

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Інститут інформатики та радіoeлектроніки,
Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

бакалавр

(ступінь вищої освіти)

на тему РОЗРОБКА ПРОГРАМНО-АПАРАТНОГО ЗАБЕЗПЕЧЕННЯ
ДЛЯ РЕАЛІЗАЦІЇ ТЕХНОЛОГІЇ ІНТЕРНЕТУ РЕЧЕЙ
В НАВЧАЛЬНІЙ ЛАБОРАТОРІЇ
SOFTWARE AND HARDWARE DEVELOPMENT
FOR IMPLEMENTING THE IOT TECHNOLOGY
IN A TRAINING LABORATORY

Виконав: студент(ка) 4 курсу, групи КНТ-127
Спеціальності 121 Інженерія програмного
забезпечення

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Інженерія програмного забезпечення

Бережний О.Ю.

(прізвище та ініціали)

Керівник Пархоменко А.В.

(прізвище та ініціали)

Рецензент Морщавка С.В.

(прізвище та ініціали)

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

Інститут, факультет ІРЕ, ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(КОД І НАЙМЕНУВАННЯ)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 2021 року

З А В Д А Н Н Я

НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

Бережного Олександра Юрійовича

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Розробка програмно-апаратного забезпечення для реалізації технології Інтернету речей в навчальній лабораторії Software and Hardware Development for Implementing the IoT Technology in a Training Laboratory

керівник проєкту (роботи) Пархоменко Анжеліка Володимирівна, к.т.н., доцент

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “30” березня 2021 року № 103

2. Строк подання студентом проєкту (роботи) 17 травня 2021 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. Проектування програмно-апаратного комплексу. 3. Конструювання програмно-апаратного комплексу. 4. Керівництво програміста. 5. Керівництво користувача.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-5 Основна частина	Пархоменко А.В., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “28” лютого 2021 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	2 – 3 тижні	Розділ 1
3	Розробка архітектури програми	4 тиждень	Розділ 2
4	Розробка програми	5 – 6 тижні	Розділ 3
5	Тестування та експериментальне дослідження програми	7 тиждень	Розділ 3
6	Оформлення пояснювальної записки та документів до неї. Нормоконтроль та рецензування	8 тиждень	Розділ 4, розділ 5, додатки
7	Захист роботи	9 тиждень	

Студент(ка)

(підпис) Бережний О.Ю.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Пархоменко А.В.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до випускної кваліфікаційної роботи бакалавра: 103 с., 59 рис., 24 табл., 3 дод., 35 джерел.

ВІДДАЛЕНА ЛАБОРАТОРІЯ, ВІДЕОМОНІТОРИНГ, ЗАХОПЛЕННЯ ЗОБРАЖЕННЯ, ПРОГРАМНО-АПАРАТНИЙ КОМПЛЕКС, WI-FI МОДУЛЬ, СТИСНЕННЯ ЗОБРАЖЕННЯ, ESP-32, ESP-32-CAM, ARDUINO, C++.

Об'єкт дослідження – процес організації відеомоніторингу.

Предмет дослідження – методи та засоби захоплення, трансляції, отримання й відтворення відеозображення для організації відеомоніторингу.

Мета роботи – розробка програмно-апаратного комплексу відеомоніторингу для розширення функціональних можливостей віддаленої навчальної лабораторії.

Матеріали, методи та технічні засоби: структурне та об'єктно-орієнтоване програмування, модулі ESP-32-CAM, модуль ESP-32, камери, TFT-дисплей, мова програмування C++, платформа Arduino, персональний комп'ютер з процесором Intel Core 3 під управлінням операційної системи Microsoft Windows 10.

Результати. Створено програмно-апаратний комплекс на основі SoC модулів ESP-32, який надає можливість проведення відеотрансляції та відеоспостереження для віддаленої навчальної лабораторії.

Висновки. Розроблений програмно-апаратний комплекс відеомоніторингу дозволяє розширити функціональні можливості віддаленої навчальної лабораторії, оскільки надає можливість відеоспостереження за обладнанням стенду MyHome та симуляції відеоспостереження за обладнанням стенду Smart House&IoT.

Галузь використання – віддалені навчальні лабораторії, телемедицина, системи відеоспостереження.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 103 pages, 59 figures, 24 tables, 3 appendices, 35 sources.

REMOTE LABORATORY, VIDEO MONITORING, IMAGE CAPTURE, SOFTWARE AND HARDWARE COMPLEX, WI-FI MODULE, IMAGE COMPRESSION, ESP-32, ESP-32-CAM, ARDUINO, C++

The object of research is the process of organizing video monitoring.

The subject of research is methods and tools of capturing, broadcasting, receiving and reproducing video images for the organization of video monitoring.

The goal of the work is the development of software and hardware complex of video monitoring to expand the functionality of a remote training laboratory.

Materials, methods and technical means: structural and object-oriented programming, ESP-32-CAM modules, ESP-32 module, cameras, TFT-display, C++ programming language, Arduino platform, personal computer with Intel processor Core 3 running Microsoft Windows 10.

Results. The software and hardware complex based on SoC modules ESP-32 has been created, which provides the possibility of performing video broadcasting and video surveillance for a remote training laboratory.

Conclusions. The developed software and hardware complex of video monitoring allows to expand the functionality of the remote training laboratory, as it provides video surveillance of the equipment of the MyHome stand and simulation of video surveillance of the equipment of the Smart House & IoT stand.

The fields of application are remote training laboratories, telemedicine, video surveillance systems.

ЗМІСТ

Перелік скорочень та умовних познач.....	8
Вступ.....	9
1 Аналіз предметної області.....	10
1.1 Аналіз особливостей технології Інтернету речей.....	10
1.2 Дослідження хмарних та туманних технологій як основи концепції IoT.....	17
1.3 Огляд пристроїв та протоколів для реалізації IoT	23
1.3.1 Пристрої для реалізації IoT	23
1.3.2 Протоколи для реалізації IoT	26
1.4 Огляд пристроїв відеомоніторингу та відеотрансляції	31
1.4.1 Огляд існуючих рішень для організації відеомоніторингу	31
1.4.2 Проблематика організації відеомоніторингу для віддаленої навчальної лабораторії.....	33
1.5 Постановка задач дипломної кваліфікаційної роботи.....	36
2 Проєктування програмно-апаратного комплексу	37
2.1 Аналіз вимог	37
2.2 Розробка архітектури.....	39
2.3 Вибір засобів розробки	40
2.3.1 Апаратні засоби.....	41
2.3.2 Програмні засоби	46
2.4 Висновки до розділу 2	53
3 Конструювання програмно-апаратного комплексу.....	54
3.1 Алгоритм функціонування.....	54
3.2 Особливості програмної реалізації.....	57
3.3 Тестування та експериментальне дослідження програми	64
3.4 Висновки до розділу 3	66
4 Керівництво програміста.....	67
4.1 Призначення та умови виконання програми.....	67

4.2	Характеристики програми.....	67
4.3	Звернення до програми.....	68
4.4	Вхідні та вихідні дані.....	71
4.5	Повідомлення і обробка виняткових ситуацій.....	71
5	Керівництво користувача	73
5.1	Призначення програми	73
5.2	Умови виконання програми	73
5.3	Повідомлення оператору	73
5.4	Виконання програми.....	75
	Висновки	77
	Перелік джерел посилання	78
	Додаток А Текст програми.....	83
	Додаток Б Слайди презентації	91
	Додаток В Фотозвіт процесу розробки ПАК.....	101

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

API	– Application Programming Interface;
AMQP	– Advanced Message Queuing Protocol;
BLE	– Bluetooth Low Energy;
CoAP	– Constrained Application Protocol;
GPIO	– General Purpose Input Output;
HTTP	– HyperText Transfer Protocol;
IaaS	– Infrastructure as a Service;
IDE	– Integrated Development Environment;
IoT	– Internet of Things;
IP	– Internet Protocol;
LPWAN	– Low-power Wide-area Network;
LTE-M	– Long-Term Evolution Machine Type Communication;
MP	– Mirco Python;
MQTT	– Message Queuing Telemetry Transport;
NB-IoT	– Narrowband Internet of Things;
NFC	– Near-Field Communication;
OTA	– Over The Air;
Paas	– Platform as a Service;
REIoT	– Remote Laboratory for Design of Embedded Systems and Smart House & Internet of Things;
RFID	– Radio Frequency Identification;
RTOS	– Real Time Operating System;
SaaS	– Software as a Service;
SoC	– System on a Chip;
TCP	– Transmission Control Protocol;
TFT	– Thin Film Transistor;
ОС	– операційна система;
ПАК	– програмно-апаратний комплекс.

ВСТУП

Сьогодні спостерігається бурхливий розвиток технологій Інтернету речей (Internet of Things, IoT). Перелік пристроїв, які використовують IoT постійно розширюється. IoT технології активно реалізуються в поєднанні з хмарними та туманними технологіями, оскільки вони здатні суттєво підвищити розрахункові потужності й прискорити швидкість обробки інформації.

Вже розроблено та пропонується на ринку велика кількість пристроїв, які надають можливість комунікації та налаштування через мережу Інтернет. Існують приклади реалізованих проєктів як регіонального, так і державного рівня, в тому числі в Україні.

Незважаючи на зростаючу популярність, IoT технологіям поки що притаманні певні недоліки та невирішені питання, які стримують їх всеохоплюючу інтеграцію у навколишнє середовище. Це такі питання як впровадження стандартизації, забезпечення належного рівня безпеки й конфіденційності, вартість впровадження IoT систем, поєднання з хмарними технологіями. Тому, дослідження та практична реалізація IoT технологій є актуальною задачею.

Віддалена навчальна лабораторія REIoT, що розроблена на кафедрі програмних засобів, дозволяє досліджувати програмні та апаратні засоби реалізації IoT технології. Однією з функцій віддаленої лабораторії є можливість спостереження за експериментом онлайн. При цьому відеомоніторинг надає найбільш інформативні дані, точно передає зовнішній стан обладнання віддаленої лабораторії в реальному часі, не спотворюючи його. Але методи та засоби забезпечення якісного відеомоніторингу потребують подальшого розвитку.

Мета роботи – розробка програмно-апаратного комплексу відеомоніторингу для розширення функціональних можливостей віддаленої навчальної лабораторії.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз особливостей технології Інтернету речей

IoT – технологія, за якою повсякденні предмети можуть бути оснащені можливостями ідентифікації, зондування, роботи в мережі та обробки даних. Дані можливості дозволять пристроям спілкуватися між собою через мережу Інтернет, координувати дії для досягнення певної мети [1].

Іншими словами, IoT описує мережу фізичних об'єктів (так званих "речей"), які мають вбудовані датчики, програмне забезпечення та інші технології з метою підключення та обміну даними з іншими пристроями за допомогою мережі Інтернет. Ці пристрої варіюються від звичайних побутових приладів до складних промислових інструментів [2]. Наприклад, лампочка, якою можна керувати за допомогою застосунка на смартфоні, є прикладом інтеграції приладу з вбудованою технологією IoT у побут.

Міжнародна асоціація Інститут інженерів електрики та електроніки (Institute of Electrical and Electronics Engineers, IEEE) зазначає, що метою впровадження приладів з технологією IoT є взаємозв'язок усіх речей – побутових та промислових приладів, з метою зробити їх розумними, програмованими, здатними взаємодіяти з людьми і один з одним на більш високому рівні [3]. Можна сказати, що IoT – це технологія, що оцифровує фізичний світ [4].

Прототип концепції IoT з'явився у сімдесятих роках минулого сторіччя, проте активного розвитку індустрія почала набувати лише у двадцять першому сторіччі [5]. З кожним роком все більше країн починають займатись розробкою IoT технологій, які швидко розповсюджуються у всі сфери життя суспільства, оскільки «розумні» пристрої суттєво полегшують і автоматизують рутинні процеси [6].

Безумовно, розвиток IoT зумовлено не лише чисельними фінансовими інвестиціями, а й істотним технічним прогресом, розробками нових технологій і активному поширенню вже існуючих (рис.1.1).



Рисунок 1.1 – Чинники що сприяли розвитку IoT

До основних чинників, які вплинули на розвиток IoT індустрії відносять появу і розповсюдження:

- недорогих, компактних і малопотужних датчиків, оскільки відносно дешеві та надійні датчики роблять технології IoT доступними для більшої кількості розробників і кінцевих користувачів;
- хмарних платформ, які дозволили споживачам зберігати і взаємодіяти з даними без необхідності впровадження додаткових апаратних пристроїв;
- машинного навчання, оскільки воно суттєво підвищило точність і якість аналітичних розрахунків, прогнозування даних;
- технології штучного інтелекту і голосових асистентів, які надали зрозумілий та відносно легкий користувацький інтерфейс для керування IoT системами [2].

IoT технології мають високий потенціал, та вже знайшли застосування у багатьох галузях, таких як: комунальна діяльність, забезпечення безпеки, виробничий сектор, автомобільна промисловість, торгівля, охорона здоров'я та інші. Інформаційне агентство Gartner оцінювало ринок кількості пристроїв

з IoT у 2020 році у 5,8 мільярдів одиниць, що на 21% більше в порівнянні до 2019 року (табл. 1.1) [7].

Таблиця 1.1 – Розподіл пристроїв з технологією IoT за галузями застосування

Галузь застосування	Витрати (у млрд. одиниць)	
	2019 рік	2020 рік
Комунальна діяльність	1,17	1,37
Безпека	0,95	1,09
Державний сектор	0,53	0,70
Виробництво та природні ресурси	0,40	0,49
Автомобільна галузь	0,36	0,47
Роздрібна та оптова торгівля	0,36	0,44
Автоматизація будівель	0,31	0,44
Інформаційна галузь	0,37	0,37
Охорона здоров'я	0,28	0,36
Транспорт та логістика	0,07	0,08
Усього	4,81	5,81

Комунальні підприємства використовують системи на основі IoT технологій, щоб сповіщати своїх користувачів про зміни в роботі постачання послуг, про ремонтні роботи. Розроблено і вже відбувається впровадження пристроїв автоматизованого збору даних щодо кількості споживаних послуг з можливістю миттєвої оплати через безготівковий розрахунок – так звані «розумні» лічильники.

З метою підвищення рівня безпеки і швидкого реагування на злочинні дії активно застосовуються мобільні комплекси безпеки, які дозволяють виявляти спроби несанкціонованого доступу до приміщень, миттєво повідомляти про такі випадки службу охорони, активувати розроблені програми захисту.

Державний сектор впроваджує системи типу Розумне місто, які дозволяють проводити моніторинг навколишнього середовища,

автоматизувати системи освітлення, підвищувати якість управління транспортними потоками [8].

Впровадження IoT у виробництво здатне надати конкурентну перевагу. Наприклад, постійний моніторинг здатний забезпечити завчасне технічне обслуговування обладнання, підвищити рівень контролю якості продукції. Це дозволяє компаніям зменшити експлуатаційні витрати, примножити час безвідмовної роботи виробництва і, відповідно, збільшити об'єми продукції, зменшити її собівартість.

Автомобільна промисловість усвідомлює значні переваги від використання IoT рішень: хмарне оновлення програмного забезпечення авто дозволяє швидко виправляти недоліки програмної частини без проведення відзивної компанії. Обмін даними через бездротові технології полегшує взаємодію водія з органами керування авто. Транспортні та логістичні системи також користуються пристроями IoT: маршрут вантажних автомобілів, кораблів може бути динамічно змінено в залежності від погодних умов, наявності заторів, ремонтних робіт. Вантаж може бути додатково оснащено розумними датчиками з метою забезпечення контролю його місцезнаходження, попередження випадків порушення умов транспортування, своєчасного інформування клієнтів про терміни його доставки.

У роздрібній торгівлі IoT системи дозволяють компаніям контролювати запас товару на складах, оптимізувати ланцюги поставок, зменшувати експлуатаційні витрати. Наприклад, розумні полиці, оснащені датчиками ваги, можуть здійснювати автоматичний контроль кількості товарів і сповіщати про необхідність оновлення асортименту.

Моніторинг системами IoT стає у нагоді і в галузі охорони здоров'я. Як приклад – інколи лікарям, для призначення процедур, необхідно мати детальні данні про динаміку самопочуття пацієнта. Для таких задач вже розроблено розумні браслети цілодобового моніторингу. Використання IoT пристроїв надає медичному персоналу можливість постійного нагляду за

хворими і, як наслідок, миттєвого реагування на критичні ситуації, наприклад за допомогою відеомоніторингу [9]. Це дозволяє зменшити необхідну кількість медсестер-доглядальниць та задіяти їх на інших, більш критичних напрямках медичного обслуговування.

Слід зазначити, що реалізація IoT технологій ускладнюється через певні недоліки, які стримують їх всеохоплюючу інтеграцію у навколишнє середовище (рис. 1.2).



Рисунок 1.2 – Недоліки IoT систем

Проблема конфіденційності і безпеки викликана тим, що IoT системи мають доступ до значної кількості персональних даних високої деталізації та ведуть обмін ними. Несанкціонований доступ до даних може мати серйозні наслідки, оскільки надасть зловмисникам вичерпну інформацію про об'єкт, який відстежується, дозволить спостерігати його стан в режимі реального часу. Забезпечення належного рівня безпеки ускладнено великою кількістю пристроїв і їх малою розрахунковою потужністю.

Окрім цього, сьогодні відкритим залишається питання сумісності пристроїв, оскільки немає єдиного стандарту передачі і обробки даних. Це накладає певні обмеження на розробку IoT систем та на розширення вже існуючих, що негативно відображається на часі, складності і вартості впровадження та модернізації IoT комплексів.

При впровадженні IoT слід враховувати, що відмовостійкість обладнання пов'язана не тільки з якістю виробництва складових пристроїв

системи, відсутністю помилок у програмному забезпеченні, а й з стабільністю роботи мережі. Оскільки IoT технології спираються на обмін даними – у разі відсутності Інтернет з'єднання можливе виникнення критичної для системи помилки. Тобто, при встановленні IoT системи, слід забезпечити надійне з'єднання з мережею (можливо впроваджуючи дублюючий канал зв'язку) або передбачити можливість автономної роботи.

Як і у більшості технологій вагомою перепорою є вартість впровадження системи. Адже економія ресурсів, яку здатні забезпечити IoT пристрої, можлива лише за умови розгортання якісної системи чималої вартості.

Незважаючи на існуючі недоліки, провідні держави світу не лише підтримують розвиток IoT галузі, а й займаються активним її впровадженням. За оцінками Gartner, у 2020 році за державними замовленнями було витрачено 14,7 мільярди доларів на придбання пристроїв IoT, що на 6% більше порівняно з 2019 роком (табл. 1.2) [10].

Таблиця 1.2 – Розподіл витрат державного сектору на впровадження IoT технології за галузями застосування

Галузь застосування	Витрати (у млрд. доларів США)		
	2019 рік	2020 рік	2021 рік
Зовнішнє спостереження	6,2	6,7	7,6
Управління дорожнім рухом	1,9	1,6	2,0
Вуличне та зовнішнє освітлення	2,0	1,7	1,9
Відстеження активів міста	1,4	1,6	2,0
Збір поліцейських доказів	0,6	0,9	1,3
Інше	1,9	2,1	2,5
Загальний об'єм	13,9	14,7	17,4

Міжнародна консалтингова компанія International Data Corporation прогнозує, що до 2025 року в усьому світі буде підключено 55,7 мільярдів пристроїв до мережі Інтернет, 75% з яких будуть підключені до платформи IoT [11].

Сьогодні в Україні провідні компанії стільникового зв'язку надають можливість придбання спеціалізованих тарифів для пристроїв за технологією IoT, що, безумовно, полегшує реалізацію великих проєктів. Вже існують приклади успішного впровадження в Україні IoT проєктів обласного та державного рівнів (рис. 1.3), а саме:

- організація дистанційної передачі даних з приладів обліку електроенергії у Львові, водопостачання у Луцьку, газопостачання у Кропивницькому [8];
- розвиток телекомунікаційної інфраструктури для забезпечення функціонування телемедицини;
- впровадження проєкту розумного землеробства на полях компанії Agromino у Харківській області [12];
- розгортання станції автономного моніторингу якості повітря в Києві [13].

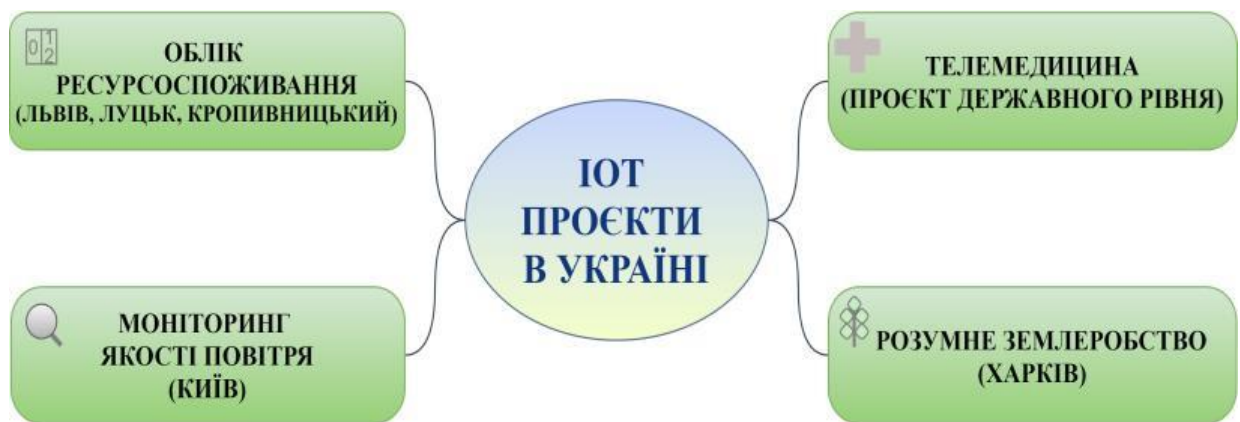


Рисунок 1.3 – Реалізовані IoT проєкти в Україні

Як показало проведене дослідження, IoT технології стрімко розвиваються. Є багато реалізованих проєктів, включно й в Україні, але наразі наявні певні недоліки, які стримують впровадження даних технологій.

1.2 Дослідження хмарних та туманних технологій як основи концепції IoT

IoT технології породили мережу взаємопов'язаних пристроїв та датчиків, які зробили революцію в виконання повсякденних завдань. Проте, самі по собі пристрої не приносять великих переваг – значущу інформацію представляють саме зібрані ними дані.

Хоча IoT і хмарні технології – різні розробки, існує тісний зв'язок між ними. Для більшої частини IoT так звана «голова» системи знаходиться в хмарі: датчики та пристрої збирають дані і виконують дії, а обробка, керування і аналітика, як правило, відбуваються в хмарі.

Хмара представляє собою централізовану систему, яка допомагає транспортувати, зберігати і аналізувати дані через Інтернет мережу [14]. Сьогодні послуги хмарних обчислень пропонують такі високотехнологічні компанії, як Amazon, Google, Microsoft та Oracle. IDC вважає, що 90% даних для IoT та Data Analytics будуть розміщені у хмарі в недалекому майбутньому [5].

Систему за хмарною технологією можна розділити на два рівні (рис. 1.4), які взаємодіють між собою:

- інтерфейсний рівень, складається з клієнтських пристроїв;
- аналітичний рівень, складається із систем зберігання та обробки даних, які розташовані далеко від клієнтських пристроїв і складають саму хмару.

Технологія хмарних обчислень надає різні види послуг, які поділяються на три групи:

- інфраструктура як послуга (IaaS – Infrastructure as a Service);
- платформа як сервіс (Paas – Platform as a Service);
- програмне забезпечення як сервіс (SaaS – Software as a Service) [15].



Рисунок 1.4 – Структура системи, побудованої за хмарною технологією

Порівняльний аналіз послуг хмарних обчислень наведено у таблиці 1.3.

