

**Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення**

**КВАЛІФІКАЦІЙНА РОБОТА
ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»**

**РОЗРОБКА МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВІДДАЛЕНОГО
КЕРУВАННЯ ДИНАМІЧНОЮ ПАНЕЛЛЮ ВІЗУАЛІЗАЦІЇ**

**DEVELOPMENT OF A MOBILE APPLICATION FOR REMOTE
CONTROL OF A DYNAMIC VISUALIZATION PANEL**

спеціальність 121 «Інженерія програмного забезпечення»
освітня програма «Інженерія програмного забезпечення»

Виконав: здобувач вищої освіти
Групи ІПЗз-41
Повстяна Соломія Олександрівна

Керівник:
Ящук Андрій Анатолійович

Кваліфікаційну роботу
допущено до захисту
«__» _____ 2026 р.
к.т.н., доцент
Гарант освітньої програми:
К.т.н., доцент
Ліщина Наталія Миколаївна

(підпис)

Луцьк – 2026 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет комп'ютерних та інформаційних технологій

Кафедра інженерії програмного забезпечення

Ступінь вищої освіти: бакалавр

Галузь знань: 12 Інформаційні технології

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітня програма: «Інженерія програмного забезпечення»

ЗАТВЕРДЖУЮ

Завідувач кафедри

«__» _____ 20__ р.

ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Повстяній Соломії Олександрівні

(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: «Розробка мобільного застосунку для віддаленого керування динамічною панеллю візуалізації»

Керівник роботи: к.т.н., доцент, Ящук Андрій Анатолійович

затверджені наказом закладу вищої освіти від «31» грудня 2026 р. № 541/01-02

2. Строк подання здобувачем вищої освіти кваліфікаційної роботи «29» травня 2026 р.

3. Вихідні дані до роботи: предметна область мобільних систем керування світлодіодними матрицями, технології розробки Android-застосунків (Kotlin, Android SDK), засоби бездротового зв'язку Bluetooth, мікроконтролерна платформа Arduino, алгоритми обробки користувацького введення та передачі даних, вимоги до програмно-апаратних систем реального часу, засоби забезпечення надійності та захисту інформації.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що потрібно розробити): аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення, опис функціонального наповнення об'єкта проектування, практична реалізація об'єкта проектування.

5. Перелік графічного матеріалу: 11 рисунків, 9 лістингів, 3 таблиці

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	Ящук А. А.		
Специфікації вимог до розробленої системи	Ящук А. А.		
Розробка об'єкта проектування	Ящук А. А.		
Нормоконтроль	Суринович О. М.		
Гарант ОП	Ліщина Н. М.		
Показник запозичень тексту	_____ %		
Академічна доброчесність	Ящук А. А.		

7. Дата видачі завдання «03» лютого 2026 р.

№	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	Огляд літературних джерел по темі кваліфікаційної роботи бакалавра	до 14.02.2026 р.	
2	Аналіз проблеми розробки та впровадження об'єкту проектування	до 03.03.2026 р.	
3	Обґрунтування вибору шляхів, технологій і засобів вирішення поставленого завдання	до 14.03.2026 р.	
4	Розробка функціонально-структурної схеми роботи об'єкта проектування та проектування бази даних	до 14.04.2026 р.	
5	Практична реалізація об'єкта проектування та розробка бази даних	до 20.04.2026 р.	
6	Тестування та налагодження об'єкта проектування	до 04.05.2026 р.	
7	Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру	до 29.05.2026 р.	

Здобувач вищої освіти

Соломія ПОВСТЯНА

Керівник кваліфікаційної роботи

Андрій ЯЩУК

АНОТАЦІЯ

Повстяна С. О. Розробка мобільного застосунку для віддаленого керування динамічною панеллю візуалізації. Рукопис. Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення» спеціальності «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2026.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків, списку використаних джерел.

У першому розділі проведено аналіз предметної області керування світлодіодними матрицями, досліджено існуючі програмні рішення та визначено їхні переваги і недоліки, а також обґрунтовано доцільність розробки власного мобільного застосунку.

У другому розділі сформовано функціональні та нефункціональні вимоги до програмного забезпечення, здійснено проектування системи із використанням UML-нотації, а також обґрунтовано вибір технологій, методів і засобів реалізації.

У третьому розділі описано процес розробки мобільного застосунку, реалізовано алгоритми взаємодії з LED-матрицею через Bluetooth, розглянуто питання захисту інформації, а також наведено результати тестування і налагодження програмного забезпечення.

Ключові слова: мобільний застосунок, LED-матриця, Arduino, Bluetooth, Android, Kotlin, вбудовані системи.

ABSTRACT

Povstiana S. O. Development of a Mobile Application For Remote Control Of a Dynamic Visualization Panel. Manuscript. Bachelor's qualification work in Software Engineering, specialty "Software engineering". Lutsk National Technical University. Lutsk, 2026.

The bachelor's qualification work consists of an introduction, three chapters, conclusions, a list of references and appendices.

The first chapter analyzes the problem domain of LED matrix control systems, reviews existing software solutions, and identifies their advantages and disadvantages, as well as justifies the need for developing a new mobile application.

The second chapter defines functional and non-functional requirements, presents system design using UML, and justifies the choice of development technologies and tools.

The third chapter describes the implementation of the mobile application, including algorithms for interaction with the LED matrix via Bluetooth, information security considerations, and the results of testing and debugging.

Keywords: mobile application, LED matrix, Arduino, Bluetooth, Android, Kotlin, embedded systems.

ЗМІСТ

ВСТУП	7
РОЗДІЛ 1 АНАЛІЗ СИСТЕМ КЕРУВАННЯ LED-МАТРИЦЯМИ ТА ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ	10
1.1 Аналіз сучасного стану проблеми.....	10
1.2 Постановка завдання на кваліфікаційну роботу бакалавра.....	15
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ	17
2.1 Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання.....	17
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання	20
РОЗДІЛ 3 РОЗРОБКА ОБ'ЄКТА ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВІДДАЛЕНОГО КЕРУВАННЯ ДИНАМІЧНОЮ ПАНЕЛЛЮ ВІЗУАЛІЗАЦІЇ	25
3.1 Практична реалізація об'єкта проектування	25
3.2 Захист інформаційно-комп'ютерної системи.....	35
3.3 Ергономічна оцінка функціональних компонентів інформаційно-комп'ютерної системи	38
3.4 Техніко-економічне обґрунтування розробки.....	40
ВИСНОВКИ.....	43
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ	46

ВСТУП

У сучасних умовах стрімкого розвитку інформаційних технологій та цифровізації суспільства значну увагу приділяють створенню інтерактивних систем візуалізації, які забезпечують ефективне представлення інформації у режимі реального часу. Одним із перспективних напрямів у цій сфері є використання світлодіодних (LED) матриць, що дозволяють відображати текстову, графічну та анімаційну інформацію з високою гнучкістю та адаптивністю. Такі системи широко застосовуються у рекламній індустрії, інформаційних табло, освітніх платформах, інтерактивних стендах, а також у рішеннях класу Internet of Things.

Актуальність теми зумовлена зростанням потреби у створенні зручних, мобільних та доступних засобів керування візуалізаційними пристроями. Традиційні підходи до керування LED-матрицями часто передбачають використання стаціонарних комп'ютерів або складних програмних інтерфейсів, що обмежує мобільність користувача та ускладнює процес взаємодії з системою. Водночас сучасні мобільні пристрої, зокрема смартфони, мають значний обчислювальний потенціал, розвинені засоби введення та підтримку бездротових технологій, що дозволяє використовувати їх як універсальний інтерфейс керування.

Особливого значення набуває використання бездротових технологій, зокрема Bluetooth, який забезпечує надійний зв'язок між мобільним застосунком і мікроконтролерною системою без необхідності підключення до мережі Інтернет. Це дозволяє створювати автономні, енергоефективні та прості у використанні системи керування, що можуть функціонувати в умовах обмеженої інфраструктури.

Незважаючи на наявність існуючих рішень, аналіз сучасних програмних продуктів показує, що більшість із них мають обмежений функціонал, не забезпечують достатнього рівня інтерактивності та не відповідають сучасним вимогам до зручності користувацького інтерфейсу. Це обумовлює

необхідність розробки спеціалізованих мобільних застосунків, які поєднують простоту використання, розширений функціонал та ефективну взаємодію з апаратною частиною.

Метою кваліфікаційної роботи бакалавра є розробка мобільного застосунку для віддаленого керування світлодіодною матрицею на базі мікроконтролера Arduino з використанням технології Bluetooth, який забезпечує інтерактивну взаємодію користувача з системою та відображення інформації у режимі реального часу.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- обґрунтувати вибір архітектури системи, засобів та технологій розробки мобільного застосунку і апаратної складової;
- розробити функціонально-структурну схему системи керування LED-матрицею, що відображає взаємодію її основних компонентів;
- виконати проєктування програмного забезпечення мобільного застосунку із використанням методів об'єктно-орієнтованого моделювання, зокрема побудувати діаграму класів;
- розробити алгоритми обробки користувацького введення, формування керувальних команд та їх передачі через Bluetooth-канал;
- реалізувати мобільний застосунок для керування LED-матрицею на платформі Android із використанням мови програмування Kotlin;
- забезпечити взаємодію мобільного застосунку з мікроконтролером Arduino та реалізувати передачу і обробку керувальних команд;
- провести тестування розробленої системи та виконати оцінку її ергономічних характеристик;
- виконати техніко-економічне обґрунтування доцільності розробки інформаційно-комп'ютерної системи.

