НАЦІОНАЛЬНИЙ УНІВЕРСТИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	Завідувач кафедри
Інформаційних технологій	Комп'ютерних систем, мереж і кібербезпеки
Болбот І.М., д.пед.н., проф.	Касаткін Д.Ю., к.пед.н., доц.
підпис ПІБ, вчене звання і ступінь	підпис ПІБ, вчене звання і ступінь
«_____»___2025 р.	«_____»___2025 р.

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

На тему:« Розробка комп‘ютеризованої системи обліку відвідувачів»

Спеціальність – 123 «Комп’ютерна інженерія»

Гарант освітньої програми

д.т.н., доц. Шкарупило В.В.

(науковий ступінь та вчене звання) (підпис) (ПІБ)

Керівник бакалаврської кваліфікаційної роботи

д пед.н.,проф. Мамченко С.М.

(науковий ступінь та вчене звання) (підпис) (ПІБ)

Виконав _____ Коваль М.С.

(підпис) (ПІБ студента)

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

Факультет (ННІ) інформаційних технологій

АТВЕРДЖУЮ
Завідувач кафедри
к.п.н., доцент Касаткін Д.Ю.

_____ _____ _____
(науковий ступінь, вчене звання) (підпис) (ім'я ПРІЗВИЩЕ)
"____" _____2025року

З А В Д А Н Н Я

ДО ВИКОНАННЯ МАГІСТЕРСЬКОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ ЗДОБУВАЧУ

Коваль Марк Станіславович
(прізвище, ім'я, по батькові)

Спеціальність
Комп'ютерна інженерія
(код і найменування)

Освітня програма
Комп'ютерні системи і мережі
(назва)

Орієнтація освітньої програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Тема магістерської кваліфікаційної роботи Дослідження комп'ютерної системи обліку відвідувачів

затверджена наказом від "_29_"_10_2024_р. № 1941-С

Термін подання завершеної роботи на кафедру 2025.11.12_
(рік, місяць, число)

Вихідні дані до магістерської кваліфікаційної роботи _____
_____
_____

Перелік питань, що підлягають дослідженню:

1. Опрацювати питання отримання інформації про вхід/вихід людини. Обрати відповідний датчик прийому сигналу.
2. Розробити схему підключення датчика да контролера та написання коду програми._
3. Описати отримані результати та проблеми, що були вирішені при розроблення проекту.

Перелік графічного матеріалу (за потреби) _____
_____

Дата видачі завдання "__03__"__лютого_2025 р.

Керівник магістерської кваліфікаційної роботи _____ Мамченко С.М.
(підпис) (ім'я ПРІЗВИЩЕ)

Завдання прийняв до виконання _____ Коваль М.С.
(ім'я ПРІЗВИЩЕ)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз вимог до системи	05.07.2025 р.	Виконано
2	Проєктування системи	15.08.2025 р.	Виконано
3	Реалізація системи	02.09.2025 р.	Виконано
4	Тестування розробленої системи	15.09.2025 р.	Виконано
5	Оформлення пояснювальної записки	25.10.2024 р.	Виконано
6	Оформлення графічного матеріалу	03.11.2025 р.	Виконано

Студент Коваль М.С.

(підпис) (ініціали та прізвище)

Керівник проекту (роботи) Мамченко С.М.

РЕФЕРАТ

Пояснювальна записка: 57 сторінок , 22 рисунків, 21 джерел.

РОЗРОБКА КОМП’ЮТЕРИЗОВАНОЇ СИСТЕМИ.

Об'єктом дослідження є розробка комп’ютеризованої системи обліку відвідувачів з використанням двох інфрачервоних датчиків E18-D80NK, спрямованої на оптимізацію обліку відвідувачів у приватних будинках.

У роботі розглядаються чотири розділи.

У першому розділі роботи проведено аналіз існуючих систем обліку відвідувачів, їх функціональних можливостей та технічних характеристик. Було розглянуто основні підходи до автоматизації обліку, включаючи використання пропускних систем, RFID-технологій, а також систем на базі відеоспостереження. Аналіз показав, що більшість систем мають певні недоліки, які впливають на їх ефективність та зручність використання.

Другий розділ присвячений проєктуванню комп'ютеризованої системи обліку відвідувачів. Описано архітектуру системи, яка складається з мікроконтролера, датчиків руху та дисплея для виведення інформації. Було розроблено структурну схему системи та описано принципи її роботи. Особливу увагу приділено вибору компонентів та їх інтеграції для забезпечення надійної роботи системи.

У третьому розділі детально описано процес реалізації системи. Представлено програмний код, який забезпечує функціонування системи на базі мікроконтролера Arduino. Описано алгоритми роботи, зокрема обробку сигналів від датчиків, підрахунок відвідувачів та виведення інформації на дисплей. Також розглянуто питання енергоспоживання та надійності системи.

Четвертий розділ присвячений тестуванню розробленої системи та оцінці її ефективності. Проведено тестові випробування в різних умовах, що дозволило оцінити точність підрахунку відвідувачів та стабільність роботи системи. Результати тестувань підтвердили високу ефективність системи та її відповідність поставленим вимогам. На основі отриманих даних зроблено висновки про можливість впровадження системи в практичну експлуатацію.

ABSTRACT

Explanatory note: 57 pages, 22 figures, 15 sources.

DEVELOPMENT OF A COMPUTERIZED SYSTEM.

The object of the research is the development of a computerized system of accounting for visitors using two infrared sensors E18-D80NK, aimed at optimizing the accounting of visitors in private houses.

The work deals with four sections.

In the first part of the work, an analysis of the existing visitor registration systems, their functionality and technical characteristics was carried out. The main approaches to accounting automation were considered, including the use of access systems, RFID technologies, as well as systems based on video surveillance. The analysis showed that most systems have certain shortcomings that affect their efficiency and usability.

The second section is devoted to the design of a computerized visitor registration system. The architecture of the system, which consists of a microcontroller, motion sensors and a display for outputting information, is described. A structural diagram of the system was developed and the principles of its operation were described. Special attention was paid to the selection of components and their integration to ensure reliable system operation.

The third chapter describes the system implementation process in detail. The program code that ensures the functioning of the system based on the Arduino microcontroller is presented. Work algorithms are described, in particular, processing signals from sensors, counting visitors and displaying information on the display. The issue of energy consumption and system reliability is also considered.

The fourth chapter is devoted to testing the developed system and evaluating its effectiveness. Tests were conducted in various conditions, which allowed us to assess the accuracy of the visitor count and the stability of the system. The results of the tests confirmed the high efficiency of the system and its compliance with the set requirements. Based on the obtained data, conclusions were made about the possibility of introducing the system into practical operation.

ЗМІСТ

ВСТУП

Сучасні технології інформаційних систем та автоматизації відіграють ключову роль у розвитку різних галузей, забезпечуючи ефективне управління ресурсами, покращення якості обслуговування та підвищення безпеки. Одним із таких важливих напрямів є автоматизований облік відвідувачів, який дозволяє значно спростити процес контролю доступу до різних установ, а також забезпечує надійне збереження даних та підвищує рівень безпеки.

Актуальність теми дипломної роботи зумовлена потребою у створенні ефективних та надійних систем обліку відвідувачів, які б відповідали сучасним вимогам і стандартам. Традиційні методи обліку, такі як ручний запис відвідувачів або використання простих електронних систем, часто не відповідають вимогам часу через свою недостатню функціональність, низький рівень безпеки та складність в управлінні великими обсягами даних. В цьому контексті розробка комп'ютеризованої системи обліку відвідувачів є надзвичайно актуальною та важливою задачею.

Метою даної дипломної роботи є розробка комп'ютеризованої системи обліку відвідувачів, яка базується на використанні сучасних технологій, зокрема мікроконтролерів Arduino, та забезпечує автоматизацію процесів контролю доступу і обліку відвідувачів. Основними задачами, які ставляться в рамках цього дослідження, є аналіз існуючих рішень, проектування системи, розробка програмного забезпечення, а також тестування та оцінка ефективності розробленої системи.

У першому розділі роботи здійснено огляд існуючих технологій та методів, які використовуються для автоматизації обліку відвідувачів. Визначено основні недоліки та обмеження традиційних систем та запропоновано концепцію нового підходу, що базується на використанні сучасних інформаційних технологій.

Другий розділ присвячений детальному опису процесу проектування системи обліку відвідувачів, включаючи вибір апаратного та програмного забезпечення, розробку структури бази даних та алгоритмів обробки інформації.

Третій розділ містить результати розробки та тестування системи, аналіз її функціональності та ефективності. Також розглянуто питання безпеки даних та можливості інтеграції з іншими системами.

У четвертому розділі представлено висновки та рекомендації щодо впровадження розробленої системи у практику, а також визначено перспективи подальшого розвитку та вдосконалення системи.

Таким чином, дана дипломна робота має на меті не лише розробку конкретного технічного рішення, але й сприяння розвитку галузі інформаційних технологій, зокрема у сфері автоматизації та безпеки.

1. ЗАГАЛЬНІ ВІДОМОСТІ ПРО СИСТЕМИ ОБЛІКУ ВІДВІДУВАЧІВ

1.1 Поставлення мети роботи

Розробка системи обліку відвідувачів є важливим завданням, яке має велике значення для різних галузей, таких як комерційні об'єкти, освітні установи, музеї, виставкові зали, офіси та багато інших.

Метою цієї роботи є створення ефективної, надійної та легкої у використанні системи, яка дозволяє точно підраховувати кількість відвідувачів у визначеному просторі, а також надавати цю інформацію у зручному для аналізу форматі.

1.2 Опис мови програмування

Мова програмування C/C++ для Arduino.

Мова програмування C/C++ є основною мовою, яка використовується для програмування мікроконтролерів, зокрема платформи Arduino. Мова C була розроблена на початку 1970-х років і швидко стала однією з найбільш популярних мов програмування для системного і прикладного програмування. Вона забезпечує високу ефективність, контроль над апаратним забезпеченням і зручний синтаксис, що робить її ідеальною для розробки вбудованих систем.

Особливості мови програмування для Arduino:

Синтаксис C/C++: програми для Arduino пишуться на C/C++ з використанням спеціальних бібліотек, що спрощують роботу з апаратними компонентами.

Стандартні структури, такі як цикли, умовні оператори, функції і масиви, забезпечують потужність і гнучкість в розробці програм.

Бібліотеки Arduino: Arduino IDE включає велику кількість бібліотек, які надають готові функції для роботи з різними датчиками, модулями зв'язку, дисплеями та іншими компонентами.

Бібліотека LiquidCrystal дозволяє легко працювати з LCD-дисплеями, надаючи функції для ініціалізації дисплея, виведення тексту та налаштування курсора.

Простота у використанні:

Arduino IDE (інтегроване середовище розробки) забезпечує простий у використанні інтерфейс для написання коду, його компіляції та завантаження на плату Arduino.

Наявність прикладів і великої кількості документації полегшує початок роботи навіть для новачків.

