Міністерство освіти і науки України
Харківський національний університет радіоелектроніки

Факультет _____Автоматики і комп'ютеризованих технологій_____
(повна назва)

Кафедра Комп'ютерно-інтегрованих технологій автоматизації та робототехніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти _____перший (бакалаврський)_____
Розроблення автоматизованого контролю замикання дверей із застосуванням IoT-технологій
(тема)

Виконав:
здобувач ____4____ року навчання,
групи _АКТАКІТ-21-1_____
_Владислав МОМОТ_____
(власне ім'я, прізвище)

Спеціальність _151 Автоматизація та комп'ютерно-інтегровані технології_____
(код і повна назва спеціальності)
Тип програми ____освітньо-професійна_____
Освітня програма _Автоматизація та комп'ютерно-інтегровані технології_____
(повна назва освітньої програми)

Керівник _асистент Ганна САМОЙЛЕНКО_
(посада, власне ім'я, прізвище)

Допускається до захисту

Завідувач кафедри _____ _____ __Ігор НЕВЛЮДОВ___
(підпис) (власне ім'я, прізвище)

2025 р.

Я, Момот Владислав Олександрович, як здобувач вищої освіти ХНУРЕ, розумію і підтримую політику закладу із академічної доброчесності. Я не надавав і не одержував недозволену допомогу під час підготовки кваліфікаційної роботи. Я не використовував штучний інтелект для підготовки кваліфикаційної роботи. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

“22” червня 2025р.  Владислав Момот

Харківський національний університет радіоелектроніки

Факультет_____Автоматики і комп'ютеризованих технологій_____

Кафедра_Комп'ютерно-інтегрованих технологій автоматизації та робототехніки_

Рівень вищої освіти_____перший (бакалаврський)_____

Спеціальність_151 Автоматизація та комп'ютерно-інтегровані технології___

Тип програми _____освітньо-професійна_____

Освітня програма__Автоматизація та комп'ютерно-інтегровані технології__

ЗАТВЕРДЖУЮ:

Зав. кафедри _____
(підпис)

« 28 »_____червня____ 2025р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

здобувачеві_____Момоту Владиславу Олександровичу_____
(прізвище, ім'я, по батькові)

1. Тема роботи Розроблення автоматизованого контролю замикання дверей із застосуванням IoT-технологій_____
затверджена наказом університету від _19_ ___травня_____2025 р. _№390Ст___
2. Термін подання здобувачем роботи до екзаменаційної комісії _25_ __червня___ 2025 р.
3. Вихідні дані до роботи _Функціональні вимоги: можливість дистанційного керування; підтримка Wi-Fi; відображення поточного стану замка у мобільному застосунку. Мікроконтролер ESP8622(NodeMCU)(емітація через ESP32); мобільний застосунок створений у середовищі Flutter_____
_____
_____
4. Перелік питань, що потрібно опрацювати в роботі _Аналіз сучасних систем контролю доступу, аналіз мов програмування та фреймворків для мобільного застосунку, розробка архітектури системи замикання, тестування системи_____
_____
_____

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _Презентація 12 слайдів_____

_____

_____

6. Консультанти розділів роботи (п.6 включається до завдання за наявності консультантів згідно з наказом, зазначеним у п.1)

Найменування розділу	Консультант (посада, прізвище, ім'я, по батькові)	Позначка консультанта про виконання розділу	
		підпис	дата

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Строк / терміни виконання етапів	Примітка
1	Аналіз технічного завдання	31.04.2025	Виконано
2	Аналіз літератури за темою роботи	08.05.2025	Виконано
3	Виконання розділу 1 Аналіз сучасних рішень систем контролю доступу	15.05.2025	Виконано
4	Виконання розділу 2 Обґрунтування архітектури системи контролю замикання	17.05.2025	Виконано
5	Виконання розділу 3 Розробка мобільного додатка для керування системою	22.05.2025	Виконано
6	Виконання розділу 4 Симуляція та тестування роботи системи	23.05.2025	Виконано
7	Подання роботи на перевірку Інтернет-сервісом StrikePlagiarism	15.06.2025	Виконано
8	Оформлення пояснювальної записки	17.06.2025	Виконано
9	Подання роботи на рецензію	17.06.2025	Виконано
10	Подання роботи на підпис зав. кафедри	20.06.2025	Виконано
11	Подання кваліфікаційної роботи в ЕК	25.06.2025	Виконано

Дата видачі завдання _28_ _квітня_2025 р.

Здобувач _____Владислав МОМОТ_____
(підпис)

Керівник роботи _____ _асистент Ганна САМОЙЛЕНКО_
(підпис) (посада, власне ім'я, прізвище)

РЕФЕРАТ

Пояснювальна записка: 79 с., 2 табл., 16 рис., додатків 3, джерел. 21

ІНФОРМАЦІЙНО-КЕРУЮЧА СИСТЕМА, ПРИСТРОЇ ЗАМИКАННЯ, WI-FI, ІНТЕРНЕТ РЕЧЕЙ(IOT) АВТОМАТИЗОВАНА СИСТЕМА КОНТРОЛЮ ЗАМИКАННЯ ДВЕРЕЙ.

Об’єкт розробки – процес створення інформаційно-керуючої системи забезпечення фізичної безпеки приміщень, що працює на основі безконтактного дистанційного управління пристроями замикання.

Предмет розробки – Апаратне та програмне забезпечення частини системи замикання дверей на основі мікроконтролера ESP8622 з використанням мобільного додатку, Wi-Fi / Bluetooth зв’язку та технологій Інтернету речей.

Мета роботи – розроблення автоматизованої системи контролю замикання дверей на базі IoT-технологій. Система повинна керуватись за допомогою мобільного додатку через Wi-Fi, підтримувати авторизацію користувачів та дистанційне управління в реальному часі без фізичної моделі, з реалізацією симуляції у середовищі Wokwi.

У першому розділі виконано аналітичний огляд сучасних рішень у сфері інтелектуальних систем контролю доступу. Проаналізовано існуючі типи електронних замків, їх принципи роботи, а також ключові елементи побудови подібних систем на базі IoT. Розкрито, як саме реалізуються системи з віддаленим керуванням та які протоколи й мови програмування найчастіше використовуються.

У другому розділі проведено обґрунтування вибору компонентів, технологій і архітектури розроблюваної системи. Визначено, чому саме ESP8622 є найбільш ефективною платформою для реалізації цього завдання. Побудовано логічну схему взаємодії всіх елементів, розглянуто мережеву взаємодію, а також обрано способи реалізації взаємодії з користувачем. Створено структуру мобільного інтерфейсу

У третьому розділі реалізовано функціональну частину проєкту. Розроблено мобільний застосунок на Flutter, який підтримує авторизацію, перемикання стану замка.

У четвертому розділі було протестована логіка схеми на базі ESP8622(ESP32 відсутність підтримки плати потрібної плати). Реалізовано виконування команд /lock, /unlock, /status та продеманстровано відповідь плати на данні команди.

У результаті виконання дипломної роботи реалізовано роздріблену модель IoT- системи контролю замикання дверей. Система функціонує у віртуальному середовищі Wokwi. Мобільний застосунок повноцінно підтримує логіку користувача, а серверна частина — не може підтримувати комунікацію та обробку команд через обмеження середовища Wokwi. Створене рішення буде реалізовано тільки з наявністю фізичної моделі.

ABSTRACT

Explanatory note: 79 p., 2 tables, 16 figures, 3 appendices, sources. 21

INFORMATION AND CONTROL SYSTEM, LOCKING DEVICES, WI-FI, INTERNET OF THINGS (IOT) AUTOMATED DOOR LOCKING CONTROL SYSTEM.

The object of development is the process of creating an information and control system for ensuring the physical security of premises, operating on the basis of contactless remote control of locking devices.

The subject of development is the hardware and software of a part of the door locking system based on the ESP8622 microcontroller using a mobile application, Wi-Fi / Bluetooth communication and Internet of Things technologies.

The purpose of the work is to develop an automated door locking control system based on IoT technologies. The system should be controlled using a mobile application via Wi-Fi, support user authorization and remote control in real time without a physical model, with the implementation of simulation in the Wokwi environment.

The first section provides an analytical review of modern solutions in the field of intelligent access control systems. Existing types of electronic locks, their principles of operation, as well as key elements of building such systems based on IoT are analyzed. It is revealed how remote control systems are implemented and which protocols and programming languages are most often used.

The second section provides a justification for the selection of components, technologies and architecture of the developed system. It is determined why ESP8622 is the most effective platform for implementing this task. A logical diagram of the

interaction of all elements is built, network interaction is considered, and methods for implementing interaction with the user are selected. The structure of the mobile interface is created

In the third section, the functional part of the project is implemented. A mobile application on Flutter is developed that supports authorization, switching the lock state.

In the fourth section, the logic of the circuit based on ESP8622 was tested (ESP32 lack of support for the required board). The execution of the /lock, /unlock, /status commands is implemented and the board's response to these commands is demonstrated.

As a result of the thesis, a distributed model of the IoT door locking control system was implemented. The system operates in the Wokwi virtual environment. The mobile application fully supports the user's logic, and the server part cannot support communication and command processing due to the limitations of the Wokwi environment. The created solution will be implemented only with the presence of a physical model.

Also, the results of the work can be attributed to Sustainable Development Goal 9 “Industry, Innovation and Infrastructure”, namely paragraphs 9.1, 9.4, 9.5. Automation of door locking processes using IoT directly contributes to the modernization of existing infrastructure facilities, ensures the efficient and safe functioning of buildings, contributing to the implementation of Goal 9.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ

API - Application Programming Interface

BLE - Bluetooth Low Energy

DIY - Do It Yourself

ERP - Enterprise Resource Planning

HTTP - Hypertext Transfer Protocol

IDE - Integrated Development Environment

IoT - Internet of Things

IP - Internet Protocol

JSON - JavaScript Object Notation

MITM - Man-in-the-middle

MQTT - Message Queuing Telemetry Transport

MVVM - Model–View–ViewModel

NFC - Near Field Communication

OLED - Organic Light-Emitting Diode

PIN - Personal Identification Number

PWA - Progressive Web App

PWM - Pulse-Width Modulation

SDK - Software Development Kit

SoC - System-on-a-Chip

UI - User Interface

ВСТУП

Актуальність теми. Безпека людини, її майна та комфорту у наші часи стала одним із пріоритетів розвитку інтелектуальних систем, зокрема в межах концепції «розумного дому». Зі стрімким розвитком цифрових технологій, а особливо – технологій Інтернету речей (IoT), з’явилися широкі можливості для створення автоматизованих систем, що дозволяють здійснювати контроль, моніторинг і управління різними об’єктами в режимі реального часу, без фізичної присутності людини. Одним із актуальних напрямів таких розробок є системи контролю доступу, які дозволяють суттєво підвищити рівень безпеки приміщень, забезпечуючи гнучке керування замиканням та відкриттям дверей із віддаленого пристрою.

Технологія IoT, у свою чергу, розширює можливості подальших досліджень для автоматизації, оскільки вона дозволяє фізичним пристроям обмінюватися даними, отримувати команди з мобільного або веб-додатку, зберігати log файли, здійснювати авторизацію користувачів тощо. При цьому використання доступних компонентів, таких як мікроконтролери ESP8622, дозволяє реалізувати функціональні прототипи, які відповідають сучасним вимогам до безпеки, надійності та зручності. Кваліфікаційна робота виконана згідно ДСТУ 3008 – 15 [1], керуючись навчальним посібником з дипломного проєкту та методичними вказівками[2].