Об'єктом кваліфікаційної роботи бакалавра є процеси взаємодії мобільного застосунку з мікроконтролерною системою у задачах керування світлодіодними пристроями.

Предметом кваліфікаційної роботи бакалавра є методи, засоби та алгоритми реалізації мобільного застосунку для керування LED-матрицею, а також способи організації бездротового обміну даними та обробки користувацького введення.

Розробка мобільного застосунку для віддаленого керування динамічною панеллю візуалізації на базі LED-матриці є актуальним та перспективним напрямом у сфері інформаційних технологій. Реалізація такого програмно-апаратного комплексу дозволяє поєднати можливості мобільних пристроїв, бездротових технологій та мікроконтролерних систем для створення зручного, ефективного та інтерактивного засобу керування візуалізаційними процесами. Запропонована система може бути застосована у навчальному процесі для демонстрації принципів роботи мікроконтролерів, бездротового зв'язку та вбудованих систем, а також у прикладних IoT-рішеннях, рекламних інсталяціях, декоративному освітленні та інтерактивних візуалізаційних системах.

Крім того, використання доступних апаратних компонентів (Arduino, LED-матриця, Bluetooth-модуль) та відсутність необхідності у серверній частині забезпечують низьку вартість впровадження, що робить розробку економічно доцільною та доступною для широкого кола користувачів.

РОЗДІЛ 1

АНАЛІЗ СИСТЕМ КЕРУВАННЯ LED-МАТРИЦЯМИ ТА ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

У сучасних інформаційно-комп'ютерних системах значного поширення набувають інтерактивні засоби візуалізації, зокрема світлодіодні (LED) матриці, які використовуються для відображення текстової, графічної та анімаційної інформації у режимі реального часу. Такі системи активно застосовуються у рекламній сфері, інформаційних табло, освітніх проєктах, системах «розумного дому» та IoT-рішеннях. Як зазначено у роботі [1], світлодіодні матриці на базі мікроконтролерів забезпечують гнучкість відображення інформації та можливість її оперативного оновлення, що робить їх ефективним інструментом динамічної візуалізації. Разом із тим традиційні підходи до керування такими системами часто передбачають використання стаціонарних комп'ютерів або складних програмних інтерфейсів, що обмежує мобільність користувача та ускладнює процес взаємодії.

Подальший розвиток систем керування LED-матрицями пов'язаний із впровадженням бездротових технологій та мобільних пристроїв як основного інтерфейсу користувача. У дослідженні [2] розглянуто реалізацію системи керування світлодіодною матрицею за допомогою смартфона через Bluetooth-з'єднання. Автори підкреслюють, що використання мобільного застосунку дозволяє значно підвищити зручність керування, забезпечити інтуїтивну взаємодію та реалізувати передачу текстових і графічних даних у режимі реального часу. Смартфон у такому підході виступає універсальним інструментом взаємодії, який поєднує функції інтерфейсу, контролера та засобу введення даних. Водночас у роботі відзначається, що існуючі реалізації подібних систем часто мають обмежений функціонал, зокрема недостатню гнучкість налаштування режимів відображення, відсутність розвиненого

користувачького інтерфейсу та складність масштабування під різні конфігурації матриць.

Важливим аспектом побудови таких систем є вибір технології бездротового зв'язку. У роботі [3] обґрунтовується доцільність використання Bluetooth як ефективного засобу організації зв'язку між мобільним застосунком і мікроконтролерною системою. Зокрема, зазначається, що Bluetooth забезпечує достатню швидкість передачі даних, низьке енергоспоживання та простоту інтеграції з більшістю сучасних мобільних пристроїв. Крім того, на відміну від Wi-Fi або хмарних рішень, використання Bluetooth не потребує підключення до мережі Інтернет, що підвищує автономність системи та зменшує її залежність від зовнішньої інфраструктури. Це особливо важливо для локальних систем керування, де необхідна швидка реакція на дії користувача та стабільна робота в умовах обмежених ресурсів.

Незважаючи на наявність досліджень у цій сфері, аналіз сучасних рішень показує, що існує низка невирішених проблем. Зокрема, багато систем орієнтовані на виконання обмеженого набору функцій, таких як відображення тексту або простих графічних ефектів, і не забезпечують повноцінної інтерактивної взаємодії користувача з LED-матрицею. Крім того, інтерфейси існуючих застосунків часто не відповідають сучасним вимогам ергономіки та UX/UI-дизайну, що ускладнює їх використання для широкого кола користувачів. Також недостатньо уваги приділяється оптимізації передачі даних та обробці користувачьких дій у режимі реального часу, що може призводити до затримок і зниження якості взаємодії.

У процесі дослідження сучасних підходів до керування світлодіодними матрицями було проаналізовано ряд існуючих програмних рішень, які реалізують віддалене керування LED-пристроями за допомогою мобільних застосунків. Метою такого аналізу є визначення функціональних можливостей наявних систем, виявлення їхніх переваг і недоліків, а також обґрунтування необхідності розробки власного програмного продукту.

Одним із найбільш поширених рішень є застосунок Pixel LED Art [4], який призначений для керування адресними LED-матрицями через Bluetooth-з'єднання (рис. 1.1). Основною функціональною можливістю цього застосунку є створення піксельних зображень у реальному часі з подальшою передачею їх на світлодіодну панель. Користувач може малювати, змінювати кольори, а також відтворювати прості анімації. Перевагою даного рішення є простота використання та швидка передача даних. Однак застосунок має обмежену функціональність, зокрема відсутність розширених режимів керування, слабку адаптацію інтерфейсу та недостатню підтримку складних сценаріїв взаємодії з матрицею.



Рисунок 1.1 – Застосунок Pixel LED Art [4]

Іншим прикладом є застосунок LED Matrix Editor [5], який орієнтований на редагування піксельних зображень та формування графічних шаблонів для LED-матриць (рис. 1.2). Цей застосунок надає можливість створення бінарних або RGB-патернів і передачі їх на пристрій. Його перевагою є точність налаштування пікселів та можливість роботи з деталізованими зображеннями.

Водночас інтерфейс застосунку є складним для сприйняття, що обмежує його використання для широкого кола користувачів, а відсутність інтерактивних режимів знижує його практичну цінність у реальних сценаріях.

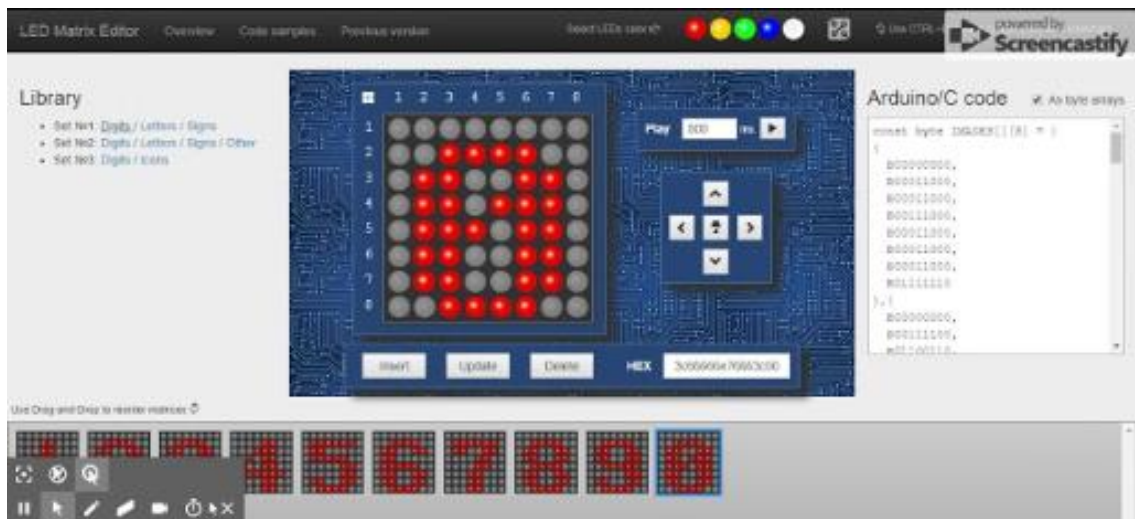


Рисунок 1.2 – Застосунок LED Matrix Editor [5]

Серед універсальних IoT-рішень варто виділити платформу Blynk IoT [6], яка дозволяє створювати кастомні панелі керування різноманітними пристроями, включаючи світлодіодні системи. Застосунок (рис. 1.3) підтримує зміну параметрів роботи пристроїв у режимі реального часу, забезпечує гнучке налаштування інтерфейсу та інтеграцію з різними мікроконтролерами. Основною перевагою Blynk є універсальність і масштабованість. Однак дана платформа не орієнтована на спеціалізовану роботу з LED-матрицями, зокрема не забезпечує зручного інструментарію для створення графічних сцен або інтерактивного малювання, що обмежує її застосування в задачах візуалізації.

Ще одним популярним рішенням є система WLED [7], яка призначена для керування адресними світлодіодними стрічками та матрицями через Wi-Fi або локальну мережу. Застосунок підтримує широкий набір світлових ефектів, налаштування кольорів, яскравості та динамічних режимів. Його перевагами є висока стабільність роботи, широкий функціонал і зручний інтерфейс. Водночас WLED орієнтований переважно на декоративні світлові ефекти, а не

на інтерактивну роботу з піксельною графікою або передачу структурованої інформації, що обмежує можливості його використання у задачах динамічної візуалізації.



