

```

```

        if (getBody.length() > 0) break;
    }
    clientTCP.stop();
    Serial.println(getBody);
}
else {
    getBody = "Connected to api.telegram.org failed.";
    Serial.println("Connected to api.telegram.org failed.");
}
return getBody;
}

void app_facenet_main()
{
    face_id_name_init(&st_face_list,    FACE_ID_SAVE_NUMBER,
ENROLL_CONFIRM_TIMES);
    aligned_face = dl_matrix3du_alloc(1, FACE_WIDTH, FACE_HEIGHT,
3);
    read_face_id_from_flash_with_name(&st_face_list);
}

static inline int do_enrollment(face_id_name_list *face_list, dl_matrix3d_t
*new_id)
{
    ESP_LOGD(TAG, "START ENROLLING");
    int left_sample_face = enroll_face_id_to_flash_with_name(face_list,
new_id, st_name.enroll_name);
    ESP_LOGD(TAG, "Face ID %s Enrollment: Sample %d",
        st_name.enroll_name,
        ENROLL_CONFIRM_TIMES - left_sample_face);
    return left_sample_face;
}

static esp_err_t send_face_list(WebsocketsClient &client)
{
    client.send("delete_faces");
    face_id_node *head = st_face_list.head;
    char add_face[64];
    for (int i = 0; i < st_face_list.count; i++)
    {
        sprintf(add_face, "listface:%s", head->id_name);
        client.send(add_face);
        head = head->next;
    }
}

```

```

static esp_err_t delete_all_faces(WebsocketsClient &client)
{
    delete_face_all_in_flash_with_name(&st_face_list);
    client.send("delete_faces");
}

void handle_message(WebsocketsClient &client, WebsocketsMessage
msg)
{
    if (msg.data() == "stream") {
        g_state = START_STREAM;
        client.send("STREAMING");
    }
    if (msg.data() == "detect") {
        g_state = START_DETECT;
        client.send("DETECTING");
    }
    if (msg.data().substring(0, 8) == "capture:") {
        g_state = START_ENROLL;
        char person[FACE_ID_SAVE_NUMBER * ENROLL_NAME_LEN] =
{0,};
        msg.data().substring(8).toCharArray(person, sizeof(person));
        memcpy(st_name.enroll_name, person, strlen(person) + 1);
        client.send("CAPTURING");
    }
    if (msg.data() == "recognise") {
        g_state = START_RECOGNITION;
        client.send("RECOGNISING");
    }
    if (msg.data().substring(0, 7) == "remove:") {
        char person[ENROLL_NAME_LEN * FACE_ID_SAVE_NUMBER];
        msg.data().substring(7).toCharArray(person, sizeof(person));
        delete_face_id_in_flash_with_name(&st_face_list, person);
        send_face_list(client);
    }
    if (msg.data() == "delete_all") {
        delete_all_faces(client);
    }

    void check(){
        checking = true;
        Serial.println("start check");
    }
}

```

```

pinMode(pir_pin, INPUT);
  attachInterrupt(digitalPinToInterrupt(pir_pin), motionDetected, check);
}

void loop() {
  auto client = socket_server.accept();
  client.onMessage(handle_message);
  dl_matrix3du_t *image_matrix = dl_matrix3du_alloc(1, 320, 240, 3);
  http_img_process_result out_res = {0};
  out_res.image = image_matrix->item;

  send_face_list(client);
  client.send("STREAMING");

  if (millis() > lastTimeBotRan + botRequestDelay) {
    int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    while (numNewMessages) {
      Serial.println("got response");
      handleNewMessages(numNewMessages);
      numNewMessages = bot.getUpdates(bot.last_message_received + 1);
    }
    lastTimeBotRan = millis();
  }

  while (client.available()) {
    client.poll();

    fb = esp_camera_fb_get();

    if (g_state == START_DETECT || g_state == START_ENROLL ||
g_state == START_RECOGNITION)
    {
      out_res.net_boxes = NULL;
      out_res.face_id = NULL;

      fmt2rgb888(fb->buf, fb->len, fb->format, out_res.image);

      out_res.net_boxes = face_detect(image_matrix, &mtmn_config);

      if (out_res.net_boxes)
      {
        if (align_face(out_res.net_boxes, image_matrix, aligned_face) ==
ESP_OK)

```