1 АНАЛІЗ СУЧАСНИХ РІШЕНЬ СИСТЕМ КОНТРОЛЮ ДОСТУПУ

1.1 Характеристика та класифікація систем доступу

Системи контролю та обмеження доступу є складовою інфраструктури безпеки сучасних об’єктів, починаючи від житлових будинків і офісів до промислових підприємств, урядових установ і стратегічних об’єктів. Їх призначення полягає у забезпеченні регульованого проходу осіб, запобіганні несанкціонованому проникненню та забезпеченні моніторингу дій користувачів у зоні контролю. В умовах стрімкої цифровізації, зростання кіберзагроз та загальної потреби в мобільності та автоматизації, питання організації доступу перестають бути виключно технічними й переходять у площину комплексного управління безпекою об’єкта в інтегрованому цифровому середовищі.

Класична система доступу передбачала фізичне блокування проходу через двері, турнікети або ворота за допомогою механічних елементів, таких як ключ, кодовий замок або пропускна система з охоронцем. Втім, подальший розвиток мікроелектроніки та автоматизації дозволив перейти до електронних систем, які забезпечують гнучкі сценарії доступу, автоматичний облік переміщення, обмеження за часом або ролями, а також інтеграцію з відеоспостереженням, сигналізацією та навіть ERP-системами підприємства [3].

Сучасні системи контролю доступу умовно поділяють на автономні, мережеві та інтегровані. Автономні системи працюють без постійного зв’язку з центральним сервером – вони мають обмежений набір функцій, зазвичай передбачають авторизацію через магнітну картку, PIN-код або NFC-мітку. Такі рішення доцільні для локального використання в умовах, де не передбачена складна ієрархія доступів чи інтеграція з іншими підсистемами. Натомість мережева система дозволяє здійснювати централізоване керування доступом на

великій кількості точок контролю, вести журнал подій у реальному часі, встановлювати розклад або обмеження для кожного користувача. Інтегровані системи – це найбільш функціональні комплекси, що об’єднують доступ, охоронну сигналізацію, відеоспостереження, управління освітленням та іншими підсистемами розумного середовища в єдину інформаційну платформу (рис.1.1).

Рисунок 1.1 – Складові сучасних систем доступу

За способом ідентифікації користувача системи контролю доступу класифікують на ті, що базуються на знанні (наприклад, введення пароля чи PIN-коду), на володінні певним пристроєм (магнітна картка, брелок, смартфон) та на біометричних ознаках (відбиток пальця, сітківка ока, розпізнавання обличчя). Найбільш прості та доступні системи використовують кодові замки, хоча вони мають низький рівень захисту. Значно ефективнішими є рішення, що базуються на біометричних параметрах, однак вони потребують якісного обладнання та надійного захисту даних [4].

Зі зростанням популярності технологій Інтернету речей, на ринку з’явилася нова генерація систем доступу – так звані «розумні замки» та IoT-платформи управління доступом. Такі рішення забезпечують доступ до механізмів замикання через мобільний додаток, веб-інтерфейс або голосові

команди. Вони можуть працювати через Wi-Fi, Bluetooth або інші бездротові протоколи, що дозволяє керувати замком з будь-якої точки світу, де є інтернет-з'єднання. У більшості випадків вони також інтегруються з хмарною інфраструктурою для зберігання журналів подій, резервного копіювання налаштувань та централізованого управління пристроями. Зокрема, у побутовому застосуванні розумні замки набули популярності завдяки можливості автоматичного відкриття дверей при наближенні смартфону власника або після підтвердження в мобільному додатку. У разі багатокористувацького доступу система може надавати тимчасові або постійні права доступу іншим користувачам, що особливо зручно для офісів, орендованих квартир або готелів.

Крім того, IoT-системи контролю доступу дозволяють реалізувати принцип «zero trust», коли кожна спроба доступу оцінюється як потенційно ризикована і підлягає додатковій перевірці, наприклад, через двофакторну авторизацію або контроль геолокації користувача. У промислових та корпоративних рішеннях такі системи поєднуються з контролем робочого часу, статистикою переміщення, енергоменеджментом та штучним інтелектом для виявлення аномалій. Наприклад, система може автоматично блокувати доступ, якщо виявлено спробу входу в неробочий час, або запускати протокол тривоги при відкритті дверей без авторизації.

Характерною рисою сучасних рішень є модульність та масштабованість: систему можна поступово нарощувати, додаючи нові точки контролю, інтегруючи нові технології або оновлюючи програмне забезпечення. У цьому контексті особливо актуальною є тенденція до використання відкритих протоколів та API, що дозволяє гнучко поєднувати пристрої від різних виробників і створювати власні сценарії поведінки. Водночас, відкритість системи підвищує вимоги до інформаційної безпеки, зокрема до захисту каналів зв'язку, автентифікації користувачів та шифрування даних [5].

У навчальній, побутовій та малому бізнес-середовищі набули поширення спрощені рішення на базі платформ Arduino або ESP8622/ESP32. Такі контролери дозволяють реалізовувати прості та водночас функціональні системи доступу за мінімальних витрат.

1.2 Принцип дії, типи та переваги електронних замків

У структурі сучасних систем контролю доступу електронні замки відіграють роль ключової виконавчої ланки, яка безпосередньо реалізує дію з фізичним обмеженням або наданням доступу до об’єкта. Від механічних замків їх відрізняє наявність активної логіки керування, яка приймає сигнали ззовні – від користувача, мобільного пристрою, модуля ідентифікації або хмарного сервісу – і трансформує їх у відповідну дію: замикання або розблокування. Вони не лише слугують засобом фізичної ізоляції, а й забезпечують функціонал інтерактивного контролю, що робить їх інструментом для цифрового керування безпекою.

Принцип дії більшості електронних замків полягає в керуванні рухом механізму блокування за допомогою електроприводу, який, у свою чергу, активується електричним сигналом, що подається з контролера. Залежно від конфігурації пристрою та умов його експлуатації, замикання може здійснюватись соленоїдним, сервомоторним або електромагнітним механізмом. Подача або зняття живлення запускає лінійний або обертальний рух елемента блокування, який механічно фіксує або вивільняє ригель. У деяких моделях використовується також двостороннє керування, коли той самий механізм може як замикати, так і розблоковувати доступ за командами ззовні [6].

У реальних умовах експлуатації електронний замок не є ізольованим компонентом. Його ефективність безпосередньо залежить від зв’язку з системою автентифікації, програмного забезпечення, сенсорної периферії

(наприклад, датчиків відкриття дверей), а також від стабільного джерела живлення. У разі зникнення електроенергії або збою в логіці керування, можливе виникнення ситуацій, коли доступ буде або повністю втрачено, або залишиться відкритим без контролю. Для уникнення таких інцидентів використовують резервні системи живлення, зокрема акумулятори або суперконденсатори, а також механічні аварійні ключі.

На сьогоднішній день на ринку існує велика кількість електронних замків, які відрізняються за способом керування, типом механізму замикання, типом ідентифікації, функціональністю та призначенням. Найпростішим типом можна вважати замки з кодовою клавіатурою, де користувач вводить цифрову комбінацію, яка порівнюється з еталонною. Така схема не забезпечує високого рівня захисту, однак часто використовується у побутових умовах через простоту реалізації. Значно більш досконалими є замки з картковим доступом, що працюють на основі RFID або NFC технологій, де роль ключа відіграє електронний носій з унікальним ідентифікатором. Такі пристрої здатні зберігати великі бази даних користувачів, журналювати події та працювати в умовах багаторазового використання без втрати надійності.

Окрему категорію становлять біометричні замки, які ідентифікують користувача за унікальними фізіологічними параметрами, такими як відбиток пальця, форма обличчя або райдужна оболонка ока. Біометрія забезпечує високий рівень захисту, однак вимагає якісного сенсорного обладнання та відповідного програмного забезпечення, здатного обробляти і порівнювати біометричні шаблони в режимі реального часу. В умовах великого навантаження або неякісного освітлення можливе виникнення помилкових спрацювань, що потребує продуманої системи резервування [7].

Найперспективнішим напрямом є розумні замки з можливістю підключення до Інтернету та віддаленого керування. Такі замки зазвичай оснащуються модулями Wi-Fi або Bluetooth, що дозволяє керувати ними зі

смартфона або планшета. Крім відкривання та закривання, розумні замки можуть надсилати повідомлення про спроби доступу, фіксувати час ідентифікації, будувати статистику використання, а також взаємодіяти з іншими елементами екосистеми «розумного дому» – наприклад, системами відеоспостереження або освітлення. Завдяки використанню хмарних платформ управління, такі пристрої дозволяють власнику мати повний контроль над безпекою свого об’єкта незалежно від місця перебування (рис. 1.2).

Рисунок 1.2 – Керування розумним замком через програму на телефоні [10]

З точки зору конструктивного виконання електронні замки можуть бути накладними або врізними. Накладні моделі встановлюються на поверхню дверей і, як правило, мають простіший монтаж, тоді як врізні вбудовуються в полотно дверей, забезпечуючи кращу естетику та надійність. Окремі моделі передбачають встановлення поверх звичайного механічного замка, таким чином утворюючи гібридну систему, яка дозволяє користувачеві обрати

найбільш зручний спосіб доступу в залежності від ситуації. У таких випадках електронна частина активує або блокує механізм повороту ключа.

Перевагами електронних замків, у порівнянні з механічними, є значно вищий рівень захисту, гнучкість у налаштуванні сценаріїв доступу, можливість багатокористувацької авторизації, ведення журналу подій, дистанційне керування та автоматичне блокування при підозрілих діях. Крім того, вони дозволяють значно зменшити ризики, пов’язані з втратою ключів, оскільки замість фізичного ключа використовується програмна ідентифікація, яку можна в будь-який момент змінити, відкликати або обмежити в правах. У разі втрати смартфона або картки доступу достатньо оновити параметри системи, не вдаючись до фізичної заміни пристрою.

У розрізі розробки власного пристрою на базі платформи ESP8622 особливу увагу слід приділити вибору приводу замикання. Найчастіше використовуються сервоприводи малого розміру, які обертаються на заданий кут і повертають механічну тягу в положення «замкнено» або «відкрито». Такі пристрої мають низьке енергоспоживання, швидкий час реакції та можуть керуватись за допомогою простого PWM-сигналу з мікроконтролера. При цьому важливо забезпечити механічну фіксацію у випадку втрати живлення або збоїв системи. Альтернативою є використання електромагнітного замка, який розблоковується при знятті живлення. Такий тип часто застосовується в офісах, однак потребує постійного живлення і не підходить для аварійних ситуацій.

1.3 Інтелектуальні системи замикання в контексті IoT

Застосування технологій Інтернету речей (IoT) стало потужним каталізатором еволюції класичних систем контролю доступу в напрямку створення інтелектуальних рішень, що здатні не лише виконувати функції замикання та розблокування, а й динамічно адаптувати свою поведінку залежно

від контексту, середовища та користувача. У цьому аспекті інтелектуальні системи замикання можна розглядати як результат конвергенції трьох базових складових: фізичних механізмів замикання, засобів цифрової обробки інформації та можливості віддаленої інтеграції через мережу.