Рисунок 1.3 – Платформа Blynk IoT [6]

Проведений аналіз сучасних програмних рішень у сфері керування LED-матрицями показав наявність ряду суттєвих обмежень. Зокрема, частина застосунків орієнтована лише на базове керування або відображення простих візуальних ефектів, тоді як інші характеризуються складним або недостатньо адаптованим інтерфейсом, що ускладнює взаємодію користувача із системою. Більшість існуючих рішень реалізують окремі функціональні можливості, такі як передача тексту чи графічних даних, однак не забезпечують комплексної інтерактивної взаємодії у режимі реального часу.

Крім того, відсутні універсальні системи, які поєднували б зручний мобільний інтерфейс, можливість інтерактивного малювання, підтримку різних режимів роботи та ефективне керування LED-матрицею через простий канал зв'язку, зокрема Bluetooth. Це обумовлює необхідність розробки спеціалізованого мобільного застосунку на базі Arduino, який забезпечить

розширений функціонал, адаптивність інтерфейсу та стабільну роботу системи у режимі реального часу.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

На основі проведеного аналізу сучасних систем керування світлодіодними матрицями встановлено, що існуючі рішення не повною мірою задовольняють вимоги щодо інтерактивності, зручності використання та функціональної гнучкості. Більшість застосунків орієнтовані на виконання обмеженого набору функцій, не забезпечують ефективної обробки користувацьких дій у режимі реального часу та мають недостатньо розвинений інтерфейс. Це обумовлює необхідність розробки спеціалізованого програмного забезпечення, яке поєднуватиме зручність мобільного керування, розширений функціонал та ефективну взаємодію з апаратною частиною.

З огляду на це метою кваліфікаційної роботи є розробка мобільного застосунку для віддаленого керування динамічною панеллю візуалізації на базі LED-матриці з використанням мікроконтролера Arduino та технології Bluetooth.

Для досягнення поставленої мети необхідно вирішити такі завдання:

- провести аналіз сучасних інформаційно-комп'ютерних систем керування світлодіодними (LED) матрицями та визначити їх основні функціональні можливості, переваги та недоліки;
- обґрунтувати вибір архітектури системи, засобів та технологій розробки мобільного застосунку і апаратної складової;
- розробити функціонально-структурну схему системи керування LED-матрицею, що відображає взаємодію її основних компонентів;
- виконати проектування програмного забезпечення мобільного застосунку із використанням методів об'єктно-орієнтованого моделювання, зокрема побудувати діаграму класів;

- розробити алгоритми обробки користувацького введення, формування керувальних команд та їх передачі через Bluetooth-канал;
- реалізувати мобільний застосунок для керування LED-матрицею на платформі Android із використанням мови програмування Kotlin;
- забезпечити взаємодію мобільного застосунку з мікроконтролером Arduino та реалізувати передачу і обробку керувальних команд;
- провести тестування розробленої системи та виконати оцінку її ергономічних характеристик;
- виконати техніко-економічне обґрунтування доцільності розробки інформаційно-комп'ютерної системи.

РОЗДІЛ 2

СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання

Реалізація мобільного застосунку для віддаленого керування динамічною панеллю візуалізації потребує інтеграції програмних і апаратних засобів, які забезпечують надійний обмін даними, інтерактивну взаємодію користувача з віддаленим пристроєм та адаптацію інтерфейсу до обмежень мобільних платформ. З огляду на це архітектуру системи побудовано за моделлю «клієнт – вбудований пристрій», у межах якої мобільний застосунок виконує функції інтерфейсу користувача та генератора керувальних команд, а мікроконтролерна система – функції обробки цих команд і безпосереднього керування адресною світлодіодною матрицею. Такий розподіл відповідальності дозволяє забезпечити гнучкість системи, зменшити навантаження на мобільний пристрій та спростити подальше розширення функціоналу без необхідності зміни апаратної частини.

Запропонована модель реалізується у вигляді дворівневої архітектури, що включає програмну та апаратну частини, пов'язані між собою каналом бездротового зв'язку (рис. 2.1). Програмна частина представлена мобільним застосунком на платформі Android, який складається з користувацького інтерфейсу, модулів обробки введення (дотиків, жестів і сенсорних даних) та підсистеми формування і передачі команд. Всі дії користувача перетворюються у стандартизовані текстові повідомлення, що містять тип операції та відповідні параметри, і передаються через Bluetooth-підсистему застосунку.

Апаратна частина системи включає Bluetooth-модуль та мікроконтролер, який приймає передані команди, виконує їх декодування та реалізує логіку керування адресною світлодіодною матрицею на рівні прошивки. Таким чином, мобільний застосунок відповідає за високорівневу інтерпретацію дій

користувача та генерацію керувальних сигналів, тоді як апаратна частина забезпечує їх безпосереднє виконання та формування візуального результату [8]. Такий підхід дозволяє ізолювати апаратно-залежну логіку у прошивці мікроконтролера та спростити модифікацію інтерфейсу й функціональності клієнтської частини без втручання в апаратну конфігурацію системи.

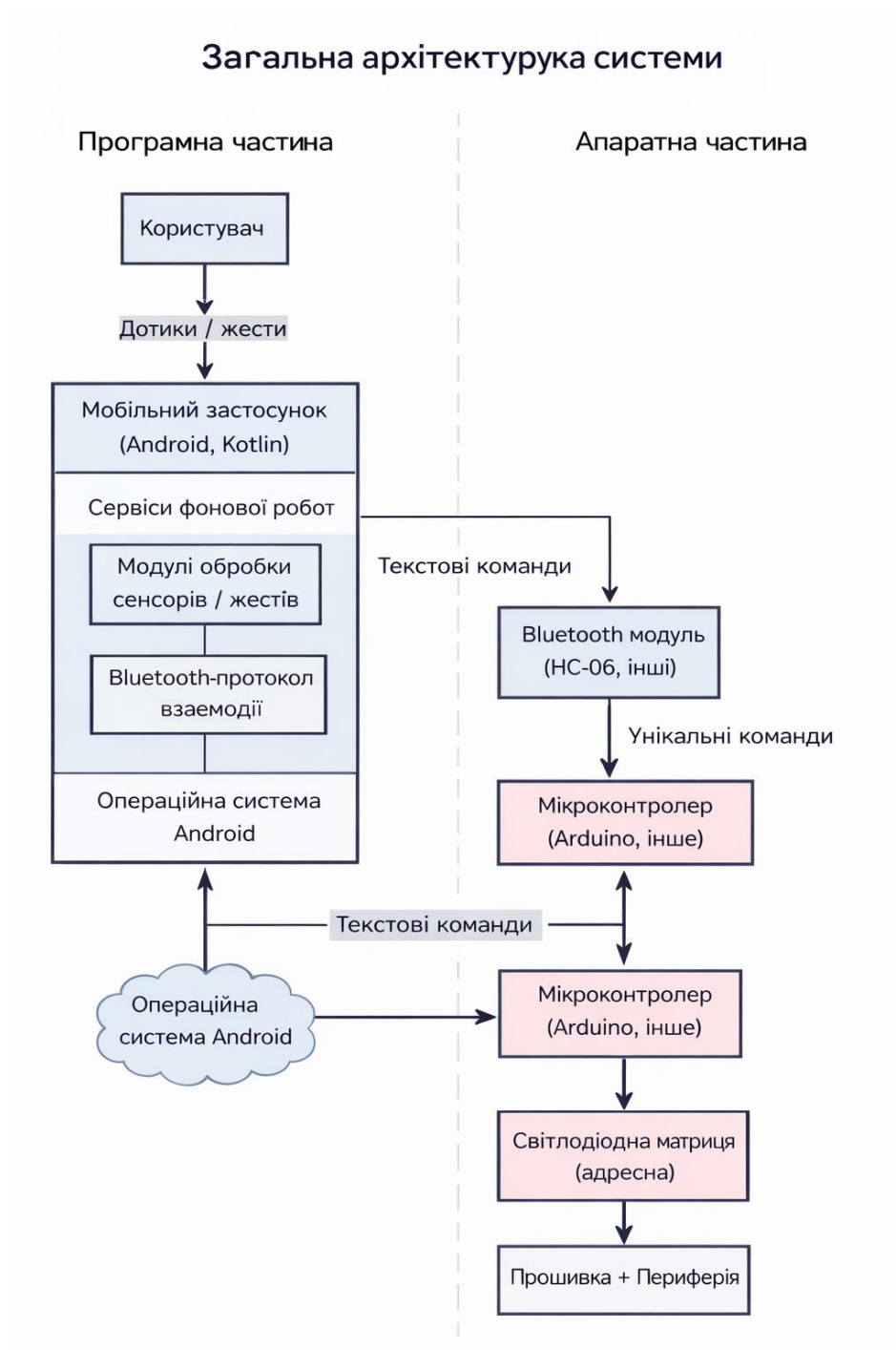


Рисунок 2.1 – Загальна архітектура системи керування LED-матрицею

Для реалізації клієнтської частини обрано платформу Android та мову програмування Kotlin, що зумовлено її офіційною підтримкою з боку розробника платформи, високою надійністю типізації, підтримкою сучасних парадигм програмування та повною сумісністю з екосистемою Android SDK [9]. Використання Kotlin дозволяє зменшити кількість помилок, пов'язаних із керуванням пам'яттю та багатопоточністю, а також спростити реалізацію асинхронних операцій, необхідних для роботи з Bluetooth-з'єднанням та сенсорними даними в режимі реального часу.

Для організації бездротового зв'язку між мобільним пристроєм і матрицею застосовано протокол Bluetooth, який забезпечує достатню пропускну здатність, низьке енергоспоживання та широку апаратну підтримку на сучасних смартфонах. Передача інформації здійснюється у вигляді стандартизованих текстових команд, що кодують тип дії та її параметри, зокрема координати пікселів, значення кольору в шістнадцятковому форматі, режими роботи та динамічні налаштування. Такий підхід спрощує інтерпретацію команд на стороні мікроконтролера, зменшує ризик помилок декодування та забезпечує прозорість протоколу обміну даними.