Підтримка апаратних переривань:мова C/C++ для Arduino підтримує роботу з апаратними перериваннями, що дозволяє реагувати на події, такі як зміна стану датчика, в реальному часі.

Це забезпечує можливість створення ефективних і швидкодіючих програм.

Можливість низькорівневого програмування:мова C/C++ дозволяє програмістам працювати безпосередньо з регістрами мікроконтролера, що забезпечує високий рівень контролю над апаратними ресурсами.

1.3 Принципи роботи системи обліку відвідувачів.

Основний принцип роботи систем обліку відвідувачів полягає у виявленні та реєстрації кожного відвідувача, що входить або виходить з приміщення. Це може здійснюватися різними способами в залежності від типу використовуваних датчиків і технологій.

Інфрачервоні датчики працюють за принципом виявлення змін інфрачервоного випромінювання, яке випромінює тіло людини. Коли людина проходить через зону контролю, датчик реєструє зміну рівня інфрачервоного випромінювання, що дозволяє фіксувати факт входу чи виходу.

Камери спостереження з технологією розпізнавання облич аналізують зображення в реальному часі, порівнюють обличчя відвідувачів з базою даних та ідентифікують їх. Це дозволяє не лише вести облік, а й підвищувати рівень безпеки.

RFID/NFC технології забезпечують швидку та точну ідентифікацію відвідувачів за допомогою спеціальних карток або смартфонів. Кожна картка має унікальний код, який зчитується при вході чи виході, що дозволяє точно вести облік.

На рис. 1 зображено блок-схема системи обліку відвідувачів:

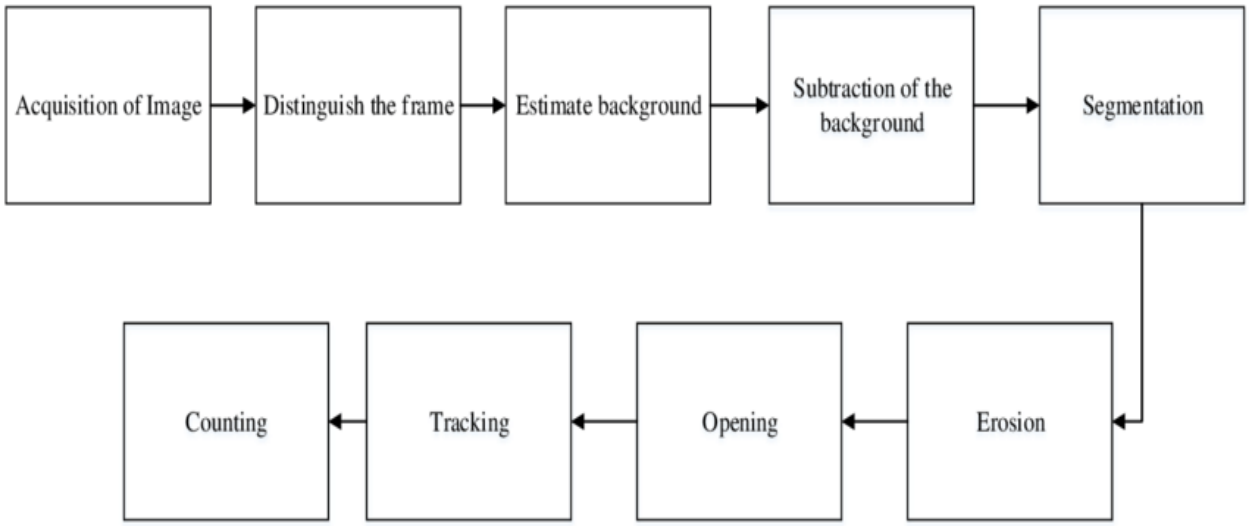


Рис. 1 Блок-схема системи обліку відвідувачів

1.4 Переваги використання систем обліку відвідувачів

Використання систем обліку відвідувачів має численні переваги:

Підвищення рівня безпеки: Системи дозволяють ефективно контролювати доступ до приміщень, виявляти несанкціонованих відвідувачів та запобігати потенційним загрозам. Завдяки точному обліку відвідувачів можна виявляти підозрілу активність і швидко реагувати на можливі загрози.

Оптимізація управління простором: Завдяки точному обліку відвідувачів, можна краще розуміти, як використовується простір, і приймати обґрунтовані рішення щодо його оптимізації. Це може включати аналіз часу пікових навантажень та визначення популярних зон у будівлі.

Комфорт для мешканців: Автоматизація процесів обліку відвідувачів дозволяє зменшити втручання в особисте життя мешканців і забезпечити їм більший комфорт. Наприклад, система може автоматично відчиняти двері для зареєстрованих мешканців або гостей, що підвищує зручність користування.

Ефективність управління: Центральне управління всіма компонентами системи з використанням програмного забезпечення дозволяє зменшити витрати на обслуговування та підвищити загальну ефективність системи. Автоматизовані звіти та аналітика допомагають управляти ресурсами більш раціонально.

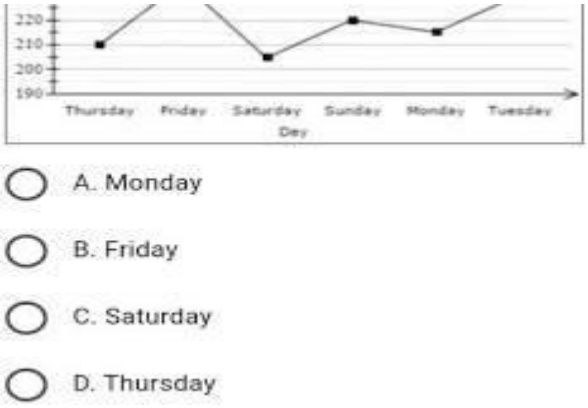


Рис.2 графік залежності кількості відвідувачів від часу

1.5 Основні структурні моделі системи обліку відвідувачів

Системи обліку відвідувачів стають все більш популярними, оскільки вони допомагають підприємствам та організаціям краще розуміти, хто відвідує їхні приміщення, коли вони приходять і скільки часу вони проводять там. Ця інформація може бути використана для покращення безпеки, оптимізації ресурсів та прийняття більш обґрунтованих бізнес-рішень.

Структурні моделі системи обліку відвідувачів

Існує декілька основних структурних моделей системи обліку відвідувачів, кожна з яких має свої переваги та недоліки.

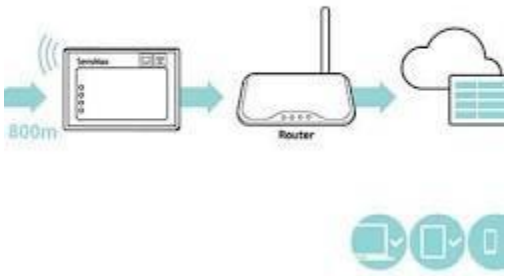


Рис.3 Централізована модель

У цій моделі всі компоненти системи обліку відвідувачів розташовані в одному центральному місці, як правило, на сервері.

Переваги:

- Простота налаштування та обслуговування
- Легкий доступ до даних
- Високий рівень безпеки

Недоліки:

- Може бути дорогою у впровадженні та експлуатації
- Не масштабується добре для великих систем
- Може бути вразливою до збоїв

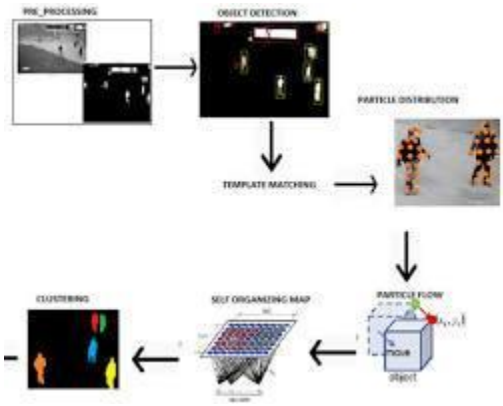


Рис. 4 Розподілена модель

1.6 Опис та принцип роботи системи обліку відвідувачів

Система обліку відвідувачів призначена для автоматичного підрахунку кількості людей, що входять та виходять з приміщення. Вона використовується в різних місцях, таких як магазини, офіси, музеї та інші громадські місця для моніторингу потоку відвідувачів та оптимізації управління ресурсами.

Основні компоненти системи включають мікроконтролер (зазвичай платформа Arduino), інфрачервоні (ІЧ) датчики, LCD-дисплей для виведення інформації, а також додаткові елементи, такі як світлодіоди для індикації.

Опис системи: **мікроконтролер (Arduino):** Мікроконтролер є центральним елементом системи, який збирає дані з датчиків, обробляє їх і керує іншими компонентами. Arduino обирається за свою простоту, доступність і велику кількість бібліотек, які спрощують розробку.

Інфрачервоні датчики (IR): Датчики встановлюються на вході та виході з приміщення і виявляють рух, коли людина перетинає їхній промінь. Кожен датчик підключений до цифрового входу Arduino і забезпечує сигнал, коли виявляє об'єкт.

LCD-дисплей: Дисплей використовується для виведення інформації про кількість відвідувачів, що увійшли та вийшли з приміщення. Це забезпечує зручний спосіб моніторингу даних у реальному часі.

Світлодіоди (LED): Світлодіоди можуть використовуватися для індикації стану системи, наприклад, для сигналізації про виявлення руху або про зміну кількості людей у приміщенні.

2. СТРУКТУРА ТА ФУНКЦІОНАЛ СИСТЕМИ

2.1 Етапи розробки системи

Процес розробки системи обліку відвідувачів включає кілька ключових етапів, кожен з яких є важливим для створення ефективної та надійної системи. Нижче наведено детальний опис кожного з цих етапів.

Аналіз вимог - першим етапом у розробці системи є аналіз вимог.

Цей етап включає:

- Визначення цілей системи: з'ясування, які конкретно задачі має вирішувати система, зокрема облік кількості відвідувачів, фіксація часу їх входу та виходу, а також додаткові функції, як-от керування освітленням.
- Аналіз користувачів: виявлення потенційних користувачів системи, їхніх потреб та очікувань від системи.
- Технічні вимоги: визначення необхідних апаратних та програмних компонентів, що забезпечуватимуть функціональність системи.

Проектування системи - на етапі проектування розробляються структурна та функціональна архітектура системи:

- Вибір компонентів: вибір відповідних датчиків (наприклад, E18-D80NK), контролера (Arduino Uno), дисплеїв (LCD), та інших компонентів.
- Схематичне проектування: розробка електричних схем для з'єднання всіх компонентів.
- Структурна архітектура: визначення основних модулів системи, їх взаємодії та обміну даними між ними.

Розробка програмного забезпечення - цей етап включає написання коду, який забезпечує роботу системи:

- Програмування на Arduino: розробка алгоритмів для обробки сигналів від датчиків, підрахунку відвідувачів, управління дисплеєм та іншими компонентами.

- Інтеграція бібліотек: використання бібліотек, таких як LiquidCrystal.h, для спрощення роботи з апаратними компонентами.