```

    {

        out_res.face_id = get_face_id(aligned_face);
        last_detected_millis = millis();
        if (g_state == START_DETECT) {
            client.send("FACE DETECTED");
        }

        if (g_state == START_ENROLL)
        {
            int left_sample_face = do_enrollment(&st_face_list,
out_res.face_id);
            char enrolling_message[64];
            sprintf(enrolling_message, "SAMPLE NUMBER %d FOR %s",
ENROLL_CONFIRM_TIMES - left_sample_face, st_name.enroll_name);
            client.send(enrolling_message);
            if (left_sample_face == 0)
            {
                ESP_LOGI(TAG, "Enrolled Face ID: %s", st_face_list.tail-
>id_name);
                g_state = START_STREAM;
                char captured_message[64];
                sprintf(captured_message, "FACE CAPTURED FOR %s",
st_face_list.tail->id_name);
                client.send(captured_message);
                send_face_list(client);

            }
        }

        if (check){
            if (g_state == START_RECOGNITION && (st_face_list.count >
0))
            {
                face_id_node *f = recognize_face_with_name(&st_face_list,
out_res.face_id);
                if (f)
                {
                    checking=false;
                }
            }
            else
            {
                sendPhotoTelegram();
                String msg = "!!!Приближення невідомої особи";
                bot.sendMessage(userid, msg, "");
            }
        }
    }
}

```

```

        checking = false;

    }
}
dl_matrix3d_free(out_res.face_id);
}

}
else
{
    if (g_state != START_DETECT) {
        sendPhotoTelegram();
        String msg = "!!Виявлено рух, обличчя не знайдено";
        bot.sendMessage(userid, msg, "");
        checking = false;

    }
}

}
}

client.sendBinary(((const char *)fb->buf, fb->len);