Перетворення користувацьких дій у керувальні сигнали здійснюється за допомогою низки алгоритмів обробки введення, зокрема визначення напрямку свайпу на основі аналізу різниці координат дотику, інтерпретації даних акселерометра для керування ігровими об'єктами, дискретизації зображень відповідно до розміру матриці та поетапної передачі пікселів з використанням таймера для уникнення перевантаження каналу зв'язку. Алгоритмічна обробка забезпечує коректну інтерпретацію жестів, стабільну передачу графічної інформації та адекватну реакцію системи на дії користувача.

З метою збереження параметрів роботи системи та підвищення зручності використання реалізовано механізм локального зберігання налаштувань, який дозволяє автоматично відновлювати попередні значення параметрів і виконувати автопідключення до останнього відомого пристрою. Це забезпечує

безперервність роботи користувача та зменшує час, необхідний для підготовки системи до використання.

Інтерфейс застосунку розроблено з урахуванням принципів адаптивності, мінімізації когнітивного навантаження та швидкого зворотного зв'язку. Візуальні елементи організовані у вигляді контекстних режимів, що дозволяє користувачеві швидко перемикатися між функціями малювання, керування іграми, передавання тексту та налаштування параметрів матриці, при цьому кожна дія супроводжується миттєвою передачею відповідної команди та візуальним підтвердженням виконання.

Отже, обрані шляхи та засоби вирішення поставленого завдання, а саме використання платформи Android із мовою Kotlin, бездротового протоколу Bluetooth, текстового формату команд і алгоритмів обробки сенсорної та жестової взаємодії, є обґрунтованими з точки зору функціональних вимог системи, обмежень мобільних платформ і потреб користувачів. Вони забезпечують ефективну реалізацію мобільного застосунку для віддаленого керування динамічною панеллю візуалізації та створюють надійну основу для подальшого розвитку та масштабування системи.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

У процесі розробки мобільного застосунку для керування світлодіодною матрицею на базі Arduino через Bluetooth ключовим етапом є обґрунтований вибір програмних засобів, технологій та алгоритмів, що забезпечують ефективну реалізацію функціональних можливостей системи. Вибір інструментів здійснюється з урахуванням вимог до продуктивності, стабільності роботи в режимі реального часу, зручності користування та сумісності з апаратною частиною.

Для реалізації клієнтської частини системи було проаналізовано два основні підходи до розробки мобільних застосунків: нативний та

кроссплатформений. Кроссплатформені технології, зокрема Flutter, React Native та Xamarin, дозволяють створювати застосунки для різних операційних систем на основі єдиного коду, однак можуть мати обмеження щодо доступу до апаратних ресурсів пристрою та роботи з Bluetooth-з'єднанням. Натомість нативний підхід із використанням Android SDK забезпечує повний доступ до функціональних можливостей пристрою, вищу продуктивність і стабільність роботи.

З огляду на необхідність ефективної взаємодії з Bluetooth-модулем, для реалізації мобільного застосунку було обрано нативну платформу Android із використанням мови програмування Kotlin. Такий вибір обумовлений офіційною підтримкою Kotlin, високим рівнем типізації, зручністю реалізації асинхронних процесів та ефективною інтеграцією з бібліотеками Android SDK.

Для організації бездротового зв'язку між мобільним застосунком і мікроконтролером було розглянуто технології Wi-Fi, Bluetooth та ZigBee. Використання Wi-Fi забезпечує високу швидкість передачі даних, однак потребує мережевої інфраструктури та підключення до Інтернету. ZigBee характеризується низьким енергоспоживанням, проте має обмежену підтримку на мобільних пристроях. У результаті було обрано технологію Bluetooth, яка забезпечує достатню швидкість передачі даних, низьке енергоспоживання та простоту реалізації, що є оптимальним для локальних систем керування.

Апаратну основу системи становить мікроконтролер Arduino, який виконує функції приймання, обробки та виконання керувальних команд. Arduino обрано завдяки його доступності, простоті програмування та широкій підтримці периферійних пристроїв. Для відображення інформації використовується адресна світлодіодна матриця, яка дозволяє керувати кожним пікселем окремо, що забезпечує реалізацію графічних, текстових та динамічних ефектів.

Важливим елементом системи є протокол обміну даними між мобільним застосунком і мікроконтролером. У розроблюваній системі використовується текстовий формат команд, що містить тип операції та відповідні параметри,

такі як координати пікселів, значення кольору або параметри режиму. Такий підхід забезпечує простоту формування та декодування команд, підвищує надійність передачі даних і зменшує складність реалізації.

Для реалізації функціональних можливостей застосунку використовуються алгоритми обробки користувацького введення та формування керувальних сигналів. У режимі малювання застосовується алгоритм перетворення координат дотику у координати світлодіодної матриці, який базується на масштабуванні площини екрана до розмірів матриці. У режимах керування за допомогою жестів використовується алгоритм визначення напрямку свайпу на основі аналізу різниці координат початкової та кінцевої точок дотику. Для обробки даних акселерометра застосовується алгоритм згладжування сигналу (low-pass filter), що дозволяє зменшити вплив шумів та забезпечити стабільність керування.

Узагальнену логіку обробки дій користувача та формування керувальних команд представлено у вигляді блок-схеми алгоритму (рис. 2.2).

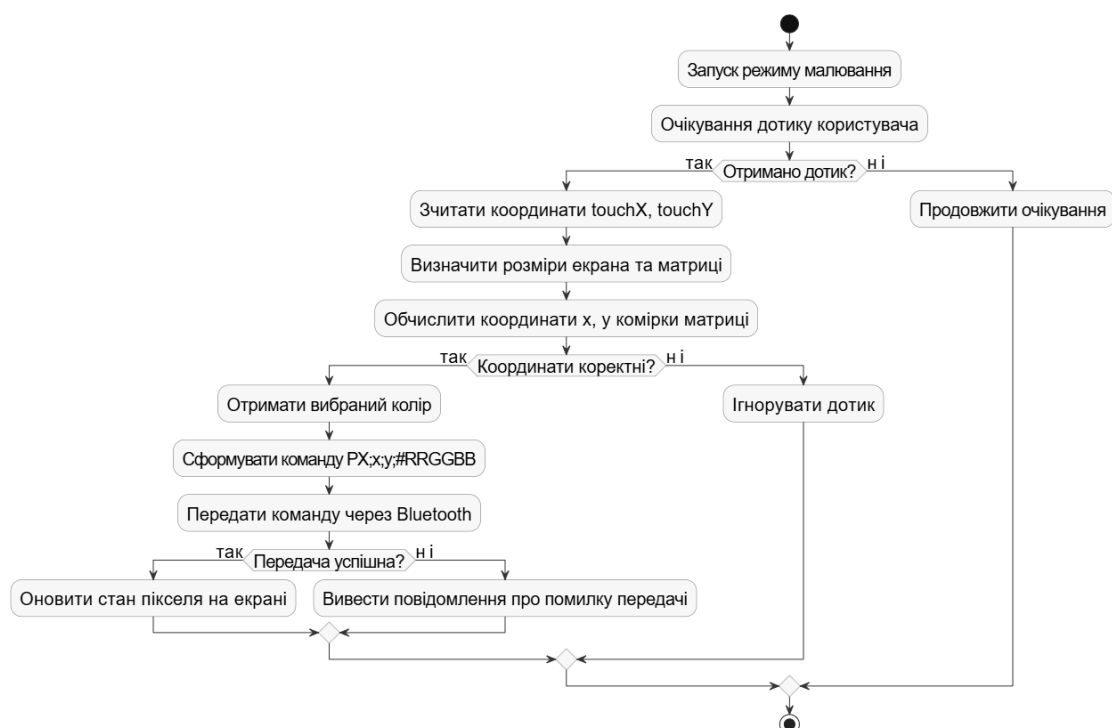


Рисунок 2.2 – Блок-схема алгоритму обробки дотику та формування команди керування пікселем

Для формалізації структури програмного забезпечення та логіки функціонування системи використано методи об'єктно-орієнтованого моделювання. Побудовано діаграму класів, яка відображає основні програмні компоненти мобільного застосунку та їх взаємозв'язки (рис. 2.3). Дана діаграма демонструє модульну структуру системи та розподіл функціональних обов'язків між її складовими. Застосування UML-моделювання дозволяє підвищити зрозумілість архітектури системи та спростити її подальше розширення.