IoT-середовище забезпечує для таких систем двосторонню комунікацію. З одного боку, пристрій передає дані про свій стан, параметри навколишнього середовища, історію подій, сигнали з датчиків; з іншого – отримує команди на зміну стану, оновлення конфігурації або навіть нову логіку поведінки. Це дозволяє користувачу або адміністратору в будь-який момент втрутитись у процес, змінити політику доступу або реагувати на загрозу, навіть перебуваючи за тисячі кілометрів від об’єкта. Застосування хмарних платформ дозволяє централізовано зберігати журнал подій, надавати різні рівні прав доступу для кількох користувачів, а також організовувати резервне копіювання конфігурації пристроїв.

Один з найважливіших моментів інтелектуальних систем замикання – це інтеграція з мобільними пристроями. Смартфон, планшет або носимий гаджет виступає не лише інтерфейсом керування, а й засобом автентифікації. Сучасні додатки дозволяють керувати замком у реальному часі, встановлювати тимчасовий або одноразовий доступ для інших користувачів, переглядати лог подій, змінювати налаштування без підключення до локальної мережі (рис. 1.3).

Рисунок 1.3 – Мобільний застосунок для керування електронним замком [10]

У поєднанні з біометричною ідентифікацією на самому пристрої, мобільний додаток формує подвійний бар'єр захисту – програмний і фізіологічний.

Не менш важливу роль відіграє вибір протоколу обміну даними. У побутових реалізаціях найчастіше застосовуються Wi-Fi та Bluetooth завдяки їхній поширеності, відносно простій реалізації та підтримці з боку мобільних платформ. У промислових та корпоративних системах перевага надається протоколам з низьким енергоспоживанням та високою масштабованістю – таких як ZigBee або LoRaWAN. Протоколи MQTT або HTTP дозволяють реалізовувати асинхронну взаємодію між пристроєм і сервером, що особливо актуально у випадках із затримками в мережі або обмеженою пропускною здатністю каналу [13].

Інтелектуальні замки також активно впроваджують алгоритми штучного інтелекту для аналізу поведінкових моделей. Наприклад, система може

навчитися розпізнавати стандартні дії користувача (час приходу, частоту використання) і на основі цього прогнозувати або блокувати аномальні дії. Такий підхід дозволяє суттєво підвищити ефективність захисту, особливо в умовах, коли класичні методи автентифікації можуть бути обійдені.

З огляду на потребу у фізичному захисті, важливим компонентом системи є датчики зворотного зв'язку. Наприклад, геркон або інфрачервоний датчик дозволяє визначити, чи були двері фізично відчинені, навіть якщо замок не отримував команди розблокування. Це дає можливість оперативно реагувати на злом або несанкціоноване втручання. У більш складних системах використовується кілька датчиків – положення, вібрації, магнітного поля – що дає змогу будувати точну модель подій.

З огляду на вищезазначене, варто відзначити, що інтелектуальні замки – це не лише технічний пристрій, а складна кіберфізична система, що працює на межі між апаратним забезпеченням, програмною логікою, мережею та людським фактором. Їх застосування є виправданим у контексті розбудови розумних будівель, корпоративної безпеки, житлової інфраструктури та індустрії готельного обслуговування. У деяких країнах такі системи вже сьогодні замінюють класичні ключі на рівні державних установ та об'єктів критичної інфраструктури.

1.4 Аналіз існуючих IoT-проєктів з контролю доступу

Інтенсивний розвиток технологій IoT та доступність мікроконтролерів, сенсорів і засобів бездротового зв'язку призвели до широкого розповсюдження проектів у сфері автоматизованого контролю доступу. Як у комерційних продуктах, так і в аматорських розробках можна простежити спільні підходи до побудови архітектури, взаємодії компонентів та організації логіки авторизації. Серед комерційних пристроїв слід виділити такі бренди, як August Smart Lock,

Nuki, Yale Linus, Ultraloq, а також AQARA Smart Lock – рішення, які вже зарекомендували себе на ринку як надійні інструменти доступу до житла або офісу. Більшість із них підтримують підключення до хмарних платформ, забезпечують багаторівневу авторизацію та легко інтегруються з іншими елементами екосистеми «розумного дому». У свою чергу, серед DIY-рішень популярності набули розробки на базі ESP8622/ESP32, що реалізують керування реле або сервоприводом через Wi-Fi, з використанням мобільних застосунків, Telegram-ботів, MQTT або навіть простих веб-серверів.

Однією з переваг власноручного проєктування є можливість повної адаптації логіки пристрою до специфічних потреб користувача. Наприклад, пристрій може бути орієнтованим виключно на авторизацію через Bluetooth (BLE), або ж поєднувати кілька методів входу – через NFC, пароль і мобільний додаток. У багатьох аматорських проєктах застосовується Arduino IDE як основне середовище розробки, а також платформи Firebase або Blynk як хмарні сервіси для зберігання та обміну даними. У більш складних рішеннях за основу береться NodeMCU з доступом до MQTT-брокера (наприклад, Mosquitto), що дозволяє реалізувати асинхронне оброблення подій у локальній або глобальній мережі [11].

Щоб краще окреслити характерні риси сучасних IoT-рішень для контролю доступу, доцільно порівняти кілька найбільш поширених варіантів – як комерційних, так і DIY. У наведеній таблиці 1.1 узагальнено функціональні можливості, переваги та обмеження кожного з підходів.

Таблиця 1.1 – Порівняння типових IoT-рішень контролю доступу

Характеристика	**August Smart Lock**	**Nuki Smart Lock 2.0**	**DIY на ESP8622 (Wi-Fi)**	**DIY на ESP32 (BLE+Wi-Fi)**
Тип живлення	Батарейки	Батарейки	Зовнішнє або акумулятор	Акумулятор або USB
Підключення до мережі	Wi-Fi, Bluetooth	Wi-Fi, Bluetooth	Wi-Fi (локальний)	Wi-Fi + BLE
Авторизація користувача	Смартфон, ключ, код	Смартфон, код	Вебінтерфейс, пароль	BLE-токен, Telegram
Хмарна інтеграція	Так (через моб. додаток)	Так (через Bridge)	Можлива через Firebase	Підтримка MQTT
Ведення журналу подій	Так	Так	Обмежене (локально)	Так (через MQTT або Blynk)
Зворотний зв’язок (стан дверей)	Так (магнітний сенсор)	Так	Так (геркон)	Так (геркон або датчик кута)
Вартість реалізації	Висока (200–300$)	Висока (180–250$)	Низька (10–20$)	Середня (15–40$)
Можливість кастомізації	Низька	Середня	Висока	Висока
Надійність у побуті	Висока	Висока	Залежить від реалізації	Залежить від реалізації
Відкритість коду/прошивки	Ні	Частково	Повна	Повна

Як видно з таблиці, комерційні пристрої переважно орієнтовані на кінцевого користувача і забезпечують високу стабільність роботи, підтримку офіційних додатків, захищене зберігання даних та постійну технічну підтримку. Водночас вони практично не дозволяють змінювати прошивку або додавати нові функції, що робить їх малопридатними для інженерних експериментів або нестандартного використання.

На противагу, DIY-рішення дозволяють повністю контролювати всі аспекти архітектури, починаючи від фізичного підключення компонентів і закінчуючи логікою авторизації. Зокрема, використання ESP8622 у поєднанні з

простим веб-сервером дозволяє створити систему, яка імітує роботу комерційного пристрою: користувач вводить пароль через браузер, а контролер, отримавши запит, активує реле, що розблокує механізм замикання. За бажанням, цю систему можна ускладнити додаванням багатофакторної авторизації, таймера автозамикання або функції тимчасового доступу [12].

Особливо привабливим для освітніх і дослідницьких цілей є варіант із застосуванням ESP32, який підтримує Bluetooth Low Energy. Це дає змогу реалізовувати авторизацію не лише через інтернет, а й через пряме з’єднання зі смартфоном без доступу до Wi-Fi, що суттєво підвищує автономність і безпеку системи. Така гібридна схема дозволяє працювати навіть у повністю офлайн-середовищі, що може бути корисно для тимчасових або мобільних об’єктів, таких як склади або будівельні майданчики.

2 ОБҐРУНТУВАННЯ АРХІТЕКТУРИ СИСТЕМИ КОНТРОЛЮ ЗАМИКАННЯ

2.1 Розробка загальної структура IoT-системи замикання

Розробка ефективної автоматизованої системи контролю замикання на основі технологій Інтернету речей вимагає не лише добору окремих функціональних елементів, а й побудови чіткої, узгодженої архітектури, яка забезпечувала б цілісність, надійність та масштабованість системи. У цьому контексті поняття «структура» охоплює як фізичну побудову пристрою, так і логіку його взаємодії з користувачем, мережевими протоколами та зовнішніми платформами.

IoT-система замикання дверей являє собою розподілений програмно-апаратний комплекс, у якому взаємодіють три основні рівні: апаратний (hardware), програмний (software) та мережевий (network). Кожен з них виконує чітко визначені функції і взаємопов’язаний з іншими через логіку системного управління.

На апаратному рівні ядром системи виступає мікроконтролер, до якого підключені основні периферійні пристрої: сервопривід або електромагнітний замок, датчики стану дверей (геркони або оптичні сенсори), індикатори стану (світлодіоди або дисплеї), а також комунікаційні модулі (Wi-Fi, Bluetooth). Саме апаратна частина забезпечує фізичну взаємодію з навколишнім середовищем та реалізує виконавчі дії – замикання, розблокування, фіксацію стану, індикацію. При цьому важливо враховувати електричні характеристики компонентів, потребу в стабільному живленні, а також енергоспоживання системи в активному та пасивному режимах.

Програмний рівень є тим, що перетворює набір електронних компонентів на цілісну логічну систему. Тут реалізується обробка подій, прийняття рішень, управління апаратними засобами, а також комунікація з мобільним застосунком чи веб-інтерфейсом. У типовому сценарії програмна логіка виконує наступні завдання: очікування команд від користувача, перевірка автентичності, керування приводом замка, фіксація події у локальному або віддаленому журналі, реагування на помилки чи виняткові ситуації. У реалізації цієї частини використовуються мови програмування низького та середнього рівня – зокрема, C/C++ у середовищі Arduino IDE або PlatformIO – а також бібліотеки для роботи з мережевими інтерфейсами, сенсорами, криптографічними алгоритмами [17].

Мережевий рівень забезпечує можливість віддаленої взаємодії з системою через Інтернет або локальну мережу. Це може бути прямий HTTP-доступ до вбудованого веб-сервера, MQTT-зв’язок з брокером повідомлень або Bluetooth-з’єднання зі смартфоном. Саме цей рівень є основою IoT-парадигми – адже без передачі даних у мережу система перетворюється на звичайний електронний замок. У разі реалізації повноцінного віддаленого доступу також необхідно вирішити питання безпеки переданих даних, уникнення перехоплення та несанкціонованого доступу до системи. Зазвичай це досягається через авторизацію користувачів, шифрування, токени доступу або VPN-з’єднання.

Із структурної точки зору систему доцільно представити у вигляді блок-схеми, в якій логічно відображено взаємозв’язки між основними функціональними модулями. Центральним елементом є мікроконтролер ESP8622, до якого підключено:

– виконавчий механізм замикання (сервопривід Tower Pro MG90S або реле для керування електромагнітним замком);

– герконовий датчик для визначення відкриття дверей;

– індикаторний світлодіод або OLED-дисплей;

– Wi-Fi-модуль (вбудований в ESP8622) для комунікації з мобільним застосунком;

– логічний блок програмного контролю, який реагує на команди та сигнали з сенсорів.