- Тестування та налагодження коду: виявлення та виправлення помилок, оптимізація роботи програмного забезпечення.

Збірка та тестування системи - на цьому етапі здійснюється фізична збірка системи:

- Монтаж компонентів: з'єднання всіх апаратних частин згідно з розробленими схемами.
- Первинне тестування: перевірка працездатності системи у статичних умовах для виявлення можливих помилок у з'єднаннях або компонентах.
- Функціональне тестування: тестування системи в реальних умовах для перевірки її здатності точно рахувати відвідувачів та виконувати всі необхідні функції.

Збірка та тестування системи - на цьому етапі здійснюється фізична збірка системи:

- Монтаж компонентів: з'єднання всіх апаратних частин згідно з розробленими схемами.
- Первинне тестування: перевірка працездатності системи у статичних умовах для виявлення можливих помилок у з'єднаннях або компонентах.
- Функціональне тестування: тестування системи в реальних умовах для перевірки її здатності точно рахувати відвідувачів та виконувати всі необхідні функції.

Впровадження та підтримка - завершальний етап передбачає введення системи в експлуатацію та її підтримку:

- Встановлення системи: монтаж системи на об'єкті, налаштування та запуск у роботу.
- Навчання користувачів: надання інструкцій та навчання персоналу, який буде працювати з системою.

- Технічна підтримка: забезпечення постійної підтримки та обслуговування системи для забезпечення її стабільної роботи та оперативного вирішення виникаючих проблем.

Етапи розробки системи обліку відвідувачів вимагають ретельного планування та виконання, адже тільки так можна досягти високої якості та надійності кінцевого продукту. Кожен етап є взаємопов'язаним та важливим для успішного завершення проекту.

2.2 Алгоритм системи обліку відвідувачів

Алгоритм системи обліку відвідувачів є серцем функціонування системи. Він відповідає за обробку сигналів від датчиків, підрахунок кількості відвідувачів, управління вивідними пристроями та збереження даних. Нижче описано покроковий процес функціонування алгоритму системи обліку відвідувачів.

Ініціалізація системи - на початковому етапі роботи системи здійснюється ініціалізація всіх апаратних компонентів та налаштування початкових умов:

- Ініціалізація контролера: встановлення всіх необхідних параметрів для роботи контролера Arduino, таких як швидкість серійного з'єднання, режим роботи пінів (вхідний або вихідний).
- Ініціалізація датчиків: налаштування датчиків руху (наприклад, E18-D80NK) для виявлення відвідувачів.
- Ініціалізація дисплея: використання бібліотеки LiquidCrystal.h для налаштування LCD-дисплея, що буде використовуватися для відображення інформації про кількість відвідувачів.

Збір та обробка даних від датчиків - алгоритм постійно зчитує дані від датчиків руху, аналізуючи їх для виявлення подій входу та виходу відвідувачів:

- Зчитування даних: Регулярне опитування датчиків для отримання їх поточного стану (активний або неактивний).

- Аналіз даних: Перевірка змін стану датчиків для визначення, чи перетнув відвідувач контрольну зону. Це може включати фільтрацію помилкових спрацьовувань та шумів.
- Визначення подій: Виявлення подій входу та виходу на основі послідовності спрацьовування датчиків (наприклад, спочатку активується датчик на вході, потім на виході — це означає вхід відвідувача).

Підрахунок кількості відвідувачів - алгоритм відповідає за коректний підрахунок кількості відвідувачів на основі даних від датчиків:

- Збільшення лічильника входів: при виявленні події входу, алгоритм збільшує лічильник вхідних відвідувачів.
- Збільшення лічильника виходів: при виявленні події виходу, алгоритм збільшує лічильник вихідних відвідувачів.
- Обчислення поточної кількості відвідувачів: різниця між лічильниками входів та виходів дає поточну кількість відвідувачів у приміщенні.

Збереження даних - для зберігання історичних даних та аналізу відвідуваності система може використовувати зовнішні носії інформації або серійний порт для передачі даних:

- Збереження в EEPROM: Збереження важливих даних у внутрішній пам'яті контролера Arduino, що дозволяє зберігати дані навіть після вимкнення живлення.
- Передача даних на комп'ютер: Використання серійного порту для передачі даних до комп'ютера, де вони можуть бути збережені та проаналізовані.
- Інтеграція з базами даних: Можливість інтеграції з базами даних для довготривалого зберігання та аналізу даних про відвідувачів.

Обробка помилок та виключень - для забезпечення надійної роботи система повинна мати механізми обробки помилок та виключних ситуацій:

- Виявлення помилок датчиків: Моніторинг стану датчиків для виявлення можливих збоїв або некоректних даних.
- Обробка нештатних ситуацій: Введення алгоритмів, що дозволяють відновити роботу системи у випадку виникнення помилок.

Відображення інформації про помилки на дисплеї або передача відповідних повідомлень через серійний порт. Таким чином, алгоритм системи обліку відвідувачів є багатокомпонентним процесом, що включає ініціалізацію компонентів, збір та обробку даних, підрахунок відвідувачів, відображення інформації, збереження даних та обробку помилок. Кожен з цих кроків є критично важливим для забезпечення коректної та надійної роботи системи.

2.3 Вибір компонентів для розробик системи.Опис технічної складової.

Arduino Uno – це один з найпопулярніших і найширше використовуваних мікроконтролерів у сімействі Arduino. Він відзначається своєю простотою у використанні, широкими можливостями для підключення різноманітних датчиків та пристроїв, а також великою спільнотою користувачів, що забезпечує легкий доступ до документації та прикладів коду.

Arduino Uno базується на мікроконтролері ATmega328P від компанії Atmel. Він має 14 цифрових входів/виходів (з яких 6 можуть використовуватися як виходи PWM), 6 аналогових входів, кварцовий генератор на 16 МГц, роз'єм для програмування через USB, роз'єм для живлення, роз'єм ICSP і кнопку перезавантаження.

Може живитися як від USB-порту комп'ютера, так і від зовнішнього джерела живлення, наприклад, адаптера на 9V або 12V. Вбудований регулятор напруги забезпечує стабільне живлення всіх компонентів плати. Плата має 14 цифрових входів/виходів, які можна налаштувати як входи або виходи, та шість з них можуть використовуватися для виходу сигналів PWM, що дозволяє керувати такими пристроями, як світлодіоди або мотори.

У системі обліку відвідувачів Arduino Uno виконує роль головного контролера, який зчитує сигнали з датчика руху E18-D80NK, обробляє ці дані та керує іншими компонентами системи, такими як дисплей для відображення інформації про кількість відвідувачів.

Простота програмування та багатофункціональність Arduino Uno дозволяють легко налаштовувати і модифікувати систему відповідно до конкретних вимог проекту.

Arduino Uno є ідеальним вибором для даного проекту завдяки своїй надійності, доступності і широкій підтримці від спільноти розробників, що забезпечує великий обсяг навчальних матеріалів і прикладів для реалізації різноманітних проектів.

Рис 4 - Arduino Uno

Датчик E18-D80NK – це інфрачервоний датчик руху, який використовується для виявлення об'єктів на певній відстані. Його принцип роботи базується на випромінюванні інфрачервоного світла і виявленні його відбиття від об'єкта.

Цей датчик є надзвичайно популярним завдяки своїй точності, надійності і простоті інтеграції в різні проекти.

Технічні характеристики датчика E18-D80NK:

- вхідна напруга: 5V DC (постійний струм);
- споживання струму: > 25mA (min) ~ 100mA (max)
- розміри: 1.7 см (діаметр) x 4.5 см (довжина)
- довжина кабелю: 45 см;
- виявлення об'єктів: як прозорих, і непрозорих;
- діапазон виявлення: від 3 до 80 см (залежно від положення гвинта на задній стороні датчика);
- тип виходу: NPN (normally high);
- діапазон робочих температур: -25 °C ~ 55 °C.

Рис. - 5 датчик E18D80NK

Один з головних плюсів цього датчика – його висока точність і швидкість реагування. Він здатний миттєво виявляти об'єкти, що дозволяє використовувати його в реальному часі. Також E18-D80NK стійкий до впливу зовнішніх факторів, таких як зміни освітлення або температури, що робить його надійним вибором для використання в різних умовах.

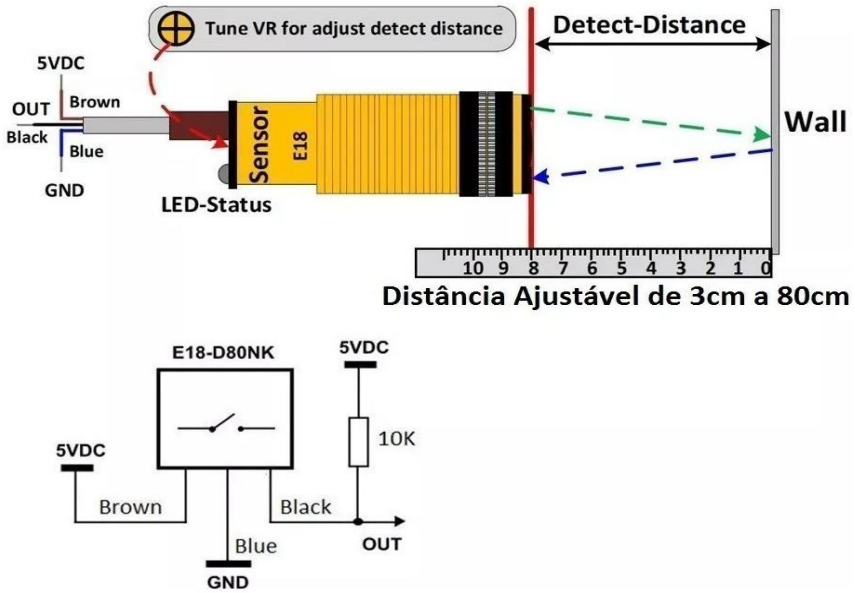


Рис. 6 - схема E18-D80NK

LCD 1602 - це широко використовуваний модуль рідкокристалічного дисплея (LCD), який знаходить застосування в різних електронних проектах.

Опис: Розмір: 16 символів у рядку та 2 рядки (дисплей 16x2) Підсвічування: зелене також і синє, забезпечує освітлення для перегляду символів при слабкому освітленні.

Набір символів: підтримує стандартний алфавітно-цифровий набір символів разом із деякими символами.

Опис його ключових характеристик та функціональних можливостей: Відображає текст і числа**:** можна запрограмувати LCD для відображення текстових повідомлень, чисел або навіть символів на його екрані.

Інтерфейс керування**:** використовують в базових інтерфейсах, які потребують декількох контактів керування з вашої мікроконтролерної плати для управління дисплеєм.

Рис. – 7 LCD 1602 дисплей

Джерело живлення: Arduino Uno має спеціальний роз'єм для підключення адаптера живлення. Цей роз'єм підтримує діапазон напруги від 7V до 12V, що робить адаптери на 9V або 12V ідеальними для живлення плати.