IaaS – віртуальний еквівалент традиційного центру обробки даних. Постачальники хмарної інфраструктури використовують технологію віртуалізації, щоб доставити своїм клієнтам масштабовані обчислювальні ресурси, такі як сервери, мережі та сховище. Це вигідно для клієнтів, оскільки їм не потрібно купувати особисте обладнання та керувати ним.

Постачальник послуг IaaS відповідає за всю інфраструктуру, але користувачі мають повний контроль над нею. У свою чергу, користувачі відповідають за встановлення та обслуговування застосунків і операційних систем, а також, за безпеку, час роботи, проміжне програмне забезпечення та дані.

PaaS – це основа для створення програмного забезпечення, що постачається через Інтернет, пропозиція платформи із вбудованими програмними компонентами та інструментами, за допомогою яких розробники можуть створювати, налаштовувати, тестувати та запускати програми. Постачальники PaaS керують серверами, оновленнями операційної системи, виправленнями та резервними копіями даних. Клієнти

зосереджуються на розробці застосунків та даних, не турбуючись про інфраструктуру, проміжне програмне забезпечення та обслуговування ОС.

SaaS надає доступ до хмарного програмного забезпечення постачальника. Користувачам не потрібно завантажувати та встановлювати програми SaaS на локальних пристроях, але, іноді, їм можуть знадобитися додаткові плагіни. Програмне забезпечення SaaS знаходиться у віддаленій хмарній мережі і доступ до нього можливий лише через API постачальника. SaaS – найпоширеніша категорія хмарних обчислень. Постачальник SaaS управляє всім: від стабільності обладнання до функціонування застосунків. Клієнти не відповідають ні за що в цій моделі, вони використовують програми лише для виконання своїх завдань.

Таблиця 1.3 – Порівняльний аналіз IaaS, PaaS, SaaS

Параметр	Послуга		
	IaaS	PaaS	SaaS
Цільова аудиторія	Системні адміністратори	Розробники	Кінцеві користувачі
Надається користувачам	Віртуальний центр обробки даних для зберігання інформації та створення платформ для розробки служб та застосунків і їх тестування	Віртуальна платформа та інструменти для створення, тестування і розгортання програм та служб	Програмне забезпечення для виконання бізнес-завдань
Контролює постачальник	Сервери, мережу, систему віртуалізації	Сервери, мережу, систему віртуалізації, операційну систему, проміжне програмне забезпечення, середовище виконання	Сервери, мережу, систему віртуалізації, операційну систему, проміжне програмне забезпечення, середу виконання, кінцеві застосунки, данні
Контролює користувач	Операційну систему, проміжне програмне забезпечення, середовище виконання, кінцеві застосунки, данні	Кінцеві застосунки, данні	—

Оскільки пристрої за технологією IoT мають обмежений обсяг пам'яті та лімітовану обчислювальну потужність, інтеграція з хмарними обчисленнями допомагає:

- покращити продуктивність – зв'язок між датчиками IoT та системами обробки даних стає стабільніше і не обмежений за відстанню;
- підвищити ємність – надається масштабований та майже необмежений простір для зберігання даних і їх обміну;
- збільшити обчислювальний ресурс – віддалені центри обробки даних надають можливості віртуальної обробки даних, які, зазвичай, набагато вище ніж у побутових IoT пристроїв;
- зменшити витрати розробки і підтримки проєктів – плата за хмарні технології нижча за вартість локального обладнання та його постійне обслуговування.

Проте, хмарні технології мають певні недоліки при застосуванні з IoT системами:

- латентність – деякі з IoT пристроїв вимагають дуже низької затримки, а хмарні технології не можуть цього гарантувати через велику відстань між клієнтськими пристроями та центрами обробки даних;
- вірогідність простою – технічні проблеми та перебої з Інтернет мережею залишають клієнтів без можливості використання пристроїв;
- безпека та конфіденційність – приватні дані зберігаються на хмарних сховищах, які вразливі до кібератак або втрати даних.

З метою усунення недоліків хмарних технологій було розроблено туманні системи [16]. Як у природі туман знаходиться між хмарами та землею – так само і у технологічному світі: туман наближено до кінцевих споживачів з метою знизити хмарне навантаження, проте туман і хмарні технології мають тісний взаємозв'язок (рис. 1.5). Можна сказати, що туман – це розширення хмарних обчислень, яке складається з безлічі граничних вузлів, безпосередньо підключених до фізичних пристроїв.

Граничні вузли знаходяться ближче до кінцевих пристроїв, у порівнянні з хмарними центрами обробки даних, тому вони здатні забезпечувати миттєве з'єднання. Значна обчислювальна спроможність граничних вузлів дозволяє їм виконувати об'ємні обчислення даних самостійно, не надсилаючи їх на віддалені сервери.

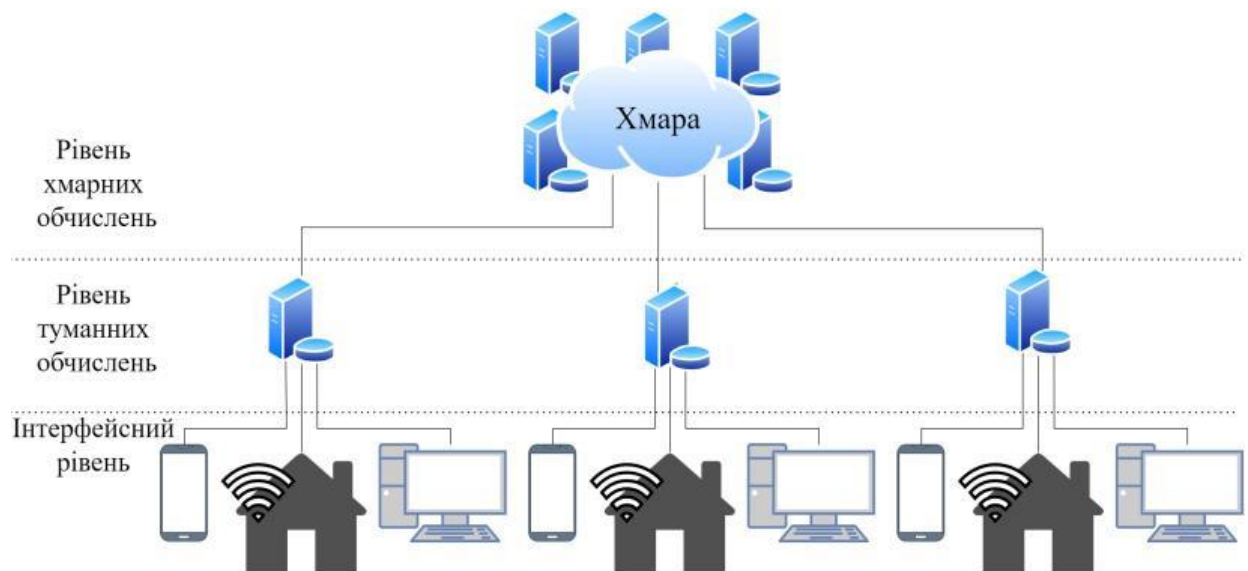


Рисунок 1.5 – Структура системи, побудованої за технологією туману

Основна різниця між обчисленнями туману та хмарними обчисленнями полягає в тому, що хмара – це централізована система, тоді як туман – розподілена децентралізована інфраструктура.

Туман є посередником між обладнанням та віддаленими серверами. Він розподіляє і регулює яку інформацію слід надсилати на сервер, а яку можна обробляти локально. Таким чином можна визначити, що туман – це розумний шлюз, який розвантажує хмару, забезпечуючи більш ефективне зберігання, обробку та аналіз даних. Варто звернути увагу, що туман не є окремою архітектурою і, відповідно, не замінює хмарні обчислення, а доповнює їх, наближаючись до джерела інформації наскільки це можливо.

Туманний підхід має багато переваг для IoT, великих даних та аналітики в режимі реального часу. Основними перевагами обчислень туману перед хмарними обчисленнями є:

- низька затримка – туман розташовано ближче до користувачів, що забезпечує можливість надавати миттєві відповіді;
- збільшення пропускну здатності – інформація збирається по частинах в різних точках, на відміну від хмарного підходу, де відбувається відправка даних разом в один центр за одним каналом;
- зменшена вірогідність втрати зв'язку – інформація передається через безліч взаємопов'язаних каналів;
- енергоефективність – кінцеві пристрої використовують енергоефективні протоколи (Bluetooth, Zigbee або Z-Wave).

Серед недоліків застосування туманних технологій можна виділити:

- складність архітектури – туман є додатковим шаром в системі обробки та зберігання даних;
- обмежена масштабованість – туман не такий масштабований, як хмара і вимагає більше зусиль для розширення системи.

Отже, хмарні і туманні технології певною мірою схожі між собою, проте існує ряд відмінностей між ними (табл. 1.4).

Таблиця 1.4 – Порівняння хмарних і туманних систем

Параметр	Система	
	Хмара	Туман
Архітектура	Централізована	Розподілена
Обробка даних	На великій відстані від джерела інформації	В безпосередній близькості до джерела інформації
Обчислювальні потужності	Присутні у великій кількості	Присутні
Кількість вузлів	Мінімально можлива	Відносно велика кількість
Терміни обробки інформації	Тривалий період	Короткий термін
Підключення	Інтернет	Різні протоколи та стандарти

Хмарна архітектура централізована і складається з великих центрів обробки даних, які можуть бути розташовані по всьому світу, за тисячами кілометрів від клієнтських пристроїв. Архітектура туману розподілена і складається з невеликих вузлів, розташованих якомога ближче до клієнтських пристроїв.

Туман виступає посередником між центрами обробки даних та обладнанням, а, отже, він ближче до кінцевих користувачів. Якщо шару туману немає, хмара безпосередньо спілкується з пристроями, що займає багато часу.

У хмарних обчисленнях обробка даних відбувається у віддалених центрах обробки даних. Обробка та зберігання за туманними технологіями здійснюється поблизу джерела інформації, що має вирішальне значення для контролю в режимі реального часу.

Хмара є потужнішою за туман щодо обчислювальних можливостей та ємності зберігання і націлена на довгостроковий глибокий аналіз, тоді як туман виконує короткочасний аналіз, забезпечуючи миттєву реакцію.

Система, яка застосовує хмарну технологію припиняє роботу без з'єднання з мережною Інтернет. Обчислення в туманних системах мають менший ризик виникнення зупинки, завдяки використанню різних протоколів та стандартів передачі даних.

Отже, хмарні і туманні технології, незважаючи на подібність, володіють власними перевагами і недоліками, тож вибір конкретного рішення залежить від задачі, що вирішується, її умов і потреб.

1.3 Огляд пристроїв та протоколів для реалізації IoT

1.3.1 Пристрої для реалізації IoT

Пристрої для реалізації IoT поділяються на два типи: інтелектуальні пристрої і перетворювачі (рис.1.6).



Рисунок 1.6 – Пристрої для реалізації IoT

Інтелектуальні пристрої – пристрої на основі мікроконтролера або мікропроцесора, які мають можливість проводити обчислення і здатні здійснювати керування певними процесами [17].

Перетворювачі – це пристрої, які перетворюють одну форму енергії в іншу, включають внутрішні датчики та виконавчі механізми (актуатори).

Датчики – пристрої, які виявляють зміни у своєму середовищі та створюють електричні імпульси для передачі значень параметрів, що вимірюються. Зазвичай призначені для виявлення і вимірювання зміни стану навколишнього середовища: температури, вологості, тиску, загазованості, освітленості, рівню шуму, вібрації, швидкості, прискорення, положення GPS, висоти та інших змінних. Датчики розділяються за типами вимірювальних величин, в межах одного типу відрізняються точністю вимірювань, роздільною здатністю, умовами роботи, надійністю, вартістю.

Актуатори – приводи, які виконують фізичні дії при отриманні відповідних команд від центра управління, що надходять, як правило, у відповідь на зміни, виявлені датчиками.

Розробка пристроїв за технологією IoT стає доступнішою завдяки появі недорогих, доступних плат модульної розробки апаратного забезпечення і великої кількості різноманітних актуаторів та датчиків, уніфікованих за інтерфейсом під'єднання. Модульні конструкції

забезпечують велику гнучкість розробки [18]. Така архітектура надає можливість відносно швидко власноруч оновлювати апаратну конструкцію, відповідно до потреб розробника. Завдяки широкому вибору компонентів розробники в короткі терміни можуть провести тестування різних за характеристиками датчиків, актуаторів та обрати найоптимальніші.

Багато модульних апаратних пристроїв розроблені навколо інтегральної схеми System-on-a-Chip (SoC), яка поєднує на одному чіпі такі можливості як: обробка даних, зберігання даних, мережеве з'єднання.

SoC уявляє собою інтегральну схему, яка має один або декілька мікроконтролерів, блок пам'яті (оперативна, постійна, програмована), регулятор живлення та інші. Датчики та актуатори підключаються до модулю за допомогою цифрових або аналогових пінів GPIO (General Purpose Input / Output) або через апаратну шину. Прикладами SoC систем є ESP-8266, ESP-32, ESP-32-CAM та інші (рис. 1.7).



Рисунок 1.7 – Приклади SoC систем

За результатами аналізу виявлено, що класична IoT система складається з інтелектуальних модулів, датчиків і актуаторів. В якості головного інтелектуального модулю найчастіше використовують модуль з SoC архітектурою.

1.3.2 Протоколи для реалізації IoT

Протоколи для реалізації IoT класифікуються на дві категорії (рис. 1.8):

- протоколи даних IoT, які отримують реалізацію на програмному рівні взаємодії;
- мережеві протоколи IoT, які реалізуються на фізичному (апаратному) рівні взаємодії [19].

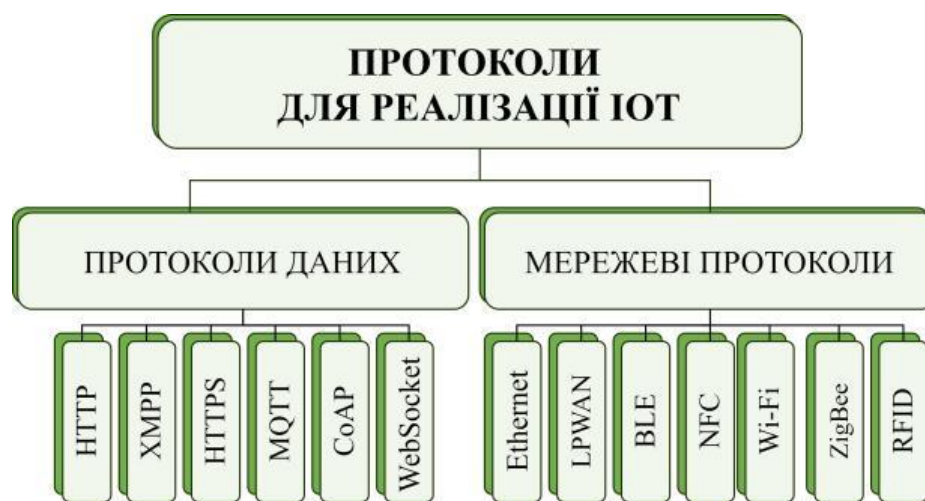


Рисунок 1.8 – Протоколи для реалізації IoT

Протоколи даних IoT охоплюють обмін повідомленнями і інформацією на рівні програмного забезпечення. Найчастіше в IoT мережах використовуються: протоколи HTTP, HTTPS, CoAP (полегшена версія HTTP), MQTT, AMQP та XMPP (рис. 1.9) [20].

MQTT (Message Queue Telemetry Transport) – легкий протокол передачі даних IoT. Має модель обміну повідомленнями видавець-підписник і забезпечує простий обмін даними між різними пристроями. Протокол було розроблено для використання в мережах з низькою пропускну здатністю, що стало необхідним через збільшення кількості дрібних та малопотужних пристроїв, які з'явилися в мережі за останні роки. MQTT властива відносно легка масштабованість, простий у використанні.

AMQP (Advanced Message Queuing Protocol) – відкритий стандартний протокол, який використовується для обміну повідомлень між серверами. Основними функціями цього протоколу IoT є отримання і розміщення повідомлень у чергах, зберігання повідомлень. Завдяки високому рівню надійності найчастіше застосовується в сферах з підвищеними вимогами до безпеки, наприклад у банківській галузі. Важкість архітектури протоколу не дуже підходить для пристроїв-датчиків, яким властиво мати невеликий об'єм пам'яті. Через це його використання в IoT має обмежений характер.

XMPP (eXtensible Messaging and Presence Protocol) – розширюваний протокол обміну повідомленнями та присутності. Протокол розроблено на мові XML розмітки для зв'язку в режимі реального часу шляхом миттєвого обміну повідомленнями. Через спорідненість з XML форматом має високу наочність, проте його повідомленням властива перевантаженість зайвими даними, що зумовлено структурою формату.

HTTP (HyperText Transfer Protocol) – протокол, розроблений для передачі гіпертекстових документів, наразі дозволяє передавати довільні дані. Через складність архітектури і великий час роботи не є кращим, як стандарт IoT для передачі коротких повідомлень, але використовується в випадках, коли треба забезпечити обмін великими за об'ємом даними. Наприклад, забезпечує підключення у мережі ПК до тривимірних принтерів і передачу тривимірних моделей для друку.

CoAP (Constrained Application Protocol) – це протокол прикладного рівня (для взаємодії мережі і користувача). Розроблено для задоволення потреб IoT систем на основі HTTP протоколу. Передача через протокол HTTP може бути реалізована більшістю пристроями IoT, проте таке рішення є важким та енергоємним. CoAP вирішив проблеми об'ємності та енергоємності, які властиві HTTP протоколу, окрім цього має вбудовану можливість багатоадресної передачі даних. Проте протокол передає данні великого обсягу гірше ніж HTTP.

WebSocket – це протокол, що призначений для одночасного двунправленого обміну інформацією. Рекомендовано застосувати до системи, в якій дані передаються безперервно на декілька пристроїв, систем моніторингу (тих, які вимагають швидкого та постійного оновлення).

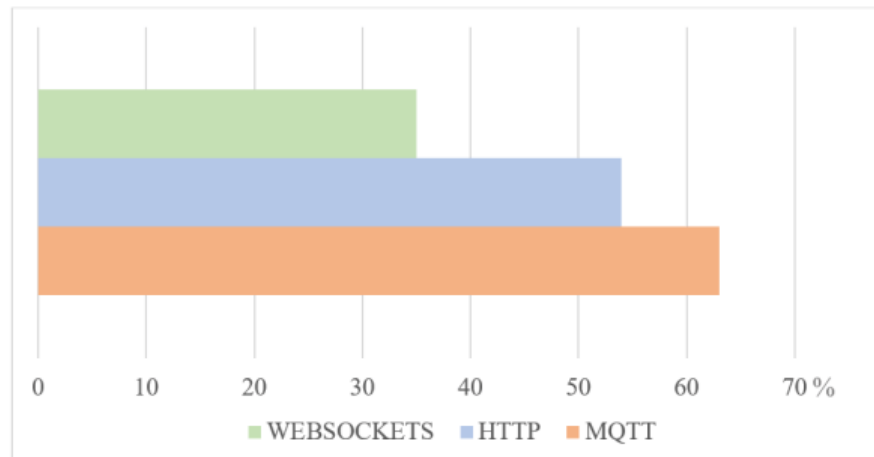


Рисунок 1.9 – Частка найпопулярніших протоколів обміну в IoT системах [21]

Серед мережевих технології IoT слід звернути увагу на стільникові мережі, Wi-Fi та Ethernet, а також більш спеціалізовані рішення, такі як LPWAN, Bluetooth Low Energy (BLE), ZigBee, NFC та RFID (табл. 1.5) [20].

Таблиця 1.5 – Порівняння мережевих протоколів

Назва	Частота	Швидкість даних	Відстань передачі	Енергоспоживання	Вартість
NB-IoT	Стільникові частоти	0.1-1 Мб/с	Декілька кілометрів	середнє	висока
LTE-M	Стільникові частоти	1-10 Мб/с	Декілька кілометрів	середнє	висока
BLE	2.4 ГГц	1-3 Мб/с	до 100 м	низьке	середня
ZigBee	2.4 ГГц	250 Кб/с	~ 100 м	низьке	висока
NFC	13.56 МГц	424 Кб/с	до 4 см	помірне	середня
RFID	433 МГц	-	до 100 м	відсутнє або низьке	низька
Wi-Fi	2.4 ГГц, 5 ГГц	від 0.1 Мб/с	менше 100 м	високе	середня
Ethernet	-	10 Мб/с	необмежено	помірне	низька

LPWAN (low-power wide-area network) – тип комунікації, призначений для бездротового зв'язку з низьким енергоспоживанням. Підходить для широкомасштабного розгортання малопотужних пристроїв IoT, таких як бездротові датчики.

LPWAN реалізовано у стандартах NB-IoT (Narrowband IoT) та LTE-M (Long-Term Evolution – Machine Type Communication), які забезпечують низькопотужний, варіант зв'язку IoT пристроїв завдяки використанню стільникових мереж. NB-IoT може забезпечити гнучке управління пристроїв і безпеку мережі [22]. LTE-M, в порівнянні з NB-IoT, забезпечує вищі швидкості передачі даних через мережу, але вимагає більшої пропускної здатності і є дорожчим при використанні. У березні 2019 року асоціація постачальників послуг мобільного зв'язку GSA повідомила, що понад 100 операторів стільникового зв'язку розгорнули мережі NB-IoT або LTE-M [23].

BLE (Bluetooth Low Energy) – версія протоколу бездротового зв'язку Bluetooth з низьким енергоспоживанням. Розроблено для зв'язку на малу дальність – не більше 100 метрів. Використовується, як правило, у зірковій конфігурації з одним основним пристроєм, який управляє декількома підрядними. BLE найкраще підходить для пристроїв, які передають серії малих обсягів даних і мають необхідність переходити у режим зниженого енергоспоживання, так званого «сну». Прикладом застосування можуть виступати «розумні» годинники для фітнесу.

ZigBee – протокол зв'язку, який працює на частоті 2,4 ГГц. В порівнянні з BLE має низьке енергоспоживання і більшу дальність зв'язку, проте має складнощі при роботі з пристроями, які здатні переходити у режим «сон». Відносно простий протокол обміну пакетними даними орієнтовний на безпечну передачу даних при малих швидкостях, легко масштабується.

NFC (Near-field communication) – протокол зв'язку ближнього поля, використовується для зв'язку на малих відстанях (до 4 см). Пропонує низькошвидкісне з'єднання з простим налаштуванням. NFC часто використовується для платіжних систем, систем реєстрації.

RFID (Radio-frequency identification) – протокол радіочастотної ідентифікації. Теги RFID зберігають ідентифікатори та дані, прикріплені до пристроїв і зчитуються за допомогою зчитувача RFID. Типовий діапазон дії RFID тегу становить не більше метра. Теги RFID можуть бути активними, пасивними або допоміжними пасивними. Пасивні мітки ідеально підходять для пристроїв без джерела живлення, оскільки ідентифікатор пасивно зчитується читачем. Активні теги періодично транслюють свій ідентифікатор, тоді як допоміжні пасивні теги стають активними лише у присутності зчитувача RFID. Активні теги розповсюджують сигнал до 100 метрів. Подібно до NFC, типовим випадком використання RFID є відстеження предметів у роздрібних та промислових застосунках IoT.

Wi-Fi – це протокол стандартної бездротової мережі, який пропонує одну з найбільших пропускних спроможностей для даних, але потребує великого енергоспоживання. Wi-Fi застосовується в багатьох пристроях IoT і є найрозповсюдженішим на сьогоднішній день.

Ethernet – найпоширеніший протокол у сучасних локальних дротових мережах. Завдяки популярності і доступності інколи використовується для під'єднання датчиків. Наприклад, при під'єднання датчиків, які встановлені в системі автоматизації житлових будівель чи офісів.

При обранні протоколу даних і мережевого протоколу слід враховувати, що трансляція відеопотоку є передачею даних великого обсягу.