У разі розширення системи можливе підключення додаткових компонентів – наприклад, модуля зчитування NFC-тегів, біометричних сенсорів або камери з функцією розпізнавання облич. Це не змінює фундаментальної архітектури системи, однак вимагає додаткових обчислювальних ресурсів та зміни у програмному забезпеченні.

Загальна структура також передбачає існування інтерфейсу взаємодії з користувачем – ним може виступати мобільний застосунок, створений у середовищі Flutter або React Native, який з’єднується з пристроєм безпосередньо через Wi-Fi або передає команди через хмарний сервер. Мобільний застосунок повинен мати функціонал автентифікації, візуального контролю стану замка, журналу подій та можливості змінювати параметри системи (наприклад, задавати таймер автоматичного замикання, створювати тимчасові ключі тощо) (рис. 2.1).

Рисунок 2.1 – Загальна структура IoT-системи замикання

Особливістю такої структури є її масштабованість і модульність. Тобто, до вже реалізованої системи в будь-який момент можна додати нові вузли, змінити логіку або інтегрувати її з іншими пристроями – наприклад, через стандарт MQTT або за допомогою Google Firebase. Такий підхід є ідеальним для проєктів, які спочатку створюються як прототип, а в подальшому можуть бути адаптовані до комерційного застосування.

У рамках даної роботи реалізується саме така структурна модель – з використанням ESP8622 як головного контролера, серводвигуна як виконавчого елемента, геркона як сенсора стану дверей, веб-сервера як інтерфейсу керування та мобільного додатку для авторизованого доступу. Система передбачає віртуальну симуляцію у середовищі Wokwi, що дозволяє перевірити її логіку, безпеку та працездатність без фізичного стенду.

2.2 Обґрунтування використання мікроконтролера ESP8622 та пояснення його функцій

У процесі розробки системи автоматизованого контролю замикання дверей, ключовим елементом, від якого залежить як функціональність, так і масштабованість всієї архітектури, є мікроконтролер. Саме він виконує центральну роль у зборі даних з сенсорів, управлінні виконавчими механізмами, реалізації мережевих протоколів та обробці команд, що надходять від користувача через мобільний застосунок або веб-інтерфейс. Тому вибір відповідної апаратної платформи є критично важливим і має спиратися не лише на технічні характеристики, а й на сумісність з іншими компонентами системи, підтримку бібліотек, доступність симуляторів та документації.

У межах цієї роботи було обґрунтовано доцільність застосування мікроконтролера ESP8622, зокрема – його популярної модифікації NodeMCU ESP8622 v3. Даний вибір базується на ряді технічних, функціональних і практичних переваг, які роблять цю платформу ідеальною для реалізації IoT-рішень на початковому та середньому рівні складності.

Мікроконтролер ESP8622 є однокристальною системою-на-чипі (SoC), розробленою компанією Espressif Systems, що включає в себе 32-бітний процесор Tensilica L106, який працює на частоті до 80/160 МГц, модуль Wi-Fi 802.11 b/g/n, а також низку периферійних інтерфейсів, серед яких GPIO, SPI, UART, I2C, PWM та ADC. Такий набір характеристик дозволяє використовувати ESP8622 як автономний пристрій, що виконує повний цикл обробки подій – від зчитування сигналів з датчиків до формування HTTP/MQTT-запитів (рис. 2.2) [13].

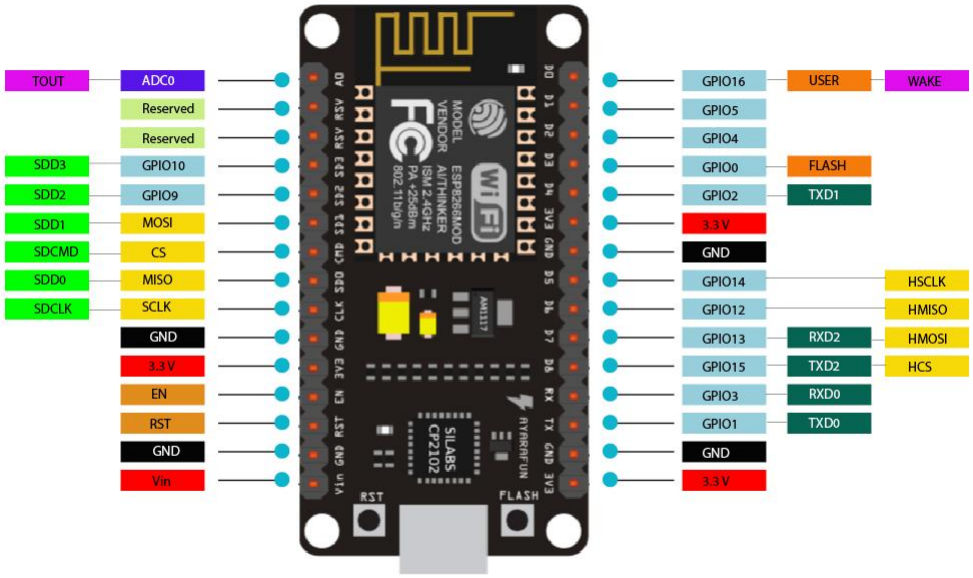

Рисунок 2.2 – NodeMCU ESP8622 v3

Основна причина вибору ESP8622 – це його вбудований Wi-Fi-модуль, який дозволяє організувати як підключення до локальної мережі, так і створення власної точки доступу (Access Point). Це особливо актуально для розробки системи замикання, оскільки саме Wi-Fi є основним каналом зв’язку між пристроєм і мобільним застосунком. Крім того, ESP8622 підтримує створення простого веб-сервера, на якому можна реалізувати базовий інтерфейс керування замком або систему автентифікації.

Перевагою NodeMCU-модифікації є наявність інтегрованого USB-UART перетворювача, що значно спрощує прошивку та налагодження пристрою, а також доступ до всіх цифрових пінів через зручну гребінку. Це робить платформу зручною для розробників, які використовують Arduino IDE або середовище PlatformIO. Завдяки відкритому SDK та широкій підтримці з боку спільноти, розробка під ESP8622 є стабільною, добре задокументованою та гнучкою.

Ще одним важливим фактором є енергоефективність. ESP8622 підтримує кілька режимів енергозбереження, зокрема Deep Sleep, що дозволяє переводити мікроконтролер у стан мінімального споживання енергії в періоди

бездіяльності. Це особливо важливо для автономних систем замикання, які живляться від батарейок або акумуляторів. Завдяки низькому енергоспоживанню у режимі сну та швидкому пробудженню пристрій здатен забезпечити тривалий період автономної роботи без заміни джерела живлення.

Функціонально ESP8622 здатен обслуговувати як прості логічні задачі (наприклад, активацію реле або сервоприводу при надходженні запиту), так і більш складні сценарії, включаючи шифрування даних, взаємодію з базами даних через інтернет або комунікацію з хмарними платформами. Крім того, мікроконтролер може використовуватись для збору та обробки даних з різних сенсорів – наприклад, герконів, термодатчиків, інфрачервоних або ультразвукових сенсорів наближення.

З точки зору віртуального тестування, ESP8622 не підтримується середовищем Wokwi – симуляційною платформою для Arduino-проєктів, яка дозволяє перевіряти працездатність прошивки, взаємодію з периферією, логіку обробки подій, як і інші середовища. Але повністю підтримує ESP32, через яку і було проведене тестування схеми. Цей інструмент дозволяє значно скоротити цикл розробки та мінімізувати ризики апаратних помилок [16, 18].

Порівняно з альтернативами (такими як ESP32, STM32 або Raspberry Pi Pico W), ESP8622 виграє за параметрами ціна-функціональність для завдань середнього рівня складності. Наприклад, ESP32 має більшу обчислювальну потужність та Bluetooth-модуль, але у більшості сценаріїв базового контролю доступу ці ресурси залишаються незатребуваними. До того ж, ESP8622 має менше споживання енергії та більш просту архітектуру, що знижує вимоги до обслуговування та налагодження системи.

2.3 Аналіз і вибір способів зв’язку

У системах Інтернету речей, зокрема в автоматизованих системах контролю замикання дверей, вибір технології зв’язку є критично важливим етапом проєктування. Саме від способу комунікації між користувачем і виконавчим пристроєм залежить як зручність взаємодії, так і безпека, енергоефективність та швидкодія системи. Найпоширенішими бездротовими технологіями у сфері IoT залишаються Wi-Fi та Bluetooth, кожна з яких має свої переваги, недоліки та сценарії використання.

У контексті даної розробки, яка передбачає як локальну взаємодію з мобільним додатком, так і можливість віддаленого доступу до системи замикання, доцільно розглянути гібридну архітектуру, що комбінує обидві технології зв’язку.

WiFi дозволяє двом пристроям взаємодіяти один з одним за допомогою радіохвиль (RF). Найчастіше використовується для підключення інтернет-маршрутизаторів до пристроїв або для з’єднання двох апаратних компонентів. Обидва радіодіапазони UHF 2.4 ГГц і 5 ГГц SHF ISM доступні для використання з WiFi. У мікроконтролері ESP8622 Wi-Fi-модуль вбудований безпосередньо в кристал, що дозволяє економити місце на платі та не потребує зовнішніх компонентів. Завдяки цьому з’являється можливість організовувати веб-сервер на самому мікроконтролері та отримувати HTTP-запити від мобільного застосунку або браузера.

Діапазон мереж WiFi, як правило, набагато ширший, охоплюючи цілі будинки, офіси або навіть великі відкриті території в деяких випадках. Це розширене покриття робить його ідеальним для середовищ, де користувачі багато рухаються, гарантуючи, що вони залишаються на зв’язку.

Ще однією перевагою є сумісність з мобільними платформами, адже всі сучасні смартфони мають вбудований Wi-Fi-модуль, а підключення до

локального веб-інтерфейсу не вимагає встановлення додаткових драйверів. Крім того, Wi-Fi підтримує передачу значного обсягу даних, що дозволяє, у разі необхідності, реалізовувати додаткові функції – наприклад, передачу відео з камери або логів подій.

До недоліків можна віднести вищий рівень енергоспоживання у порівнянні з Bluetooth, а також більший час виходу на зв’язок після пробудження пристрою. У автономних пристроях, що живляться від акумуляторів, це може стати критичним фактором. Також Wi-Fi є більш вразливим до атак типу MITM (людина посередині), якщо не застосовується додаткове шифрування або автентифікація.

Bluetooth, зокрема його енергоощадна модифікація Bluetooth Low Energy (BLE), є альтернативним способом зв’язку, який також активно використовується в IoT-проєктах. BLE забезпечує надійне точкове з’єднання на коротких відстанях (до 10 метрів), що робить його ідеальним для застосувань, де користувач фізично перебуває біля дверей.

Крім того, BLE дозволяє створити модель так званого «безключового доступу», коли користувач підходить до дверей зі смартфоном, і замок автоматично розблоковується, якщо виявлено авторизований пристрій у зоні дії. Така функціональність часто застосовується в сучасних автомобілях і «розумних» будинках.

До обмежень Bluetooth можна віднести необхідність використання ESP32 замість ESP8622, адже останній не має вбудованого BLE-модуля. Це створює потребу в переході на іншу мікроконтролерну платформу, яка хоч і потужніша, але споживає більше енергії та є дещо складнішою у налаштуванні. Крім того, з’єднання по Bluetooth зазвичай не дозволяє реалізувати віддалений контроль, якщо не використовуються додаткові проксі-механізми або шлюзи.