Важливо, щоб полярність підключення була правильною: центральний пін роз'єму має бути підключений до позитивного полюсу адаптера, а зовнішній – до негативного.

Малюнок – 8 (9V,12V джерело живлення)

Переваги використання адаптера живлення 9V або 12V

- Стабільність живлення: Адаптери забезпечують стабільне електроживлення, що дуже важливо для надійної роботи мікроконтролера та підключених компонентів.
- Безперервна робота: Використання адаптера дозволяє системі працювати безперервно без необхідності заміни батарей.
- Простота підключення: Адаптери легко підключаються до роз'єму живлення на платі Arduino.
- Достатня потужність: Вихідна потужність адаптерів 9V або 12V зазвичай достатня для живлення не тільки самого Arduino, але і додаткових компонентів, таких як датчики, дисплеї та інші периферійні пристрої.
- Недоліки використання адаптера живлення 9V або 12V
- Залежність від електромережі: Для роботи адаптера необхідний доступ до електричної розетки, що обмежує портативність системи.
- Вимоги до якості: Вибір низькоякісного адаптера може призвести до нестабільності роботи або навіть пошкодження компонентів системи через коливання напруги.

Натискна кнопка: натискна кнопка є механічними перемикачами, які замикають або розмикають електричний ланцюг при натисканні. Вони мають два основні стани: відкритий (розімкнутий) і закритий (замкнутий). Коли кнопка не натиснута, контакти кнопки розімкнуті, і через неї не проходить струм.

При натисканні контакти замикаються, дозволяючи струму проходити через ланцюг. Натискна кнопка підключаються до цифрових входів мікроконтролера Arduino. Один контакт кнопки підключається до землі (GND), а інший – до цифрового входу і через резистор до живлення (5V).

Цей резистор, який називається підтягуючим резистором (pull-up resistor), забезпечує стабільний рівень напруги на вході, коли кнопка не натиснута.

Рис. – 9 натискна кнопка

З'єднувальні дроти: за допомогою дротів можна підключити :підключення датчика руху E18-D80NK: вивід "Vcc" датчика підключається до 5V на Arduino за допомогою з'єднувального дроту.Вивід "GND" датчика підключається до землі (GND) на Arduino.Сигнальний вивід датчика підключається до одного з цифрових входів на Arduino.

Підключення LCD-дисплея:Виводи живлення LCD-дисплея (Vcc і GND) підключаються до відповідних виводів на Arduino. Керуючі виводи дисплея (RS, E, D4-D7) підключаються до відповідних цифрових виходів на Arduino за допомогою з'єднувальних дротів.

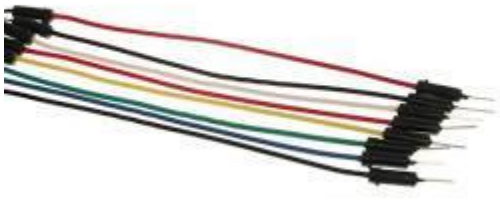


Рис. – 10 Джамперні дроти
(мама-папа, мама-мама, папа-папа)

Резистори: при підключенні світлодіодів до Arduino завжди використовуються резистори для обмеження струму. Резистор номіналом 220 Ом використовується для обмеження струму через світлодіод до безпечного рівня.

Кнопку ч підключаємодо Arduino через підтягуючі резистори. Це забезпечує стабільний високий або низький рівень напруги, коли кнопка не натиснута.

Резистори використовуються для створення дільників напруги, які дозволяють вимірювати напругу за допомогою аналогових входів Arduino.

Підключення світлодіод:

- Один вивід світлодіод підключається до цифрового виходу Arduino через резистор номіналом 220 Ом.
- Інший вивід світлодіода підключається до землі (GND).
- Підключення кнопки:
- Один вивід кнопки підключається до цифрового входу Arduino.
- Інший вивід підключається до землі (GND).
- Між цифровим входом і живленням (5V) встановлюється підтягуючий резистор номіналом 10 кОм.

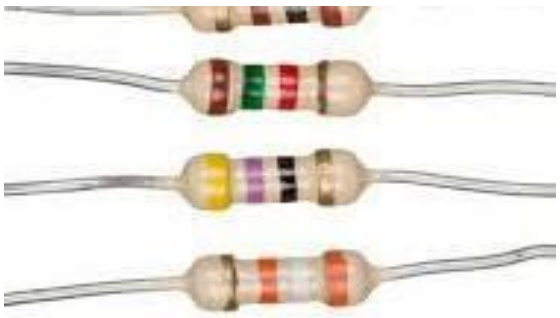


Рис. – 11 резистори

Макетна плата Breadboard Half: макетна плата складається з пластикової основи, в яку вмонтовані рядки і стовпці контактних отворів. Контактні отвори з'єднані між собою металевими доріжками, що дозволяє створювати електричні з'єднання між компонентами без використання пайки.

Зазвичай плати прототипування мають наступну структуру: живильні шини (power rails): дві вертикальні лінії з контактних отворів, розташовані з обох боків плати, призначені для підключення живлення (5V і GND). Це дозволяє легко підключати живлення до різних компонентів на платі.

Робоча область (terminal strips): горизонтальні ряди контактних отворів, розташовані в центральній частині плати, де компоненти підключаються один до одного. Кожен ряд зазвичай складається з п'яти отворів, з'єднаних між собою.

Підключення до Arduino: плата прототипування часто використовується разом з мікроконтролерами Arduino. Виводи Arduino підключаються до макетної плати за допомогою з'єднувальних дротів, що дозволяє легко тестувати і модифікувати схеми.

Підключення датчика руху:виводи датчика руху E18-D80NK підключаються до відповідних контактних отворів на платі. Живлення (5V і GND) підключається до живильних шин, а сигнальний вивід датчика підключається до цифрового входу Arduino через робочу область плати.

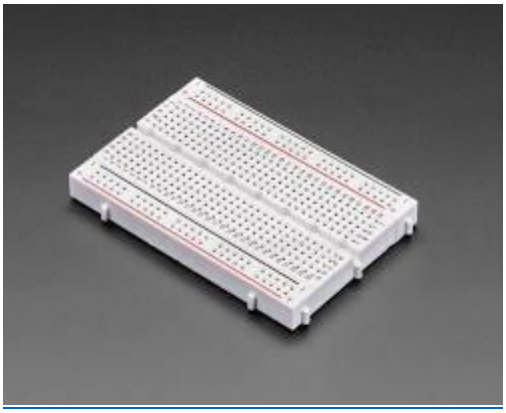


Рис. – 12 макетна плата Breadboard Half

Зовнішній корпус:корпус захищає компоненти системи від пошкоджень та надати їй більш естетичний вигляд.

Рис. – 13 Прототип зовнішнього корпусу

Виявлення руху та облік відвідувачів: датчик руху E18-D80NK є важливим компонентом системи обліку відвідувачів, оскільки саме він відповідає за виявлення руху. Датчик використовує інфрачервоне випромінювання для виявлення змін у середовищі. При потраплянні об'єкта в зону дії датчика, відбувається зміна інтенсивності відбитого інфрачервоного випромінювання, що і фіксується датчиком.

E18-D80NK має кілька переваг, таких як висока чутливість та можливість налаштування робочої відстані. Він здатний виявляти рух на відстані до 80 см, що дозволяє реєструвати проходження людей через зону виявлення. Виявлення руху реалізується шляхом безперервного сканування простору і надсилання сигналу до мікроконтролера Arduino при виявленні змін.

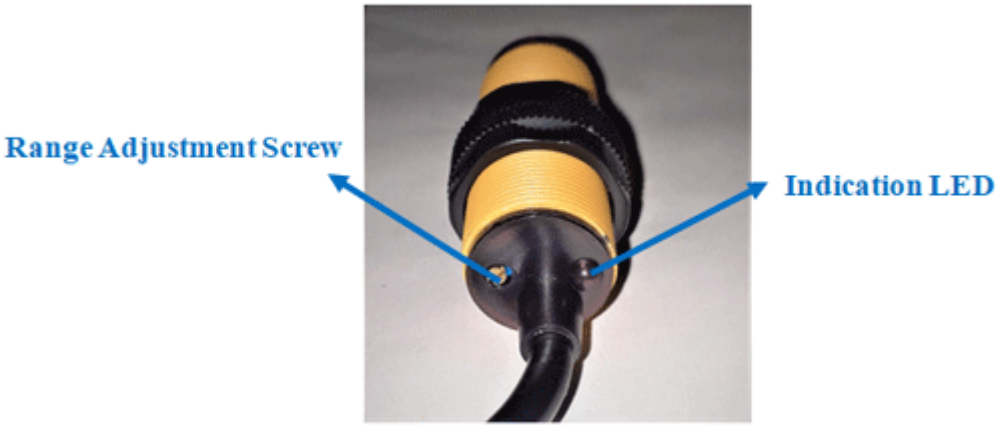


Рис. 14 - регулювання датчика

Діапазон виявлення може бути індивідуально відрегульований для кожного застосування за допомогою спеціального гвинта, розміщеного на звороті датчика. Сигнал на виході датчика змінюється залежно від перешкод. Коли ніяких перешкод не виявлено, він має високий рівень (high), при виявленні перешкод він змінює свій стан на low (низький рівень). На звороті датчика крім гвинта регулювання діапазону виявлення також розташований світлодіод червоного кольору, який завжди включається при виявленні перешкод.

Датчик E18 працює від 5V напруги і в режимі спокою споживає струм від 5mA до 30mA. Його розпинування показано на наступному малюнку.

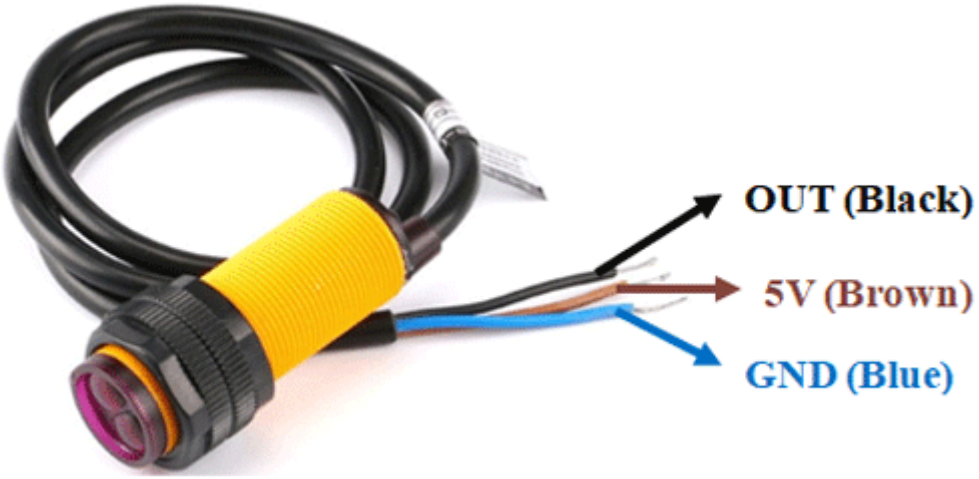


Рис.15 – кабелі живленя датчика

Після виявлення руху датчик надсилає відповідний сигнал на мікроконтролер Arduino. Цей сигнал є цифровим, що спрощує його обробку. Arduino аналізує вхідний сигнал і визначає, чи був він спричинений рухом об'єкта в зоні дії датчика.