Беручи до уваги складові лабораторії RELDES, а саме – наявність роутера, в якості мережевого протоколу для ПАК обрано Wi-Fi протокол. Окрім зниження вартості розробки і зменшення складності інтеграції системи таке рішення зумовлено тим, що, згідно з проведеним аналізом, Wi-Fi з'єднання є одним з найкращих засобів передачі даних великого обсягу.

Через чималий розмір даних неможливо застосувати найпоширеніший IoT протокол передачі даних MQTT. Враховуючі те, що поточна камера і сервер лабораторії REIoT використовують HTTP протокол, який, до речі, дозволяє передавати дані значного об'єму – для передачі відеопотоку обрано

саме його. З метою розширення дослідницьких можливостей лабораторії, а також, беручи до увагу високу популярність протоколу WebSocket і властиві йому функціональні можливості – його обрано в якості основного в системі симуляції відеоспостереження.

1.4 Огляд пристроїв відеомоніторингу та відеотрансляції

Відеомоніторинг – це процес віддаленого спостереження за певним об'єктом шляхом застосування відеозахоплюючої та відеовідтворюючої апаратури.

1.4.1 Огляд існуючих рішень для організації відеомоніторингу

Пристрої забезпечення відеомоніторингу розроблюються за допомогою залучення сучасних технологій виробництва засобів захоплення зображення в поєднанні з розробкою та застосуванням програмного забезпечення, яке відповідає вимогам, що висунуті до використання пристрою. У найпростішому випадку задача пристрою полягає в трансляції зображення через один з існуючих інтерфейсів передачі даних.

Сьогодні на ринку представлено доволі широкий асортимент відеокамер для забезпечення моніторингу. Одним з найсуттєвіших параметрів є спосіб передачі даних. Відеокамери поділяються за способом передачі даних на аналогові і цифрові (так звані IP відеокамери). Обидва типи застосовуються при реалізації відеомоніторингу і, як показали проведені дослідження, кожен з них має переваги і недоліки [24].

Аналогова відеокамера – пристрій, призначений для перетворення оптичного зображення в аналоговий відеосигнал в системах відеомоніторингу. На відміну від цифрових відеокамер, аналогові мають меншу вартість, але для передачі й перегляду відеозображення на комп'ютері потребують додаткового аналогово-цифрового перетворювача [25].

Цифрові камери позбавлені такого недоліку, оскільки зображення транслюється безпосередньо у цифровому форматі. Деякі камери такого типу мають вбудований Wi-Fi адаптер, який дозволяє спростити налаштування і встановлення камери, підключивши її бездротовим способом до мережі Інтернет. Вихідне зображення з цифрових камер має більшу чіткість і роздільну здатність, ніж зображення з аналогічної аналогової камери. Кадр можна збільшувати без суттєвих спотворень для перегляду дрібних деталей, подібно до того, як сьогодні працюють камери на смартфоні [26]. Результати порівняння основних характеристик відеокамер зведено у таблицю 1.6.

Таблиця 1.6 – Порівняння IP і аналогових відеокамер [27]

Параметр	IP відеокамера	Аналогова камера
Доступ та управління	Можлива розподілена або закрита (локальна) архітектура побудови системи. У разі відритої системи можна переглядати зображення з камери в реальному часі і управляти через веббраузер	Тільки закрита (локальна) архітектура системи. Немає можливості керувати камерою віддалено без додаткового обладнання
Якість	Цифровий сигнал не втрачає якості під час його передачі по каналах зв'язку і зберіганні	Якість зображення з аналогової камери часто спотворюється при великих обсягах комутаційного кабелю
Масштабованість системи	Доволі просте додавання камери в існуючу мережу	Складність підключення додаткових камер, яка пов'язана з необхідністю прокладати окремий кабель від пристроїв обробки до камери

Слід зазначити, що цифрові відеокамери мають більшу вартість ніж аналогові [28]. Проаналізувавши в березні 2021 цінову політику відеокамер у декількох провідних інтернет-магазинах (Rozetka, Comfy), отримано дані щодо мінімальної ціни на аналогові та цифрові відеокамери (табл. 1.7). Результати аналізу продемонстрували, що відмінність в ціні в залежності від типу камери досягає майже двократного розміру.

Таблиця 1.7 – Мінімальна ціна аналогових та цифрових відеокамер у Інтернет магазинах

	Comfy.ua	Rozetka.com.ua
Аналогова відеокамера	395 грн. (Dahua DH-HAC-T1A11P)	480 грн. (Atis MHD AMW-2MIR-20W)
Цифрова відеокамера	739 грн. (Hikvision DS-2CE16D0T-IRF)	738 грн. (Green Vision GV-073-IP-H-DOA14-2)

Враховуючи відмінності у якості вихідного зображення, а також той факт, що різницю у вартості різного типу камер буде зменшено через потребу у придбанні аналогово-цифрового перетворювача – при виборі пристроїв відеомоніторингу доцільним є звернути увагу на цифрової відеокамери.

1.4.2 Проблематика організації відеомоніторингу для віддаленої навчальної лабораторії

Інтеграція віддаленої лабораторії в навчальний процес розширює можливості дистанційного навчання і надає можливість перевірки на практиці теоретичних знань [29].

Одним з завдань віддалених лабораторій є забезпечення можливості відслідковування ходу експерименту, тобто підвищення наочності досліджень. Найкращім способом забезпечити спостереження є впровадження відеомоніторингу, оскільки він надає найбільш інформативні дані, точно передає зовнішній стан системи в реальному часі, не спотворюючи його.

Лабораторія REIoT (Remote Laboratory for Design of Embedded Systems and Smart House & Internet of Things) – це комплекс, який розроблено на платформах Raspberry, Arduino. Він може використовуватись для проведення досліджень та виконання навчальних завдань.

REIoT об'єднує дві підсистеми: RELDES (віддалена лабораторія з розробки вбудованих систем) та Smart House&IoT (віддалена лабораторія дослідження технологій розумного будинку). На сайті лабораторії доступна

можливість передачі команд на пристрої комплексу й спостереження за їх виконанням через відеотрансляцію в режимі реального часу, яка здійснюється завдяки IP-камері D-Link DCS-2121.

Камера за будовою становить повноцінну систему з процесором і веб-сервером. Під'єднання до комп'ютеру реалізовано через протокол Ethernet. Здійснення відеотрансляції відбувається з залученням утиліти ffmpeg, яка захоплює відео роздільною здатністю 1280 x 1024 з камери, кодує його в формат MPEG з частотою 10 кадрів на секунду і бітрейтом 800 КБіт/с для того, щоб не створювати надлишкового навантаження на канал передачі даних. Після кодування через HTTP протокол камера відправляє зображення на локальний сервер, який направляє відеострім кінцевому користувачу.

Окрім RELDES і Smart House&IoT лабораторій існує мобільний стенд домашньої автоматизації MyHome від компанії Legrand (рис. 1.10), який здатен розширити функціонал існуючого комплексу, оскільки дозволяє ознайомитись з промисловими рішеннями реалізації IoT технологій. Але, через відсутність можливості відеотрансляції, стенд не може бути під'єднаний до SREIoT.

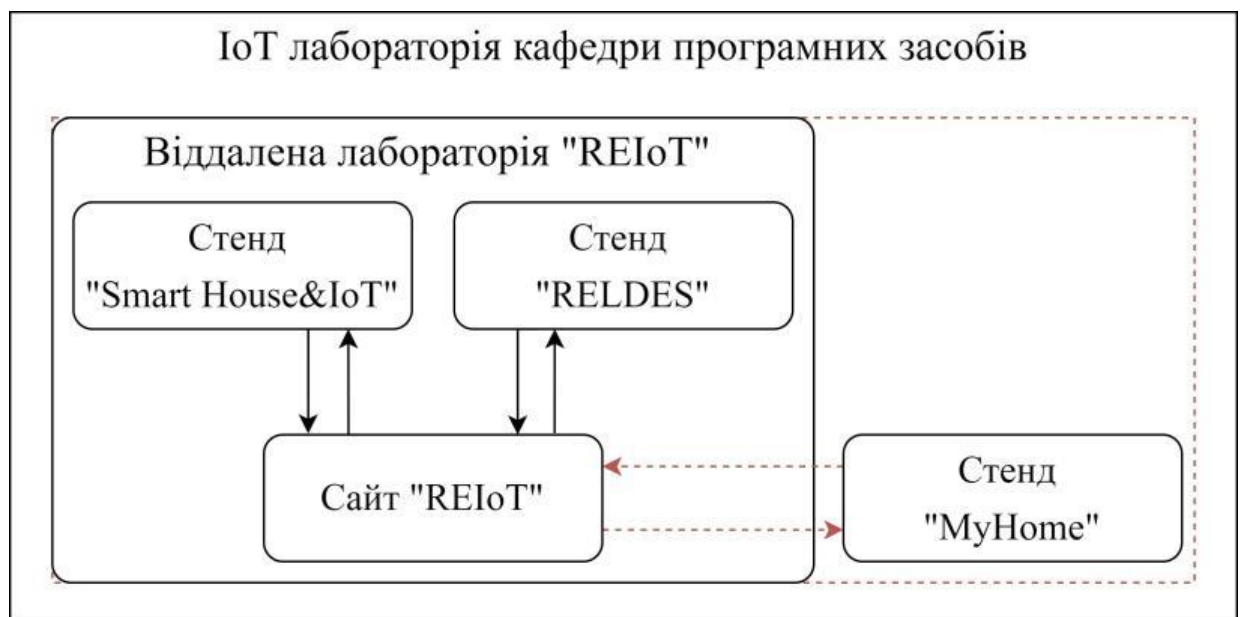


Рисунок 1.10 – Структура віддаленої лабораторії

Підключення стенду MuHome до лабораторії REIoT призведе до збільшення площі, яку необхідну транслювати на сайт лабораторії. Розширити зону відеозахоплення можна декількома шляхами (табл. 1.8):

- збільшити області захоплення поточної відеокамери;
- впровадити додаткову відеокамеру.

Розширити область захоплення поточної відеокамери можна двома способами: збільшити відстань до об'єктів спостереження або застосувати широкоформатний об'єктив. Збільшення відстані призведе до зменшення рівня деталізації, що ускладнить спостереження за проведенням експериментів та зменшить їх наочність. Широкоформатні об'єктиви характеризуються дуже великим кутом захоплення навколишнього простору, але, через надшироке охоплення, у вихідному зображенні будуть присутні суттєві оптичні викривлення, що не є прийнятним для сприйняття людським оком [30].

Залучання додаткового пристрою відеофіксації дозволить зберегти високу деталізацію зображення об'єктів дослідження, уникнути викривлень. Проте, слід зазначити, що таке рішення потягне чималі фінансові витрати через відносно високу вартість обладнання.

Таблиця 1.8 – Порівняння методів розширення області відеозахоплення

Параметр порівняння	Поточна відеокамера		Додаткова відеокамера
	Збільшення відстані	Зміна об'єктиву	
Зображення відносно поточного	Зменшено деталізацію	Викривлено	Без змін
Витрати на впровадження	Незначні	Середні	Високі
Наочність експериментів	Ускладнено	Ускладнено	Без змін

Отже, незважаючи на фінансові витрати, найбільш прийнятним для розширення області відеозахоплення є впровадження додаткового пристрою відеофіксації, оскільки лише даний варіант забезпечує збереження поточної якості зображення.

1.5 Постановка задач дипломної кваліфікаційної роботи

Як показали проведені дослідження, IoT технології стрімко розвиваються в усьому світі. Існують приклади реалізованих проєктів як регіонального, так і державного рівня, в тому числі в Україні. На сьогодні розроблено багато пристроїв, протоколів, технологій, які дозволяють впроваджувати IoT системи у багатьох галузях, але, при цьому, існують поки що невирішені питання. Отже, вивчення IoT технологій є актуальною задачею. Віддалена лабораторія кафедри програмних засобів спрямована на вивчення цих технологій, але потребує модернізації, а саме:

- забезпечення якісної відеотрансляції виконуваних експериментів стенду MuHome;
- розширення функціоналу віддаленої лабораторії підсистемою симуляції відеоспостереження.

Метою дипломної кваліфікаційної роботи є розробка програмно-апаратного комплексу відеомоніторингу для розширення функціональних можливостей віддаленої навчальної лабораторії.

Для досягнення мети роботи необхідно вирішити такі завдання:

- виконати аналіз вимог та розробити архітектуру ПАК;
- обрати апаратні та програмні засоби розробки;
- виконати програмну реалізацію;
- провести тестування ПАК;
- розробити програмну документацію.

2 ПРОЄКТУВАННЯ ПРОГРАМНО-АПАРАТНОГО КОМПЛЕКСУ

2.1 Аналіз вимог

Засновуючись на проведеному аналізі поточного стану ІоТ лабораторії, було складено функціональні і нефункціональні вимоги (рис. 2.1), реалізація яких сприятиме розширенню функціоналу лабораторії.

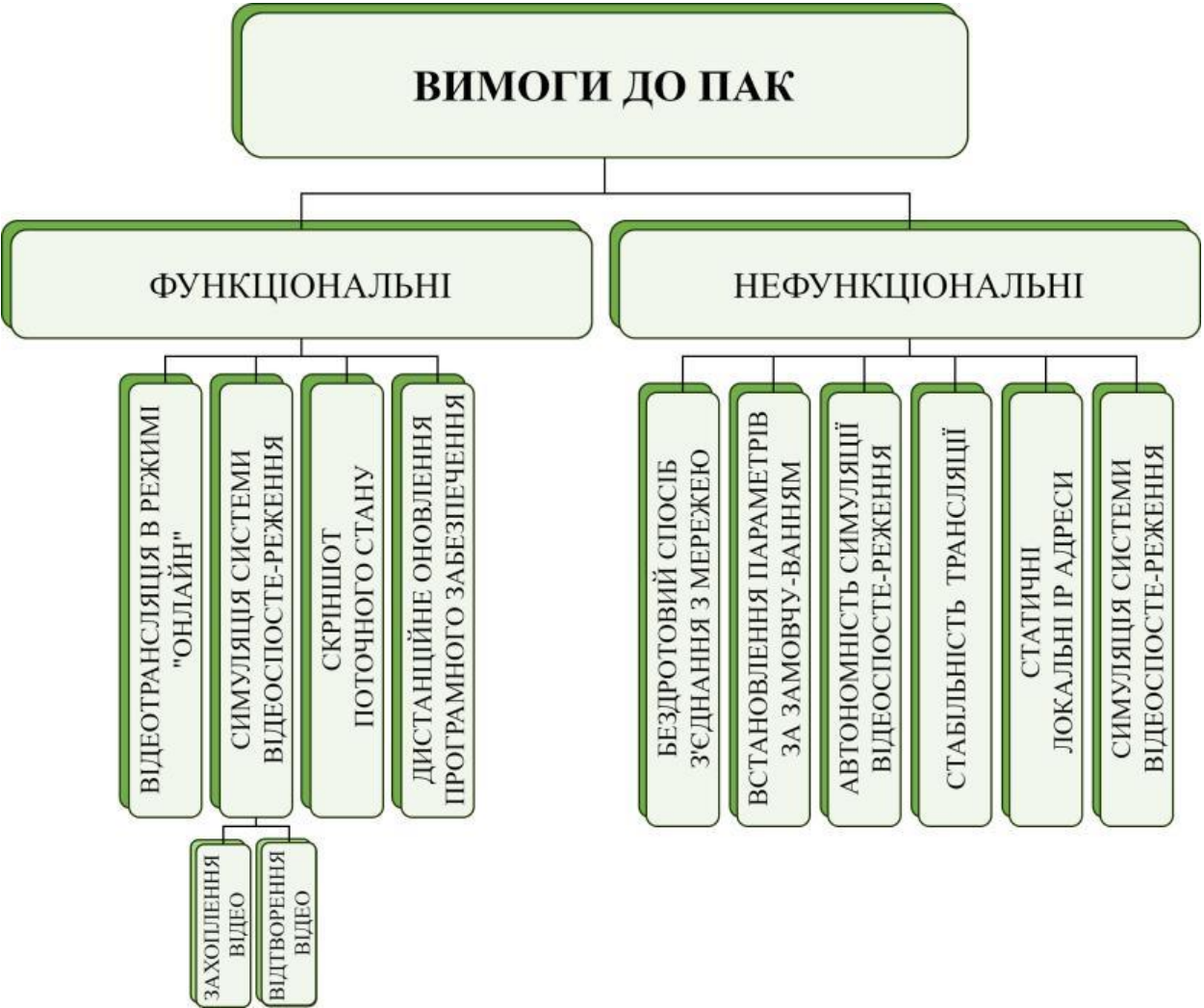


Рисунок 2.1 – Вимоги до ПАК

Для забезпечення користувачам лабораторії можливості спостереження за роботою стенду MuNome, ПАК повинен мати функціонал для трансляції відео в режимі онлайн.

Оскільки тестування розроблених програм може займати велику кількість часу ПАК повинен мати функціонал, щоб за відповідним запитом надавати зображення поточного стану системи – робити скріншот.

Враховуючи актуальність питання забезпечення відеомоніторингу, виникає потреба у створенні навчальних пристроїв для його вивчення. Отже, необхідно, щоб ПАК у своєму складі мав підсистему симуляції відеоспостереження, а саме: блок захоплення відео та блок відтворення відео в режимі реального часу. Це розширить дослідницькі можливості лабораторії, надавши користувачам лабораторії можливість ознайомитись з основами організації відеомоніторингу.

Окрім цього, з метою полегшення модернізації програмного забезпечення комплексу, необхідно впровадити функцію дистанційного оновлення програмного забезпечення.

Враховуючи складність монтажу комунікацій для дротового з'єднання з мережею, а також з метою збереження мобільності стенду MuNote, з'єднання ПАК з мережею Інтернет повинно бути реалізовано бездротовим способом (див. п. 1.3.2).

Складові блоки ПАК повинні надавати можливість встановлення за замовчуванням налаштувань основних параметрів (яrkість, контрастність, баланс білого). Зображення, що транслюється, повинно бути належної якості, тобто відеопотік повинен бути стабільним, без переривань.

Для того, щоб уникнути можливості конфлікту ПАК з існуючими пристроями в локальній мережі і для полегшення його налаштування – блоки комплексу повинні мати статичні локальні IP адреси.

Частина ПАК, яка є симуляцією підсистеми відеоспостереження повинна бути мобільною і незалежною, тобто, взаємодія модулю захоплення і модулю відтворення відео повинна бути реалізована без залучання потужностей існуючих стендів).

Враховуючі проаналізовані вимоги складено діаграму прецедентів взаємодії користувача з ПАК (рис. 2.2).

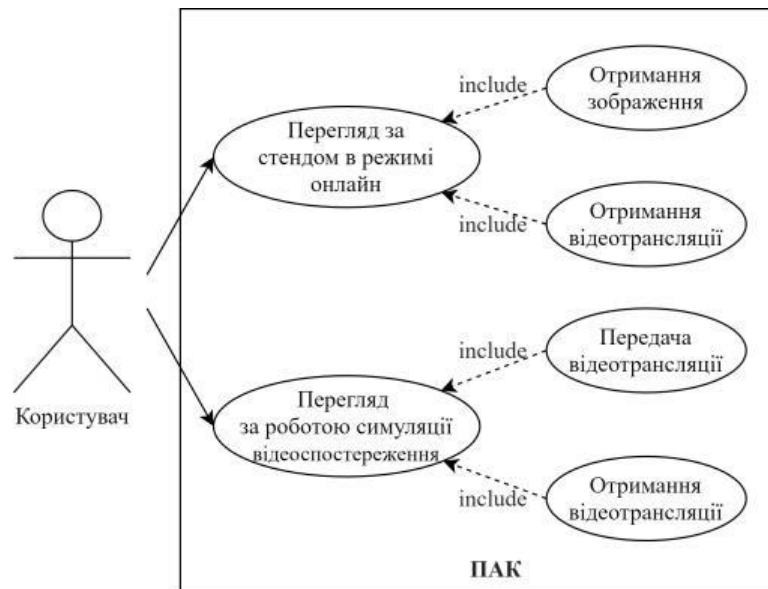


Рисунок 2.2 – Діаграма прецедентів взаємодії користувача з ПАК

На основі проведеного аналізу вимог може бути спроектована архітектура ПАК.

2.2 Розробка архітектури

Розроблену архітектуру ПАК наведено на рисунку 2.3.

Архітектуру розроблено, враховуючи результати аналізу принципу взаємодії компонентів. ПАК складається з:

- блоку захоплення та трансляції відеозображення на сервер;
- підсистеми симуляції відеоспостереження.

У свою чергу підсистема симуляції відеоспостереження складається з двох блоків:

- блоку захоплення та трансляції відеозображення
- блоку отримання і відтворення відеозображення.

Отже, ПАК у своєму складі має два блоки захоплення й трансляції відеозображення. Кожен з блоків складається з модулю камери та модулю обробки і передачі зображення.

Блок отримання і відтворення відеозображення один в ПАК і в своєму складі має модуль отримання та обробки зображення і модуль відтворення зображення.

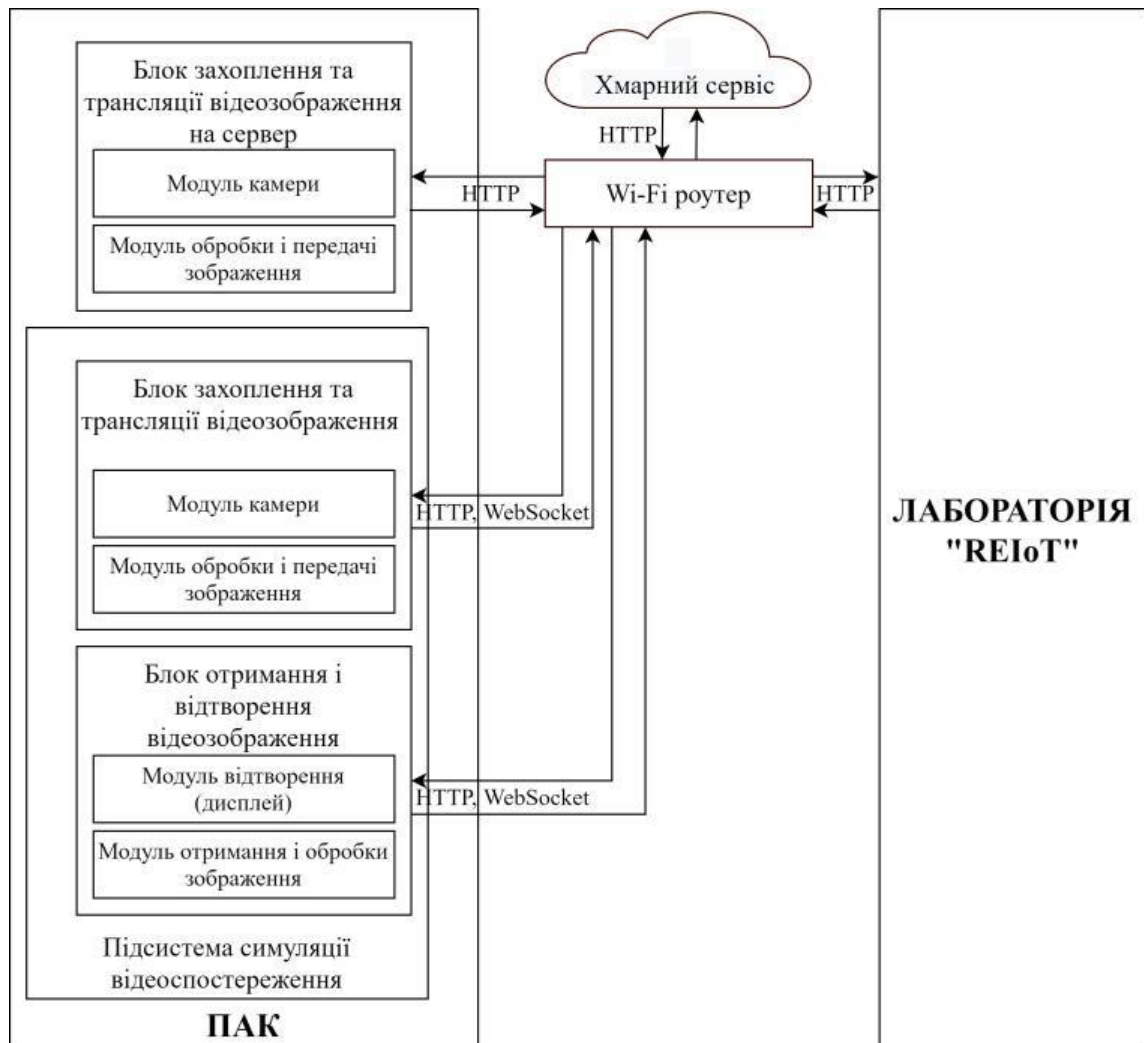


Рисунок 2.3 – Архітектура ПАК

2.3 Вибір засобів розробки

Розробку ПАК слід розпочинати з визначення апаратних засобів, які будуть застосовані, після чого – обирати програмні засоби. Такий порядок зумовлений тим, що більшість апаратних засобів, через істотні технічні обмеження, підтримують обмежену кількість протоколів та технологій і, відповідно, тим самим накладають обмеження на перелік доступних для застосування програмних засобів.