esp_camera_fb_return(fb);
fb = NULL;
}
}

```

ДОДАТОК Б
Слайди презентації

Національний університет «Запорізька політехніка»
Кафедра програмних засобів
Дипломна кваліфікаційна робота бакалавра
на тему:

**РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ
БЕЗПЕКИ НА ОСНОВІ БЕЗДРОТОВИХ ТЕХНОЛОГІЙ**

Виконав
студент групи КНТ-211сп
Кирило ЩЕРБАКОВ

Керівник
к.т.н доцент
Анжеліка ПАРХОМЕНКО

2024

Рисунок Б.1 – Слайд 1



Рисунок Б.2 – Слайд 2

МЕТА ТА ЗАВДАННЯ ДИПЛОМНОЇ РОБОТИ

Метою роботи є розробка програмно-апаратного рішення на основі бездротових технологій для системи безпеки з розпізнаванням облич.

Завдання роботи:

- ♦ провести аналіз програмно-апаратних засобів для забезпечення системи безпеки “розумного” будинку;
- ♦ здійснити проєктування програмного забезпечення;
- ♦ створити програмне забезпечення для системи безпеки “розумного” будинку;
- ♦ виконати тестування розробленого програмного забезпечення для перевірки його ефективності.

3

Рисунок Б.3 – Слайд 3

АНАЛІЗ ПРОГРАМ ВІДЕОСПОСТЕРЕЖЕННЯ

Критерій порівняння	Назва програми		
	Blue Iris	Хеспа	Ізру
Велика вартість	+	+-	-
Підтримка різних платформ	-	+	+-
Сумісність з різними обладнанням	+	+-	-
Великі вимоги до системи	+	-	+-
Велика спільнота	+	+-	+
Відкритий код	-	-	+
Віддалений доступ	+	+	+
Великий функціонал	+	+-	-
Споживання	-	+-	+-

4

Рисунок Б.4 – Слайд 4

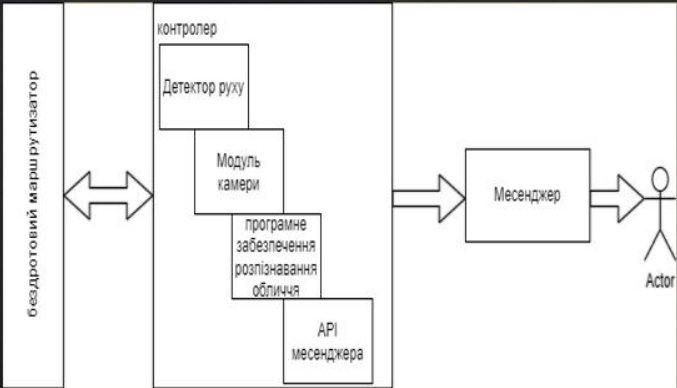
АНАЛІЗ СТАНДАРТІВ ПЕРЕДАЧІ ДАНИХ

Характеристика	Назва стандарту		
	Wi-Fi	Bluetooth	ZigBee
Частотний діапазон	2.4 ГГц, 5 ГГц	2.4 ГГц, 2.1 ГГц	2.4 ГГц
Швидкість передачі даних	До 9.6 Гбіт/с	До 3 Мбіт/с	До 250 Кбіт/с
Радіус дії	До 100 м (в приміщенні), до 1 км (на відкритому повітрі)	До 100 м	До 100 м
Масштабованість	Висока	Обмежена	Висока
Призначення	Високошвидкісна передача даних	Передача даних між пристроями на короткі відстані	Сенсорні, мережі, автоматизація

5

Рисунок Б.5 – Слайд 5

ПРОЄКТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ



6

Рисунок Б.6 – Слайд 6

ВИБІР АПАРАТНИХ ЗАСОБІВ

Характеристика	Назва модуля		
	ESP32-CAM	<u>Arduino Nano</u>	Raspberry Pi Pico
Тактова частота процесору	240MHz	16MHz	133MHz
Кількість флеш-пам'яті	4MB	32KB	133MHz
Кількість оперативної пам'яті	520KB	2KB	264KB
Комунікаційні інтерфейси	I2C, SPI, UART, <u>Wi-Fi, Bluetooth</u>	I2C, SPI, UART	I2C, SPI, UART, <u>Wi-Fi, Bluetooth</u>
Робоча напруга	2.2 - 3.6 V	5 V	1.8 - 5.5 V
Ціна \$	3-10	5-20	4-10

Характеристика	Назва модуля		
	ESP32-CAM	<u>Arduino Nano</u>	<u>Raspberv Pi Pico</u>
Можливість бездротового з'єднання	+	-	+
Легке підключення модулів камери	+	-	-
Легка робота з бібліотеками для роботи з розпізнаванням на камері	+	-	-
Достатня потужність для обробки зображень	+	-	+
Невелике споживання енергії	+-	+	+-

7

Рисунок Б.7 – Слайд 7

ВИБІР МОВИ ПРОГРАМУВАННЯ

Критерій порівняння мов програмування	Назва мови		
	C	Rust	Arduino C
Висока складність при програмуванні контролерів	+-	+-	-