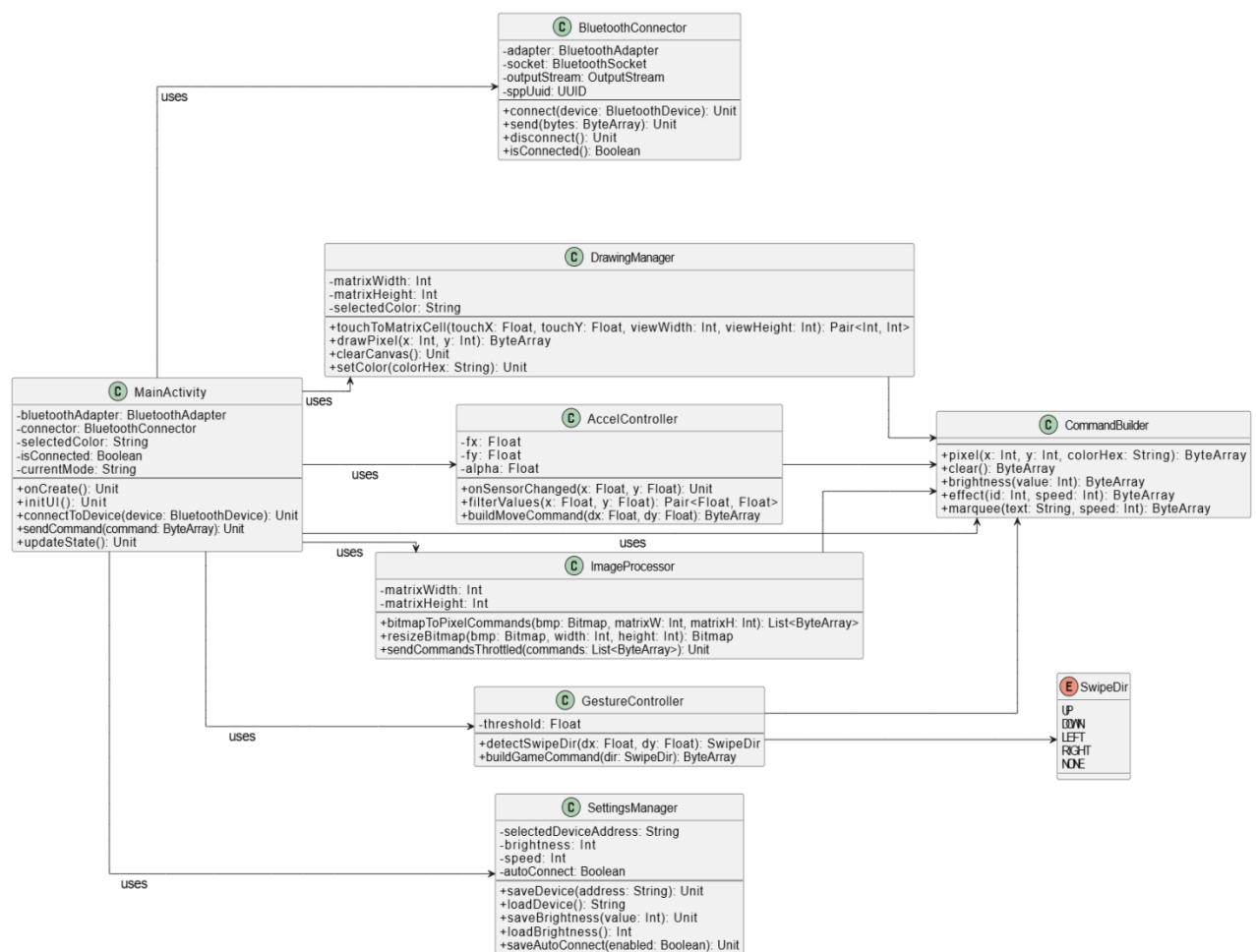


Рисунок 2.3 – Діаграма класів мобільного застосунку для керування LED-матрицею

Таким чином, обрані засоби, методи та алгоритми забезпечують ефективну реалізацію мобільного застосунку для керування LED-матрицею.

Використання платформи Android, технології Bluetooth та мікроконтролера Arduino дозволяє досягти високої продуктивності, надійності та зручності використання системи, а застосовані алгоритми забезпечують коректну обробку введення та стабільну роботу у режимі реального часу.

РОЗДІЛ 3

РОЗРОБКА ОБ’ЄКТА ПРОЕКТУВАННЯ МОБІЛЬНОГО ЗАСТОСУНКУ ДЛЯ ВІДДАЛЕНОГО КЕРУВАННЯ ДИНАМІЧНОЮ ПАНЕЛЛЮ ВІЗУАЛІЗАЦІЇ

3.1 Практична реалізація об’єкта проектування

Практична реалізація мобільного застосунку для віддаленого керування динамічною панеллю візуалізації виконана у середовищі розробки Android Studio з використанням мови програмування Kotlin та стандартних бібліотек Android SDK. Розроблений застосунок орієнтований на роботу в режимі реального часу, де дії користувача, такі як натискання, жести або сенсорні дані, перетворюються у керувальні команди, що передаються через Bluetooth на мікроконтролер Arduino.

Для забезпечення коректної роботи Bluetooth-з’єднання в сучасних версіях операційної системи Android необхідно реалізувати механізм перевірки та отримання дозволів користувача. У лістингу 3.1 наведено приклад реалізації перевірки та запиту необхідних дозволів для сканування пристроїв і встановлення з’єднання.

Лістинг 3.1 – Перевірка/запит Bluetooth-дозволів

```
import android.Manifest
import android.content.pm.PackageManager
import androidx.activity.result.contract.ActivityResultContracts
import androidx.core.content.ContextCompat
import androidx.fragment.app.Fragment

class PermissionsFragment : Fragment() {

    private val btPermissions = arrayOf(
        Manifest.permission.BLUETOOTH_SCAN,
        Manifest.permission.BLUETOOTH_CONNECT
    )

    private val requestPermissionsLauncher =
```

```

registerActivityResult(ActivityResultContracts.RequestMultiplePermissions
()) { result ->
    val granted = result.values.all { it }
    // оновлення інтерфейсу відповідно до результату надання
    // дозволів
}

fun ensureBluetoothPermissions() {
    val allGranted = btPermissions.all { perm ->
        ContextCompat.checkSelfPermission(requireContext(), perm) ==
        PackageManager.PERMISSION_GRANTED
    }
    if (!allGranted) requestPermissionsLauncher.launch(btPermissions)
}
}

```

Кінець лістингу 3.1

Для встановлення з'єднання з мікроконтролером використовується клас `BluetoothConnector`, який реалізує створення RFCOMM-сокета, підключення до пристрою та передачу даних у вигляді байтових масивів. Приклад реалізації цього класу наведено у лістингу 3.2.

Лістинг 3.2 – Клас `BluetoothConnector`

```

import android.annotation.SuppressLint
import android.bluetooth.BluetoothAdapter
import android.bluetooth.BluetoothDevice
import android.bluetooth.BluetoothSocket
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.withContext
import java.io.OutputStream
import java.util.UUID
class BluetoothConnector(
    private val adapter: BluetoothAdapter
) {
    // SPP UUID (стандартний для більшості модулів)
    private val sppUuid: UUID = UUID.fromString("00001101-0000-1000-8000-00805F9B34FB")
}

```

```

private var socket: BluetoothSocket? = null
private var out: OutputStream? = null

@SuppressLint("MissingPermission")
suspend fun connect(device: BluetoothDevice) = withContext(Dispatchers.IO)
{
    disconnect()
    val s = device.createRfcommSocketToServiceRecord(sppUuid)
    adapter.cancelDiscovery()
    s.connect()
    socket = s
    out = s.outputStream
}

suspend fun send(bytes: ByteArray) = withContext(Dispatchers.IO) {
    out?.write(bytes)
    out?.flush()
}

suspend fun disconnect() = withContext(Dispatchers.IO) {
    runCatching { out?.close() }
    runCatching { socket?.close() }
    out = null
    socket = null
}

fun isConnected(): Boolean = socket?.isConnected == true
}

```

Кінець лістингу 3.2

Інтерфейс вибору Bluetooth-пристрою та стан підключення зображено на рисунку 3.1. Як показано на рисунку 3.1, користувач має можливість вибрати доступний Bluetooth-пристрій для встановлення з'єднання.



Рисунок 3.1 – Екран сканування/списку Bluetooth-пристроїв

Для спрощення взаємодії між мобільним застосунком і мікроконтролером використано уніфікований текстовий протокол обміну командами. Кожна команда містить тип операції та відповідні параметри. Наприклад, команда встановлення пікселя має вигляд `PX;x;y;#RRGGBB`, а команда очищення – `CLR`. Формування таких команд реалізовано у вигляді окремого класу (лістинг 3.3).

Лістинг 3.3 – Формування команд керування

```
object CommandBuilder {
```

```

fun pixel(x: Int, y: Int, colorHex: String): ByteArray {
    require(x >= 0 && y >= 0) { "Координати мають бути невід'ємні" }
    require(colorHex.matches(Regex("^#[0-9A-Fa-f]{6}$"))) { "Колір у
форматі #RRGGBB" }
    return "PX;$x;$y;$colorHex\n".encodeToByteArray()
}

fun clear(): ByteArray = "CLR\n".encodeToByteArray()

fun brightness(value: Int): ByteArray {
    val v = value.coerceIn(0, 255)
    return "BR;$v\n".encodeToByteArray()
}

fun effect(id: Int, speed: Int): ByteArray {
    val sp = speed.coerceIn(1, 20)
    return "EF;$id;$sp\n".encodeToByteArray()
}

fun marquee(text: String, speed: Int): ByteArray {
    val safeText = text.replace("\n", " ")
    val sp = speed.coerceIn(1, 20)
    return "TXT;$sp;$safeText\n".encodeToByteArray()
}
}

```

Кінець лістингу 3.3

Однією з основних функціональних можливостей розробленого мобільного застосунку є реалізація інтерактивного режиму малювання, який забезпечує можливість формування графічних елементів безпосередньо користувачем у режимі реального часу. Для цього в системі реалізовано спеціалізований користувацький інтерфейс, що поєднує робочу область у вигляді координатної сітки та засоби вибору параметрів відображення, зокрема кольору пікселів. Інтерфейс режиму малювання та вибору кольору представлено на рисунку 3.2.