2.3.1 Апаратні засоби

Спершу розглянемо блок захоплення і трансляції відеозображення. Однією з складових блоку є модуль обробки та передачі зображення, який, для забезпечення розрахункових операцій, повинен містити вбудований мікроконтролер. Розглянувши існуючі апаратні рішення, а саме – модулі на базі мікроконтролерів ESP, AVR (табл. 2.1), було обрано ESP-32 через те, що він відрізняється великою обчислювальною потужністю, широким спектром вбудованого функціоналу, відносно низькою ціною. З метою поліпшення процесу розробки та виробництва для ПАК обрано модуль ESP-32-CAM (рис. 2.4), параметри якого аналогічні до модулю ESP-32, але він містить вбудований роз'єм для під'єднання камери, ліхтарик та ін.

Таблиця 2.1 – Порівняння модулів з мікроконтролерами [31]

	Arduino UNO R3	Arduino Mega	ESP-8266	ESP-32
Мікроконтролер	ATmega4809	ATmega2560	ESP-8266	ESP-32
Робоча напруга	5 В	5 В	3,3 В	3,3 В
Поточне споживання	50 мА - 150 мА	50 мА - 200 мА	0,5 мкА	5 мкА
Флеш-пам'ять	48 КБ	256 КБ	4 МБ	4 МБ
SRAM	6 КБ	8 КБ	—	520 КБ
EEPROM	256 Байт	4096 Байт	512 Байт	512 Байт
Тактова частота	16 МГц	16 МГц	52 МГц	80 МГц / 160 МГц
Wi-Fi	є	немає	є	є
Ціна	\$ 44,90	\$ 38,50	\$ 6	\$ 11

Окрім модулю обробки блок захоплення і трансляції відеозображення має модуль камери. ESP-32-CAM дозволяє під'єднати камери на базі OV2640. Вони існують двох типів: «стандарт» та «риб'яче око». «Риб'яче око» є широкоформатним об'єктивом з властивою даному типу проблемою – спотворення зображення. Оскільки для дослідження експериментів потрібна точність зображення, перевагу надано об'єктиву «стандарт».

Отже, найкращім рішенням для блоку захоплення і трансляції відеозображення є застосування модулю ESP-32-CAM з об'єктивом «стандарт».



Рисунок 2.4 – Модуль ESP-32-CAM з об'єктивом «стандарт»

Оскільки ESP-32-CAM не має вбудованого USB інтерфейсу, для завантаження програмного забезпечення та перегляду повідомлень послідовного порту через комп'ютер слід під'єднати UARTtoUSB адаптер (табл. 2.2). Для завантаження програмного забезпечення через адаптер на ESP-32-CAM потрібно з'єднати піни U00 і GND (рис. 2.5, позначено фіолетовим кольором).

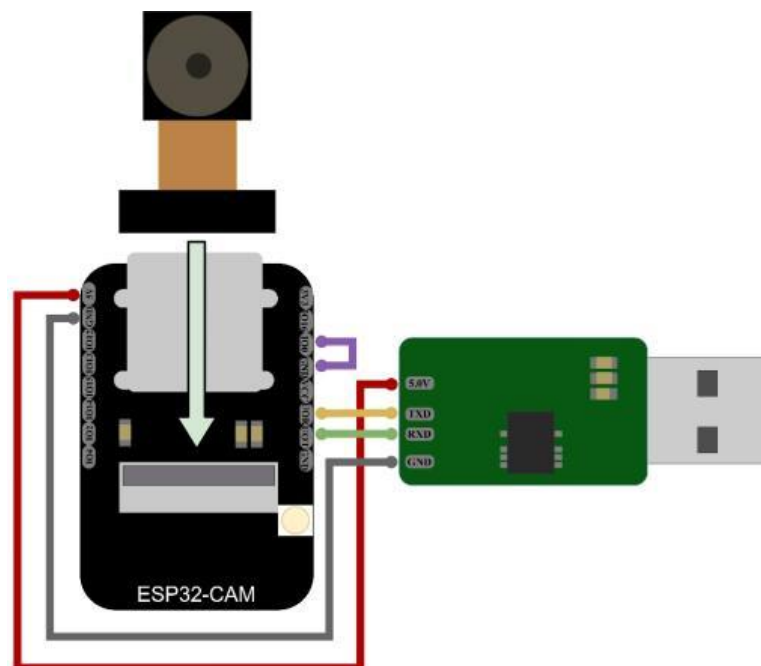


Рисунок 2.5 – Схема підключення ESP-32-CAM до UART адаптеру

Таблиця 2.2 – Зіставлення пінів для з'єднання ESP-32-CAM і UART адаптера

Піни	
ESP-32-CAM	UART адаптер
5B	5B
GND	GND
UOR	TXD
UOT	RXD

Блок отримання і відтворення відеозображення складається з модуля прийому даних і модуля відображення даних – дисплею.

Модуль прийому даних повинен виконувати обробку отриманих даних, мати підтримку бездротового Wi-Fi з'єднання та дозволяти під'єднання сторонніх модулів (дисплею). Враховуючи суміжні з блоком обробки і трансляції відеозображення вимоги до розрахункової потужності – найоптимальнішим рішенням є модуль ESP-32 (рис. 2.6).

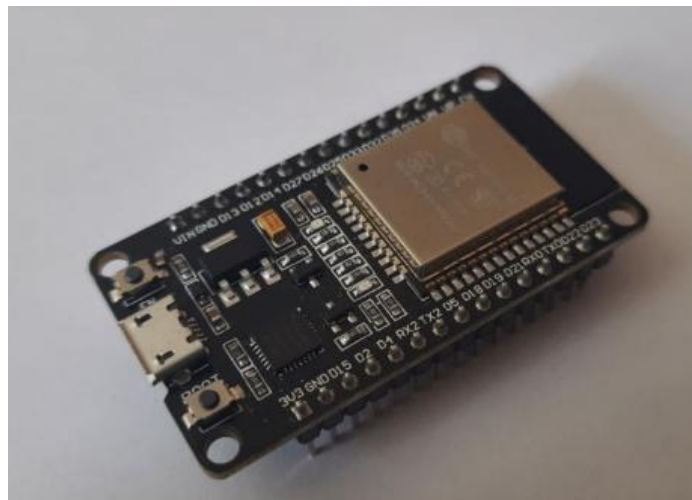


Рисунок 2.6 – Модуль ESP-32

Дисплей повинен надавати можливість динамічного відображення зображення та бути сумісним з модулем прийому даних. Серед існуючих дисплеїв, які сумісні з ESP модулями, для відтворення зображення з високою роздільною здатністю найкраще підходять дисплеї TFT типу. Хоча вони споживають більше енергії, ніж дисплеї іншого типу, але забезпечують

значно вищу якість зображення. Серед TFT дисплеїв найпопулярнішими є дві моделі: 2.4 SPI ILI9341 SKU:MSP2402 і 2.4 16BIT ILI9341 SKU:MRB2408 (табл. 2.3) [31].

Оскільки для відеомоніторингу важливим показником є швидкість передачі зображення, враховуючи, що ESP-32 має достатню кількість пінів для під'єднання обох типів TFT дисплеїв – було обрано дисплей TFT типу 2.4 16BIT ILI9341 SKU:MRB2408 (рис. 2.7).

Таблиця 2.3 – Порівняння TFT дисплеїв SPI ILI9341 SKU:MSP2402 і 16BIT ILI9341 SKU:MRB2408

	2.4 SPI ILI9341 SKU:MSP2402	2.4 16BIT ILI9341 SKU:MRB2408
Розмір дисплею	2.4'' (240x320)	2.4'' (240x320)
Кількість пінів для з'єднання	15	4
Швидкість передачі даних	висока	середня
Вартість	середня	висока

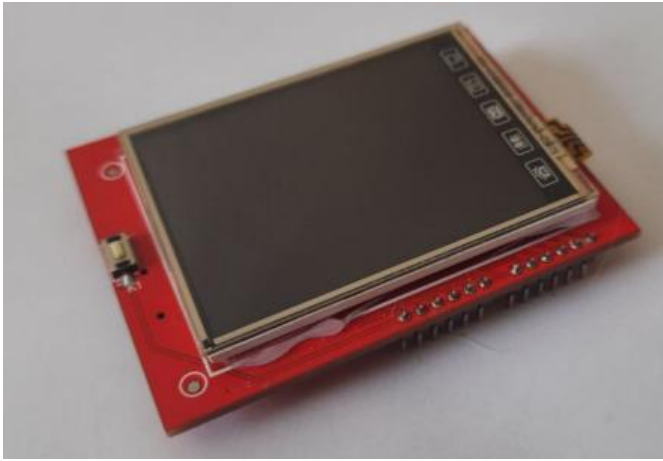


Рисунок 2.7 – TFT дисплей 2.4 16BIT ILI9341 SKU:MRB2408

Отже, найкращім рішенням для блоку отримання і обробки відеотрансляції є застосування модулю ESP-32 з TFT-дисплеєм 2.4 16BIT ILI9341 SKU:MRB2408. Схему підключення зображено на рисунку 2.8, зіставлення пінів для з'єднання зазначено у таблиці 2.4.

Таблиця 2.4 – Зіставлення пінів для з’єднання ESP-32-CAM і TFT дисплею 16BIT ILI9341MRB2408

Піни	
ESP-32	TFT дисплей 16BIT ILI9341MRB2408
G16	LCD_D0
G4	LCD_D1
G23	LCD_D2
G22	LCD_D3
G21	LCD_D4
G19	LCD_D5
G18	LCD_D6
G17	LCD_D7
G26	LCD_RST
G27	LCD_CS
G14	LCD_RS
G12	LCD_WR
G13	LCD_RD
GND	GND
3V3	3V3

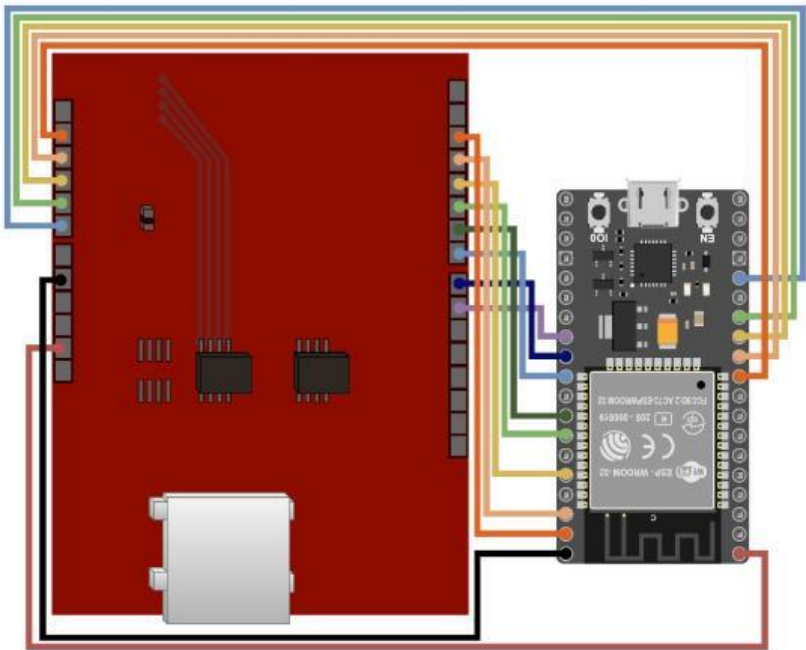


Рисунок 2.8 – Схема підключення ESP-32 до TFT дисплею 2.4 16BIT ILI9341 SKU:MRB2408

2.3.2 Програмні засоби

За результатами аналізу апаратних засобів в якості програмованих модулів було обрано мікроконтролери серії ESP-32, тож вибір програмних засобів буде орієнтовано на найкращу спорідненість з даними мікроконтролерами.

Зазвичай, при використанні конкретної моделі мікроконтролера, у розробників відсутня можливість обрання мови програмування. Але зростаюча популярність використання ESP призвела до розвитку декількох програмних платформ, які дозволяють користувачам обирати мову для розробки програмного забезпечення. Сьогодні для ESP модулів можна вибрати між мовами програмування C, C++, Lua, MicroPython (MP) та використанням AT-команд (рис. 2.9).



Рисунок 2.9 – Засоби програмування ESP-32

Керування ESP за допомогою AT команд, дозволяє лише проводити тестування модулю і його периферії, що, звісно, не є повноцінним засобом програмування. Через це даний спосіб не є доцільним при обранні засобів програмування ПАК [33].

Одним з рішень є обрання C++ – об'єктно орієнтовної мови програмування загального призначення, яка застосовується з платформою

Arduino. Висока популярність мови і розповсюдженість платформи зумовлює наявність у вільному доступі широкої кількості посібників, бібліотек і прикладів, які здатні значно полегшити і пришвидшити розробку програмного продукту. Серед бібліотек існують рішення, які підтримують управління GPIO, зв'язок Wi-Fi за допомогою TCP, протоколи передачі даних HTTP, MQTT, WebSocet та інші можливості. Дане рішення має підтримку у декількох середовищах розробки, і, на сьогодні, є найпопулярнішим. Недоліком використання є відсутність вбудованої підтримки багатозадачності і відносно складний синтаксис.

Іншим підходом до програмування ESP є використання прошивки NodeMCU, з мовою Lua. На ринку доступні плати ESP-32 із уже встановленою прошивкою NodeMCU, але зазвичай її слід завантажувати власноруч. Lua - це асинхронна мова подій. У Lua, посилаючись на стиль програмування керований подіями, завдання асоціюються із заданими подіями. Усі події ставляться в чергу, потім платформа виконує по черзі завдання, що пов'язані з подіями, які виникли. Планувальник виконує завдання, які знаходяться у черзі, за формулою FIFO. Її недоліком можна вважати необхідність дотримання обмежень за часом, оскільки завдання, які довго працюють, можуть порушити стабільність інших, наприклад, Wi-Fi з'єднання.

MP – мова інтерпретації та об'єктно-орієнтованого сценарію, що використовується на мікроконтролерах, заснована на ядрі Python. Перевага MP полягає в тому, що її спрямовано на максимальне спрощення коду для зручності початківцям і підвищенні наочності. На противагу Lua, MP є процедурною мовою. За необхідністю, MP може бути розширено мовами C або C ++, тобто користувачі можуть змішувати їх, з метою пришвидшити роботу, коли це потрібно. Простота синтаксису має свої недоліки, які виражаються у меншій швидкості роботи, необхідності більшого об'єму пам'яті порівняно з C++ або Lua.

Ще одним розповсюдженим рішенням є використання мови С в комбінації з операційною системою реального часу RTOS. Система є багатозадачною, що дозволяє виконувати декілька завдань псевдоодночасно. Для розподілу часу на певні завдання RTOS використовує планувальник. Він дозволяє створювати багатозадачні програми, заплановані системою. Кожне завдання RTOS розподіляється у визначеному просторі стека, що дозволяє передбачити і розподілити використання пам'яті. Мова набагато потужніша за С++ або МР. Проте вона має високу складність синтаксису, потребує знання роботи з покажчиками і розуміння алгоритму роботи RTOS.

Польські науковці провели порівняльне тестування швидкості обробки відтворення зображень при реалізації алгоритму різними мовами програмування (табл. 2.5). Тестування складалось з відображення напису та його очищення якомога швидше [33].

Таблиця 2.5 – Порівняння швидкості обробки й відтворення зображення при використанні різних мов програмування

Мова	Кількість кадрів за секунду
МР	7 – 8
Lua	4
С	27
С++	35

Реалізація мовою МР виявилась найповільнішою. Lua, через архітектурні особливості, не надає можливості швидко проводити виконання завдання, тому посіла передостанню сходинку. С виявилась повільнішою за С++, оскільки RTOS, під час виконання програми, постійно переключає потоки. Найкращі показники продемонструвала мова С++ з бібліотеками Arduino [33].

За результатами проведеного аналізу виявлено, що для програмної реалізації відеомоніторингу найоптимальнішим рішенням є використання мови С++ з бібліотеками Arduino.

Найпоширенішими середовищами розробки для ESP на мові C++ є Arduino IDE (рис. 2.10) і PlatformIO (рис. 2.11).

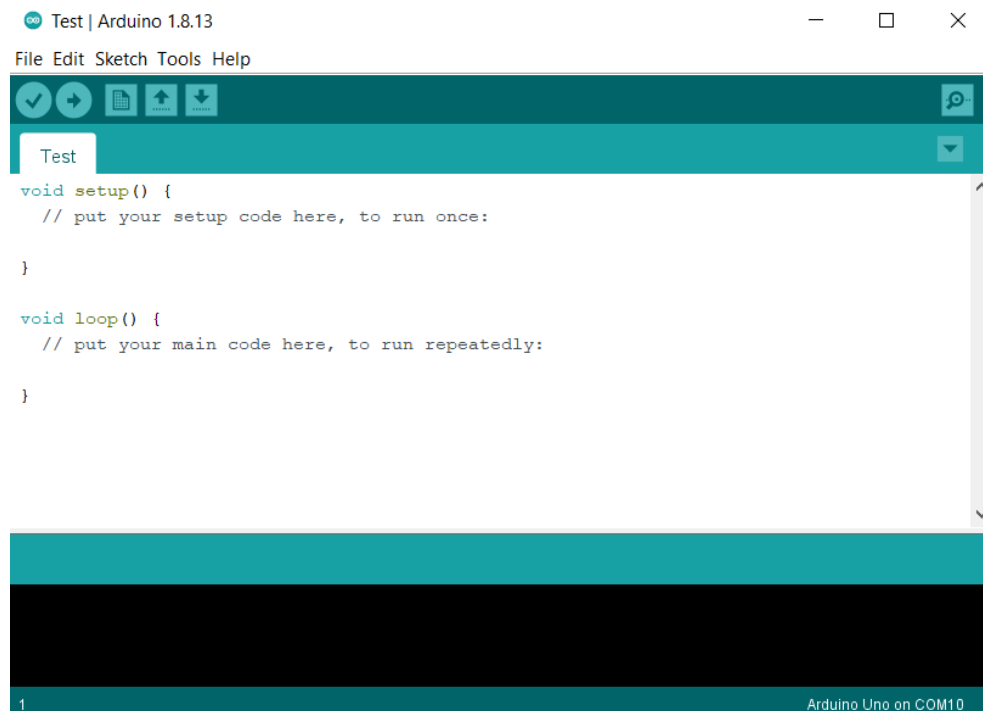


Рисунок 2.10 – Інтерфейс Arduino IDE

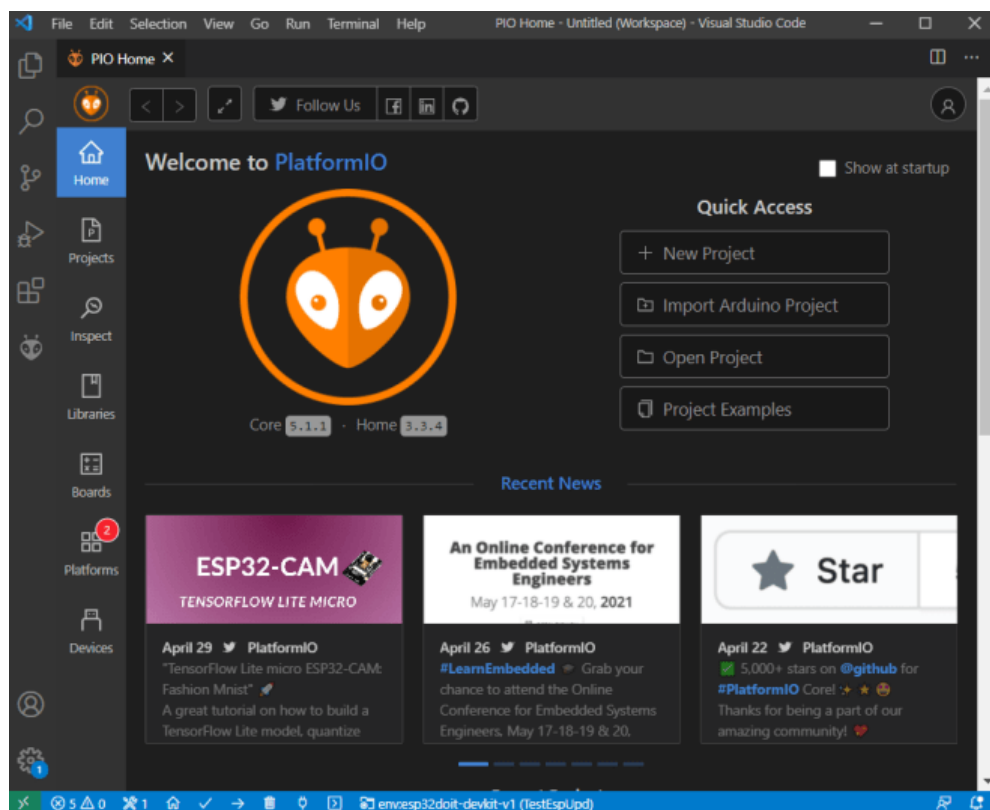


Рисунок 2.11 – Інтерфейс PlatformIO IDE у редакторі VS-Code

Arduino IDE – кросплатформене середовище розробки, призначене для розробки і завантаження програм до Arduino-сумісних плат та до плат інших виробників за умови встановлення відповідних драйверів. Середовище якісно відрізняється легкою установкою, зручністю використання, простим користувальницьким інтерфейсом. Воно має вбудований менеджер бібліотек, за допомогою якого можна встановлювати необхідні застосунки і розширення. Проте, слід зазначити, що Arduino IDE має низьку недоліків, які пов’язані з його простотою: відсутність автодоповнення коду, кодової навігації, менеджера залежностей і вбудованої системи керування версіями [34].

Отже, Arduino IDE є гарним інструментом для початківців, проте на серйозних проєктах здатне викликати складнощі в використанні.

PlatformIO IDE – кросплатформене середовище розробки з вбудованим уніфікованим налагоджувачем, яке призначене для розробки і завантаження програмного забезпечення у IoT прилади. PlatformIO може бути розгорнуте на базі декількох програм редагування коду, проте на сайті розробника IDE рекомендовано використовувати редактор VS-Code від компанії Microsoft. PlatformIO в поєднанні з VS-Code має широкі налаштування і функціонал:

- можливість налаштування інтерфейсу;
- навігацію за кодом;
- автодоповнення та інтелектуальний аналіз коду з порадами;
- вбудовану систему відслідковування версій Git.

Серед недоліків, в порівнянні до Arduino IDE, можна виділити дещо складніший процес встановлення та навантажений інтерфейс.

Отже, враховуючі широкі функціональні можливості і вірогідну об’ємність проєкту, для розробки ПАК доцільним є використання PlatformIO IDE.