Беручи до уваги цілі даного проєкту, було прийнято рішення зосередитися на Wi-Fi-зв’язку як основному каналі взаємодії між мобільним

застосунком та системою замикання. Це забезпечує простоту реалізації, сумісність з ESP8622, можливість симуляції у Wokwi, а також зручність налагодження та тестування. Крім того, Wi-Fi дозволяє підключати пристрій як до локальної мережі, так і створювати режим точки доступу, що відкриває більшу гнучкість у сценаріях використання.

Водночас, у перспективі, розглядається можливість розширення системи за рахунок BLE-компонентів на основі ESP32. Це дозволить впровадити додаткові сценарії взаємодії з користувачем, зокрема безконтактне розблокування або обмежений локальний доступ без підключення до Інтернету. Такий підхід забезпечує масштабованість проєкту та створює заділ на майбутнє вдосконалення системи.

Таким чином, обраний спосіб зв’язку – Wi-Fi – є найраціональнішим з точки зору функціональності, доступності, сумісності з обраним мікроконтролером і можливістю повноцінної симуляції в навчальних умовах. Він забезпечує достатній баланс між зручністю, енергоефективністю та безпекою для реалізації задач, поставлених у дипломному проєкті.

2.4 Аналіз компонентів апаратної частини

Апаратна частина системи замикання дверей є основою її фізичного функціонування. Саме електронні та електромеханічні компоненти забезпечують виконання дій, без яких неможливо досягти цілей контролю доступу. У рамках цієї дипломної роботи реалізовано компактну, модульну конфігурацію, яка легко відтворюється в симуляційних середовищах і може масштабуватись до реального прототипу.

Кожен компонент виконує певну роль у загальному ланцюгу функціонування:

– сервопривід або реле виконує функцію механічного або електромагнітного блокування;

– геркон або датчик положення забезпечує фіксацію стану дверей;

– світлодіоди або OLED-дисплей надають зворотній зв’язок користувачу;

– кнопки або сенсори можуть слугувати резервним способом локального керування.

Залежно від конкретної моделі системи, набір компонентів може варіюватися, але в основі залишається саме ця функціональна тріада: виконавчий елемент – сенсор – контролер.

Для фізичного замикання використовується стандартний сервопривід Tower Pro MG90S, який є оптимальним вибором для демонстраційних прототипів. Він працює на 5В, має низьке енергоспоживання і забезпечує поворот на 180°, що дозволяє механічно зафіксувати або відкрити блокіратор.

Особливістю Tower Pro MG90S є простота підключення до ESP8622: він керується за допомогою одного PWM-піна, не потребує складної калібровки та легко тестується в симуляторах на кшталт Wokwi. Для забезпечення надійної роботи необхідно використовувати зовнішнє джерело живлення, адже живлення від USB може бути недостатнім при максимальному навантаженні [14].

У варіантах реального впровадження сервопривід може бути замінений на електромагнітне реле, яке керує замком постійного струму або механічним засувом.

Для визначення факту відчинення або зачинення дверей у системі застосовується герконовий датчик – замкнений або розімкнений контакт, який спрацьовує при наближенні магнітного поля. Цей датчик не вимагає живлення, має високу надійність і не потребує складного кріплення. Стан геркона фіксується мікроконтролером через цифровий вхід і використовується для

журналювання подій, запобігання помилковому блокуванню та зворотного контролю.

В альтернативних варіантах можна використовувати інфрачервоні або ультразвукові сенсори наближення, однак вони значно складніші в реалізації та мають більший рівень споживання енергії.

Світлодіоди використовуються як прості індикатори стану системи: червоний – замкнено, зелений – розблоковано, жовтий – помилка. У розширеному варіанті застосовується OLED-дисплей 0.96" на контролері SSD1306, який відображає поточний стан, IP-адресу пристрою, час або результати автентифікації [15].

Крім цього, у системі можуть використовуватися кнопки або сенсорні панелі для ручного резервного керування, а також зумери для подачі акустичних сигналів. У таблиці 2.1 наведено порівняння основних компонентів.

Таблиця 2.1 – Огляд електронних компонентів для управління доступом

Компонент	**Модель / Тип**	**Функція**	**Переваги**	**Обмеження**
Сервопривід	Tower Pro MG90S	Механічне блокування	Легкість керування, дешевизна	Обмежена сила тяги, малий ресурс
Сенсор дверей	Геркон	Виявлення відкриття	Висока надійність, без живлення	Може помилково спрацювати від удару
Індикація	LED / OLED SSD1306	Відображення стану	Простота підключення, універсальність	Займає GPIO, потребує бібліотек
Реле (альтернатива)	SRD-05VDC-SL-C	Керування замком пост. струму	Сумісність із більшістю замків	Потребує транзистора і діода

Таким чином, апаратна частина системи побудована на принципах доступності, надійності та сумісності з ESP8622. Усі компоненти підібрано з урахуванням можливості симуляції, низького енергоспоживання, простоти

підключення та підтримки на рівні бібліотек Arduino. Такий підхід дозволяє забезпечити стабільну та передбачувану роботу всієї системи контролю замикання дверей.

2.5 Вибір середовища розробки, бібліотек і протоколів обміну

Ефективна реалізація системи замикання дверей на основі IoT залежить не лише від апаратного забезпечення, а й від обраного програмного інструментарію. Правильно підібране середовище розробки, оптимальні бібліотеки та надійні протоколи обміну даними істотно впливають на стабільність, безпеку та функціональну гнучкість проєкту. У цьому підрозділі обґрунтовується вибір відповідних інструментів, які використовуються при розробці програмного забезпечення мікроконтролера, мобільного додатку та для симуляції роботи системи.

Основним середовищем програмування для прошивки ESP8622 у цій роботі обрано VSCode з додатком PlatformIO [13]. Це популярна, кросплатформна інтегрована оболонка, яка підтримує як базові, так і розширені можливості розробки під мікроконтролери. Arduino IDE має такі переваги:

- простота інтерфейсу та налаштувань;
- велика спільнота користувачів;
- підтримка великої кількості бібліотек для периферійних пристроїв;
- безпосередня сумісність із платами ESP8622, завдяки встановленню плати через менеджер.

Для симуляції роботи системи було використано онлайн-платформу Wokwi, яка дозволяє моделювати взаємодію ESP32 із сервоприводами, герконами, дисплеями та іншими елементами (рис. 2.3). Особливість Wokwi – у можливості безперервного оновлення прошивки, дебагінгу в реальному часі та перегляду даних з UART-консолі без необхідності підключення фізичної плати.

Рисунок 2.3 – Головна сторінка онлайн-платформи Wokwi

Для розширених потреб (у перспективі проєкту) розглядається також середовище PlatformIO – особливо в разі переходу на ESP32 чи потреби інтеграції з Git, CI/CD або розширеними інструментами кодування.

Проєкт використовує набір відкритих бібліотек, які забезпечують швидку реалізацію базових функцій пристрою. Серед них:

– ESP8622WiFi.h – бібліотека для підключення до мережі Wi-Fi та створення точки доступу або клієнта.

– ESP8622WebServer.h – для запуску локального веб-сервера на мікроконтролері, що дозволяє передавати команди через HTTP-запити.

– Servo.h – стандартна бібліотека для управління сервоприводом.

– Adafruit_SSD1306.h, Adafruit_GFX.h – для роботи з OLED-дисплеєм у разі використання графічного інтерфейсу.

– ArduinoJson.h – для обробки запитів і відповідей у форматі JSON, зокрема під час взаємодії з мобільним додатком.

– Також активно застосовуються функції delay, millis, digitalRead, analogRead та digitalWrite для організації логіки пристрою.

Комунікація між мобільним додатком і мікроконтролером реалізується за допомогою протоколу HTTP, який забезпечує просту реалізацію REST-запитів у межах локальної мережі. В контексті ESP8622, HTTP дозволяє використовувати веб-браузер або мобільний застосунок як інтерфейс керування. У відповідь на GET або POST запити, контролер повертає JSON-повідомлення або HTML-сторінку зі станом замка.

У перспективі, система може бути розширена з використанням MQTT – легкого брокерного протоколу, що оптимально підходить для IoT. MQTT дозволяє реалізувати push-сповіщення, а також забезпечує мінімальний обсяг трафіку в бездротових мережах. Для таких задач, як сповіщення про події, обробка станів або ведення журналу, MQTT є більш ефективним, ніж HTTP.

Для шифрування трафіку при потребі може використовуватись HTTPS (через бібліотеку WiFiClientSecure), однак це ускладнює налаштування та збільшує споживання пам’яті пристрою.

Для створення мобільного інтерфейсу системи обрано фреймворк Flutter, який дозволяє створювати кросплатформні додатки на одній кодовій базі для Android та iOS. Flutter забезпечує високу швидкодію, сучасний інтерфейс користувача та можливість інтеграції з HTTP-запитами та WebSocket [17].

Для взаємодії із системою замикання Flutter-додаток використовує:

– плагін http для надсилання запитів до ESP8622;
– плагін shared_preferences для збереження токенів автентифікації;
– плагін flutter_blue (у перспективі) для BLE-взаємодії з ESP32.

Інтерфейс передбачає:

– вхід з автентифікацією;
– кнопки «Замкнути / Відчинити»;
– індикатор стану замка;
– сповіщення про помилки або збої з’єднання.

Сукупність обраних інструментів – Arduino IDE, бібліотеки ESP8622WiFi/WebServer, а також протоколи HTTP/JSON – забезпечують швидку, стабільну і масштабовану розробку системи замикання дверей. Обраний стек технологій відповідає вимогам до універсальності, доступності ресурсів, спрощення симуляції та можливості подальшого розширення.

3 РОЗРОБКА МОБІЛЬНОГО ДОДАТКА ДЛЯ КЕРУВАННЯ СИСТЕМОЮ

3.1 Вимоги до функціональності мобільного застосунку

У системі автоматизованого контролю замикання дверей важливою складовою є програмний інтерфейс, через який користувач отримує змогу взаємодіяти з виконавчими пристроями. У рамках цієї дипломної роботи передбачається реалізація мобільного застосунку, який виступає центральним засобом доступу до функціоналу IoT-системи. Такий застосунок не лише забезпечує керування процесом замикання та відмикання дверей, а й виконує функції автентифікації, реєстрації користувачів, відображення поточного стану системи та обробки зворотного зв’язку. Формування чітких функціональних вимог до нього є критичним кроком у процесі розробки, оскільки від цього залежить логіка архітектури, вибір фреймворку та технічна реалізація взаємодії з мікроконтролером.

Функціональність мобільного додатку повинна охоплювати як базові дії, що безпосередньо впливають на стан замка, так і допоміжні механізми, які гарантують безпеку та стабільність обміну даними. Ключовою функцією є можливість безперешкодного керування сервоприводом або реле, яке відповідає за механізм замикання. У цьому контексті мобільний застосунок повинен формувати HTTP-запити, що надсилаються безпосередньо до IP-адреси мікроконтролера ESP8622 у локальній мережі або через точку доступу. Відповідно, інтерфейс має містити чітко позначені елементи керування, які дозволяють ініціювати зміну стану замка, з можливістю миттєвого оновлення візуального індикатора його поточного положення.