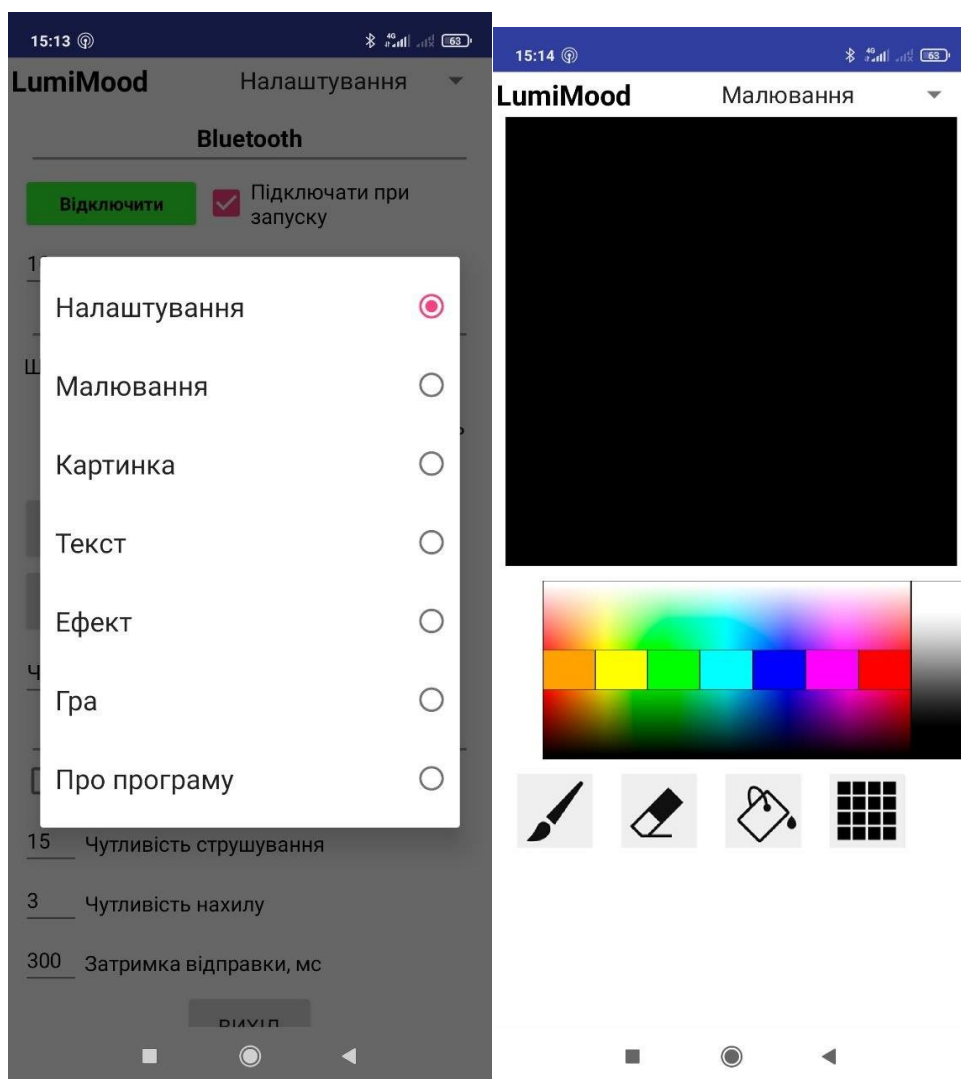


Рисунок 3.2 – Екран режиму малювання (сітка) + вибір кольору

У режимі малювання координати дотику користувача на екрані перетворюються у координати пікселя світлодіодної матриці. Це здійснюється шляхом масштабування координат відповідно до розміру матриці. Відповідний алгоритм наведено у лістингу 3.4.

Лістинг 3.4 – Перетворення координат дотику у координати матриці

```
fun touchToMatrixCell(
    touchX: Float,
    touchY: Float,
    viewWidth: Int,
    viewHeight: Int,
    matrixW: Int,
```

```

matrixH: Int
): Pair<Int, Int> {
    val cellW = viewWidth.toFloat() / matrixW
    val cellH = viewHeight.toFloat() / matrixH
    val x = (touchX / cellW).toInt().coerceIn(0, matrixW - 1)
    val y = (touchY / cellH).toInt().coerceIn(0, matrixH - 1)
    return x to y
}

```

Кінець лістингу 3.4

Результат роботи алгоритму у вигляді сформованого графічного зображення представлено на рисунку 3.3.

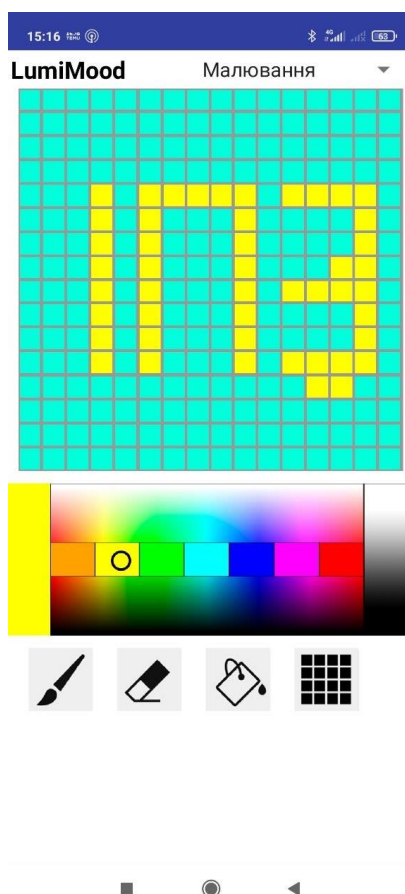


Рисунок 3.3 – Приклад «намальованого» патерну (кілька пікселів)

Для реалізації ігрових режимів використовується обробка жестів користувача. Зокрема, напрямок свайпу визначається на основі різниці

координат початкової та кінцевої точок дотику. Відповідний алгоритм наведено у лістингу 3.5.

Лістинг 3.5 – Визначення напрямку свайпу

```
enum class SwipeDir { UP, DOWN, LEFT, RIGHT, NONE }

fun detectSwipeDir(dx: Float, dy: Float, threshold: Float = 50f): SwipeDir {
    if (kotlin.math.abs(dx) < threshold && kotlin.math.abs(dy) < threshold)
return SwipeDir.NONE
    return if (kotlin.math.abs(dx) > kotlin.math.abs(dy)) {
        if (dx > 0) SwipeDir.RIGHT else SwipeDir.LEFT
    } else {
        if (dy > 0) SwipeDir.DOWN else SwipeDir.UP
    }
}
```

Кінець лістингу 3.5

Для керування об'єктами за допомогою нахилу пристрою використовується акселерометр. З метою зменшення впливу шумів застосовано алгоритм згладжування сигналу (low-pass filter), що дозволяє отримати більш стабільні значення. Реалізацію наведено у лістингу 3.6.

Лістинг 3.6 – Обробка даних акселерометра

```
import android.hardware.Sensor
import android.hardware.SensorEvent
import android.hardware.SensorEventListener

class AccelController(
    private val onMove: (dx: Float, dy: Float) -> Unit
) : SensorEventListener {

    private var fx = 0f
    private var fy = 0f
    private val alpha = 0.2f // коефіцієнт фільтра

    override fun onSensorChanged(event: SensorEvent) {
```

```

if (event.sensor.type != Sensor.TYPE_ACCELEROMETER) return
val x = event.values[0]
val y = event.values[1]

fx = fx + alpha * (x - fx)
fy = fy + alpha * (y - fy)

// інтерпретація: нахил → керування
onMove(dx = -fx, dy = fy)
}

override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) = Unit
}

```

Кінець лістингу 3.6

Приклад використання акселерометра у застосунку представлено на рисунку 3.4.



Рисунок 3.4 – Екран налаштування чутливості

У режимі відображення зображень використовується алгоритм масштабування графічного файлу до розміру LED-матриці з подальшим перетворенням кожного пікселя у відповідну команду керування. Алгоритм формування набору команд наведено у лістингу 3.7.

Лістинг 3.7 – Перетворення зображення у набір команд

```
import android.graphics.Bitmap
import android.graphics.Color

fun bitmapToPixelCommands(
    bmp: Bitmap,
    matrixW: Int,
    matrixH: Int
): List<ByteArray> {
    val scaled = Bitmap.createScaledBitmap(bmp, matrixW, matrixH, true)
    val cmds = ArrayList<ByteArray>(matrixW * matrixH)

    for (y in 0 until matrixH) {
        for (x in 0 until matrixW) {
            val c = scaled.getPixel(x, y)
            val hex = String.format("#%02X%02X%02X", Color.red(c),
Color.green(c), Color.blue(c))
            cmds.add(CommandBuilder.pixel(x, y, hex))
        }
    }
    return cmds
}
```

Кінець лістингу 3.7

Передача великої кількості команд здійснюється поетапно з використанням затримки між відправленнями, що дозволяє уникнути перевантаження Bluetooth-каналу. Відповідний фрагмент реалізації наведено у лістингу 3.8.

Лістинг 3.8 – Поетапна передача команд

```
import kotlinx.coroutines.delay

suspend fun sendCommandsThrottled(
    connector: BluetoothConnector,
    commands: List<ByteArray>,
    delayMs: Long = 5L
) {
    for (cmd in commands) {
        connector.send(cmd)
        delay(delayMs) // “дроселювання” каналу
    }
}
```

Кінець лістингу 3.8

Для підвищення зручності використання реалізовано механізм локального збереження параметрів, зокрема вибраного пристрою, яскравості та режимів роботи. Це дозволяє автоматично відновлювати стан застосунку після повторного запуску.

Таким чином, у процесі практичної реалізації було створено повнофункціональний мобільний застосунок, який забезпечує взаємодію користувача з LED-матрицею у режимі реального часу. Реалізовані алгоритми обробки введення, формування команд та їх передавання забезпечують стабільну та ефективну роботу системи, а застосовані технології дозволяють легко розширювати функціональні можливості програмного продукту.