Для дистанційного оновлення доцільним є застосування технологій хмари. Over-the-Air (OTA) – технологія оновлення програмного забезпечення за допомогою хмарних технологій (рис. 2.12). При впровадженні у IoT

системи OTA здатна полегшити взаємодію з ними: підвищити безпеку, знизити витрати на підтримку і обслуговування.

Підвищення безпеки зумовлене тим, що механізми OTA дозволяють розробникам дистанційно виправляти виявлені вразливості безпеки, замість того, щоб проводити відзивну компанію чи надсилати спеціалістів.

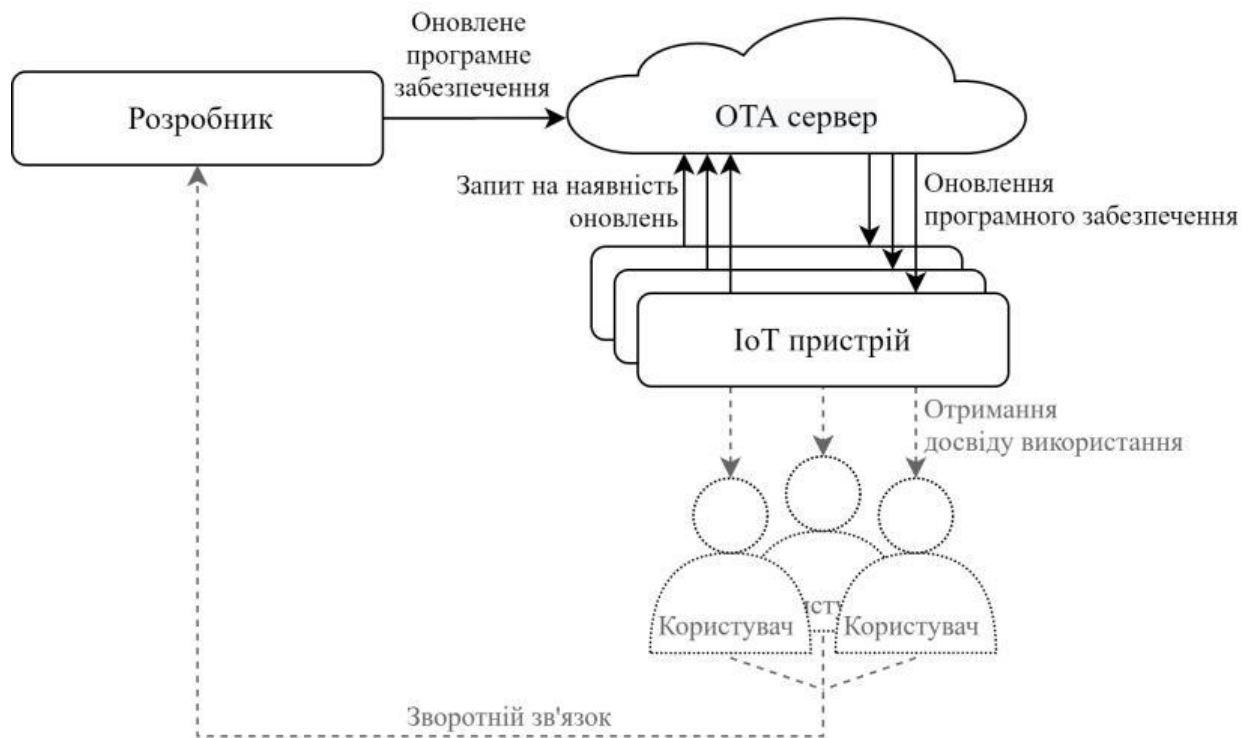


Рисунок 2.12 – Схема роботи OTA оновлень

Ручне оновлення програмного забезпечення може потребувати чималих витрат, оскільки IoT системи можуть бути розгорнуті на великій території. Застосування OTA технологій дозволяє в автоматичному режимі оновлювати прошивку на одному чи на групі пристроїв, що зменшує витрати на підтримку систем.

Одними з найрозповсюдженіших рішень онлайн OTA серверів є: OtaDrive і Google Cloud.

OtaDrive – онлайн сервіс для оновлення програмного забезпечення за OTA технологією для ESP платформ. Має відносно простий інтерфейс і не потребує складних налаштувань завдяки орієнтованості на конкретну

платформу. Дозволяє оновлювати як один конкретний, так і цілу систему пристроїв. З метою безпеки для IoT пристроїв у ньому впроваджено дворівневу авторизацію за ключом і серійним номером (рис. 2.13).

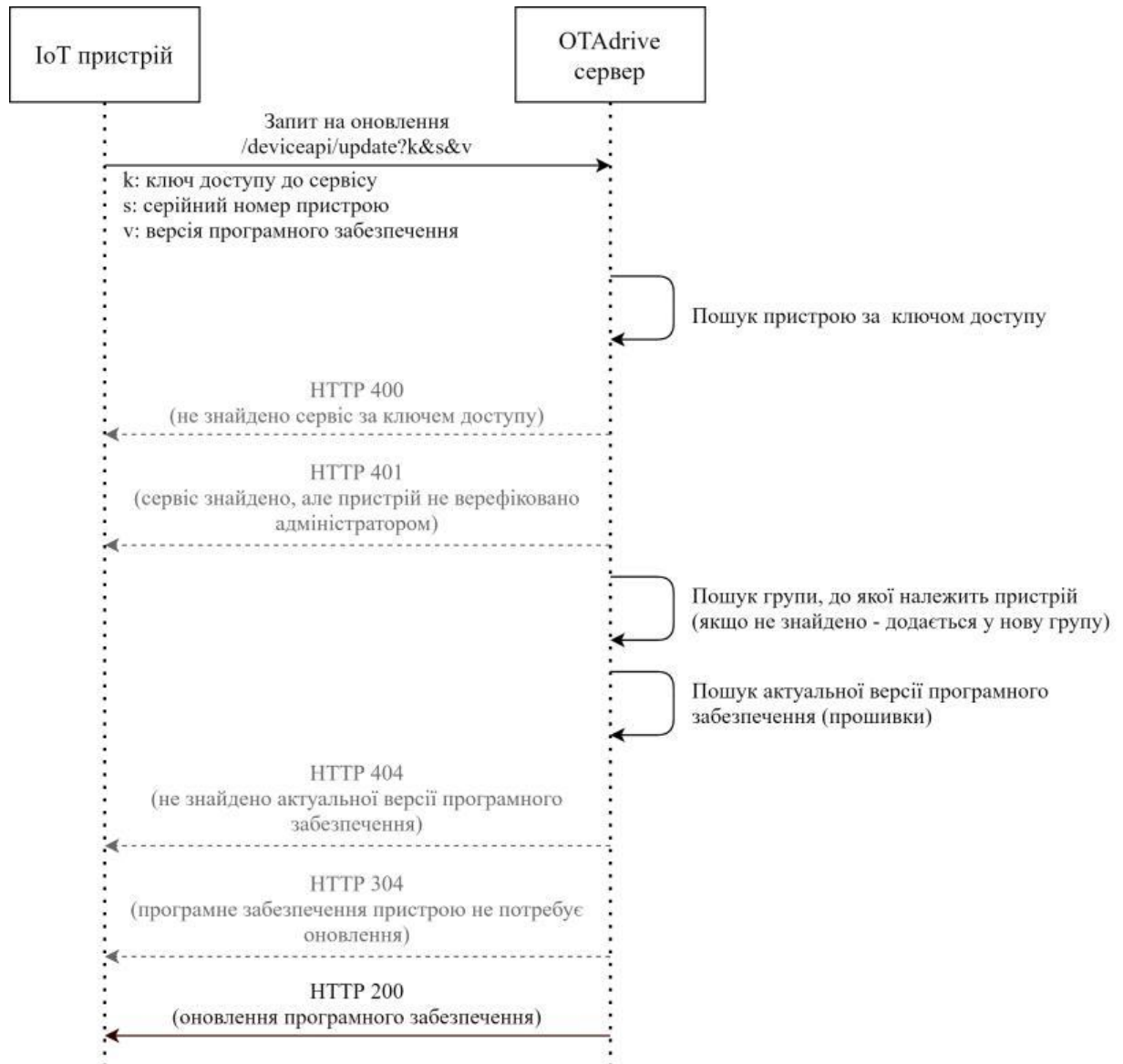


Рисунок 2.13 – Алгоритм оновлення програмного забезпечення сервісом OTAdrive

Google Cloud – хмарний сервіс, який надає широкий перелік послуг, ОТА оновлення включно. Підтримує різні платформи, проте потребує ретельного налаштування параметрів сервісу, що ускладнює його використання.

Враховуючи, що ПАК складається з блоків, які базуються виключно на ESP платформах, та той факт, що впровадження хмарного оновлення у комплекс спрямовано на поліпшення розробки і подальшої підтримки – в якості OTA сервісу доцільним є обрання OtaDrive.

2.4 Висновки до розділу 2

Відповідно до мети роботи було виконано аналіз вимог до проєкту, розроблено архітектуру, визначено апаратні та програмні засоби для розробки ПАК.

Спираючись на існуючі апаратні засоби лабораторії, взаємодію компонентів слід реалізувати через мережевий протокол Wi-Fi. Враховуючи великі об'єми інформації, в якості протоколів обміну даними обрано HTTP і WebSocet.

В якості блоків для обробки і передачі зображення обрано модулі ESP-32-CAM, для прийняття зображення і керуванням його відтворення обрано модуль ESP-32. З метою забезпечити найбільшу швидкість і якість відображення відеопотоку обрано TFT дисплей.

Мовою програмування обрано C++ та бібліотеки платформи Arduino, оскільки вони мають найкраще співвідношення швидкості оновлення зображень на дисплеї до зручності використання. Середовищем розробки визначено PlatformIO IDE. В якості сервісу дистанційного оновлення програмного забезпечення обрано OtaDrive.

з'єднання. Між спробами штучно додано затримку з метою зменшення навантаження на блоки ПАК і мережу.

Блок захоплення і трансляції зображення на сервер лабораторії після встановлення зв'язку з мережею ініціалізує роботу Wi-Fi серверу, після чого очікує на під'єднання клієнта і запит від нього. Передбачено обробку двох типів запиту: запиту на миттєве зображення з камери і запиту на відеотрансляцію. При реєстрації HTTP сервером запиту на миттєве зображення блок отримує поточне зображення з модулю камери і надсилає його клієнту. При реєстрації HTTP сервером запиту на відеотрансляцію блок починає циклічно отримувати зображення з камери і надсилати їх у відповідь MJPEG потоком клієнту.

Блок відтворення відеозображення з підсистеми симуляції відеозображення після встановлення зв'язку з мережею розгортає роботу WebSocket серверу, і очікує під'єднання клієнта. Після з'єднання з клієнтом переходить у режим отримання зображень: очікує на зображення і, після його завантаження, відображає на дисплеї.

Блок захоплення і трансляції зображення з підсистеми симуляції відеозображення після встановлення зв'язку з мережею, розгортає роботу WebSocket клієнта, і робить циклічні спроби з'єднання з WebSocket сервером. Після успішного з'єднання надсилає на сервер зображення з модулю захоплення зображень.

Кожен з блоків ПАК в програмному забезпеченні має алгоритм оновлення, який зв'язується з хмарним сховищем і, за наявності оновлення, завантажує та оновлює прошивку (рис. 3.2). При виклику алгоритму надсилається запит на хмарний сервер для перевірки наявності оновлення. Запит містить ідентифікатор блоку, версію встановленого програмного забезпечення і ключ до особового кабінету. Хмарний сервіс, після отримання запиту, проводить пошук акаунту за ключем. Якщо акаунт не знайдено – повертається відповідь «не авторизовано», алгоритм закінчує роботу. За умови, що відповідний акаунт знайдено – сервіс перевіряє за

ідентифікатором блоку чи дозволено для нього оновлення. У випадку, якщо оновлення дозволене і поточна версія програмного забезпечення на сервісі новіша за встановлену – проводиться завантаження і встановлення нового програмного забезпечення. Після встановлення оновлення передбачено автоматичне перезавантаження блоку.

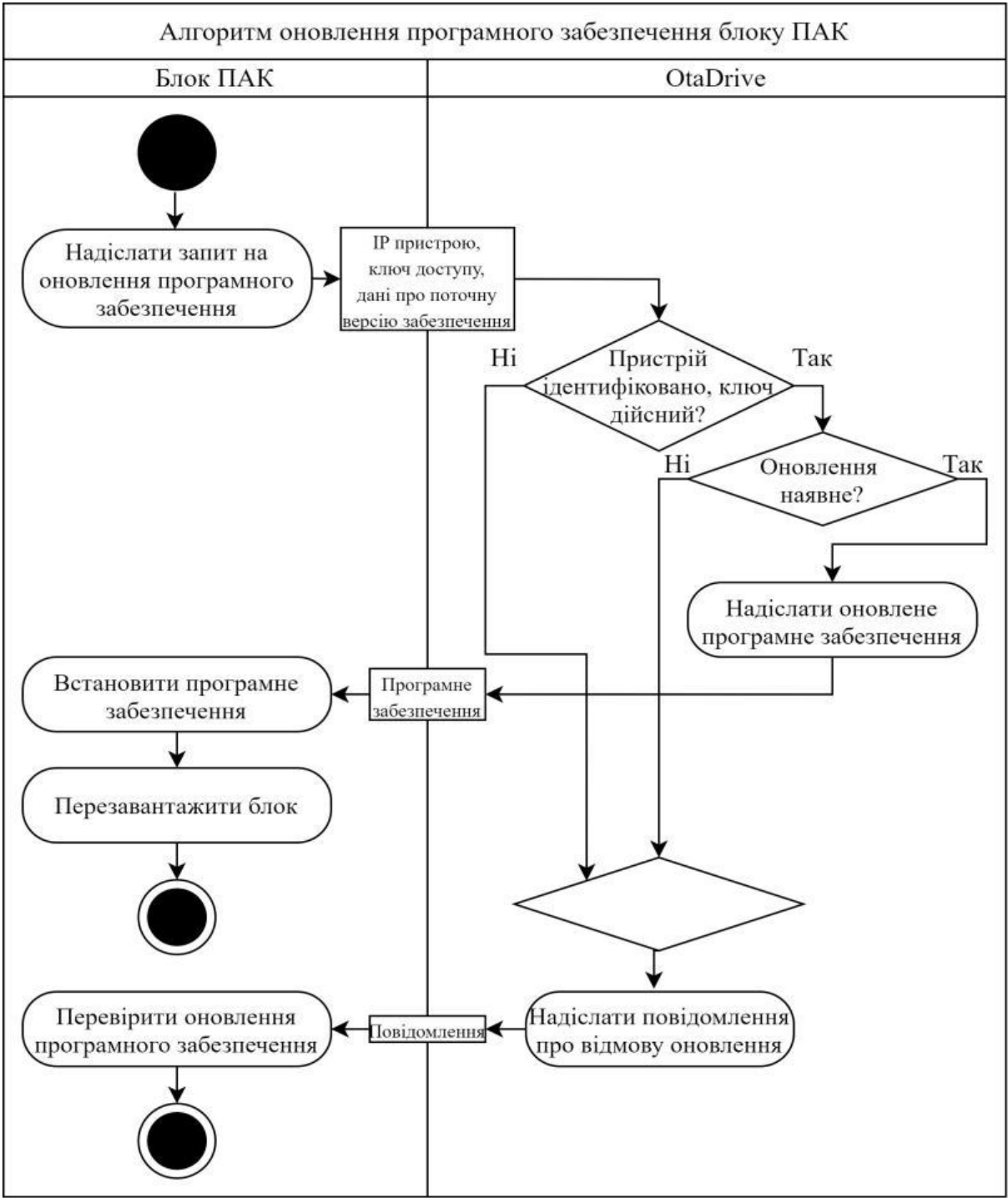
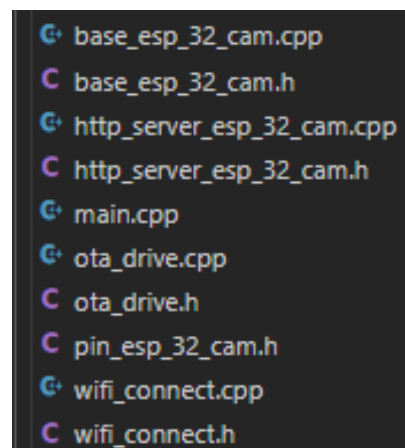


Рисунок 3.2 – Алгоритм оновлення програмного забезпечення блоку ПАК

3.2 Особливості програмної реалізації

Кожен клас, згідно рекомендаціям до обраної мови програмування та платформи, описано двома файлами: заголовочним (з розширенням .h) і вихідним (з розширенням .cpp). Заголовочні файли містять декларації змінних, прототипи функції, тоді як у вихідних написано основний код. З метою уникнення конфлікту, у кожному заголовочному файлі вбудовано конструкцію запобігання подвійного підключення файлів до проєкту.

Програмне забезпечення блоку захоплення та трансляції відеозображення на сервер складається з декількох класів, серед яких можна виділити: OtaDrive, WifiConnect, BaseEsp32Cam, HttpServerEsp32Cam (рис. 3.3).



```
base_esp_32_cam.cpp
base_esp_32_cam.h
http_server_esp_32_cam.cpp
http_server_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h
```

Рисунок 3.3 – Структура програмного забезпечення блоку захоплення та трансляції відеозображення на сервер

Клас OtaDrive надає функціонал для перевірки, завантаження і оновлення програмного забезпечення блоку. В ньому можна виділити дві функції:

- getChipId – повертає унікальний ідентифікатор блоку;
- update – перевіряє наявність оновлення у хмарному сховищі, за наявності нової версії – завантажує її і встановлює, після інсталяції блок перезавантажується.

З метою поліпшення і пришвидшення розробки було задіяно бібліотеку платформи Arduino HttpUpdate. Вона надає функціонал для завантаження двійкового файлу із веб-сервера і його встановлення на блок.

Клас WiFiConnect надає функціонал для з'єднання з Інтернет мережею за допомогою вбудованого Wi-Fi адаптеру. Функцію start можна виділити як провідну в класі: вона налаштовує зв'язок з мережею, встановлюючи статичну IP-адресу для блоку. У разі, якщо з'єднання встановити неможливо – передбачено функціонал для циклічного повтору спроби під'єднання з затримкою. При реалізації функціоналу було підключено бібліотеку Wi-Fi, яка надає інтерфейс для роботи з Wi-Fi адаптером ESP-32.

Клас BaseEsp32Cam надає функціонал ініціалізації з'єднання ESP-32-CAM з модулем камери та її налаштування. Провідними функціями класу є:

- initialize – ініціалізує зв'язок між модулем камери OV2640 і ESP-32-CAM, згідно до попередньо встановлених параметрів;
- setImageSettings – застосовує налаштування для камери, згідно до попередньо встановлених параметрів.

Для успішного налаштування зв'язку необхідно зазначити версію ESP-32-CAM, згідно до зазначеної інформації від фірми-виробника, оскільки модулі різних версій мають відмінності у призначенні пінів.

У файлі pin_esp_32_cam.h зазначено налаштування пінів для найрозповсюджених версій (табл. 3.1).

Таблиця 3.1 – Перелік параметрів для зазначення версії ESP-32-CAM

Версія ESP-32-CAM	Параметр
WROVER KIT	CAMERA_MODEL_WROVER_KIT
ESP EYE	CAMERA_MODEL_ESP_EYE
M5STACK	CAMERA_MODEL_M5STACK
M5STACK V2	CAMERA_MODEL_M5STACK_V2
M5STACK WIDE	CAMERA_MODEL_M5STACK_WIDE
TTGO JOURNAL	CAMERA_MODEL_TTGO_T_JOURNAL
AI THINKER	CAMERA_MODEL_AI_THINKER

Передбачено можливість обрати формат (табл. 3.2) і розмір (табл. 3.3) зображення.

Таблиця 3.2 – Перелік параметрів для зазначення формату зображення

Формат зображення	Параметр
YUV422	PIXFORMAT_YUV422
MONOCHROME	PIXFORMAT_GRAYSCALE
RGB565	PIXFORMAT_RGB565
JPEG	PIXFORMAT_JPEG

Таблиця 3.3 – Перелік параметрів для зазначення розміру зображення

Розмір зображення	Параметр
320 x 240	FRAME_SIZE_QVGA
352 x 288	FRAME_SIZE_CIF
640 x 480	FRAME_SIZE_VGA
800 x 600	FRAME_SIZE_SVGA
1024 x 768	FRAME_SIZE_XGA
1280 x 1024	FRAME_SIZE_SXGA
1600 x 1200	FRAME_SIZE_UXGA (1600 x 1200)

З метою покращення вихідного зображення і підвищення його якості в setImageSettings передбачено широкий перелік налаштувань камери (табл. 3.4). Це дозволить обрати найоптимальніші параметри у відповідності до місця розташування камери.

Таблиця 3.4 – Перелік параметрів для зазначення розміру зображення

Налаштування	Функція	Параметр
1	2	3
Режим балансу білого	set_wb_mode ()	0 – авто 1 – сонячно 2 – хмарно 3 – офісне освітлення 4 – домашнє освітлення
Горизонтальне віддзеркалення	set_hmirror ()	0 – вимкнути 1 – увімкнути

Продовження таблиці 3.4

1	2	3
Вертикальне віддзеркалення	set_vflip ()	0 – вимкнути 1 – увімкнути
Яскравість	set_brightness ()	Від -2 до 2 (-2 – найтемніше, 2 – найяскравіше)
Контраст	set_contrast ()	Від -2 до 2 (-2 – найтемніше, 2 – найяскравіше)
Насиченість	set_saturation ()	Від -2 до 2 (-2 – найтемніше, 2 – найяскравіше)
Художній ефект	set_special_effect ()	0 – відсутній 1 – негатив 2 – монохромний 3 – червоний відтінок 4 – зелений відтінок 5 – синій відтінок 6 – сепія

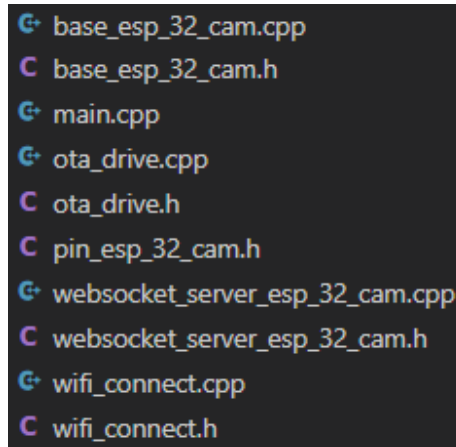
Клас `HttpServerEsp32Cam` забезпечує ініціалізацію і роботу HTTP серверу, який надає можливість отримання відеостріму або миттєвого поточного зображення з камери, в залежності від запиту. При реалізації функціоналу використано бібліотеку `esp_http_server`, яка полегшує налаштування і запуск веб-сервера на ESP-32.

Серед структури класу варто виділити функції `capture_handler`, `stream_handler`, `start`:

- `capture_handler` – повертає HTTP відповідь, яка містить поточне зображення з камери;
- `stream_handler` – повертає HTTP відповідь, яка надає потік відео у форматі MJPEG;
- `start` – розпочинає роботу серверу, реєструє запити від пристроїв-клієнтів і викликає, в залежності від запиту (`/stream` або `/screen`) відповідні функції (`capture_handler` або `stream_handler`).