Не менш важливою вимогою є наявність механізму автентифікації, що дає змогу обмежити доступ до керування пристроєм лише авторизованим користувачам. При початковому запуску мобільного додатку користувач має пройти процедуру реєстрації або входу, після чого доступ до функцій розблокування чи блокування надається лише за умови успішної ідентифікації. У цьому проєкті застосовується локальне збереження даних за допомогою стандартних засобів зберігання (наприклад, SharedPreferences або SecureStorage), що дозволяє підтримувати сесію навіть після перезапуску програми.

Ще одним важливим елементом функціоналу є можливість перегляду статусу пристрою в реальному часі. Мобільний застосунок повинен відображати інформацію про поточний стан дверей: замкнено, відчинено або не вдається з’єднатися з пристроєм. Для цього реалізується періодичний запит до мікроконтролера, відповідь на який аналізується для оновлення графічного інтерфейсу. Важливо, щоб інтерфейс користувача залишався інтуїтивно зрозумілим, незалежно від технічної підготовки кінцевого користувача, і не перевантажувався зайвими опціями.

Враховуючи можливість використання кількох пристроїв одночасно (наприклад, у випадку багатоквартирного будинку чи офісу), застосунок повинен забезпечувати конфігурацію кількох точок доступу. Користувач повинен мати змогу додати новий пристрій до переліку, вказавши його ідентифікатор або IP-адресу, а також зберегти відповідні налаштування для подальшого використання.

Крім цього, мобільний застосунок повинен мати механізм повідомлень про помилки – зокрема, у випадках втрати зв’язку, неправильного паролю, спроби повторного відкриття вже відкритого замка тощо. Такі повідомлення повинні виводитися в формі спливаючих інтерфейсних елементів або системних повідомлень.

Особлива увага приділяється питанням інформаційної безпеки. Передача даних між мобільним застосунком і мікроконтролером має здійснюватися в захищеному форматі, а всі критичні операції, що змінюють стан замка, повинні бути підтверджені з боку користувача, щоб уникнути випадкових натискань.

З урахуванням наведеного, мобільний застосунок має реалізовувати такі функції:

– керування замиканням/відмиканням дверей через Wi-Fi;

– автентифікацію користувача при запуску;

– відображення поточного стану системи (замкнено/відчинено);

– конфігурацію кількох пристроїв у локальній мережі;

– повідомлення про помилки або збої зв’язку;

– захищене збереження сесійної інформації.

3.2 Порівняння нативної та кросплатформна розробки

У процесі створення мобільного застосунку для IoT-системи замикання дверей важливим є обґрунтований вибір архітектурного підходу до розробки клієнтської частини. Існує два базових підходи: нативна розробка для кожної платформи окремо (Android та iOS) та кросплатформна розробка, яка дозволяє створити єдину кодову базу для обох платформ. Рішення, яке буде ухвалено на цьому етапі, істотно впливає не лише на продуктивність застосунку, а й на його масштабованість, час розробки, легкість підтримки та сумісність з пристроєм ESP8622 [18].

Нативна розробка передбачає використання окремих технологій для кожної операційної системи: для Android це Java або Kotlin у поєднанні з Android SDK, для iOS – Swift або Objective-C із використанням Xcode. Такий підхід гарантує найвищий рівень оптимізації, повний доступ до системних API та можливість реалізувати складну анімацію, глибоку інтеграцію з апаратними

функціями смартфона, а також адаптацію під вимоги маркетплейсів. Однак недоліком є необхідність мати окремі команди або хоча б окрему гілку коду для кожної платформи. Це збільшує витрати на розробку, ускладнює тестування та подовжує цикл впровадження змін.

Кросплатформна розробка, своєю чергою, дозволяє створити універсальний застосунок, який працює одразу на кількох платформах, базуючись на єдиній кодовій базі. Серед популярних рішень у цьому напрямі варто згадати React Native, Xamarin, а також Flutter. Основна перевага такого підходу – скорочення часу розробки та спрощення процесу обслуговування програмного коду. Оновлення, оптимізація чи виправлення помилок здійснюється одночасно для всіх цільових платформ, що є суттєвим плюсом при роботі в обмежених часових рамках, наприклад, у межах дипломного проєкту.

З технічного погляду, кросплатформна модель дозволяє достатньо ефективно працювати з API пристроїв, надсилати HTTP-запити, обробляти відповіді в JSON-форматі та будувати інтерфейс, який не поступається нативним рішенням. Це цілком задовольняє потреби даного проєкту, в якому застосунок виступає як клієнтський інтерфейс до локального пристрою ESP8622. При цьому відсутність надмірної графічної складності або необхідності використання специфічного функціоналу смартфона (наприклад, NFC чи AR) робить кросплатформний варіант повністю доцільним.

Варто також врахувати, що сучасні кросплатформні фреймворки мають активну спільноту, безперервно розвиваються і підтримують плагіни для реалізації більшості типових завдань – від Bluetooth-з’єднання до керування локальними базами даних або безпечним зберіганням токенів. Таким чином, навіть у випадку масштабування проєкту або інтеграції з хмарними сервісами, обраний підхід не буде обмеженням.

3.3 Обґрунтування вибору Flutter як основного фреймворку

Вибір технологічного стеку для реалізації мобільного застосунку відіграє ключову роль у забезпеченні його надійності, продуктивності та відповідності функціональним вимогам. Після порівняння нативного і кросплатформного підходів, перевагу було надано саме кросплатформному рішенню, що дозволяє створити універсальний застосунок із єдиною кодовою базою. Серед доступних фреймворків особливу увагу було зосереджено на Flutter – технології, розробленій компанією Google, яка зарекомендувала себе як гнучкий, швидкий і стабільний інструмент для створення мобільних інтерфейсів.

Однією з основних переваг Flutter є його архітектура, побудована на використанні власного графічного рушія Skia, що дозволяє відображати інтерфейс напряму, без посередників із боку нативних компонентів платформи. Завдяки цьому мобільні застосунки, створені з використанням Flutter, демонструють високу швидкодію, стабільність і мінімальну затримку при анімації, навіть на пристроях із середніми характеристиками. Для задач проєкту, де графічна насиченість не є критичною, але стабільна робота та плавна взаємодія з користувачем важливі – це вагомий аргумент.

Крім технічної продуктивності, Flutter пропонує зручну систему відображення інтерфейсу через декларативну модель. Інтерфейс визначається у вигляді деревовидної структури віджетів, кожен із яких відповідає за певний елемент UI. Такий підхід дозволяє гнучко змінювати структуру екранів, легко адаптувати застосунок під різні роздільні здатності та орієнтації екранів. У контексті системи замикання, де необхідно реалізувати інтуїтивно зрозумілі елементи керування – наприклад, кнопки для замикання та відмикання, індикатори стану пристрою, поля автентифікації – це значно спрощує роботу з інтерфейсом.

Ще одним важливим фактором є широка підтримка мережевих функцій та плагінів. Flutter має вбудовану підтримку бібліотек для надсилання HTTP-запитів, обробки форматів JSON, збереження даних локально та інтеграції з API платформ. Зокрема, для взаємодії з ESP8622 достатньо використати базовий плагін http, який забезпечує реалізацію REST-запитів, що повністю відповідає логіці побудови зв’язку в цій роботі. Крім того, Flutter дозволяє використовувати плагіни provider, shared_preferences, secure_storage та інші для реалізації збереження сесій, обробки статусу автентифікації та логіки переходів між екранами.

Слід також відзначити, що Flutter підтримує можливість побудови прогресивних веб-застосунків (PWA) та десктопних додатків, що, хоча і не входить до безпосередньої задачі дипломної роботи, створює передумови для подальшого розширення функціональності системи. У перспективі це дозволяє створити універсальне середовище керування замками не лише зі смартфона, а й через браузер чи окрему комп’ютерну програму.

Важливим аргументом на користь Flutter є також його активна спільнота розробників, широка документація та наявність великої кількості готових рішень у відкритому доступі. Для дипломного проєкту, який виконується в обмежені терміни, це дозволяє значно прискорити процес розробки та зосередитися на реалізації прикладної логіки, а не на низькорівневому програмуванні чи адаптації під конкретну ОС.

3.4 Архітектура мобільного застосунку

Архітектура мобільного застосунку для керування IoT-системою замикання дверей визначає логіку організації програмного коду, взаємозв’язок між екранами, компонентами інтерфейсу, сервісами обміну даними та модулями автентифікації. У межах цієї дипломної роботи застосовується

архітектурна модель типу MVVM (Model–View–ViewModel), яка є типовою для застосунків на базі Flutter і дозволяє відокремити бізнес-логіку від інтерфейсу.

На верхньому рівні структура застосунку поділяється на дві великі гілки: автентифікаційний модуль (login/registration) і модуль основного керування замком. Кожен модуль представлений окремим екраном (screen), які пов’язані між собою за допомогою навігаційної системи Navigator.

На початку роботи застосунок завантажує початковий екран – SplashScreen, на якому користувач бачить екран входу та реєстрації (рис. 3.1).

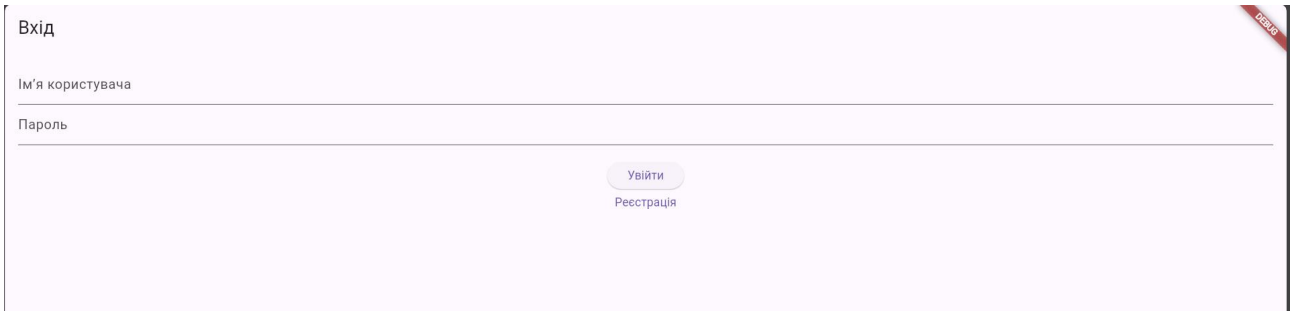

Рисунок 3.1 – Головне меню Flutter-додатку

Після авторизації користувач потрапляє до головного екрану (HomeScreen), де відображається основна інформація про пристрій: стан замка (відчинено / замкнено), індикатор з’єднання, кнопки «Відчинити» і «Замкнути». Весь функціонал реалізовано через звернення до відповідного API мікроконтролера за допомогою HTTP-запитів. У разі зміни стану пристрою користувач отримує візуальне підтвердження, а також можливість оновити дані вручну (рис. 3.2).