3.2 Захист інформаційно-комп'ютерної системи

Розроблювана інформаційно-комп'ютерна система здійснює керування фізичним виконавчим пристроєм через бездротовий канал зв'язку та оперує локальними конфігураційними даними користувача. Незважаючи на відсутність передавання даних через мережу Інтернет і обмежений масштаб

застосування, система підпадає під вплив низки потенційних загроз, пов'язаних із несанкціонованим доступом, підміною команд і порушенням цілісності керування. У зв'язку з цим у процесі проектування було передбачено комплекс заходів захисту, спрямованих на мінімізацію ризиків та підвищення надійності функціонування системи.

Основними потенційними загрозами для системи є:

- несанкціоноване підключення стороннього пристрою до LED-матриці через Bluetooth-інтерфейс;
- підміна або ін'єкція керувальних команд у канал передавання;
- перехоплення або модифікація даних у бездротовому каналі;
- некоректне або шкідливе введення параметрів користувачем;
- витік або пошкодження локальних конфігураційних даних.

З огляду на ці загрози було сформовано набір організаційних та програмно-технічних заходів захисту.

Обмеження доступу до керування пристроєм. Для запобігання несанкціонованому доступу передбачено механізм обмеження підключення лише до відомих Bluetooth-пристроїв. Мобільний застосунок зберігає MAC-адресу останнього авторизованого пристрою та дозволяє автоматичне підключення лише до нього. Усі інші пристрої ігноруються або потребують підтвердження користувача.

Аутентифікація на рівні Bluetooth. Для підвищення безпеки використовується стандартний механізм парінгу Bluetooth із PIN-кодом або підтвердженням користувача. Це унеможливорює автоматичне підключення сторонніх пристроїв без участі власника системи.

Валідація команд перед передаванням. Перед формуванням та передаванням кожної команди здійснюється перевірка її формату та параметрів (ліст. 3.9). Це дозволяє запобігти відправленню некоректних або потенційно небезпечних значень (наприклад, координат поза межами матриці або некоректних кольорових кодів).

Лістинг 3.9 – Приклад валідації параметрів команди

```
private val COLOR_REGEX = Regex("^#[0-9A-Fa-f]{6}$")
```

```
fun validatePixelCommand(x: Int, y: Int, colorHex: String, maxW: Int, maxH:
Int): Boolean {
    if (x !in 0 until maxW) return false
    if (y !in 0 until maxH) return false
    if (!COLOR_REGEX.matches(colorHex)) return false
    return true
}
```

Кінець лістингу 3.9

Мінімізація поверхні атаки. Система не використовує передавання даних через Інтернет та не підключається до зовнішніх серверів, що істотно зменшує кількість потенційних точок доступу для атак. Уся взаємодія здійснюється в межах локального Bluetooth-з'єднання.

Захист локальних даних. Локально зберігаються лише мінімально необхідні дані: ідентифікатор пристрою, параметри відображення та налаштування інтерфейсу. Дані не містять персональної інформації та не передаються третім сторонам. Для доступу до них використовується стандартний механізм захищеного сховища Android (DataStore/SharedPreferences), доступ до якого обмежений межами застосунку.

Забезпечення цілісності та стабільності. Для підвищення надійності передбачено обробку виняткових ситуацій, пов'язаних із втратою з'єднання, помилками передавання та некоректним введенням даних. У таких випадках система або повторює спробу виконання операції, або повідомляє користувача про помилку та переводить систему у безпечний стан.

Реалізований комплекс заходів захисту забезпечує достатній рівень безпеки для інформаційно-комп'ютерної системи даного класу. Він запобігає несанкціонованому доступу до керування фізичним пристроєм, зменшує ризик

підміни або пошкодження команд та забезпечує цілісність локальних даних користувача. З огляду на автономний характер системи та відсутність мережевої взаємодії із зовнішніми сервісами, реалізовані механізми є достатніми та адекватними для забезпечення надійного функціонування розроблюваного програмно-апаратного комплексу.

3.3 Ергономічна оцінка функціональних компонентів інформаційно-комп'ютерної системи

Ергономічна оцінка розроблюваної інформаційно-комп'ютерної системи здійснюється з метою визначення відповідності інтерфейсу користувача вимогам зручності, ефективності та безпеки взаємодії людини з програмним продуктом. Оскільки система призначена для керування візуальними процесами в режимі реального часу, інтерфейс повинен забезпечувати швидкий доступ до основних функцій, мінімізувати кількість дій, необхідних для виконання типових операцій, та зменшувати когнітивне навантаження на користувача.

Принципи ергономічного проектування інтерфейсу. Під час розробки інтерфейсу застосунку було дотримано таких базових принципів ергономіки [10, 11]:

- принцип мінімізації дій, згідно з яким основні функції (вибір режиму, керування яскравістю, запуск ефектів, малювання) доступні не більш ніж за два–три кроки від головного екрану;
- принцип візуальної чіткості, що реалізується через достатню контрастність елементів керування, використання обмеженої кількості кольорів та уникнення дрібних текстових елементів;
- принцип адекватного розміру елементів керування, який передбачає використання кнопок і повзунків, розмір яких відповідає рекомендаціям для сенсорних екранів і забезпечує зручність взаємодії без помилкових натискань;

– принцип уникнення перевантаження інтерфейсу, що полягає у поділі функціональності на окремі контекстні режими та відображенні лише релевантних елементів у кожному з них.

Застосування зазначених принципів сприяє підвищенню швидкості виконання операцій, зменшенню кількості помилок та підвищенню загального рівня комфорту користувача.

Методика користувацького тестування. Для емпіричної перевірки ергономічних характеристик було проведено користувацьке тестування за участю п'яти респондентів віком від 14 до 17 років, які мають базовий досвід користування смартфонами, але не були залучені до розробки системи. Кожному респонденту було запропоновано виконати набір типових завдань, зокрема: підключитися до LED-матриці, змінити яскравість, намалювати простий графічний елемент, запустити текстовий ефект та від'єднатися від пристрою.

Під час тестування фіксувалися такі показники: час виконання кожного завдання, кількість допущених помилок (некоректні натискання, вибір неправильного режиму, повторні спроби), а також суб'єктивна оцінка зручності інтерфейсу за п'ятибальною шкалою.

Результати ергономічної оцінки. Аналіз результатів показав (табл. 3.1), що середній час виконання базових завдань не перевищує однієї хвилини, кількість помилок є низькою та зменшується після першої взаємодії з системою, а середня суб'єктивна оцінка зручності інтерфейсу становить понад чотири бали з п'яти можливих. Це свідчить про те, що інтерфейс є інтуїтивно зрозумілим і не потребує спеціального навчання для використання.

Таблиця 3.1 – Результати ергономічного тестування

Респондент	Час виконання (с)	К-сть помилок	Оцінка зручності (1-5)
R1	42	1	4
R2	35	0	5
R3	48	2	4
R4	40	1	4
R5	37	0	5

Для більш узагальненої оцінки було визначено середні значення показників. Середній час виконання завдань становить 40,4 с, середня кількість помилок – 0,8, а середня оцінка зручності – 4,4 бала. Це підтверджує, що інтерфейс забезпечує ефективну взаємодію користувача із системою та не викликає значних труднощів при первинному використанні.

Найбільші труднощі у користувачів викликали режими, що використовують жести або сенсорні дані, що пояснюється відсутністю явних візуальних підказок. У зв'язку з цим було рекомендовано доповнити інтерфейс короткими інструктивними підказками або анімаціями при першому запуску відповідного режиму.

Отримані результати дозволяють зробити висновок, що розроблений інтерфейс відповідає основним вимогам ергономіки, забезпечує ефективну взаємодію користувача з системою та не створює надмірного когнітивного навантаження. Разом із тим, результати тестування вказують на доцільність подальшого вдосконалення інтерфейсу в частині навчальної підтримки користувача та адаптації керування у складніших режимах.

Ергономічна оцінка здійснювалася відповідно до рекомендацій міжнародного стандарту ISO 9241 (Ergonomics of human-system interaction) [12].

3.4 Техніко-економічне обґрунтування розробки

Метою техніко-економічного обґрунтування є визначення витрат на створення розроблюваної інформаційно-комп'ютерної системи та оцінка її економічної доцільності у порівнянні з існуючими рішеннями.

Основною статтею витрат при розробці програмного забезпечення є витрати на оплату праці розробника. Загальні трудові витрати визначаються як сума витрат часу на виконання окремих етапів розробки. Оцінку трудомісткості виконано на основі фактичного часу, витраченого на розробку системи.

На основі фактичного виконання роботи оцінено, що розробка системи потребувала (табл. 3.2).

Таблиця 3.2 – Оцінка трудомісткості розробки

Етап робіт	Трудомісткість, год
Аналіз вимог	12
Проектування	18
Реалізація	35
Тестування та налагодження	15
Документування	10
Разом	90

Вартість оплати праці розробника визначається за формулою (3.1) [13]:

$$Clab = T \times R, \quad (3.1)$$

де T – трудомісткість розробки, год;

R – погодинна ставка розробника.

За умови середньої ставки розробника $R=10$ €/год:

$$Clab = 90 \times 10 = 900 \text{ €}.$$

У процесі розробки використовувалися безкоштовні програмні засоби з відкритими ліцензіями (Android Studio, Kotlin, Android SDK), тому витрати на програмне забезпечення становлять:

$$Csoft = 0.$$

Окрім програмної складової, система включає апаратні компоненти, необхідні для її функціонування. Орієнтовну вартість апаратної частини наведено у таблиці 3.3.