Програмне забезпечення блоку захоплення та трансляції відеозображення у складі симуляції відеоспостереження, завдяки вдало розробленій розподіленій архітектурі, частково складається з класів, що були

розроблені і застосовані в програмному забезпеченні блоку трансляції відеозображення на сервер: OtaDrive, WifiConnect, BaseEsp32Cam (рис. 3.4). Серед унікальних класів потрібно виділити WebSocketServerEsp32Cam.



```

base_esp_32_cam.cpp
base_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
websocket_server_esp_32_cam.cpp
websocket_server_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h

```

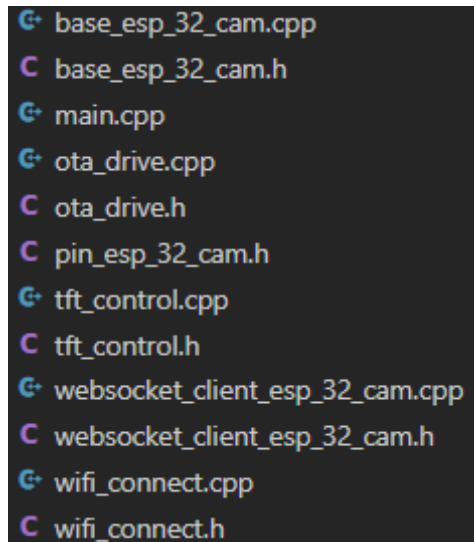
Рисунок 3.4 – Структура програмного забезпечення блоку захоплення та трансляції відеозображення у складі симуляції відеоспостереження

Клас WebSocketServerEsp32Cam забезпечує функціонал і роботу WebSoket серверу, який надає можливість отримання відеостріму чи окремого зображення. При реалізації функціоналу використано бібліотеку ArduinoWebsockets. ArduinoWebsockets – бібліотека для написання програм з застосуванням WebSockets на Arduino платформі, забезпечує відносно простий і зручний інтерфейс для роботи.

Серед структури класу варто виділити функції capture, start:

- capture – захоплює поточне зображення з камери;
- start – розпочинає роботу серверу, при під'єднанні клієнта – циклічно надсилає йому зображення.

Програмне забезпечення блоку відтворення відеозображення у складі симуляції відеоспостереження складається з декількох класів, серед яких можна виділити WebSocketClientEsp32Cam і TftControl (рис. 3.5). Частина з використаних класів є класами, які було розроблено в програмному забезпеченні блоку трансляції відеозображення на сервер: OtaDrive, WifiConnect, BaseEsp32Cam.



```

base_esp_32_cam.cpp
base_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
tft_control.cpp
tft_control.h
websocket_client_esp_32_cam.cpp
websocket_client_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h

```

Рисунок 3.5 – Структура програмного забезпечення блоку відтворення відеозображення у складі симуляції відеоспостереження

Клас `WebSocketClientEsp32Cam` забезпечує прослуховування веб-серверу, завантаження зображення (за наявності). При реалізації функціоналу використано бібліотеку `ArduinoWebsockets`. Серед структури класу варто виділити функцію `start` – вона розпочинає прослуховування серверу, при отриманні зображення викликає функцію його відмальовування.

Клас `TftControl` забезпечує функціонал відмальовування зображення на TFT дисплеї. Для реалізації його функціоналу було задіяно бібліотеки `TJpgDec` і `TFT_eSPI`.

`TJpgDec` – бібліотека візуалізації зображень JPEG, яку оптимізовано для використання у модулях малої потужності. Використовується для докомпресії зображення, отриманого від веб-серверу.

`TFT_eSPI` – бібліотека роботи з графікою та шрифтами, сумісна з Arduino платформою, орієнтована на ESP-32. Використана для забезпечення рендеренгу зображення на TFT-дисплеї. Перед її застосуванням в файлі користувацьких параметрів бібліотеки `User_Setup.h` було зазначено піни, через які здійснено під'єднання дисплею (табл. 3.5).

З метою коректного відображення зображення в налаштуваннях дисплею (табл. 3.6) було встановлено параметри за замовчанням: альбомний

режим відображення, вимкнено інвертацію кольорів, початковий колір екрану – синій.

Таблиця 3.5 – Користувацькі налаштування бібліотеки TFT_eSPI

Параметр	Значення
TFT_CS	27
TFT_DC	14
TFT_RST	26
TFT_WR	12
TFT_RD	13
TFT_D0	16
TFT_D1	4
TFT_D2	23
TFT_D3	22
TFT_D4	21
TFT_D5	19
TFT_D6	18
TFT_D7	17

Таблиця 3.6 – Перелік змінних параметрів роботи дисплею

Налаштування	Функція	Параметр
Орієнтація	setRotation	0 – книжний режим 1 – книжний перегорнутий режим 2 – альбомний режим 3 – альбомний перегорнутий режим
Інвертація кольору	invertDisplay	0 – вимкнути 1 – увімкнути
Колір екрану	fillScreen	TFT_BLACK – #000000 TFT_BLUE – # 0000FF TFT_GREEN – # 00FF00 TFT_RED – # FF0000 TFT_YELLOW – # FFFF00 TFT_WHITE – #FFFFFF

За результатами написання програмного забезпечення для блоків ПАК можна стверджувати, що розроблена архітектура має гарну структурованість

і модульність. Це дозволяє повторно використовувати частини коду (класи), що пришвидшує розробку коду, поліпшує його подальшу модернізацію.

3.3 Тестування та експериментальне дослідження програми

З метою перевірки дієздатності розробленого ПАК, розроблено методику для проведення функціонального тестування.

Функціональне тестування – це процес забезпечення якості, різновид тестування «чорна скринька», який базується на перевірці функцій шляхом співвідношення вхідних і вихідних даних. Функціональне тестування проводиться для оцінки відповідності системи попередньо зазначеним функціональним вимогам.

Враховуючи визначені до ПАК вимоги було створено тест-кейси та контрольний список для проведення функціонального тестування, після чого проведено випробування за ними.

Результати зазначено у таблицях 3.7 – 3.9.

Таблиця 3.7 – Тест-кейс перевірки дієздатності блоку трансляції відеозображення на сервер

Передумови			
Увімкнути роутер, через який налаштовано Wi-Fi з'єднання, увімкнути камеру			
№	Дія	Очікуваний результат	Статус
1	Перейти у браузері на ПК за адресою IP адреса камери/stream	Відображається відеопотік з камери в режимі онлайн	Пройдено
2	Перейти у браузері на ПК за адресою IP адреса камери/screen	Отримано зображення з камери	Пройдено
3	Перезавантажити Wi-Fi роутер через який відбувається з'єднання, зачекати 10 секунд і повторити пункти 1, 2	Аналогічно до пунктів 1 і 2 відповідно	Пройдено

Таблиця 3.8 – Тест-кейс перевірки дієздатності підсистеми симуляції відеоспостереження

Передумови			
Увімкнути роутер, через який налаштовано Wi-Fi з'єднання			
№	Дія	Очікуваний результат	Статус
1	Увімкнути блок відображення відеопотоку	Дисплей запалюється синім кольором	Пройдено
2	Увімкнути камеру	Не довше через 10 секунд на дисплеї відображається відеопотік з камери	Пройдено
3	Вимкнути Wi-Fi роутер	На дисплеї відображується останнє зображення з камери	Пройдено
4	Увімкнути Wi-Fi роутер	Аналогічно до пункту 2	Пройдено

Таблиця 3.9 – Контрольний список функціонального тестування ПАК

Вимога	Статус
Блок трансляції відеозображення на сервер	
Трансляція відео в режимі онлайн	Пройдено
Можливість миттєвого захоплення поточного зображення камери	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Блок трансляції відеозображення у складі підсистеми симуляції відеоспостереження	
Трансляція відео в режимі онлайн	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Трансляція відео в режимі онлайн	Пройдено
Незалежність від існуючих стендів	Пройдено
Блок відтворення відеозображення у складі підсистеми симуляції відеоспостереження	
Відображення відео в режимі онлайн	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Незалежність від існуючих стендів	Пройдено

За результатами проведеного тестування всі блоки розробленого ПАК характеризуються стабільною роботою, яка повністю відповідає попередньо висунутим вимогам.

3.4 Висновки до розділу 3

Отже, за результатами проведеної роботи, було розглянуто особливості і розроблено алгоритм функціонування складових частин ПАК. Враховуючи апаратні особливості компонентів, спроектовано схеми підключення блоків та їх складових частин.

Було описано структуру програмного забезпечення, основні функції і змінні. Розроблена архітектура має гарну структурованість і модульність, що дозволяє повторно використовувати частини коду для пришвидшення розробки програмного коду.

З метою перевірки дієздатності ПАК, а, також, відповідності його до висунутим вимогам, було складено тест кейси і контрольні списки тестування, за якими перевірено роботу комплексу. ПАК успішно склав усі тести, що стверджує про успішну розробку та його стабільну роботу ПАК.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови виконання програми

Програмний продукт призначено для забезпечення роботи програмно-апаратного комплексу з організації відеомоніторингу. Він складається з трьох програм:

- програма для забезпечення роботи блоку захоплення та трансляції відеозображення на сервер, для роботи на модулі ESP-32-CAM;
- програма для забезпечення роботи блоку захоплення та трансляції відеозображення (у складі підсистеми симуляції відеоспостереження), для роботи на модулі ESP32-CAM;
- програма для забезпечення роботи блоку отримання і відтворення відеозображення, для роботи на модулі ESP-32;

Програмне забезпечення розроблено на мові C++ з застосуванням бібліотек платформи Arduino.

Для експлуатації програмно-апаратного комплексу необхідні такі програмно-технічні засоби:

- джерело живлення 5В або 3.3В;
- Wi-Fi роутер;
- персональний комп'ютер з встановленим браузером Google Chrome (версії 88.0.4324.182 або вище), Firefox (версії 88.0.1 або вище), Egle(версії 46.3.4.5155 або вище);
- оперативна пам'ять 1 ГБ або більше;
- клавіатура;
- маніпулятор типу миша.

4.2 Характеристики програми

Розроблене програмне забезпечення дозволяє ПАК виконувати такі функції:

- надання відеотрансляції в режимі онлайн;
- надання скріншоту поточного стану;
- робота підсистеми симуляції відеоспостереження (автономна робота блоків захоплення і відтворення відеотрансляції);
- встановлення налаштувань за замовчуванням для кожного з блоків ПАК;
- дистанційне оновлення програмного забезпечення кожного блоку ПАК за допомогою хмарного сервісу.

4.3 Звернення до програми

В програмному забезпеченні блоків захоплення та трансляції відеозображення можна встановити налаштування камери (файл BaseEsp32Cam, функція setImageSettings), а саме:

- режим балансу білого – авто, сонячно, хмарно, офісне освітлення, домашнє освітлення (рис 4.1);
- художній ефект – відсутній, негатив, монохромний, червоний відтінок, зелений відтінок, синій відтінок, сепія (рис. 4.2).
- яскравість – 5 режимів (рис. 4.3);
- контраст – 5 режимів (рис. 4.4);
- насиченість – 5 режимів (рис. 4.5);



Рисунок 4.1 – Робота камери при різних режимах білого кольору



Рисунок 4.2 – Робота камери при різних художніх режимах



Рисунок 4.3 – Робота камери при різних режимах яскравості



Рисунок 4.4 – Робота камери при різних режимах контрастності



Рисунок 4.5 – Робота камери при різних режимах насиченості

Окрім цього також присутня можливість встановити горизонтальне або вертикальне віддзеркалення (рис. 4.6).



Рисунок 4.6 – Робота камери при різних режимах віддзеркаленнях

Для кожного блоку передбачено можливість дистанційного оновлення програмного забезпечення. Для оновлення слід пройти авторизацію у хмарному сервісі <https://otadrive.com>, відкрити потрібну групу блоків (рис. 4.7) і завантажити для неї програмне забезпечення, яке попередньо скомпільоване у bin файл.

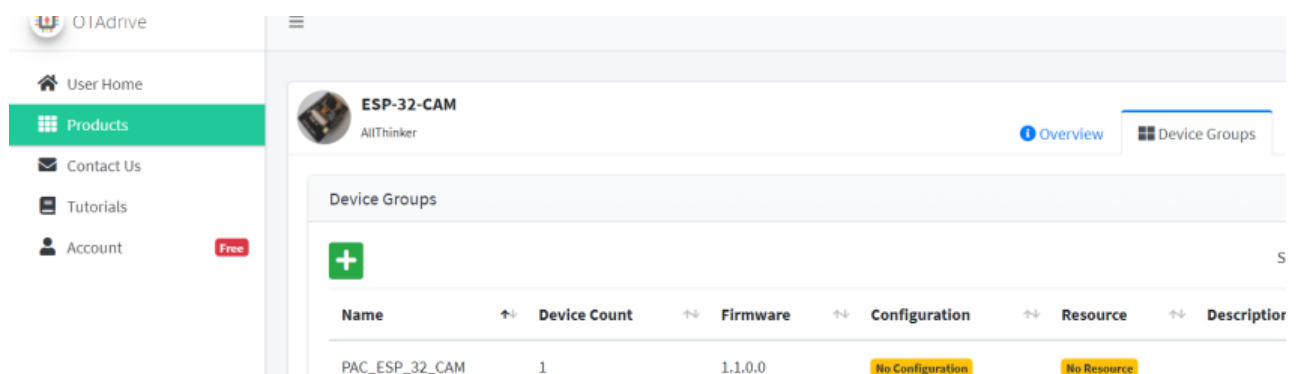


Рисунок 4.7 – Обрання групи блоків у сервісі OtaDrive

Слід зазначити, що при завантаженні на сервіс програмного забезпечення необхідно вказати його версію та можна додати додатковий опис.

Після завантаження програмного забезпечення і надання статусу «актуальне» (рис. 4.8) воно буде автоматично встановлене на обраний блок ПАК при під'єднанні блоку до хмари.

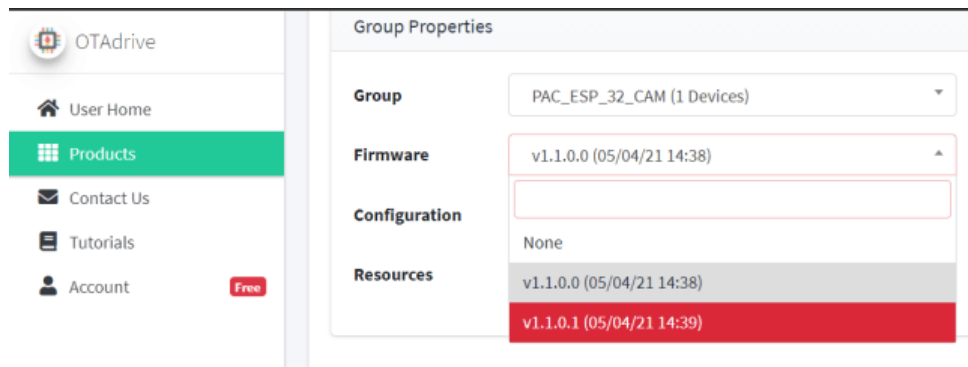


Рисунок 4.8 – Налаштування актуальної версії програмного забезпечення для групи блоків в хмарному сервісі OtaDrive

4.4 Вхідні та вихідні дані

Вхідними даними програми блоку захоплення та трансляції відеозображення на сервер є налаштування мережі і налаштування камери. Вихідними даними є відеотрансляція з камери в режимі онлайн і миттєве зображення з камери, які надаються в залежності від отриманого блоком запиту.

Вхідними даними програми блоку захоплення та трансляції відеозображення у складі підсистеми симуляції відеоспостереження є налаштування мережі і налаштування камери. Вихідними даними є відеотрансляція з камери в режимі онлайн.

Вхідними даними програми блоку отримання і відтворення відеозображення у складі підсистеми симуляції відеоспостереження є налаштування мережі, налаштування дисплею, відеотрансляція. Вихідні дані представлені відображенням відеотрансляції на дисплеї.

4.5 Повідомлення і обробка виняткових ситуацій

Розроблений ПАК забезпечує стабільне функціонування. Виняткові ситуації можуть виникнути при неправильному налаштуванні блоків. При

виникненні помилок передбачено відправлення відповідних повідомлень у послідовний порт.

У таблиці 4.1 зазначено повідомлення програмісту, які може надавати ПАК.

Таблиця 4.1 – Повідомлення програмісту від блоків ПАК

Повідомлення	Опис	Дії програміста
Блок захоплення і трансляції відеозображення на сервер		
Error. Camera initialize failed.	Модуль камери не ініціалізовано	Перевірити правильність параметрів налаштування камери
Error. Wi-Fi: failed to configure	Вбудований Wi-Fi адаптер не вдалось проініціалізувати вказаними параметрами	Перевірити правильність вказання IP блоку, шлюзу та інших параметрів налаштувань Wi-Fi з'єднання
Error. Camera capture failed.	Не вдалось обробити запит на захоплення зображення	Перевірити параметри обробника запитів, дієздатність модулю камери
Error. Video broadcast failed. Video broadcast	Не вдалось обробити запит на відеотрансляцію	Перевірити параметри обробника запитів, дієздатність модулю камери
Підсистема симуляції відеоспостереження		
Блок захоплення і трансляції відеозображення		
Error. Camera initialize failed.	Модуль камери не ініціалізовано	Перевірити правильність параметрів налаштування камери
Error. Wi-Fi: failed to configure	Вбудований Wi-Fi адаптер не вдалось проініціалізувати вказаними параметрами	Перевірити правильність вказання IP блоку, шлюзу та інших параметрів налаштувань Wi-Fi з'єднання
Error. Video broadcast failed. Video broadcast	Не вдалось обробити запит на відеотрансляцію	Перевірити параметри обробника запитів, дієздатність модулю камери
Блок отримання і відтворення відеозображення		
Error. Display initialize failed.	Модуль дисплею не ініціалізовано	Перевірити правильність налаштувань дисплею
Error. Wi-Fi: failed to configure	Вбудований Wi-Fi адаптер не вдалось проініціалізувати вказаними параметрами	Перевірити правильність вказання IP блоку, шлюзу та інших параметрів Wi-Fi

5 КЕРІВНИЦТВО КОРИСТУВАЧА

5.1 Призначення програми

Програмний продукт призначено для організації відеомоніторингу, а саме для:

- забезпечення можливості отримання відеотрансляції в режимі онлайн;
- забезпечення можливості отримання миттєвого зображення;
- забезпечення можливості розгортання підсистеми симуляції відеоспостереження – комплексу з захоплення і відтворення відеозображення.

5.2 Умови виконання програми

Для експлуатації програмно-апаратного комплексу необхідні такі програмно-технічні засоби:

- джерело живлення 5В або 3.3В;
- Wi-Fi роутер;
- персональний комп'ютер з встановленим браузером Google Chrome (версії 88.0.4324.182 або вище), або Firefox (версії 88.0.1 або вище), або Egle (версії 46.3.4.5155 або вище);
- оперативна пам'ять 1 ГБ або більше;
- клавіатура;
- маніпулятор типу миша.

Живлення 5В здатне забезпечити більш стабільну роботу блоків та підвищити допустиму максимальну дальність від Wi-Fi роутера.

5.3 Повідомлення оператору

Розроблений ПАК забезпечує відправлення повідомлень через послідовний порт про поточний стан роботи блоків.

У таблиці 5.1 зазначено повідомлення оператору, які можуть надавати блоки ПАК.

Таблиця 5.1 – Повідомлення оператору від блоків ПАК

Повідомлення	Опис
Блок захоплення і трансляції відеозображення на сервер	
Wi-Fi. Successfully configured	Вбудований Wi-Fi адаптер успішно сконфігуровано
Wi-Fi. Successfully connected	Блок успішно з'єднано з Wi-Fi мережею
Update checked	Перевірено оновлення програмного забезпечення
Camera initialized	Модуль камери успішно проініціалізовано
Started on url: http://<IP адреса>	Блок успішно розпочав роботу за зазначеною адресою
Video sended	Відеотрансляція успішно розпочата у відповідь на відповідний запит
Image sended	Зроблено миттєве зображення і відправлено у відповідь на відповідний запит
Підсистема симуляції відеоспостереження	
Блок захоплення і трансляції відеозображення	
Wi-Fi. Successfully configured	Вбудований Wi-Fi адаптер успішно сконфігуровано
Wi-Fi. Successfully connected	Блок успішно з'єднано з Wi-Fi мережею
Update checked	Перевірено оновлення програмного забезпечення
Camera initialized	Модуль камери успішно проініціалізовано
Started on url: http://<IP адреса>	Блок успішно розпочав роботу за зазначеною адресою
Connected to WebSocket server	Під'єднано до серверу
Video sended	Передачу відеотрансляції на блок відображення успішно розпочато
Блок отримання і відтворення відеозображення	
Wi-Fi. Successfully configured	Вбудований Wi-Fi адаптер успішно сконфігуровано
Wi-Fi. Successfully connected	Блок успішно з'єднано з Wi-Fi мережею
Update checked	Перевірено оновлення програмного забезпечення
Display initialized	Модуль дисплею успішно проініціалізовано
Started on url: http://<IP адреса>	Блок успішно розпочав роботу за зазначеною адресою
The client has connected	WebSocket клієнт під'єднався до серверу

5.4 Виконання програми

Для виконання програми необхідно ввімкнути роутер і під'єднати живлення до кожного блоку програмно-апаратного комплексу.

Підсистема симуляції відеоспостереження розпочинає роботу в автоматичному режимі (рис. 5.1). Спочатку дисплей блоку отримання і обробки зображення розпочне світитись синім кольором, що свідчить про готовність блоку до прийому даних. Після нетривалого очікування на дисплеї буде транслюватись відеопотік з блоку захоплення і трансляції відеозображення.

Блок трансляції відеозображення на сервер після під'єднання живлення автоматично розпочинає роботу. Для перевірки дієздатності блоку необхідно відкрити браузер і перейти за посиланням <http://<IP адреса блоку>/stream> для отримання відеопотоку в режимі онлайн, або за посилання <http://<IP адреса блоку>/capture> (рис. 5.2) для отримання миттєвого зображення з камери.

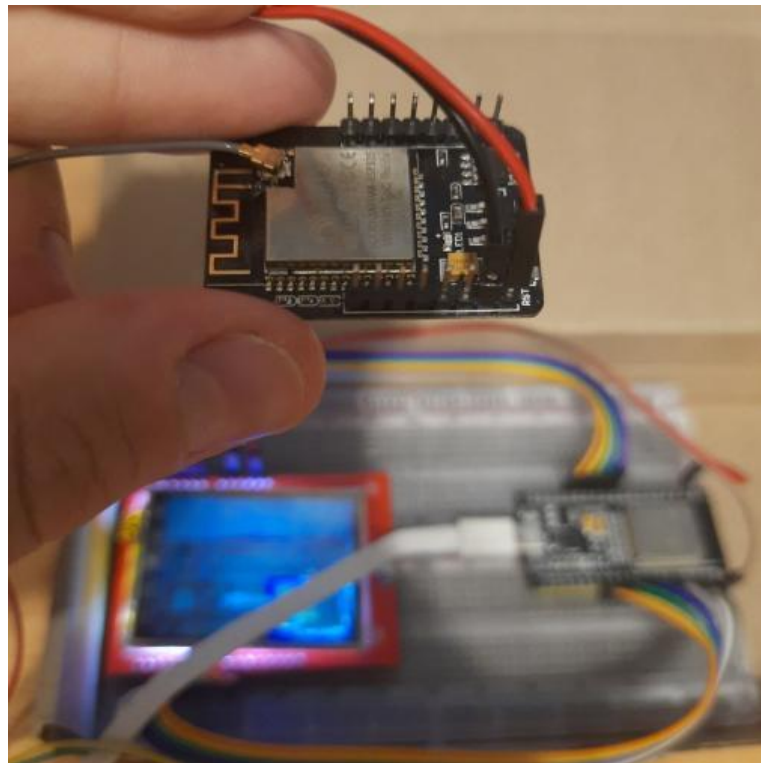


Рисунок 5.1 – Приклад роботи підсистеми симуляції відеоспостереження



Рисунок 5.2 – Відображення у браузері зображення, захопленого блоком захоплення та трансляції відеозображення на сервер

ВИСНОВКИ

За результатами виконання дипломної роботи вирішено актуальну проблему забезпечення можливості онлайн трансляції роботи стенду системи домашньої автоматизації MyHome від компанії Legrand a, також, проблему забезпечення апаратно-програмної частини для можливості ознайомлення і дослідження на практиці алгоритму роботи систем відеотрансляції.

Для вирішення даних проблем було розроблено ПАК, який має наступний функціонал:

- отримання відеотрансляції в режимі онлайн;
- отримання миттєвого зображення;
- розгортання підсистеми симуляції відеоспостереження – комплексу з захоплення й відтворення відеозображення.