Доступність бібліотек для програмування контролерів	+-	-	+
Швидкодія	+	+	+-
Вбудовані бібліотеки для роботи з контролерами	-	-	+

8

Рисунок Б.8 – Слайд 8

ВИБІР СЕРЕДОВИЩА РОЗРОБКИ

Критерій порівняння середовищ розробки	Назва середовища розробки		
	Arduino IDE	Visual Studio	Atmel studio
Вбудована підтримка Arduino C	+	-	-
Доступність бібліотек для програмування контролерів	+	+-	+
Відкритий код	+	+	-
Можливість завантажувати програму на платформу	+	+-	+
Вбудовані бібліотеки для роботи з контролерами	+	-	-

Рисунок Б.9 – Слайд 9

Схема алгоритму функціонування системи

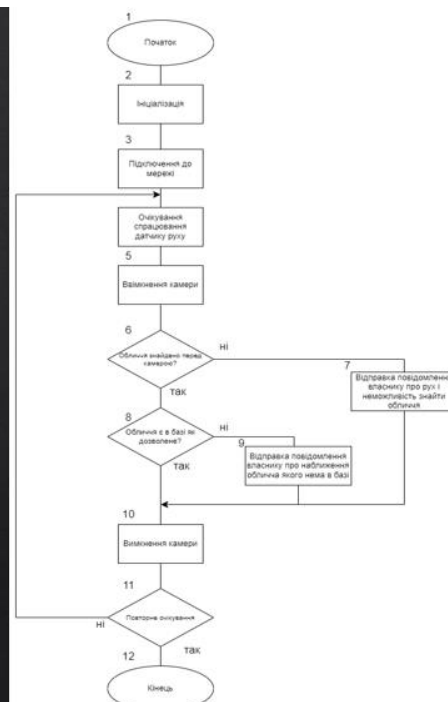
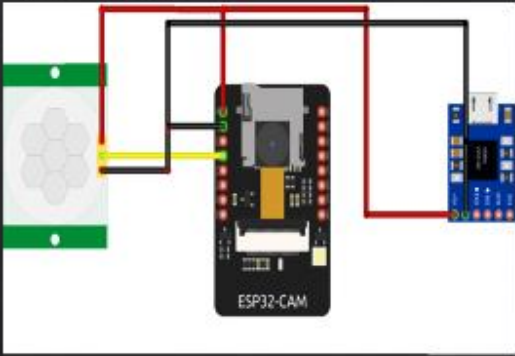


Рисунок Б.10 – Слайд 10

КОНСТРУЮВАННЯ ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ

Бібліотека	Опис бібліотеки
esp_system.h	Надає функції для взаємодії з системою, такі як скидання налаштувань пристрою або отримання ідентифікатора чіпа пристрою
esp_camera.h	Містить функції для взаємодії з модулем камери ESP32-CAM (включаючи функцію розпізнавання обличчя). Використовується для прямої трансляції через веб-сервер
esp_http_server.h	Надає функції для налаштування веб-сервера HTTP на пристрої
esp_timer.h	Містить функції для взаємодії з апаратним таймером пристрою
camera_pins.h	Визначає константи для призначень контактів для модуля ESP32-CAM.
connect_wifi.h	Містить функції для підключення пристрою до мережі Wi-Fi
UniversalTelegramBot.h	Містить функції для роботи з телеграм ботом
ArduinoJson.h	Містить функції для створення, парсингу та серіалізації JSON



11

Рисунок Б.11 – Слайд 11

ПРИКЛАД РОБОТИ



Ведити назву індикатора

Перегляд камери

Відслідковування обличчя

Додати обличчя

Список дозволених осіб

Видалити всі

З'явлено рух, обличчя не знайдено 22:14



З'явлено рух, обличчя не знайдено 22:15

Settings 21:42 ✓

Посилання для налаштування:https://192.168.1.91 21:42

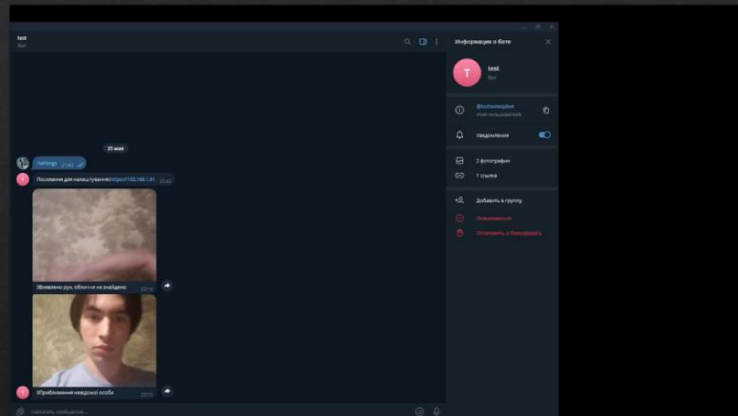


З'явлено рух, обличчя не знайдено 22:14

12

Рисунок Б.12 – Слайд 12

ДЕМОНСТРАЦІЯ РОБОТИ



13

Рисунок Б.13 – Слайд 13

ВИСНОВКИ

- ◆ Результатом виконання дипломної роботи бакалавра стало створення апаратно-програмного рішення для системи безпеки "розумного" будинку, за допомогою системи розпізнавання облич та використання бездротових технологій.
- ◆ Цінність розробки полягає в реалізації незатратного та ефективного способу забезпечення безпеки у "розумному" будинку, що підвищує рівень комфорту та безпеки мешканців будинку.
- ◆ У створеного рішення є багато можливостей розширення функціоналу: додавання більшої кількості датчиків та пристроїв для ще більш ефективного забезпечення безпеки; додавання не пов'язаних з безпекою функцію, наприклад автоматизації певних дій за допомогою зв'язування з іншими пристроями у тій самій локальній мережі.
- ◆ Результати роботи доповідались на НПК "Тиждень науки-2024".

14

Рисунок Б.14 – Слайд 14