Таблиця 3.3 – Вартість апаратних компонентів

Компонент	Вартість, €
Arduino	10
LED-матриця	20
Bluetooth-модуль	5
Разом	35

Загальна вартість розробки системи визначається за формулою (3.2) [14, 15]:

$$C_{\text{total}} = C_{\text{lab}} + C_{\text{soft}} + C_{\text{hw}}, \quad (3.2)$$

де C_{hw} – вартість апаратної складової.

Тоді:

$$C_{\text{total}} = 900 + 0 + 35 = 935 \text{ €}.$$

Отримане значення загальних витрат свідчить про відносно низьку вартість розробки інформаційно-комп'ютерної системи. Економічна доцільність розробки обумовлена відсутністю витрат на ліцензійне програмне забезпечення, можливістю багаторазового використання створеного продукту та його адаптацією до різних прикладних задач.

Крім того, у порівнянні з комерційними аналогами, розроблена система забезпечує достатній рівень функціональності при значно нижчих витратах, що підтверджує ефективність її створення.

Таким чином, реалізація мобільного застосунку для керування LED-матрицею є економічно доцільною, а отриманий програмно-апаратний комплекс може бути використаний у практичній діяльності без значних додаткових витрат.

ВИСНОВКИ

У кваліфікаційній роботі бакалавра вирішено актуальне науково-прикладне завдання розробки мобільного застосунку для віддаленого керування динамічною панеллю візуалізації на базі світлодіодної LED-матриці з використанням мікроконтролера Arduino та технології Bluetooth. Актуальність роботи обумовлена зростанням потреби у створенні мобільних, автономних та інтерактивних систем керування візуалізаційними пристроями, що забезпечують ефективну взаємодію користувача з апаратною складовою у режимі реального часу.

У процесі виконання роботи проведено аналіз сучасних інформаційно-комп'ютерних систем керування LED-матрицями, зокрема застосунків Pixel LED Art, LED Matrix Editor, Blynk IoT та WLED. За результатами аналізу встановлено, що існуючі рішення мають низку суттєвих обмежень, пов'язаних із недостатньою інтерактивністю, обмеженим функціоналом, складністю користувацького інтерфейсу та відсутністю комплексної підтримки режимів реального часу. Це підтвердило доцільність створення спеціалізованого мобільного застосунку, орієнтованого на поєднання зручного інтерфейсу, розширених можливостей керування та ефективної взаємодії з LED-матрицею через Bluetooth-з'єднання.

У роботі обґрунтовано вибір архітектури системи за моделлю «мобільний клієнт – вбудований пристрій», у межах якої мобільний застосунок виконує функції інтерфейсу користувача та генератора керувальних команд, а мікроконтролер Arduino забезпечує їх приймання, декодування та виконання. Для реалізації програмної частини обрано платформу Android, мову програмування Kotlin та засоби Android SDK, що забезпечили стабільність роботи застосунку, ефективну підтримку Bluetooth-з'єднання та можливість реалізації асинхронної взаємодії з апаратною частиною системи.

У процесі проєктування розроблено функціонально-структурну схему системи керування LED-матрицею, яка відображає взаємодію між мобільним застосунком, Bluetooth-модулем, мікроконтролером та світлодіодною матрицею. Побудовано UML-діаграму класів, що дозволило формалізувати структуру програмного забезпечення, визначити взаємозв'язки між основними програмними компонентами та забезпечити модульність і масштабованість системи.

У межах роботи розроблено алгоритми обробки користувацького введення, формування керувальних команд та їх передачі через Bluetooth-канал. Реалізовано алгоритм перетворення координат дотику у координати LED-матриці, алгоритм визначення напрямку свайпу, алгоритм згладжування сигналу акселерометра та механізм поетапної передачі команд для уникнення перевантаження каналу зв'язку. Використання зазначених алгоритмів забезпечило стабільну та коректну взаємодію користувача з системою у режимі реального часу.

Практичним результатом роботи стало створення повнофункціонального мобільного застосунку LumiMood для керування LED-матрицею на платформі Android із використанням мови програмування Kotlin. Розроблений застосунок забезпечує:

- підключення до LED-пристрою через Bluetooth;
- інтерактивне малювання;
- передавання текстових повідомлень;
- зміну кольорів і яскравості;
- запуск динамічних візуальних ефектів;
- використання акселерометра та жестового керування;
- підтримку інтерактивних режимів роботи.

У процесі реалізації забезпечено повноцінну взаємодію мобільного застосунку з мікроконтролером Arduino та реалізовано механізми передавання і обробки керувальних команд. Передача даних здійснюється у вигляді текстових команд, що містять тип операції та відповідні параметри.

Такий підхід забезпечив простоту реалізації, прозорість протоколу обміну даними та надійність функціонування системи.

Проведене тестування підтвердило коректність роботи основних функціональних компонентів системи, зокрема Bluetooth-з'єднання, алгоритмів формування команд, режимів малювання, текстового відображення та сенсорного керування. У результаті ергономічної оцінки встановлено, що середній час виконання типових операцій становить 40,4 с, середня кількість помилок – 0,8, а середня оцінка зручності інтерфейсу – 4,4 бала з 5 можливих. Отримані результати свідчать про високий рівень зручності, інтуїтивності та ефективності взаємодії користувача із системою.

У роботі також реалізовано базові механізми захисту інформаційно-комп'ютерної системи, зокрема валідацію керувальних команд, обмеження підключення до авторизованих Bluetooth-пристроїв та локальне збереження параметрів користувача. Це дозволило підвищити надійність функціонування системи та зменшити ризики несанкціонованого доступу до керування LED-матрицею.

Виконане техніко-економічне обґрунтування підтвердило доцільність розробки програмно-апаратного комплексу. Загальна трудомісткість створення системи становила 90 годин, а загальна вартість реалізації — 935 €. Використання безкоштовних програмних засобів, відкритих технологій та доступних апаратних компонентів забезпечило низьку собівартість розробки при достатньо високому рівні функціональності системи.

Отже, у роботі досягнуто поставленої мети та виконано всі визначені завдання. Розроблений мобільний застосунок є завершеним програмним продуктом, який забезпечує ефективне керування LED-матрицею у режимі реального часу та може бути використаний у навчальних, демонстраційних, рекламних і IoT-проектах. Перспективами подальшого розвитку системи є інтеграція з Wi-Fi та хмарними сервісами, розширення функціональних можливостей LED-візуалізації та підтримка складніших сценаріїв взаємодії користувача з програмно-апаратним комплексом.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wireless LED Dot Matrix Display using Arduino. International Journal for Scientific Research & Development (IJSRD). 2024. URL: <https://www.ijsrd.com/articles/IJSRDV12I30116.pdf> (дата звернення: 25.02.2026).
2. Bluetooth Controlled LED Matrix using Smartphone. Internet of Things Project Report. 2023. URL: <https://www.studocu.vn/vn/document/truong-dai-hoc-fpt/internet-of-things/bluetooth-controlled-led-matrix-final-report/123238408> (дата звернення: 28.02.2026).
3. Wireless LED Matrix Display using Bluetooth Technology. ResearchGate. 2023. URL: https://www.researchgate.net/publication/371911332Wireless_LED_Dot_Matrix_Display (дата звернення: 25.02.2026).
4. Pixel LED Art. Мобільний застосунок для керування LED-матрицями. URL: <https://play.google.com/store/apps/details?id=com.pixel.ledart> (дата звернення: 27.02.2026).
5. LED Matrix Editor. Мобільний застосунок для редагування та передачі зображень на LED-матрицю. URL: <https://play.google.com/store/apps/details?id=com.hanry.matrixled> (дата звернення: 22.02.2026).
6. Blynk IoT. Платформа для керування IoT-пристроями. URL: <https://blynk.io> (дата звернення: 25.02.2026).
7. WLED. Система керування адресними LED-пристроями. URL: <https://kno.wled.ge> (дата звернення: 25.02.2026).
8. Bluetooth SIG. Bluetooth Core Specification Version 5.0. URL: https://www.bluetooth.com/wp-content/uploads/2019/03/Bluetooth_5-FINAL.pdf?utm_source (дата звернення: 12.03.2026).

9. Monk S. Programming Arduino: Getting Started with Sketches. URL: https://agsci.colostate.edu/wp-content/uploads/sites/95/2020/03/Programming-Arduino.pdf?utm_source (дата звернення: 15.03.2026).
10. Nielsen J. Usability Engineering. URL: https://www.academia.edu/40437644/Usability_Engineering?utm_source (дата звернення: 18.04.2026).
11. Kumar R. Human Computer Interaction / Usability Engineering. URL: https://www.researchgate.net/publication/347256975_Usability_Engineering (дата звернення: 18.04.2026).
12. ISO 9241. Ergonomics of human-system interaction. URL: https://www.userfocus.co.uk/pdf/ISO9241_EN.pdf?utm_source (дата звернення: 21.04.2026).
13. Boehm B. et al. Cost Models for Future Software Life Cycle Processes. URL: <https://csse.usc.edu/TECHRPTS/2000/usccse00-505/usccse00-505.pdf> (дата звернення: 02.05.2026).
14. Методичні вказівки до економічного обґрунтування програмних розробок. URL: https://ela.kpi.ua/bitstream/123456789/27747/1/Ekonomika_programnih_rozrobok.pdf (дата звернення: 03.05.2026).
15. OECD. Measuring the Digital Economy. URL: <https://www.oecd.org/digital/measuring-the-digital-economy.html> (дата звернення: 03.05.2026).