Практична цінність розробленого ПАК полягає у тому, що він надасть можливість забезпечити відеотрансляцію стенду MyHome. Це сприятиме розширенню функціоналу віддаленої лабораторії REIoT, яка знаходиться в аудиторії 53 а. Окрім цього, ПАК дозволить студентам на практиці ознайомитись з особливостями роботи системи забезпечення відеоспостереження.

В подальшому буде підвищено зручність використання комплексу шляхом розробки веб-інтерфейсу для керування налаштуваннями через веб-браузер. Окрім цього буде розширено програмну та апаратну частину підсистеми симуляції відеоспостереження для забезпечення можливості відеотрансляції зі звуком. Також планується впровадження системи авторизації для пристроїв-клієнтів.

Основні положення й результати роботи були представлені в тезах доповіді НПК Тиждень науки – 2021.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Whitmore, A. The Internet of Things. A survey of topics and trends [Electronic resource] / Whitmore, A., Agarwal. – Access mode <https://doi.org/10.1007/s10796-014-9489-2>.
2. Official website of the company Oracle: What Is IoT? [Electronic resource]. – Access mode: <https://www.oracle.com/internet-of-things/what-is-iot>.
3. Official website of the company Abb: Internet of Things (IoT), ecosystem study [Electronic resource]. – Access mode: https://iot.ieee.org/images/files/pdf/iot_ecosystem_exec_summary.pdf.
4. Manyika, J. The Internet of Things: Mapping the Value Beyond the Hype [Electronic resource] / Manyika, J., Chui, M., Bisson, P. – Access mode: https://www.mckinsey.com/~media/McKinsey/Industries/Technology%20Media%20and%20Telecommunications/High%20Tech/Our%20Insights/The%20Internet%20of%20Things%20The%20value%20of%20digitizing%20the%20physical%20world/Unlocking_the_potential_of_the_Internet_of_Things_Executive_summary.pdf.
5. Geng, H. Internet of Things and Data Analytics / Geng H. – Wiley; 1st edition, 2016, 763 p.
6. Chen, J. China's Internet of Things John Chen [Electronic resource] / Chen J., Walz E., Lafferty B. – Access mode: <https://www.uscc.gov/files/000783>.
7. Official website of the company Gartner: 5.8 Billion Enterprise and Automotive IoT Endpoints Will Be in Use in 2020 [Electronic resource]. – Access mode: <https://www.gartner.com/en/newsroom/press-releases/2019-08-29-gartner-says-5-8-billion-enterprise-and-automotive-iot#:~:text=Gartner%2C%20Inc.,a%2021%25%20increase%20from%202019.&text=Utilities%20will%20be%20the%20highest,to%20reach%201.37%20billion%20endpoints>.
8. Офіційний сайт LifeCell: Інтернет речей [Електрон. ресурс] . – Режим доступу: https://iot.lifecell.ua/uk?gclid=CjwKCAjwmv-DBhAMEiwA7xYrd24l4Jtx1CXh-_Hfmf3dINkdOM7ufs2IvYk9h3HiLvSkll-dvIqf6xoCo4AQAvD_BwE.

9. Continuous Video Monitoring [Electronic resource]. – Access mode: https://journals.lww.com/jncqjournal/Fulltext/2021/04000/Continuous_Video_Monitoring__Implementation.9.aspx.

10. Official website of the company Gartner: Government IoT Revenue for Endpoint Electronics and Communications to Total \$15 Billion in 2020 [Electronic resource]. – Access mode: <https://www.gartner.com/en/newsroom/press-releases/2020-10-05-gartner-says-government-iot-revenue-for-endpoint-electronics-and-communications-to-total-15-billion-in-2020>.

11. Official website of the company IDC: IoT Growth Demands Rethink of Long-Term Storage Strategies, says IDC [Electronic resource]. – Access mode: <https://www.idc.com/getdoc.jsp?containerId=prAP46737220>.

12. Офіційний сайт компанії IoT Ukraine: Smart agriculture (розумне сільське господарство) [Електрон. ресурс]. – Режим доступу: <https://iotukraine.com/projects-ua/smart-agriculture-rozumne-silске-gospodarstvo>.

13. Офіційний сайт компанії IoT Ukraine: Smart environment (моніторинг якості повітря) [Електрон. ресурс]. – Режим доступу: <https://iotukraine.com/projects-ua/smart-environment-monitoring-jakosti-povitrja>.

14. McKenna, N. Explaining the Relationship Between IoT, Big Data and Cloud Computing [Electronic resource] / McKenna N. – Access mode: <https://www.mckennaconsultants.com/relationship-between-iot-big-data-and-cloud-computing/#:~:text=Cloud%20Computing%20is%20an%20easy,time%20control%20and%20data%20monitoring>.

15. IaaS vs. PaaS vs. SaaS: What's the Difference [Electronic resource]. – Access mode: <https://www.sam-solutions.com/blog/iaas-vs-paas-vs-saas-whats-the-difference>.

16. Fog Computing vs. Cloud Computing for IoT Difference [Electronic resource]. – Access mode: <https://www.sam-solutions.com/blog/fog-computing-vs-cloud-computing-for-iot-projects/#:~:text=Their%20purpose%20is%20to%20support,between%20hardware%20and%20remote%20servers>.

17. Official website of the company Microsoft: IoT technologies and protocols [Electronic resource]. – Access mode: <https://azure.microsoft.com/en-us/overview/internet-of-things-iot/iot-technology-protocols>.

18. Official website of the company IBM: Choosing the best hardware for your next IoT project [Electronic resource]. – Access mode: <https://developer.ibm.com/technologies/iot/articles/iot-lp101-best-hardware-devices-iot-project>.

19. Official website of the company Nabto: A Complete Guide to IoT Protocols & Standards In 2021 [Electronic resource]. – Access mode: <https://www.nabto.com/guide-iot-protocols-standards>.

20. Official website of the company IBM: Connecting all the things in the Internet of Things [Electronic resource]. – Access mode: <https://developer.ibm.com/technologies/iot/articles/iot-lp101-connectivity-network-protocols>.

21. Official website of the company Eclipse: IoT Developer survey result 2018 [Electronic resource]. – Access mode: <https://iot.eclipse.org/community/resources/iot-surveys/assets/iot-developer-survey-2018.pdf>.

22. Офіційний сайт компанії Vodafone: NB IoT (Narrowband Internet of Things) [Електрон. ресурс]. – Режим доступу: https://business.vodafone.ua/iot/nb-iot?utm_source=Search&utm_medium=CPC&utm_campaign=Vodafone_NB-IoT&utm_term=nb%20iot&gclid=Cj0KCQjwyZmEBhCpARIsALizmnJYDg3iLCAszVt6O_MiKA2UnfNLihEJkdHgcSttWQ_1Bx8udaYve0oaAq_AEALw_wcB.

23. Official website of the company GSA: NB-IoT and LTE-MTC Global Ecosystem and Market Status [Electronic resource]. – Access mode: <https://gsacom.com/paper/nb-iot-and-lte-mtc-global-ecosystem-and-market-status>.

24. Чим відрізняються аналогові і IP відеокамери спостереження? [Електрон. ресурс]. – Режим доступу:

<https://worldvision.com.ua/ua/articles/chem-otlichayutsya-analogovie-i-ip-videokameri-nablyudeniya>.

25. Чим відрізняються IP та аналогові камери відеоспостереження? Плюси та мінуси 2 технологій [Електрон. ресурс]. – Режим доступу: <https://e-server.com.ua/sovety/chem-otlichajutsja-ip-i-analogovye-kamery-videonabljudenija-ocenivaem-pljusy-i-minusy-2-tehnologij>.

26. Camera Installation by ADT [Electronic resource]. – Access mode: <https://smarthomestennessee.com/services/cctv>.

27. Світличний, В.А. Сучасні засоби і системи відеоспостереження [Електрон. ресурс] / Світличний В.А. – Режим доступу: <http://webcache.googleusercontent.com/search?q=cache:j9dU2gbo4M8J:lib.univd.edu.ua/%3Fcontroller%3Dservice%26action%3DdownloadRep%26id%3D97476+%&cd=13&hl=en&ct=clnk&gl=ua>.

28. Цифровые IP камеры видеонаблюдения или аналоговые видеокамеры? [Электрон. ресурс]. – Режим доступа: <https://smartel.ua/articles/ip-kamery-videonablyudeniya-analogovye-videokamery-bezopasnosti>.

29. Пархоменко, А. В., Програмно-апаратна платформа для навчання технологіям Інтернету речей: навчальний посібник / Пархоменко А. В., Туленков А.В., Соколянський О.В., Залюбовський Я.І. – Дике Поле, 2017. – 120 с.

30. Види об'єктів [Електрон. ресурс]. – Режим доступу: <https://kodak-dukak.com.ua/%D0%B2%D0%B8%D0%B4%D0%B8-%D0%BE%D0%B1%D1%94%D0%BA%D1%82%D0%B8%D0%B2%D1%96%D0%B2>.

31. Arduino vs ESP8266 vs ESP32 Comparison [Electronic resource]. – Access mode: <https://diyi0t.com/technical-datasheet-microcontroller-comparison>.

32. Choosing and Adding a Display for Arduino Projects [Electronic resource]. – Access mode: <https://blog.upverter.com/2019/12/20/choosing-and-adding-a-display-for-arduino-projects>.

33. Rak, F. Comparison of ESP programming platforms / Rak F., Wiora J. – Computer Science and Information Technologies, Poland Vol. 2, No. 2, 2021, pp. 77~86.

34. Buckley, I. Arduino Coding With VS Code and PlatformIO [Electronic resource] / Buckley I. – Access mode: <https://www.makeuseof.com/tag/better-arduino-coding-vs-code-platformio>.

35. Brown, R. Platformio for VS Code VS. Arduino IDE [Electronic resource] / Brown R. – Access mode: <https://nerdytechy.com/platformio-vscode-vs-arduino-ide>.

ДОДАТОК А
ТЕКСТ ПРОГРАМИ

A.1 Файл `wifi_connect.h`

```
#ifndef WIFI_CONNECT_H
#define WIFI_CONNECT_H

#include <Arduino.h>
#include <WiFi.h>

class WiFiConnect
{
public:
    void start(IPAddress LocalIP, IPAddress GatewayIP, IPAddress Subnet,
const char *ssid, const char *password);
};
#endif
```

A.2 Файл `wifi_connect.cpp`

```
#include "wifi_connect.h"

void WiFiConnect::start(IPAddress LocalIP, IPAddress GatewayIP, IPAddress
Subnet, const char *ssid, const char *password)
{
    if (!WiFi.config(LocalIP, GatewayIP, Subnet))
    {
        Serial.println("Wi-Fi: failed to configure");
    }

    WiFi.begin(ssid, password);

    Serial.print(".");

    while (WiFi.status() != WL_CONNECTED)
    {
        delay(500);
        Serial.print(".");
    }

    Serial.print("Wi-Fi. Successfully configured");
}
```

A.3 Файл `ota_drive.h`

```
#ifndef OTA_DRIVE_H
#define OTA_DRIVE_H

#include <Arduino.h>
#include <HTTPUpdate.h>

class OtaDrive {
```

```

private:
    String _key;
    String _firmwareVersion;
    String getChipId();
public:
    OtaDrive(String key, String firmwareVersion);
    void update();
};
#endif

```

A.4 Файл ota_drive.cpp

```

#include "ota_drive.h"

OtaDrive::OtaDrive(String key, String firmwareVersion)
{
    this->_key = key;
    _firmwareVersion = firmwareVersion;
}

String OtaDrive::getChipId()
{
    String chipIdHex = String((uint32_t)(ESP.getEfuseMac() >> 32), HEX);
    chipIdHex += String((uint32_t)ESP.getEfuseMac(), HEX);
    return chipIdHex;
}

void OtaDrive::update()
{
    String url = "http://otadrive.com/deviceapi/update?";
    url += "k=" + _key;
    url += "&v=" + _firmwareVersion;
    url += "&s=" + getChipId();
    WiFiClient client;
    httpUpdate.update(client, url, _firmwareVersion);
    Serial.print("Update checked");
}

```

A.5 Файл base_esp_32_cam.h

```

#ifndef BASE_ESP_32_CAM_H
#define BASE_ESP_32_CAM_H
#define CAMERA_MODEL_AI_THINKER
#include <Arduino.h>
#include "esp_camera.h"
#include "pin_esp_32_cam.h"

class BaseEsp32Cam {
public:
    BaseEsp32Cam();
    void initialize();
}

```

```

        void setImageSettings();
    };
#endif

```

A.6 Файл base_esp_32_cam.cpp

```

#include "base_esp_32_cam.h"

void BaseEsp32Cam::setImageSettings()
{
    sensor_t *s = esp_camera_sensor_get();
    s->set_brightness(s, 1);
    s->set_contrast(s, 0);
    s->set_saturation(s, 0);
    s->set_special_effect(s, 0);
    s->set_whitebal(s, 1);
    s->set_awb_gain(s, 1);
    s->set_wb_mode(s, 0);
    s->set_exposure_ctrl(s, 1);
    s->set_aec2(s, 0);
    s->set_ae_level(s, 0);
    s->set_aec_value(s, 300);
    s->set_gain_ctrl(s, 1);
    s->set_agc_gain(s, 0);
    s->set_gainceiling(s, (gainceiling_t)0);
    s->set_bpc(s, 0);
    s->set_wpc(s, 1);
    s->set_raw_gma(s, 1);
    s->set_lenc(s, 1);
    s->set_hmirror(s, 0);
    s->set_vflip(s, 0);
    s->set_dcw(s, 1);
    s->set_colorbar(s, 0);
}

void BaseEsp32Cam::initialize()
{
    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 = 200000000;
config.pixel_format = PIXFORMAT_JPEG;
config.frame_size = FRAMESIZE_VGA;
config.jpeg_quality = 10;
config.fb_count = 2;

esp_err_t err = esp_camera_init(&config);

if (err != ESP_OK)
{
    Serial.printf("Camera initialize failed.");
    return;
}

Serial.printf("Camera initialized.");
}

```

A.7 Файл websocket_client_esp_32_cam.h

```

#ifndef HTTP_SERVER_ESP_32_CAM_TEST_H
#define HTTP_SERVER_ESP_32_CAM_TEST_H

#include "esp_http_server.h"
#include "img_converters.h"
#include "Arduino.h"

class HttpServerEsp32Cam {

private:
    static const char *_STREAM_CONTENT_TYPE;
    static const char *_STREAM_BOUNDARY;
    static const char *_STREAM_PART;
    static esp_err_t stream_handler(httpd_req_t *req);
    static esp_err_t capture_handler(httpd_req_t *req);
public:
    static void start();
};
#endif

```

A.8 Файл websocket_client_esp_32_cam.cpp

```

#include "websocket_client_esp_32_cam.h"

#define PART_BOUNDARY "1234567890000000000000987654321"

const char *HttpServerEsp32Cam::_STREAM_CONTENT_TYPE = "multipart/x-mixed-replace;boundary=" PART_BOUNDARY;
const char *HttpServerEsp32Cam::_STREAM_BOUNDARY = "\r\n--" PART_BOUNDARY "\r\n";

```

```

const char *HttpServerEsp32Cam::_STREAM_PART = "Content-Type:
image/jpeg\r\nContent-Length: %u\r\n\r\n";

esp_err_t HttpServerEsp32Cam::stream_handler(httpd_req_t *req)
{
    camera_fb_t *fb = NULL;
    esp_err_t result = ESP_OK;
    size_t _jpg_buf_len = 0;
    uint8_t *_jpg_buf = NULL;
    char *part_buf[64];
    result = httpd_resp_set_type(req, _STREAM_CONTENT_TYPE);

    if (result != ESP_OK)
    {
        return result;
    }

    while (true)
    {
        fb = esp_camera_fb_get();

        if (!fb)
        {
            Serial.println("Error. Video broadcast failed. Video broadcast");
            result = ESP_FAIL;
        }
        else
        {
            _jpg_buf_len = fb->len;
            _jpg_buf = fb->buf;
        }

        if (result == ESP_OK)
        {
            size_t hlen = snprintf((char *)part_buf, 64, _STREAM_PART,
            _jpg_buf_len);
            result = httpd_resp_send_chunk(req, (const char *)part_buf, hlen);
        }

        if (result == ESP_OK)
        {
            result = httpd_resp_send_chunk(req, (const char *)_jpg_buf,
            _jpg_buf_len);
        }

        if (result == ESP_OK)
        {
            result = httpd_resp_send_chunk(req, _STREAM_BOUNDARY,
            strlen(_STREAM_BOUNDARY));
        }

        if (fb)
        {
            esp_camera_fb_return(fb);
            fb = NULL;
            _jpg_buf = NULL;
        }
    }
}

```

```

    }
    else if (_jpg_buf)
    {
        free(_jpg_buf);
        _jpg_buf = NULL;
    }

    if (result != ESP_OK)
    {
        break;
    }
}

Serial.println("Video sended");
return result;
}

esp_err_t HttpServerEsp32Cam::capture_handler(httpd_req_t *req)
{
    esp_err_t result = ESP_OK;
    camera_fb_t *fb = esp_camera_fb_get();

    if (!fb)
    {
        Serial.println("Camera capture failed.");
        httpd_resp_send_500(req);
        result = ESP_FAIL;
    }
    else
    {
        httpd_resp_set_type(req, "image/jpeg");
        httpd_resp_set_hdr(req, "Content-Disposition", "inline;
filename=capture.jpg");
        httpd_resp_set_hdr(req, "Access-Control-Allow-Origin", "*");
        result = httpd_resp_send(req, (const char *)fb->buf, fb->len);
        esp_camera_fb_return(fb);
        Serial.println("Camera capture sended.");
    }

    Serial.println("Image sended");
    return result;
}

void HttpServerEsp32Cam::start()
{
    httpd_handle_t stream_httpd = NULL;
    httpd_config_t config = HTTPD_DEFAULT_CONFIG();
    config.server_port = 80;

    httpd_uri_t stream_uri =
    {
        .uri = "/stream",
        .method = HTTP_GET,
        .handler = stream_handler,
        .user_ctx = NULL};

```

```
httpd_uri_t capture_uri =
{
    .uri = "/capture",
    .method = HTTP_GET,
    .handler = capture_handler,
    .user_ctx = NULL};

if (httpd_start(&stream_httpd, &config) == ESP_OK)
{
    httpd_register_uri_handler(stream_httpd, &stream_uri);
    httpd_register_uri_handler(stream_httpd, &capture_uri);
}
}
```

ДОДАТОК Б
СЛАЙДИ ПРЕЗЕНТАЦІЇ

Національний університет «Запорізька політехніка»

Кафедра програмних засобів

Дипломна кваліфікаційна робота бакалавра

Розробка програмно-апаратного забезпечення для реалізації технології Інтернету речей в навчальній лабораторії

Виконав
ст. гр. КНТ-127

О.Ю. Бережний

Керівник
к.т.н., доцент

А.В. Пархоменко

2021

Рисунок Б.1 – Слайд № 1



Рисунок Б.2 – Слайд № 2

Мета та задачі дипломної роботи

Мета роботи – розробка програмно-апаратного комплексу відеомоніторингу для розширення функціональних можливостей віддаленої навчальної лабораторії.

Задачі роботи:

- виконати аналіз вимог та розробити архітектуру ПАК;
- обрати апаратні та програмні засоби розробки;
- виконати програмну реалізацію;
- провести тестування ПАК;
- розробити програмну документацію.

Рисунок Б.3 – Слайд № 3



Рисунок Б.4 – Слайд № 4

Дослідження протоколів для реалізації IoT

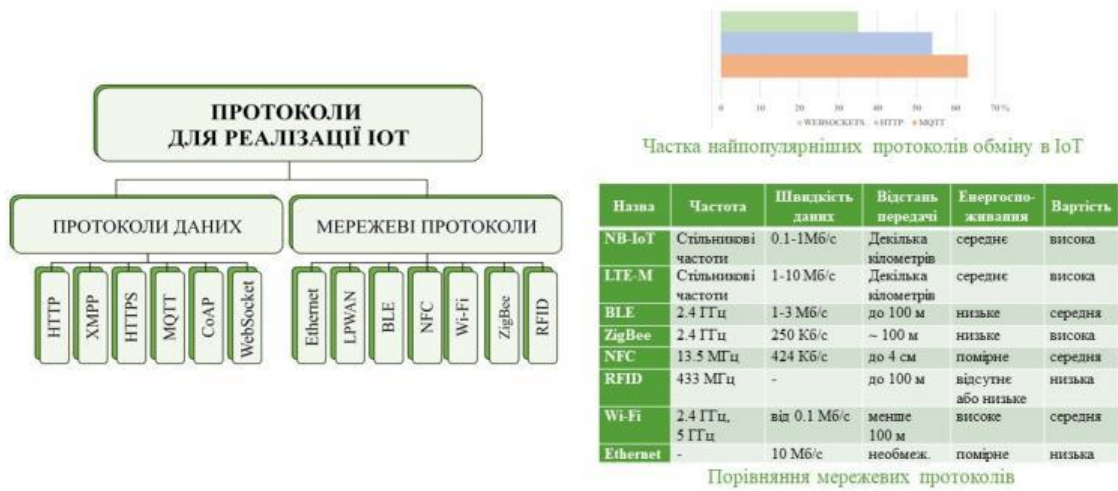


Рисунок Б.5 – Слайд № 5

Архітектура ПАК

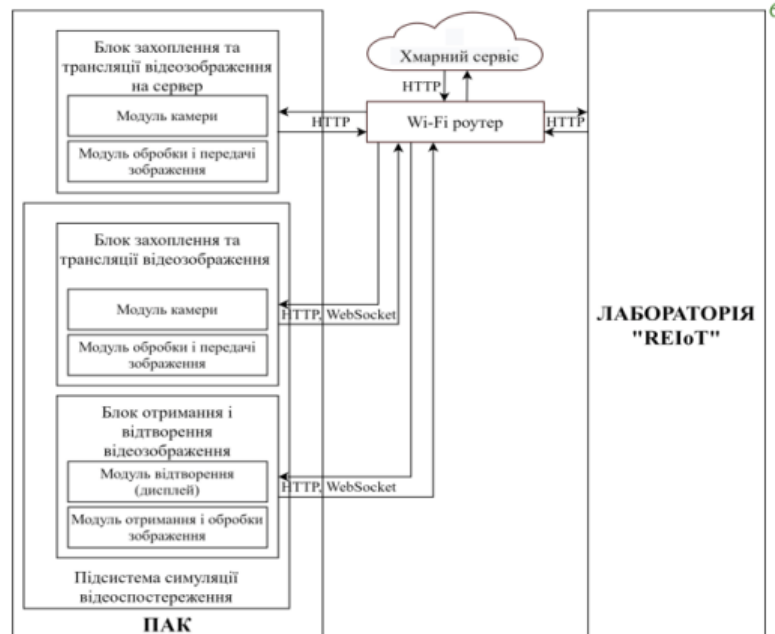


Рисунок Б.6 – Слайд № 6

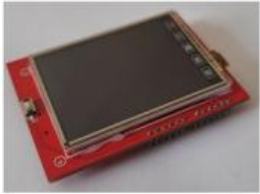
Вибір апаратних засобів



ESP-32



ESP-32-CAM
з об'єктивом «стандарт»



TFT-дисплей 2.4 16BIT ILI9341

	Arduino UNO R3	Arduino Mega	ESP-8266	ESP-32
Мікроконтролер	Atmega 4809	Atmega 2560	ESP-8266	ESP-32
Робоча напруга	5 В	5 В	3,3 В	3,3 В
Поточне споживання	50 мА - 150 мА	50 мА - 200 мА	0,5 мкА	5 мкА
Флеш-пам'ять	48 КБ	256 КБ	4 МБ	4 МБ
SRAM	6 КБ	8 КБ	—	520 КБ
EEPROM	256 Байт	4096 Байт	512 Байт	512 Байт
Тактова частота	16 МГц	16 МГц	52 МГц	160 МГц
Wi-Fi	є	немає	є	є