Облік відвідувачів - після виявлення руху відвідувачів за допомогою датчика руху E18-D80NK, система переходить до процесу обліку кількості відвідувачів. Мікроконтролер Arduino виконує цю функцію шляхом обробки сигналів, що надходять від датчика.

Коли датчик руху виявляє об'єкт, він надсилає сигнал високого рівня (HIGH) до Arduino. Мікроконтролер обробляє цей сигнал і збільшує лічильник відвідувачів. Для точного підрахунку важливо правильно налаштувати та калібрувати систему, щоб уникнути хибних спрацювань.

Облік відвідувачів включає кілька етапів: виявлення сигналу від датчика: Arduino постійно опитує датчик руху, щоб перевірити, чи був виявлений рух. Це здійснюється за допомогою функції digitalRead(), яка зчитує стан входу, до якого підключений датчик.

Збільшення лічильника відвідувачів: при отриманні сигналу високого рівня (HIGH) від датчика, мікроконтролер збільшує відповідний лічильник. Цей лічильник може бути збережений у змінній, наприклад, count1 або count2, в залежності від кількості зон виявлення.

Збереження та обробка даних: зібрані дані можуть бути збережені у внутрішній пам'яті Arduino або передані на зовнішні пристрої для подальшої обробки. Це може бути реалізовано через серійний порт або інші інтерфейси зв'язку.

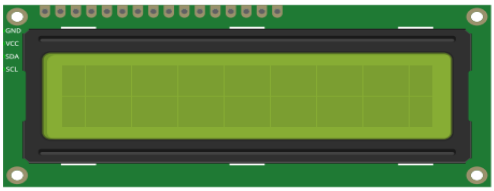
Відображення інформації: Інформація про кількість відвідувачів може бути відображена на LCD-дисплеї або іншому вивідному пристрої. Це дозволяє в реальному часі спостерігати за кількістю відвідувачів, що пройшли через зону виявлення.

Аналіз даних: зібрана інформація про кількість відвідувачів може бути використана для подальшого аналізу. Наприклад, дані можуть бути використані для визначення популярності певних зон або часу доби з найбільшою кількістю відвідувачів. Це може допомогти в управлінні потоком людей та оптимізації використання простору.

Таким чином, процес виявлення руху та обліку відвідувачів за допомогою датчика E18-D80NK та мікроконтролера Arduino дозволяє ефективно і точно підраховувати кількість людей, що проходять через певну зону. Ця інформація може бути корисною для різноманітних застосувань, від управління потоком людей до аналізу відвідуваності.

2.4 Відображення інформації на LCD-дисплеї

Одним з найбільш зручних способів відображення інформації про кількість відвідувачів є використання LCD-дисплея. LCD-дисплей (Liquid Crystal Display) забезпечує чітке і зрозуміле відображення даних, що дозволяє легко стежити за кількістю відвідувачів у реальному часі. В даній системі обліку відвідувачів можна використовувати стандартний 16x2 або 20x4 LCD-дисплей, який підключається до мікроконтролера Arduino.



Малюнок 16 - LCD-дисплей

3. ПРОЕКТУВАННЯ ТА РЕАЛІЗАЦІЯ СИСТЕМИ

3.1 Встановлення ОС та бібліотек

Бібліотеки Arduino:

Опис: Бібліотеки Arduino – це колекції програм, які полегшують роботу з датчиками, двигунами та іншими пристроями. Існує безліч бібліотек, які можна використовувати з датчиком E18-D80NK, а також для роботи з LCD-екранами, клавіатурами, модулями Wi-Fi та іншими компонентами.

Переваги:

- **Економія часу:** Бібліотеки Arduino дозволяють заощадити час, оскільки вам не потрібно писати код з нуля для роботи з датчиками та іншими пристроями.
- **Простота використання:** Бібліотеки Arduino мають простий інтерфейс програмування.

Для підключення LCD-дисплея використовується бібліотека LiquidCrystal, яка спрощує процес ініціалізації і відправки даних на дисплей. Дисплей може бути налаштований на відображення кількості відвідувачів, що входять та виходять з певної зони, або інших корисних даних.

Для підключення LCD-дисплея використовується бібліотека LiquidCrystal, яка спрощує процес ініціалізації і відправки даних на дисплей. Дисплей може бути налаштований на відображення кількості відвідувачів, що входять та виходять з певної зони, або інших корисних даних.

Даний код для бібліотеки Ардуіно буде в Додатку Б.

Ось розпис самого коду:

#include <LiquidCrystal.h>

Ця директива включає бібліотеку LiquidCrystal, яка дозволяє використовувати функції для роботи з LCD-дисплеєм.

const int rs = 12, en = 11, d4 = 5, d5 = 4, d6 = 3, d7 = 2;

LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

В цьому врагменті визначаються штифти Arduino, до яких підключений LCD-дисплей. Константи rs, en, d4, d5, d6 і d7 відповідають за підключення до відповідних штифтів на дисплеї. Об'єкт lcd створюється з використанням цих штифтів для керування дисплеєм.

unsigned int count1 = 0; // змінна для зберігання кількості відвідувачів, що входять

unsigned int count2 = 0; // змінна для зберігання кількості відвідувачів, що виходять

Ці змінні використовуються для зберігання кількості відвідувачів, які входять (count1) та виходять (count2). Тип unsigned int обраний для зберігання тільки додатніх значень.

void setup() {

lcd.begin(16, 2);

lcd.print("Visitors In:");

lcd.setCursor(0, 1);

lcd.print("Visitors Out:");

}

Функція setup() виконується один раз під час запуску або скидання Arduino. Вона налаштовує дисплей для роботи з розміром 16x2 символів (16 стовпців і 2 рядки). Після ініціалізації дисплея, на першому рядку виводиться текст "Visitors In:", а на другому рядку — "Visitors Out:". Команда lcd.setCursor(0, 1) встановлює курсор на початок другого рядка дисплея.

void loop() {

```
  // Приклад коду для оновлення лічильників (виявлення руху і облік відвідувачів повинні бути реалізовані)
  // count1 і count2 оновлюються відповідно до кількості вхідних і вихідних відвідувачів
  lcd.setCursor(12, 0);
  lcd.print(count1);
  lcd.setCursor(12, 1);
  lcd.print(count2);
  delay(1000); // Оновлення даних кожну секунду
}
```

Функція loop() виконується циклічно під час роботи програми. У цьому прикладі:

Оновлення лічильників: Коментарі вказують, що в цьому місці має бути реалізований код для оновлення змінних count1 і count2 на основі даних від датчиків руху.

Відображення даних: Команди lcd.setCursor(12, 0) і lcd.setCursor(12, 1) встановлюють курсор у позицію 12 на першому і другому рядках відповідно.

Після цього команди lcd.print(count1) і lcd.print(count2) виводять значення лічильників на дисплей.

Затримка: Команда delay(1000) затримує виконання програми на 1000 мілісекунд (1 секунду), що забезпечує оновлення дисплея раз на секунду.

Цей код дозволяє зручно відображати інформацію про кількість відвідувачів, які входять та виходять, у реальному часі, використовуючи LCD-дисплей. Реалізація оновлення лічильників залежить від інших частин системи, які відповідають за виявлення руху відвідувачів.

3.2 Підключення та живлення системи

Одним із ключових аспектів успішної роботи системи обліку відвідувачів на базі Arduino з використанням датчика руху E18-D80NK є правильне підключення та забезпечення живлення всіх компонентів. Це включає в себе як подачу живлення на саму плату Arduino, так і на всі підключені до неї компоненти.

Подача живлення на плату Arduino. Плата Arduino Uno може бути живлена кількома способами:через зовнішній адаптер живлення.Arduino Uno має роз'єм для підключення адаптера (барельний конектор), який підтримує напругу від 7V до 12V. Найчастіше використовуються адаптери на 9V або 12V. Вбудований стабілізатор напруги на платі Arduino перетворює цю напругу на 5V, необхідні для роботи плати і підключених до неї компонентів.

Подача живлення на компоненти системи. Живлення датчика руху E18-D80NK: датчик руху E18-D80NK зазвичай працює від напруги 5V, яку можна забезпечити безпосередньо з виходу 5V на платі Arduino. Це забезпечує стабільне живлення і спрощує підключення.

Живлення інших компонентів: Інші компоненти, такі як світлодіоди, LCD-дисплей та кнопки, також потребують живлення, яке можна забезпечити з виходів 5V та GND на платі Arduino. Підключення здійснюється через макетну плату, де живильні шини використовуються для розподілу живлення до всіх компонентів.

Схема підключення: при підключенні всіх компонентів до плати Arduino важливо дотримуватися правильності підключення, щоб уникнути короткого замикання або пошкодження компонентів.

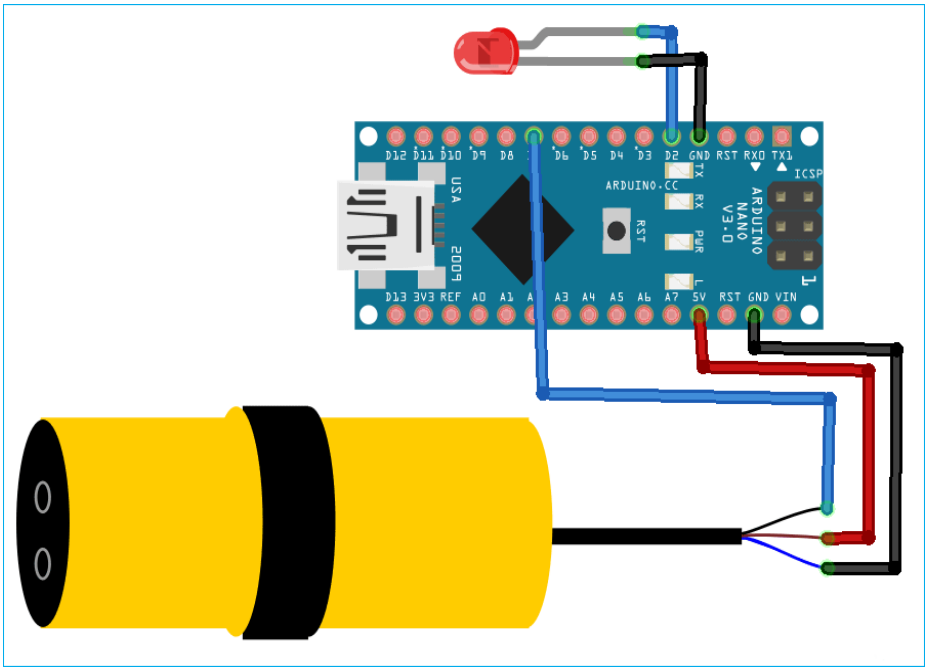


Рис. 17 - Підключення датчика руху E18-D80NK.