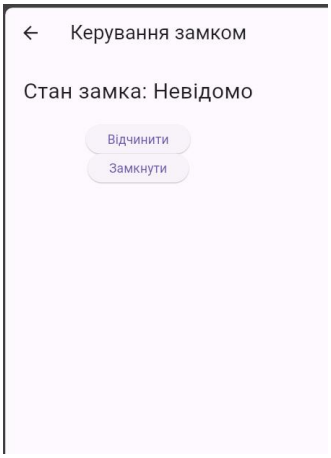

Рисунок 3.2 – Знімок головного екрану застосунку з кнопками керування та статусом замка

Комунікація між застосунком і мікроконтролером відбувається через LockService, який є частиною ViewModel. Він відповідає за логіку відправки запитів, обробку відповіді, обробку помилок та оновлення інтерфейсу через Provider або setState. Через обмежені можливості Wokwi(не підтримує ESP8622) та ESP32, стан замка завжди буде невідомим, бо немає з’єднання додатка з платою.

Для організації внутрішнього стану застосунку застосовується ChangeNotifier, що дозволяє підписаним віджетам реагувати на зміну даних у ViewModel. Завдяки цьому, наприклад, зміна статусу замка миттєво оновлює інтерфейс, не потребуючи ручного перезавантаження екрана.

Система автентифікації побудована за принципом збереження токена в SharedPreferences після успішного входу, що дозволяє зберігати авторизацію навіть після перезапуску додатку. Паролі шифруються, а серверна перевірка (для локального симулятора – умовна) здійснюється через HTTP POST-запити (рис. 3.3 – рис. 3.4).

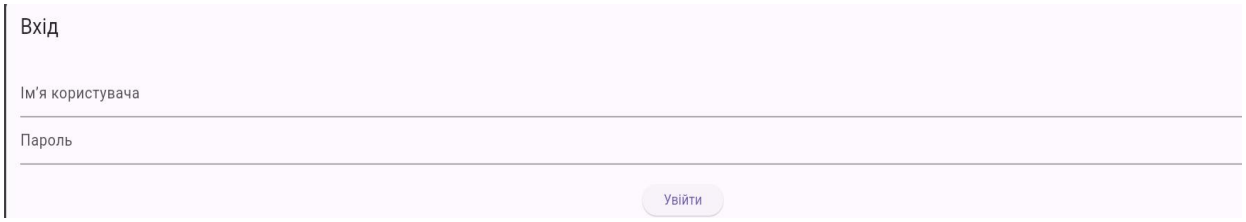

Рисунок 3.3 – Інтерфейс форми входу з полями email та password

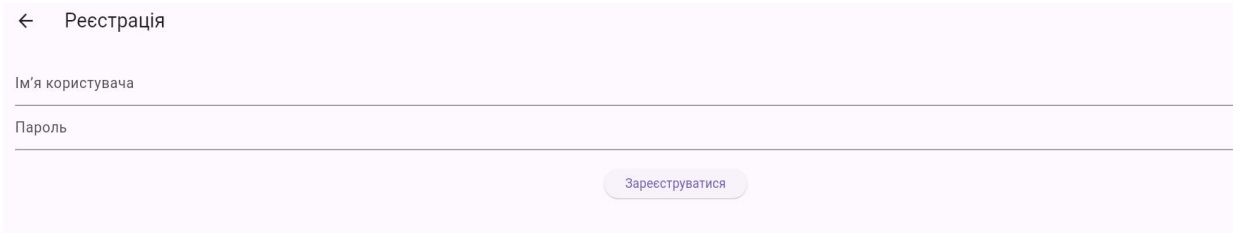

Рисунок 3.4 – Форма реєстрації нового користувача (поля, перевірка, кнопка реєстрації)

Додаток також дозволяє додавати кілька пристроїв замикання, які зберігаються у внутрішньому списку. Користувач обирає поточний пристрій зі списку, після чого основний екран оновлюється відповідно до адреси обраного ESP8622 (рис. 3.5 – рис. 3.6).

Рисунок 3.5 – Інтерфейс додавання нового пристрою замикання із вказанням IP-адреси

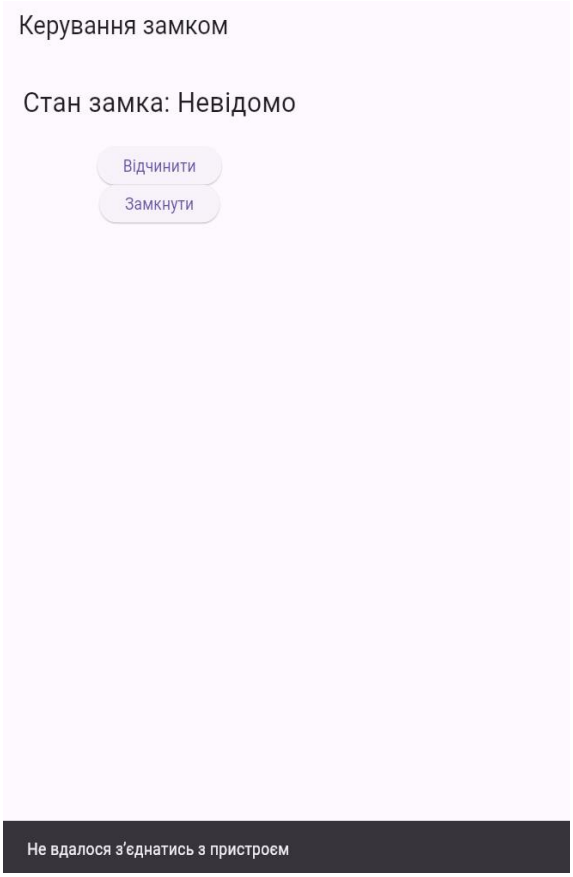

Рисунок 3.6 – Загальний вигляд застосунку на смартфоні після невдалого підключення до ESP8622

Таким чином, архітектура мобільного застосунку є чітко структурованою, розділеною на незалежні логічні компоненти, кожен із яких відповідає за свою частину функціоналу. Такий підхід спрощує тестування, підтримку та масштабування проєкту, а також дозволяє адаптувати його для майбутньої інтеграції з новими пристроями або мережевими протоколами.

3.5 Симуляція взаємодії з ESP8622 без фізичного пристрою

Основним середовищем моделювання було обрано Wokwi – онлайн-платформу для емуляції мікроконтролерів, яка не забезпечує повну підтримку ESP8622, навідміну від ESP32 за допомогою якої і перевірялась схема, дозволяє

додавати периферійні компоненти (реле, світлодіоди, серводвигуни), а також запускати Arduino-скетчі з можливістю спостерігати їхню поведінку в реальному часі. Це середовище також забезпечує вивід даних у серійну консоль, симуляцію таймерів, пінів,а також візуальне представлення підключень на макетній платі. Завдяки цьому стало можливим тестування без необхідності фізичних пристроїв чи додаткових інструментів.

Для забезпечення зв’язку з мобільним застосунком було змодельовано веб-сервер на основі бібліотеки ESP8622WebServer. У середовищі Wokwi ESP8622 працювала б в режимі точки доступу, або ж з імітованим Wi-Fi-з’єднанням (режим STA), що дозволяє обробляти запити типу GET /lock або GET /unlock. У відповідь на запит сервер повертає просту відповідь у форматі text/plain, яка фіксується застосунком Таким чином, структура команд була збережена максимально простою для забезпечення стабільної інтеграції з Flutter-застосунком. Це буде працювати з платою ESP8622, але не буде з ESP32, через що мобільний додаток не зможе підключитись до плати та показати робостоспособність напряму

З боку мобільного застосунку взаємодія повинна відбуваєтись через стандартні HTTP-запити, реалізовані за допомогою пакету http. Після чого ці запити оброблюються

4 СИМУЛЯЦІЯ ТА ТЕСТУВАННЯ РОБОТИ СИСТЕМИ

4.1 Створення віртуального середовища в Wokwi для тестування

З огляду на відсутність фізичного макету пристрою, однією з ключових задач у рамках дипломної роботи стало моделювання IoT-системи керування замиканням дверей у спеціалізованому онлайн-середовищі. Для цього було обрано Wokwi – хмарну платформу.

Створене віртуальне середовище у Wokwi повністю відтворює архітектуру реального пристрою: плата ESP32 підключена до реле (або сервоприводу), яке виконує функцію виконавчого механізму для замикання чи відмикання дверей. Додатково в макеті передбачено використання світлодіодного індикатора для візуального відображення поточного стану замка (рис. 4.1 – 4.2).

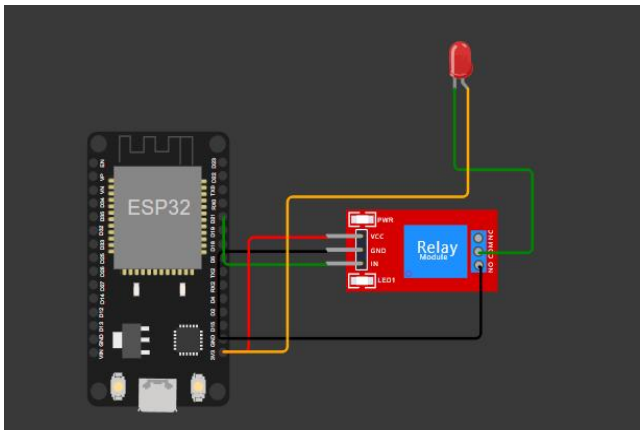

Рисунок 4.1 – Загальна схема взаємодії ESP32, реле та світлодіода у Wokwi

```
void loop() {
  if (Serial.available()) {
    command = Serial.readStringUntil('\n');
    command.trim();

    if (command == "/lock") {
      locked = true;
      digitalWrite(relayPin, HIGH);
      Serial.println("Замок заблоковано");
    } else if (command == "/unlock") {
      locked = false;
      digitalWrite(relayPin, LOW);
      Serial.println("Замок відчинено");
    } else if (command == "/status") {
      Serial.println(locked ? "Статус: заблоковано" : "Статус: відчинено");
    } else {
      Serial.println("Невідома команда");
    }
  }
```

Рисунок 4.2 – Фрагмент коду Arduino C++ з обробкою запитів /lock і /unlock

У результаті моделювання в середовищі Wokwi вдалося досягти повноцінної симуляції системи без жодного апаратного компонента, що дозволило значно пришвидшити етапи розробки, перевірки та усунення помилок. Отримані результати підтвердили коректність обраної архітектури та відповідність функціональним вимогам.

4.2 Тестування сценаріїв замикання/відмикання дверей

Звертаючи увагу на рисунок 4.3 бачимо, що у заблокованому стані вмикається світлодіод. При виконанні команди /unlock світлодіод вимикається, що означає вдалу обробку команди на відкривання дверей (рис. 4.4). За допомогою команди /status користувач може дізнатись про статус системи у реальному часі (рис. 4.3 – рис. 4.4).