	2.4 SPI ILI9341 SKU:MSP2402	2.4 16BIT ILI9341 SKU:MIRB2408
Розмір дисплею	2.4" (240x320)	2.4" (240x320)
Кількість пінів для з'єднання	15	4
Швидкість передачі даних	висока	середня
Вартість	середня	висока

Рисунок Б.7 – Слайд № 7

Вибір програмних засобів

LUA (NodeMCU)

MicroPython

C++ (ARDUINO)

C (RTOS)

ATL КОМАНДИ

ПРОГРАМУВАННЯ ESP-32



Arduino IDE



PlatformIO

Мова	Кількість кадрів за секунду
MicroPython	7 – 8
Lua	4
C	27
C++	35

Критерій / Середовище	Arduino IDE	PlatformIO
Складність інтерфейсу	низька	середня
Налаштування інтерфейсу	відсутнє	присутнє
Навігація за кодом	відсутня	присутня
Автооновнення	відсутнє	присутнє
Система відслідковування версій	відсутня	присутня

Рисунок Б.8 – Слайд № 8

Алгоритм роботи ПАК

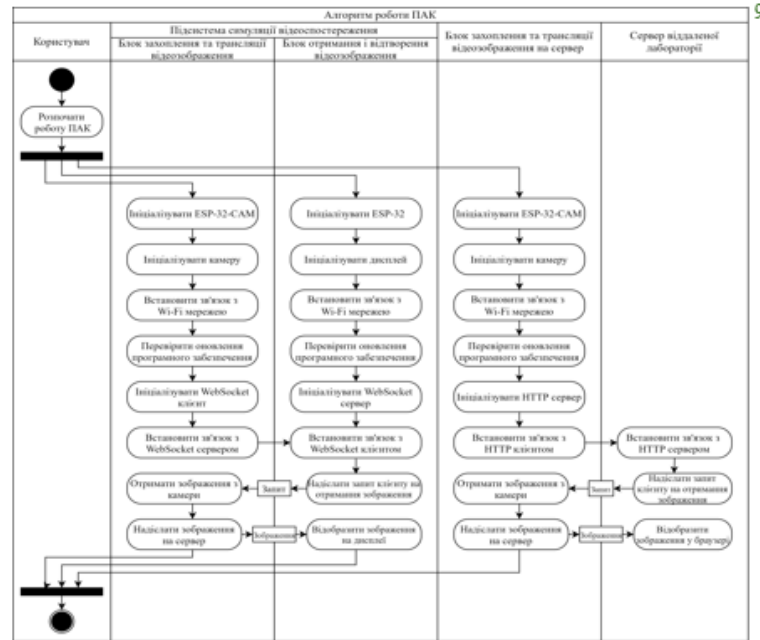


Рисунок Б.9 – Слайд № 9

Алгоритм оновлення ПЗ

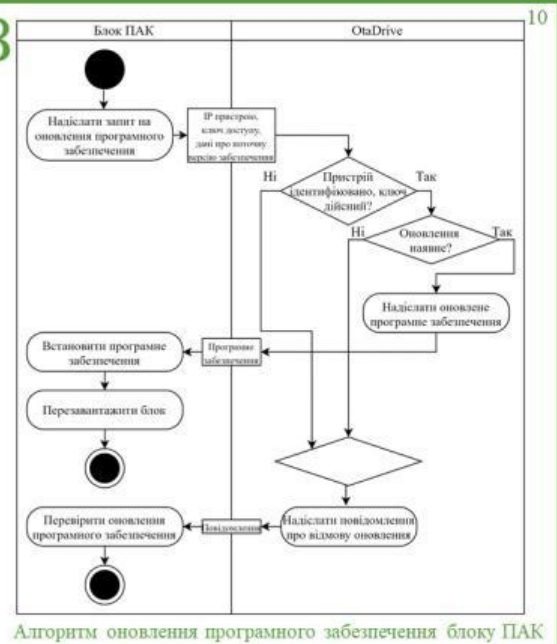


Рисунок Б.10 – Слайд № 10

Структура програмного забезпечення

```

base_esp_32_cam.cpp
base_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
websocket_server_esp_32_cam.cpp
websocket_server_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h

```

Блок отримання та відтворення відеозображення у складі симуляції відеоспостереження

```

base_esp_32_cam.cpp
base_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
tft_control.cpp
tft_control.h
websocket_client_esp_32_cam.cpp
websocket_client_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h

```

Блок захоплення та трансляції відеозображення у складі симуляції відеоспостереження

```

base_esp_32_cam.cpp
base_esp_32_cam.h
http_server_esp_32_cam.cpp
http_server_esp_32_cam.h
main.cpp
ota_drive.cpp
ota_drive.h
pin_esp_32_cam.h
wifi_connect.cpp
wifi_connect.h

```

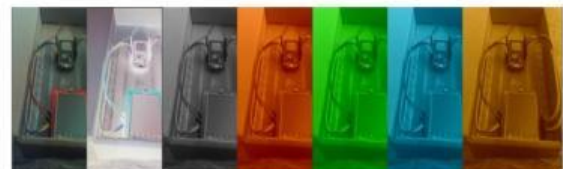
Блоку захоплення та трансляції відеозображення на сервер

Рисунок Б.11 – Слайд № 11

Робота камери за різних налаштувань



Робота камери при різних режимах балансу білого



Робота камери при різних художніх режимах



Робота камери при різних режимах яскравості



Робота камери при різних режимах насиченості

Рисунок Б.12 – Слайд № 12

Функціональне тестування

Передумови			
Увімкнути роутер, увімкнути камеру			
№	Дія	Очікуваний результат	Статус
1	Перейти у браузері на ПК за адресою «IP адреса камери»/stream	Відображається відеопотік з камери в режимі онлайн	Пройдено
2	Перейти у браузері на ПК за адресою «IP адреса камери»/screen	Отримано зображення з камери	Пройдено
3	Перезавантажити Wi-Fi роутер через який відбувається з'єднання, зачекати 10 секунд і повторити пункти 1, 2	Аналогічно до пунктів 1 і 2 відповідно	Пройдено

Тест-кейс перевірки блоку трансляції відеозображення на сервер

Передумови			
Увімкнути роутер, через який налаштовано Wi-Fi з'єднання			
№	Дія	Очікуваний результат	Статус
1	Увімкнути блок відображення відеопотоку	Дисплей запалюється синім кольором	Пройдено
2	Увімкнути камеру	Не довше через 10 секунд на дисплеї відображається відеопотік з камери	Пройдено
3	Вимкнути Wi-Fi роутер	На дисплеї відображується останнє зображення з камери	Пройдено
4	Увімкнути Wi-Fi роутер	Аналогічно до пункту 2	Пройдено

Тест-кейс перевірки дієздатності підсистеми симуляції відеоспостереження

Рисунок Б.13 – Слайд № 13

Функціональне тестування

Вимога	Статус
Блок трансляції відеозображення на сервер	
Трансляція відео в режимі «онлайн»	Пройдено
Можливість миттєвого захоплення поточного зображення камери	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Блок трансляції відеозображення у складі підсистеми симуляції відеоспостереження	
Трансляція відео в режимі «онлайн»	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Трансляція відео в режимі «онлайн»	Пройдено
Незалежність від існуючих стендів	Пройдено
Блок відтворення відеозображення у складі підсистеми симуляції відеоспостереження	
Відображення відео в режимі «онлайн»	Пройдено
Дієздатність параметрів за замовчуванням	Пройдено
Дистанційне оновлення програмного забезпечення	Пройдено
Дієздатність статичної IP адреси	Пройдено
Незалежність від існуючих стендів	Пройдено

Контрольний список функціонального тестування ПАК

Рисунок Б.14 – Слайд № 14

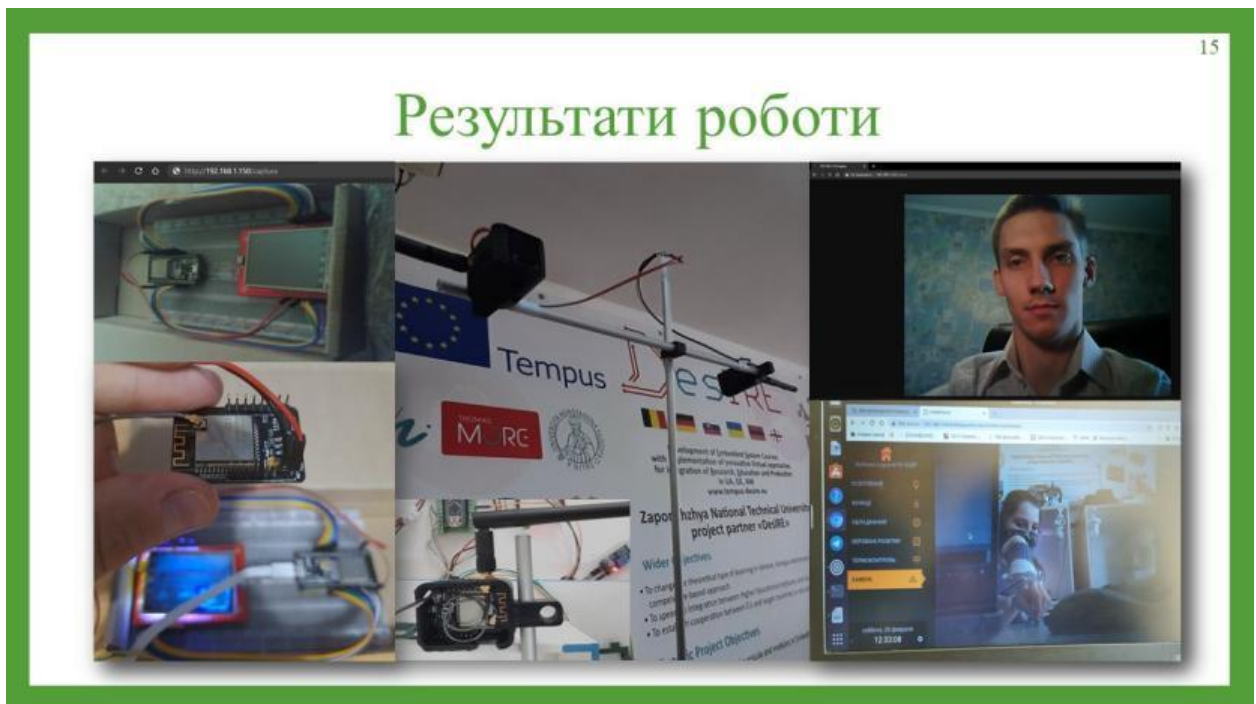


Рисунок Б.15 – Слайд № 15

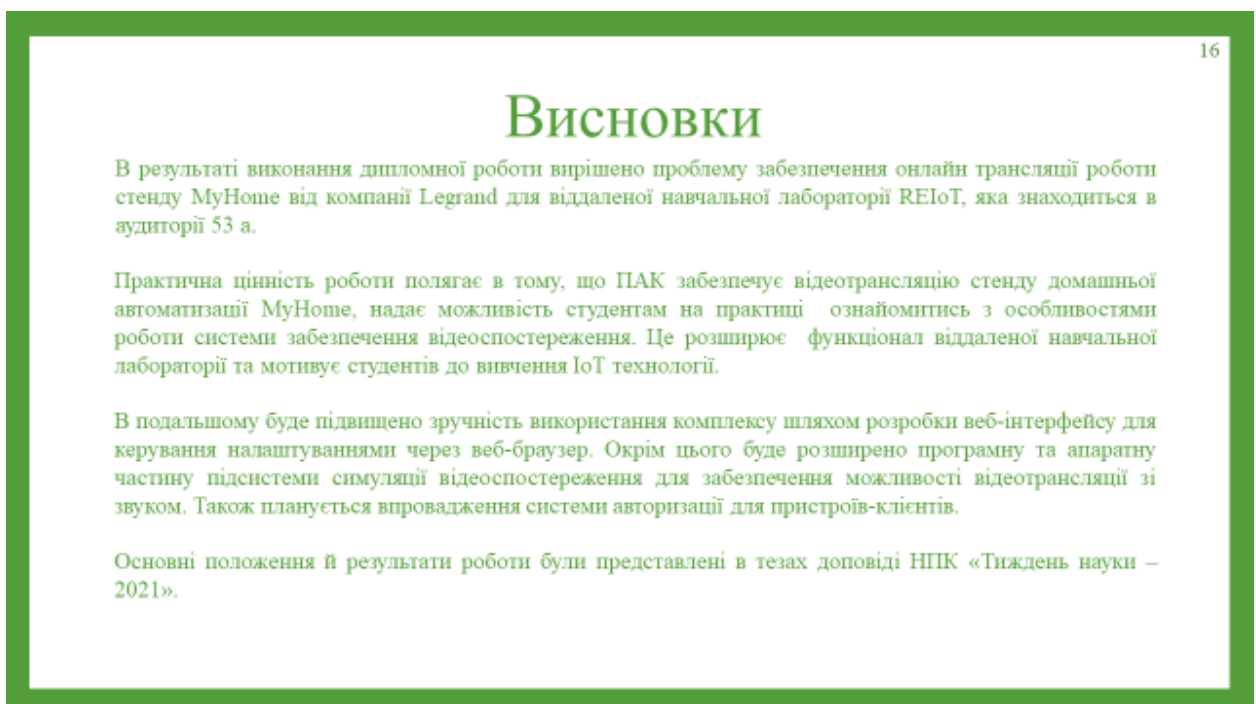


Рисунок Б.16 – Слайд № 16

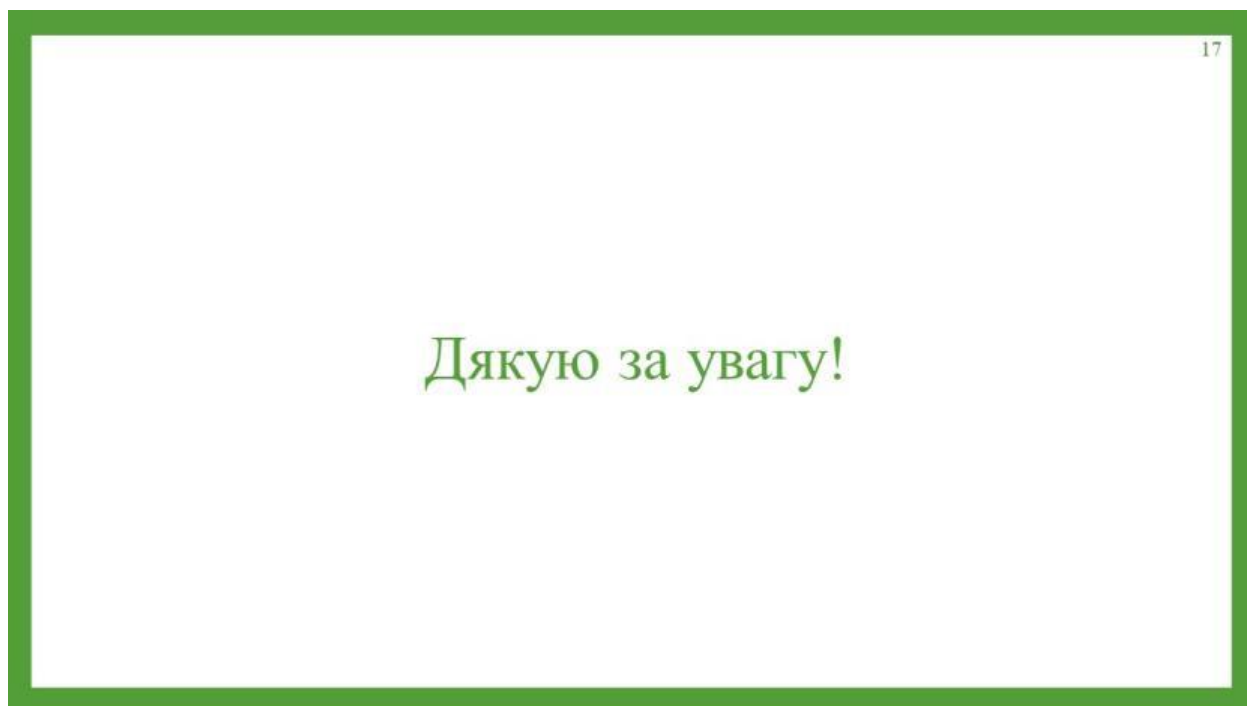


Рисунок Б.17 – Слайд № 17

ДОДАТОК В
ФОТОЗВІТ ПРОЦЕСУ РОЗРОБКИ ПАК

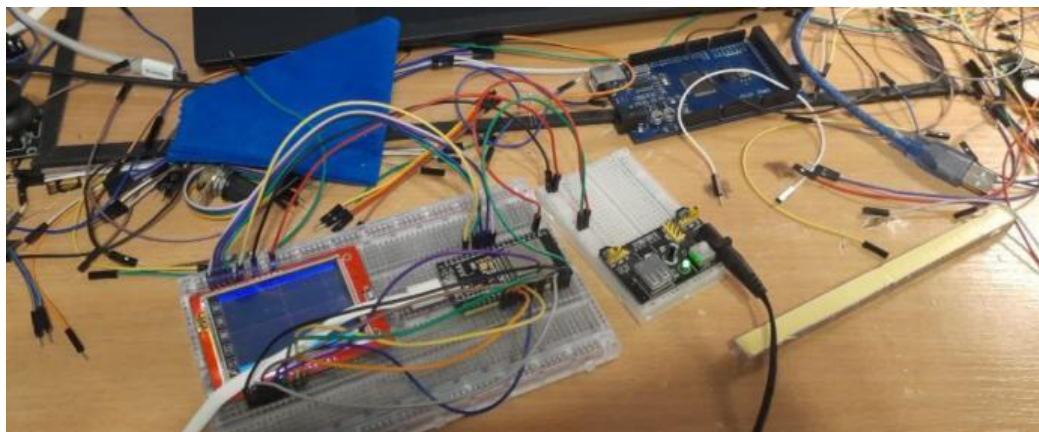


Рисунок В.1 – З'єднання модулів підсистеми симуляції відеоспостереження



Рисунок В.2 – Розробка кріплення для блоку захоплення і трансляції відеозображення

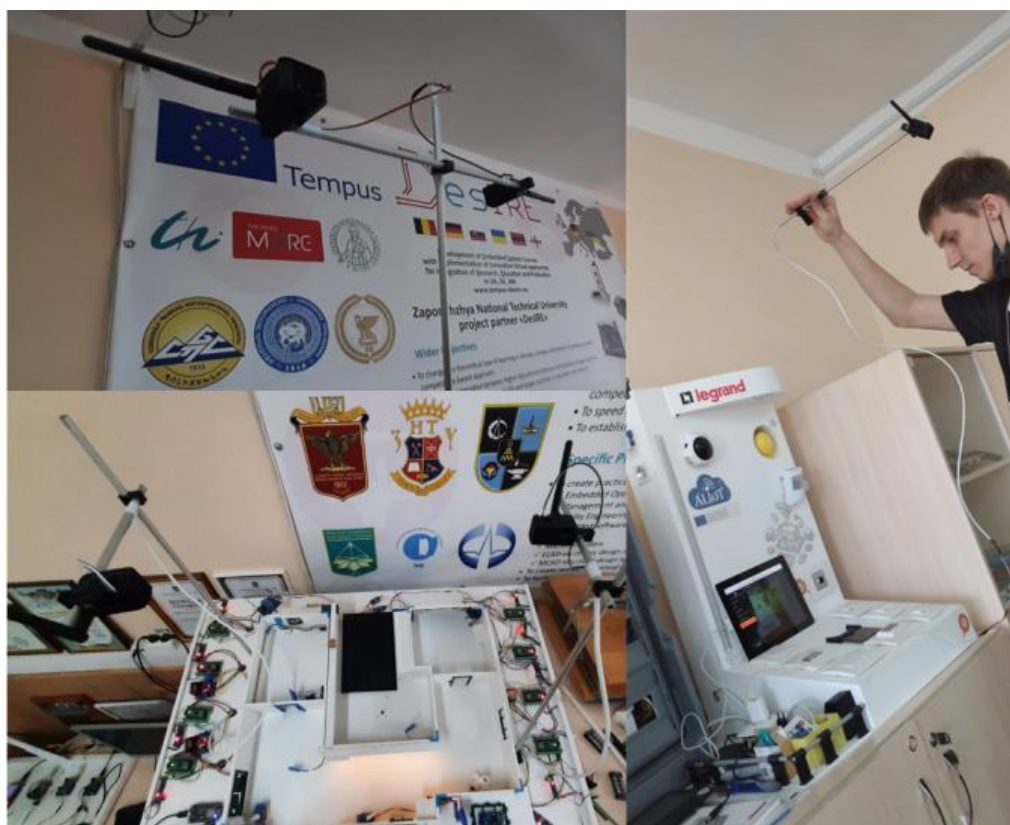


Рисунок В.3 – Встановлення блоку захоплення і трансляції відеозображення

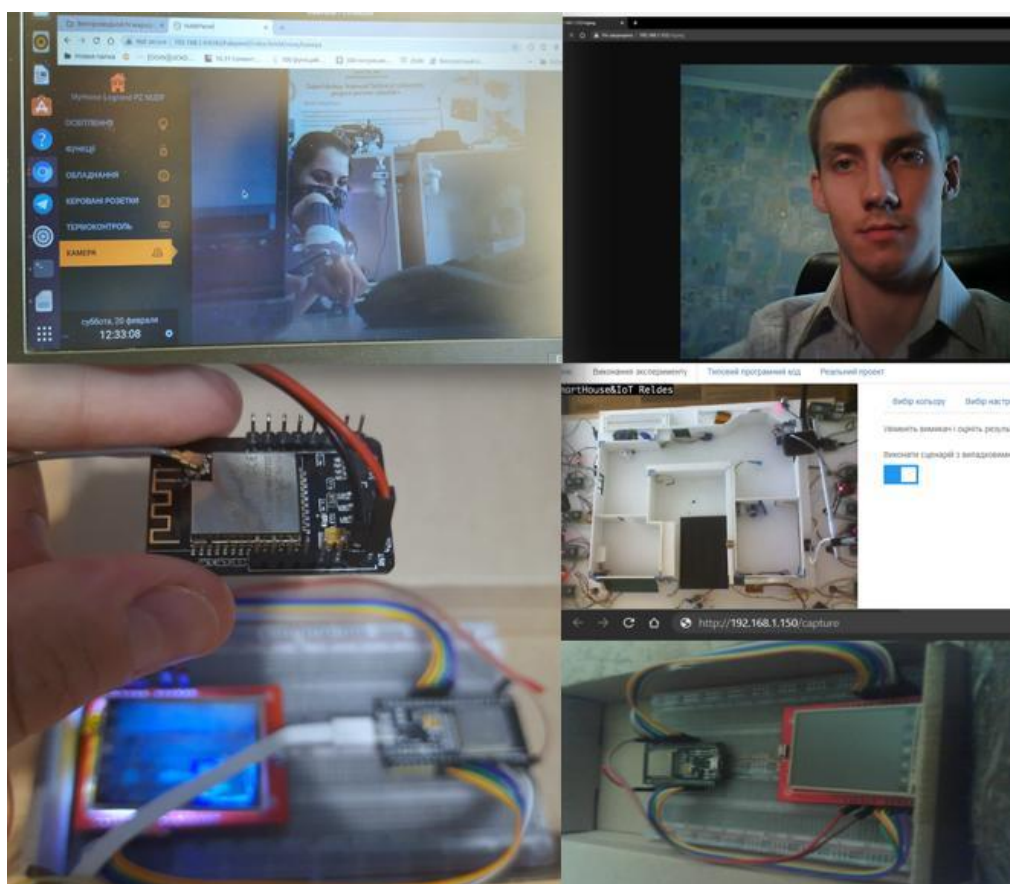


Рисунок В.4 – Тестування роботи блоків ПАК