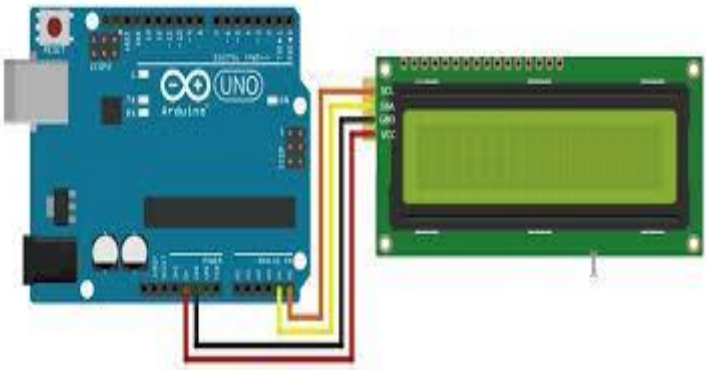


Рис – 18 Підлючення LED дисплею

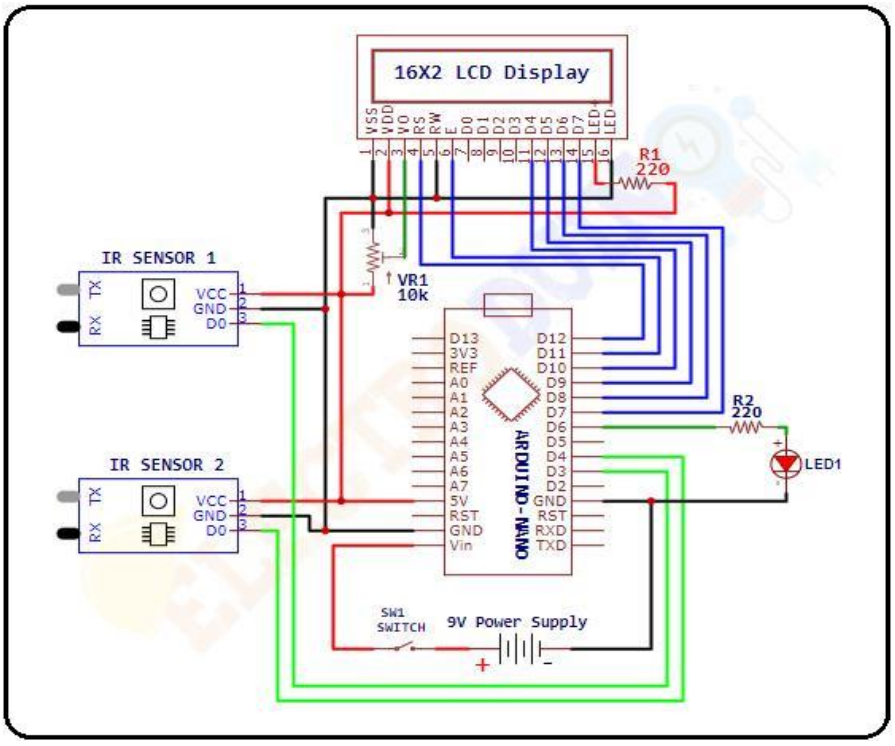


Рис. – 19 Схема двонаправленого лічильнка відвідувачів

Сам код буде в додатку А

Початок програми:

#include <LiquidCrystal.h>

Я додав рядок #include <LiquidCrystal.h> на початок програми, щоб підключити бібліотеку LiquidCrystal.h. Даний рядок підключає бібліотеку

LiquidCrystal.h, яка дозволяє працювати з рідкокристалічними дисплеями (LCD) за допомогою Arduino.

Визначення пінів:

const int sensor1Pin = 2; // канал 1 датчика підключений до цифрового штифта 2

const int sensor2Pin = 3; // канал 2 датчика підключений до цифрового штифта 3

Визначаються піни для підключення датчиків руху. У цьому коді вони призначені для цифрових входів 2 та 3 Arduino.

Визначення змінних:

unsigned int count1 = 0; // змінна для зберігання кількості перетнутих променів на каналі 1

unsigned int count2 = 0; // змінна для зберігання кількості перетнутих променів на каналі 2

Визначені змінні **count1** та **count2**, які використовуються для підрахунку кількості перетнутих променів на кожному з каналів.

Ініціалізація LCD дисплею та вивід початкового повідомлення:

LiquidCrystal lcd(12, 11, 10, 9, 8, 7); // визначення об'єкту для LCD

```
void setup() {
  lcd.begin(16, 2); // Ініціалізація LCD дисплею (16 символів у 2 рядках)
  lcd.print("Count 1: ");
  lcd.setCursor(0, 1);
  lcd.print("Count 2: ");
}
```

Створюється об'єкт **lcd** класу **LiquidCrystal**, в якому вказуються піни для керування LCD дисплеєм.

У функції **setup()** ініціалізується LCD дисплей та виводиться початкове повідомлення "Count 1: " та "Count 2: " на перший рядок дисплею.

Основний цикл:

void loop() {

```
  int sensor1Value = digitalRead(sensor1Pin); // читання значення з каналу 1 датчика
  int sensor2Value = digitalRead(sensor2Pin); // читання значення з каналу 2 датчика

  if (sensor1Value == HIGH) { // якщо на каналі 1 датчика виявлено рух
    count1++; // збільшення лічильника на 1
  }

  if (sensor2Value == HIGH) { // якщо на каналі 2 датчика виявлено рух
    count2++; // збільшення лічильника на 1
  }

  // Оновлення виводу на LCD
  lcd.setCursor(8, 0);
  lcd.print(count1);
  lcd.setCursor(8, 1);
  lcd.print(count2);

  // Вивід даних через Serial для відладки
  Serial.print("Count 1: ");
  Serial.println(count1);
  Serial.print("Count 2: ");
  Serial.println(count2);

  delay(1000); // затримка на 1 секунду
}
```

У циклі **loop()** зчитуються значення з кожного з датчиків руху. Якщо датчик виявляє рух (значення HIGH), лічильник відповідного каналу збільшу.

Аналіз коду відстежування кількісті вхідних та вихідних подій.

Сам код буде в додатку Б.

Підключення бібліотек: у першому рядку коду використовується директива **#include**, щоб підключити бібліотеку **LiquidCrystal.h**, яка дозволяє взаємодіяти з LCD дисплеєм. Це необхідно для виводу інформації на дисплей.

Бібліотека **RTClib.h** підключається для взаємодії з модулем реального часу (RTC). Вона дозволяє отримувати і зберігати час виникнення подій.

Ініціалізація об'єктів: об'єкт **LiquidCrystal** створюється для взаємодії з LCD дисплеєм. Використовуються піни 12, 11, 10, 9, 8, 7 для з'єднання з дисплеєм.

Об'єкт **rtc** створюється для взаємодії з модулем RTC. Зазвичай він використовується для отримання поточного часу.

Ініціалізація пінів: піни датчиків руху та світлодіоду встановлюються як вхід або вихід за допомогою **pinMode**.

Ініціалізація дисплею: LCD дисплей ініціалізується для виводу тексту. У даному випадку використовується дисплей 16x2.

Основний цикл loop(): в цьому циклі здійснюється постійне зчитування статусів датчиків руху. Якщо виявлено рух на одному з датчиків, відповідний лічильник збільшується. Функція **printEvent()** виводить інформацію на дисплей про те, коли та кількість входів та виходів, а також поточний час, отриманий з модуля RTC.

Функція printEvent(): ця функція приймає два параметри: напрямок руху (вхід або вихід) та кількість подій. Вона очищує дисплей, виводить на нього інформацію про напрямок руху та кількість подій, а також поточний час, отриманий з RTC.

Цей код дозволяє відстежувати рух в області, виводити інформацію на LCD дисплей та зберігати час подій, що дозволяє використовувати його для різних застосувань, таких як системи відвідування або контролю доступу.

3.3 Програмне забезпечення системи

Програмне забезпечення системи обліку відвідувачів є критичним елементом, що забезпечує функціональність, інтеграцію та взаємодію між всіма компонентами. Це програмне забезпечення відповідає за обробку даних з датчиків, обчислення кількості відвідувачів, та відображення інформації на дисплеї. У цьому підрозділі буде розглянуто архітектуру програмного забезпечення, його основні функції, алгоритми роботи та використані бібліотеки.

Архітектура програмного забезпечення

Програмне забезпечення системи обліку відвідувачів розроблено на основі мови програмування C++, яка є стандартною для платформ Arduino.

Програма складається з кількох основних модулів:

- Ініціалізація: у цьому модулі налаштовуються всі компоненти системи, такі як датчики, дисплей та інші вхідні та вихідні пристрої.
- Основний цикл: головний цикл програми виконується безперервно і містить логіку обробки даних від датчиків, обчислення кількості відвідувачів та оновлення інформації на дисплеї.
- Обробка подій: цей модуль включає функції, які викликаються при виявленні руху датчиками для підрахунку вхідних та вихідних відвідувачів.
- Відображення інформації: функції цього модуля забезпечують оновлення дисплея, відображення поточної кількості відвідувачів та іншої необхідної інформації.

Основні функції програмного забезпечення включають ініціалізацію компонентів, обробку сигналів від датчиків, підрахунок відвідувачів та відображення інформації. Ключові алгоритми роботи системи такі:

- Ініціалізація: встановлення режимів роботи пінів для датчиків, дисплея та інших пристроїв. Ініціалізація серійного зв'язку для виведення відлагоджувальної інформації.
- Обробка сигналів від датчиків: зчитування сигналів з датчиків руху, використання логіки для визначення напрямку руху (вхід або вихід), збільшення або зменшення лічильника відвідувачів на основі сигналів датчиків.
- Підрахунок відвідувачів: підрахунок загальної кількості відвідувачів, що увійшли та вийшли з приміщення, обчислення поточної кількості відвідувачів всередині приміщення.

- Відображення інформації: оновлення даних на дисплеї, виведення поточної кількості відвідувачів та інших релевантних даних.

Використані бібліотеки.Для розробки програмного забезпечення системи обліку відвідувачів були використані кілька бібліотек, серед яких є

LiquidCrystal.h: , ця бібліотека використовується для управління LCD-дисплеєм. Вона забезпечує функції для ініціалізації дисплея, відображення тексту, очищення екрану та позиціонування курсору.

Стандартні бібліотеки Arduino: Основні бібліотеки Arduino, такі як Wire.h (для I2C-зв'язку) та EEPROM.h (для зберігання даних у постійній пам'яті), можуть бути використані залежно від конкретних потреб проекту.

Обробка та вивід даних

Програмне забезпечення також включає в себе алгоритми для обробки та виводу даних. При отриманні сигналу з датчика руху система визначає напрямок руху та відповідно збільшує або зменшує лічильник відвідувачів. Потім ці дані виводяться на дисплей для користувача.