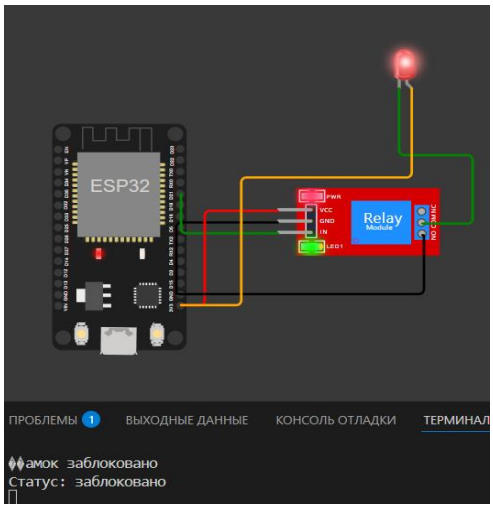

Рисунок 4.3 – Вивід повідомлень у серійну консоль під час симуляції команди /lock та підтвердження статусу командою /status

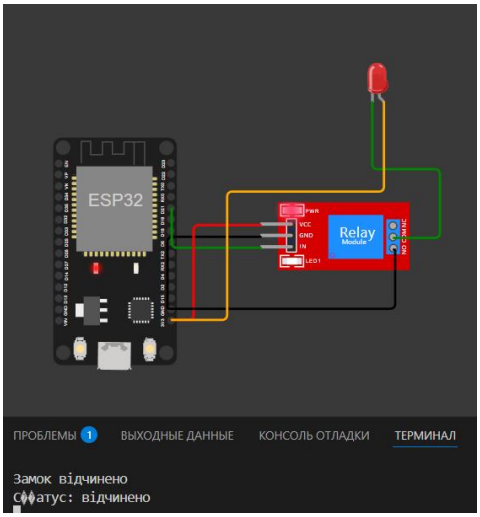

Рисунок 4.4 – Вивід повідомлень у серійну консоль під час симуляції команди /unlock та підтвердження статусу командою /status

4.3 Обмеження розробки та шляхи майбутнього вдосконалення

Через всі можливі обмеження емуляторів розроблена система віртуального контролю замикання дверей не демонструє повноцінну функціональність, варто визнати, що в межах дипломного проєкту без фізичної моделі повність емуулювати систему разом із з’єднанням до додатку неможливо. Ці обмеження є типовими для академічних розробок, що створюються без безпосереднього доступу до матеріально-технічної бази, однак їх врахування дозволяє краще оцінити потенціал подальшого вдосконалення системи.

Найбільш очевидним обмеженням стало використання симуляторів замість фізичних пристроїв: відсутність підтримки ESP8622, відсутність можливості під’єднуватись та создавати IP-адресси, котрі будуть працювати. Жоден емулятор не здатен повністю передати особливості роботи у реальному середовищі: електромагнітні перешкоди, проблеми з живленням, розбіжності у швидкості реакції тощо. У зв’язку з цим фактична поведінка системи при фізичній реалізації може потребувати додаткових калібрувань і перевірок.

Іншим аспектом є обмеження в безпеці, які у цьому проєкті були реалізовані на базовому рівні. Зокрема, автентифікація користувача відбувалася локально на стороні застосунку без використання шифрування або захищених каналів передавання даних. У контексті реального впровадження така схема не гарантує достатнього рівня захисту від несанкціонованого доступу, перехоплення команд чи атак типу «man-in-the-middle». Надалі доцільно впровадити SSL/HTTPS-з’єднання, систему токенів доступу та централізоване зберігання облікових даних з використанням безпечних протоколів.

Також обмеженим виявився функціонал самої системи. У поточному вигляді вона підтримує лише один тип пристрою та працює з фіксованою IP-адресою, що робить розгортання у більших інсталяціях менш зручним.

Розширення на багатокористувацьке середовище або інтеграція з декількома замками потребує реалізації системи керування конфігураціями, реєстрації пристроїв та їхньої взаємодії у спільній мережі. Застосунок також не має підтримки Bluetooth-режиму, попри початкову концепцію, оскільки цей функціонал потребував би глибшого переписування логіки Flutter-застосунку з використанням додаткових плагінів та API.

Нарешті, відсутність можливості тестування в умовах слабкого сигналу Wi-Fi або його повної відсутності не дозволила оцінити поведінку системи у реальних умовах польового використання – наприклад, у підвалах, на вулиці, в промислових зонах. Надалі доцільно впровадити буферизацію команд або повторні запити, щоб забезпечити надійність взаємодії у нестабільному середовищі.

Майбутній розвиток проєкту може передбачати:

– реалізацію Bluetooth-режиму;

– підтримку декількох замків та ролей користувачів;

– інтеграцію з хмарними сервісами для ведення журналу доступів;

– додавання системи push-сповіщень;

– розширення інтерфейсу мобільного застосунку з підтримкою конфігурації пристроїв.

4.4 Охорона праці

Усі етапи технологічного процесу характеризуються інформаційним навантаженням, їх виконання вимагає уваги, зосередженості, що необхідно враховувати при створенні безпечних і нешкідливих умов праці користувачів КІТ.

Комп'ютер як технічний засіб може бути джерелом небезпечних і шкідливих виробничих чинників.

З принципу роботи відеодисплейного терміналу виходить, що він є джерелом:

- для моніторів з електронно-променевою трубкою – іонізуючих випромінювань, а саме рентгенівського випромінювання, яке виникає усередині колби при різкій зупинці електронів, що швидко рухаються;
- оптичного випромінювання у видимому діапазоні, а також в УФ діапазоні, яке виникає завдяки взаємодії електронів з шаром люмінофора;
- електромагнітного випромінювання в радіочастотному діапазоні. При цьому високочастотні поля виникають під впливом електронного променя і пов'язані з частотою формування елемента зображення і інтенсивністю променя, низькочастотні поля виникають у системі горизонтальної розгортки, а дуже низькочастотні поля пов'язані з генерацією (вертикальною розгорткою);
- магнітного поля, що виникає через наявність відхиляючого пристрою;
- електростатичного поля, що виникає у зв'язку з високим потенціалом анодів ЕПТ[20].

ВИСНОВКИ

Під час дослідження було проаналізовано сучасні рішення у сфері електронних замків та систем контролю доступу, зокрема їх класифікацію, принципи дії, переваги та недоліки. Особливу увагу приділено інтеграції таких систем у середовище IoT, що дозволило сформувати концептуальну архітектуру пристрою, який здатен реагувати на віддалені команди з мобільного застосунку через Wi-Fi.

На етапі проєктування було обґрунтовано вибір мікроконтролера ESP8622, способів зв’язку (Wi-Fi та Bluetooth), а також базових апаратних компонентів – сервоприводу, реле, індикаторів.

Окремим досягненням стало створення повноцінного мобільного застосунку на базі Flutter, що підтримує авторизацію користувачів, зберігання сесії, інтуїтивно зрозумілий інтерфейс керування замком та взаємодію із мікроконтролером через протокол HTTP. Інтерфейс було адаптовано для зручної взаємодії з одним або декількома пристроями, що створює потенціал для подальшого масштабування.

Для візуалізації та тестування роботи системи побудовано детальну схему в середовищі Wokwi, що дозволило перевірити алгоритми обробки команд у режимі реального часу, не вдаючись до фізичних прототипів. Але так як Wokwi та інші його аналоги не підтримує ESP8622, було вирішино писати схему на базі ESP32 для перевірки її роботоспособності та реагування на команди /lock, /unlock, /status. Середовище Wokwi не підтримує вебсервери, тому і з’єднання з мобільним додатком неможливо.

Також, отримані результати роботи можно віднести до Цілі сталого розвитку 9 “Промисловість, інновації та інфраструктура”, а саме п.9.1, 9.4, 9.5. Автоматизація процесів замикання дверей із застосуванням IoT прямо сприяє

модернізаціїіснуючих інфраструктурних об’єктів, забезпечує ефективне й безпечне функціонування будівель, сприяючи реалізації Цілі 9[21].

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. ДСТУ 3008–15. Документація. Звіти у сфері науки та техніки. структура та правила оформлення. Введ. 2015–06–22. К. Держстандарт України, 2017. – 29 с.

2. Методичні вказівки з підготовки кваліфікаційної роботи для здобувачів першого (бакалаврського) рівня вищої освіти денної і заочної форми навчання спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» / Упоряд.: І.Ш. Невлюдов, О.І. Филипенко, О.В. Токарєва, С.П. Новоселов, О.В Сичова. Харків: ХНУРЕ, 2023. 64 с.

3. Аналіз особливостей побудови системи керування та управління доступом для банківських установ / М. В. Коломійчук // Збірник наукових праць XI Міжнародної науково-практичної конференції (Одеса, 2023). – Одеса : Державний університет інтелектуальних технологій і зв'язку, 2023. – С. 150–152. – Режим доступу: https://suitt.edu.ua/wp-content/uploads/2023/01/Chastyna-2.pdf

4. Cirani S., Ferrari G., Picone M., Veltri L. Internet of Things: Architectures, Protocols and Standards. Hoboken, NJ : Wiley, 2019. 352 p.

5. Перспективи розвитку Інтернету речей в Україні: Зелена книга. BRDO. 2019. URL: https://brdo.com.ua/wp-content/uploads/2019/12/BRDO_Green-Paper_IoT_18-12_new.pdf

6. Al-Absi M. A., Al-Absi A. A., Lee J., Lee H. A Survey on Smart Lock Systems. Journal of Information and Communication Convergence Engineering. 2021. Vol. 19, No. 1. P. 1–9. DOI: 10.6109/jicce.2021.19.1.1.

7. Domínguez-Bolaño T., Campos O., Barral V., Escudero C. J., García-Naya J. A. An overview of IoT architectures, technologies, and existing open-

source projects. arXiv preprint. 2024. arXiv:2401.13889. URL: https://arxiv.org/abs/2401.13889

8. Розумний замок. Вікіпедія : вільна енциклопедія. URL: https://uk.wikipedia.org/wiki/Розумний_замок

9. Saleem Y., Crespi N., Rehmani M. H., Copeland R. A. A Comprehensive Survey on the State-of-the-Art in IoT: Communication Technologies, Research Challenges, and Future Directions. Journal of Network and Computer Applications. 2022. Vol. 200. 103324. DOI: 10.1016/j.jnca.2022.103324.

10. Choi J. Practical ESP8622. Programming Internet of Things Projects. Berkeley, CA : Apress, 2021. 312 p.

11. Random Nerd Tutorials. ESP-NOW with ESP8622 NodeMCU. URL: https://randomnerdtutorials.com/esp-now-ESP8622-nodemcu-arduino-ide/

12. Бездротовий зв'язок. Модуль Wi–Fi ESP8622. URL: https://arduino.ptngu.com/chapters/30/

13. Arduino. Official Arduino Documentation. URL: https://www.arduino.cc/

14. Wokwi - Online Simulator for Arduino and ESP32. URL: https://wokwi.com/

15. Windmill E. Flutter in Action. Shelter Island, NY : Manning Publications, 2020. 552 p.

16. Haryanto E., Hidayat T., Putra I. K. G. D. Design of IoT-Based Temperature and Humidity Monitoring System using Flutter Framework. International Journal of Interactive Mobile Technologies (iJIM). 2021. Vol. 15, No. 15. P. 157–167. DOI: 10.3991/ijim.v15i15.24151.

17. Flutter Documentation. HTTP Requests in Flutter using the http package. URL: https://docs.flutter.dev/cookbook/networking/fetch-data

18. ENISA. Guidelines for Securing the Internet of Things. European Union Agency for Cybersecurity, November 2020. 128 p.

19. ETSI EN 303 645 V2.1.1. Cyber Security for Consumer Internet of Things: Baseline Requirements. European Telecommunications Standards Institute, 2020. 57 p.

20. Комплекс навчально-методичного забезпечення навчальної дисципліни «Безпека праці в індустрії ІТ-технологій» підготовки освітнього рівня бакалавр усіх спеціальностей та усіх напрямів університету [http://catalogue.nure.ua/knmz] / ХНУРЕ; розроб.: Т. Є. Стиценко, Г. В. Пронюк, Н. М. Сердюк. – Харків, 2017. – 122 с.

21. Цілі сталого розвитку https://www.kmu.gov.ua/storage/app/sites/1/natsionalna-dopovid-csr-Ukrainy.pdf