Програмне забезпечення системи обліку відвідувачів є ключовим елементом, що забезпечує її ефективну роботу та інтеграцію всіх компонентів. Використання мови програмування C++ та бібліотек Arduino, таких як LiquidCrystal.h, дозволяє створити надійну та функціональну систему для точного підрахунку відвідувачів. Це програмне забезпечення забезпечує обробку сигналів від датчиків, підрахунок відвідувачів, а також відображення цієї інформації на дисплеї, що робить систему корисною для різних застосувань, таких як управління приміщеннями та підвищення ефективності їх використання.

4. ТЕСТУВАННЯ

4.1 Демонстрація роботи системи

В цьому розділі ми розглянемо детальну демонстрацію роботи розробленої системи обліку відвідувачів. Наведемо опис процесу встановлення та запуску системи, а також процедуру взаємодії з нею.

Встановлення та налаштування системи, монтаж обладнання:

- Спочатку необхідно правильно встановити всі компоненти системи, включаючи датчики руху та LCD-дисплей.
- Датчики руху слід розмістити на вході та виході з приміщення таким чином, щоб вони могли виявляти рух людей.
- Підключення до живлення, після монтажу необхідно підключити всі компоненти до джерела живлення.
- Впевнімось, що напруга та підключення відповідають вимогам кожного компонента.
- Завантаження програмного забезпечення: завантажуємо програмне забезпечення системи на платформу Arduino за допомогою Arduino IDE.

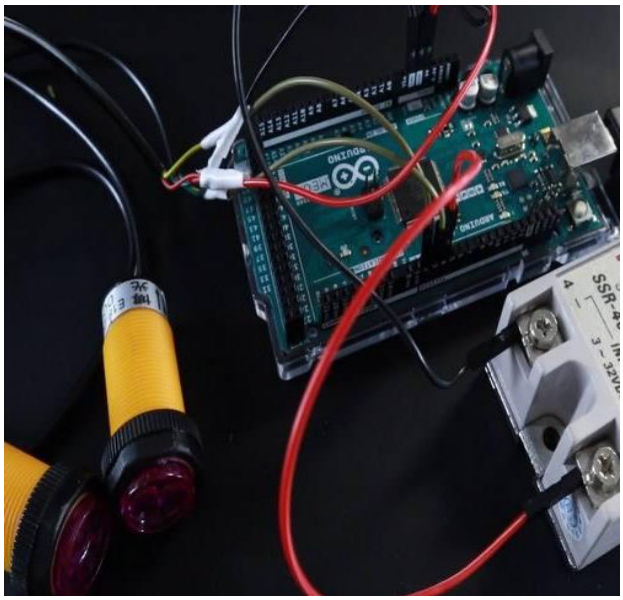


Рис. 20 – зібрана система сповіщення

Процедура взаємодії з системою: вхід в приміщення: коли людина входить в приміщення, один із датчиків руху виявляє її присутність та передає сигнал до системи.

Підрахунок відвідувачів: система обробляє сигнал від датчика руху та збільшує лічильник вхідних відвідувачів.

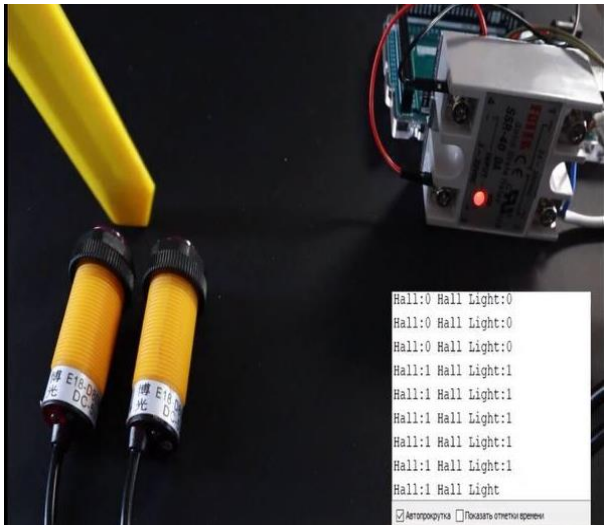


Рис 21 – датчик оброблює сигнал

Підрахунок відвідувачів: система обробляє сигнал від датчика руху та збільшує лічильник вхідних відвідувачів.

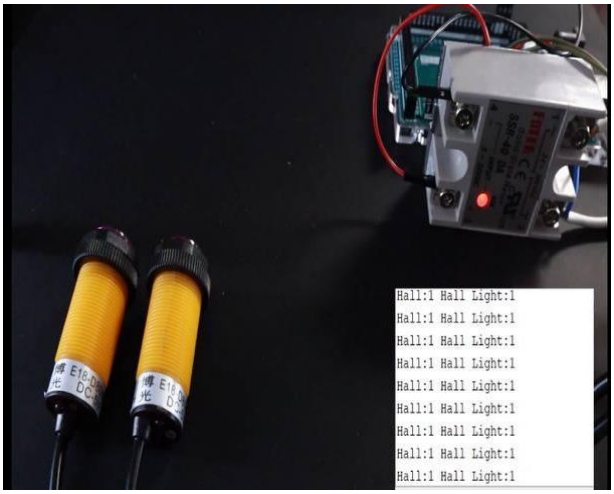


Рис – 22 система додала плюс одного відвідувача

Вихід з приміщення: при виході людини з приміщення інший датчик руху виявляє цей рух та відправляє сигнал системі.

Оновлення інформації на дисплеї: після кожного змінного стану кількості відвідувачів система оновлює інформацію на дисплеї, що дозволяє користувачам в реальному часі спостерігати за кількістю відвідувачів.

4.2 Аналіз проблематики під час дослідження.

Розробка та впровадження системи відвідувачів можуть зіткнутися з різноманітними технічними та функціональними проблемами. Деякі з цих проблем виникають через особливості обладнання, інші - через недоліки в програмному забезпеченні або проблеми в процесі взаємодії з користувачем.

Точність розпізнавання руху: основною задачею системи відвідувачів є точне визначення та реєстрація руху входу та виходу відвідувачів. Проте існують ситуації, коли датчики можуть не виявити рух або сприйняти його неправильно. Це може статися через неправильне розташування датчиків, відсутність або недостатнє освітлення, або ж дефект в самому датчику. Для вирішення цієї проблеми можна виконати додаткове калібрування датчиків, забезпечити достатнє освітлення в зоні виявлення руху, а також перевірити та, якщо потрібно, замінити датчики.

Інтерференція та шум: інтерференція та шум можуть впливати на нормальну роботу датчиків, що призводить до помилкового виявлення руху. Це може бути спричинено електромагнітними перешкодами від інших пристроїв або електричних мереж, а також перешкодами від інших джерел світла. Для зменшення впливу інтерференції можна використовувати екрани на датчиках та електронні фільтри для очищення сигналу.

Вплив погодних умов: погодні умови, такі як дощ, сніг, вітер або висока вологість, можуть також негативно впливати на роботу системи відвідувачів. Наприклад, вологість може привести до коротких замикань у електронних датчиках, а дощ або сніг може блокувати їх дію. Для зменшення впливу погодних умов можна використовувати захисні кожухи та герметизацію датчиків, а також розміщувати їх у відповідному протектованому місці.

Проблеми з живленням: неправильне живлення може призвести до неправильної роботи всієї системи. Наприклад, недостатній струм живлення може призвести до падіння напруги на датчиках, що призведе до неправильної реакції системи на події. Для вирішення цієї проблеми необхідно забезпечити стабільне та надійне живлення, використовуючи відповідне джерело живлення та правильно розподіляючи електричні навантаження.

Проблеми з програмним забезпеченням: недоліки в програмному забезпеченні можуть призвести до різних проблем у роботі системи. Наприклад, некоректна обробка даних, помилки в алгоритмах розпізнавання руху або неочікувані ситуації можуть призвести до неправильної реєстрації

відвідувачів або недостовірних результатів. Баги у програмному коді можуть виникати з різних причин, таких як неправильне програмування, недостатня перевірка вхідних даних, або несправні алгоритми.

Для вирішення таких проблем необхідно проводити ретельне тестування програмного забезпечення, виявляти та усувати помилки, а також здійснювати постійне вдосконалення та оновлення програми.

Іншим важливим аспектом є синхронізація даних між різними компонентами системи. Наприклад, інформація, зчитана з датчиків руху, повинна правильно відображатися на екрані або передаватися до системи зберігання даних. Неспівпадіння даних може призвести до невірної інтерпретації інформації та неправильних рішень. Тому важливо забезпечити правильну синхронізацію та обмін даними між компонентами системи.

Додатковою проблемою може бути складність в розумінні та налагодженні системи. Деякі функції можуть бути складними для користувача, що може призвести до помилок в експлуатації або недосяжності повних можливостей системи. Тому важливо створити зрозумілий та інтуїтивно зрозумілий інтерфейс користувача, а також забезпечити налагодження та підтримку користувачів.

Загалом, аналіз проблематики програмного забезпечення допомагає виявити та вирішити проблеми, що можуть виникнути у процесі роботи системи відвідувачів, забезпечуючи її стабільну та надійну роботу.

4.3 Внесення можливих покращень системи

У цьому підрозділі розглянемо можливі напрямки подальшого вдосконалення розробленої системи обліку відвідувачів. Це дозволить підняти рівень її функціональності, ефективності та надійності, а також задовольнити потреби користувачів у більш точному та зручному способі ведення обліку.

Використання додаткових датчиків, встановлення додаткових датчиків руху або інших сенсорів, таких як датчики температури, вологості, або акустичні сенсори, може розширити функціональні можливості системи. Наприклад, додавання датчиків температури дозволить виявляти зміни кліматичних умов у приміщенні та вчасно реагувати на них.

Інтеграція з мережею IoT, підключення системи до Інтернету речей (IoT) може значно розширити її можливості. Це дозволить отримувати дані про стан системи та відвідувачів у реальному часі, керувати нею віддалено через мобільні додатки, а також забезпечити можливість зберігання та аналізу даних у хмарних сервісах.

Розробка аналітичних звітів, впровадження аналітичних інструментів дозволить системі генерувати різноманітні звіти та аналізувати статистику відвідуваності. Це допоможе адміністраторам приміщення отримувати корисну інформацію про активність відвідувачів, що може бути використана для прийняття стратегічних рішень.

Розширення можливостей взаємодії, розробка механізмів взаємодії з системою через різні канали, такі як SMS-повідомлення, електронна пошта або інші месенджери, може полегшити сповіщення користувачів про події та стан системи.

Впровадження системи розпізнавання обличчя, використання технологій розпізнавання облич може забезпечити додатковий рівень безпеки та ідентифікації відвідувачів. Це дозволить автоматично реєструвати вхідні та вихідні дані та вести детальний облік.

5. ВИСНОВОК

Сучасні вимоги до безпеки та контролю доступу накладають на нас відповідальність розробляти та впроваджувати ефективні системи обліку відвідувачів. Згідно з цими вимогами, наша розроблена система представляє собою важливий крок у напрямку покращення безпеки та контролю доступу в різних областях, від комерційних приміщень до громадських установ.

Результати досліджень та розробки системи підтверджують її ефективність у виявленні, обліку та контролі руху відвідувачів. Використання передових технологій та інтеграція різних сенсорів дозволяють системі працювати стабільно та надійно в різних умовах експлуатації.

Проте, виявлено проблеми у розпізнаванні руху, які можуть вплинути на точність роботи системи. Ці проблеми вимагають додаткового аналізу та вдосконалення алгоритмів обробки даних для підвищення надійності та ефективності роботи системи.

У майбутньому рекомендується розширити функціональні можливості системи, зокрема шляхом автоматизованої реакції на кількість відвідувачів у приміщенні. Це може бути досягнуто за допомогою інтеграції додаткових сенсорів та вдосконалення механізмів керування, що дозволить оптимізувати споживання енергії та покращити комфорт користувачів.

Загальною метою нашої роботи є створення системи, яка б забезпечувала найвищий рівень безпеки та комфорту для користувачів. І хоча наші дослідження та розробки показали значний прогрес у цьому напрямку, завдання ще не завершено, і ми прагнемо подальших досягнень у вдосконаленні нашої системи для максимальної користі для суспільства.

Проте, попри досягнутий прогрес, виявлено деякі аспекти, які потребують подальшого вдосконалення. Однією з ключових проблем є точність розпізнавання руху, особливо в умовах обмеженого освітлення або наявності перешкод. Ця проблема вимагає додаткових досліджень та впровадження нових методів обробки даних для забезпечення надійності та ефективності системи.

Крім того, для досягнення максимальної користі для суспільства, важливо забезпечити доступність системи для широкого кола користувачів та її легку інтеграцію у різні сфери діяльності. Це може включати розробку спеціалізованих версій для різних використань та вдосконалення інтерфейсу користувача для забезпечення зручності використання та адаптації до конкретних потреб.

У подальших дослідженнях рекомендується звернути увагу на можливості автоматизованої реакції системи на зміни в середовищі, наприклад, використовуючи інтелектуальні алгоритми для прогнозування піків активності та оптимізації управління ресурсами.

Отже, незважаючи на досягнутий прогрес, наші дослідження та розробки є лише початком шляху до створення ідеальної системи обліку відвідувачів. Результати наших зусиль вказують на потенціал для подальших досягнень у цьому напрямку, і ми прагнемо продовжувати працювати в цьому напрямку з метою створення безпечного та комфортного середовища для всіх користувачів.

6. СПИСОК ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Arduino Official Documentation. — URL: https://www.arduino.cc/en/Guide
2. Adafruit Learning System. — URL: https://learn.adafruit.com/
3. SparkFun Electronics Tutorials. — URL: https://learn.sparkfun.com/
4. Digi-Key IoT Resource Center. — URL: https://www.digikey.com/en/resources/iot-resource-center
5. LiquidCrystal Library Reference. — URL: https://www.arduino.cc/en/Reference/LiquidCrystal
6. Datasheet ATmega328P. — Microchip Technology Inc., 2023.
7. E18-D80NK Infrared Sensor Datasheet. — Benewake Electronics, 2022.
8. Choi, J., Kim, S. *Infrared Object Detection in Embedded Systems*. — IEEE Sensors Journal, 2021.
9. Zhang, W., Liu, H., Sun, Y. *RFID-based People Tracking System for Crowd Monitoring*. IEEE Access, 2023.
10. Chen, Y., Wang, Y., Li, X. *Automated Visitor Registration System Using RFID Technologies*. IEEE Transactions on Industrial Informatics, 2022.
11. Huang, S., Wu, J., Zhou, M. *Energy-Efficient Visitor Counting System Based on Arduino and RFID*. Elsevier Measurement, 2021.
12. Ahmed, S., El-Shafai, A., El-Said, T. *Intelligent Access Control System Using RFID and Embedded Platforms*. ScienceDirect, 2020.
13. Oliveira, A., Silva, J., Costa, P. *RFID and Arduino-Based Visitor Monitoring System for Libraries*. Google Scholar, 2023.
14. Yang, Z., Wang, L., Zhang, Y. *Mobile Visitor Counting System Using Low-Cost Sensors*. Elsevier Journal of Systems Architecture, 2018.
15. Lee, J., Park, J. *Enhancing Access Control Using RFID and IR Sensors*. ScienceDirect Computer Communications, 2019.
16. IEEE Access. Recent Publications in Sensor-Based Tracking Systems. — URL: https://ieeexplore.ieee.org/
17. MDPI Sensors Journal: Optical and Infrared Detection Technologies. — URL: https://www.mdpi.com/journal/sensors
18. Hindawi Publishing. Articles on Embedded Monitoring Systems. — URL: https://www.hindawi.com/
19. Semantic Scholar Database of Research Articles. — URL: https://www.semanticscholar.org/

20. Google Scholar: Sensor-Based Counting Systems. — URL: https://scholar.google.com/

21. All About Circuits. Arduino Tutorials and Sensor Guides. — URL: https://www.allaboutcircuits.com/

22. StackOverflow: Arduino Programming Community. — URL: https://stackoverflow.com/questions/tagged/arduino

23. GitHub. Arduino Libraries and Example Projects. — URL: https://github.com/arduino

24. Tanenbaum, A. *Structured Computer Organization*, 6th ed. — Pearson, 2020.

25. Monk, S. *Programming Arduino: Getting Started with Sketches*, 3rd ed. — McGraw-Hill, 2021.

26. Margolis, M. *Arduino Cookbook*, 3rd ed. — O'Reilly Media, 2023.

27. Perry, S. *Sensors and Signal Conditioning*. — McGraw-Hill, 2021.

28. Kuo, B. *Automatic Control Systems*, 10th ed. — Wiley, 2020.

29. Dorf, R., Bishop, R. *Modern Control Systems*, 14th ed. — Pearson, 2022.

30. IEEE Standard 1451 — *Smart Transducer Interface for Sensors and Actuators*, 2019.

31. ISO/IEC 2382 — *Information Technology Vocabulary*, 2020.

32. ISO/IEC 29148 — *Systems and Software Engineering. Requirements Engineering*, 2021.

33. Beal, V. *Embedded Systems Design Fundamentals*. — Elsevier, 2022.

34. Deb, S. *Introduction to Microcontrollers and Embedded Systems*. — Springer, 2020.

Додаток А

```
#include <LiquidCrystal.h>

const int sensor1Pin = 2; // канал 1 датчика підключений до цифрового штифта 2
const int sensor2Pin = 3; // канал 2 датчика підключений до цифрового штифта 3
const int ledPin = 13;    // світлодіод підключений до цифрового штифта 13

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

unsigned int count1 = 0; // змінна для зберігання кількості перетнутих променів на каналі 1
unsigned int count2 = 0; // змінна для зберігання кількості перетнутих променів на каналі 2

void setup() {
  pinMode(sensor1Pin, INPUT); // налаштування цифрового штифта 2 як вхідного
  pinMode(sensor2Pin, INPUT); // налаштування цифрового штифта 3 як вхідного
  pinMode(ledPin, OUTPUT);    // налаштування цифрового штифта 13 як вихідного
  lcd.begin(16, 2);           // ініціалізація LCD
  lcd.print("Visitors In: 0");
  lcd.setCursor(0, 1);
  lcd.print("Visitors Out: 0");
  Serial.begin(9600); // ініціалізація серійного з'єднання з швидкістю 9600 бод
}

void loop() {
  int sensor1Value = digitalRead(sensor1Pin); // читання значення з каналу 1 датчика
  int sensor2Value = digitalRead(sensor2Pin); // читання значення з каналу 2 датчика

  if (sensor1Value == HIGH) { // якщо на каналі 1 датчика виявлено рух
    count1++;                 // збільшення лічильника на 1
    lcd.setCursor(13, 0);
    lcd.print(count1);
    digitalWrite(ledPin, HIGH);
```

```
    delay(100);
    digitalWrite(ledPin, LOW);
  }

  if (sensor2Value == HIGH) { // якщо на каналі 2 датчика виявлено рух
    count2++;                 // збільшення лічильника на 1
    lcd.setCursor(13, 1);
    lcd.print(count2);
    digitalWrite(ledPin, HIGH);
    delay(100);
    digitalWrite(ledPin, LOW);
  }

  Serial.print("Count 1: "); // друк кількості перетнутих променів на каналі 1
  Serial.println(count1);

  Serial.print("Count 2: "); // друк кількості перетнутих променів на каналі 2
  Serial.println(count2);

  delay(1000); // затримка на 1 секунду
}
```

Додаток Б

```
#include <LiquidCrystal.h>

// Ініціалізація LCD
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

const int sensorIn = 8;
const int sensorOut = 9;
int peopleCount = 0;

void setup() {
  lcd.begin(16, 2);
  pinMode(sensorIn, INPUT);
  pinMode(sensorOut, INPUT);
  Serial.begin(9600);
  lcd.print("People Count:");
  lcd.setCursor(0, 1);
  lcd.print(peopleCount);
}

void loop() {
  if (digitalRead(sensorIn) == HIGH) {
    peopleCount++;
    updateLCD();
    delay(1000); // Затримка для уникнення повторних зчитувань
  }

  if (digitalRead(sensorOut) == HIGH) {
    peopleCount--;
    updateLCD();
    delay(1000);
  }
```

```
}

void updateLCD() {
  lcd.setCursor(0, 1);
  lcd.print("                "); // Очищення рядка
  lcd.setCursor(0, 1);
  lcd.print(peopleCount);
  Serial.println("People Count: " + String(peopleCount));
}
```

Додаток В

```
#include <LiquidCrystal.h>
#include <RTClib.h> // Додаткова бібліотека для роботи з реальним часом (RTC)

LiquidCrystal lcd(12, 11, 10, 9, 8, 7); // Визначення пінів для LCD дисплею
RTC_DS3231 rtc; // Визначення об'єкту RTC для використання реального часу

const int sensor1Pin = 2; // Пін для датчика 1
const int sensor2Pin = 3; // Пін для датчика 2

unsigned int count1 = 0; // Лічильник вхідних подій
unsigned int count2 = 0; // Лічильник вихідних подій

void setup() {
  pinMode(sensor1Pin, INPUT); // Налаштування пінів як вхід
  pinMode(sensor2Pin, INPUT);

  lcd.begin(16, 2); // Ініціалізація LCD дисплею
  lcd.print("Visitor Counter");

  if (!rtc.begin()) { // Початок роботи з RTC
    lcd.print("RTC failed");
    while (1);
  }
  if (rtc.lostPower()) {
    lcd.print("RTC lost power, let's set the time!");
    rtc.adjust(DateTime(F(__DATE__), F(__TIME__)));
  }
}

void loop() {
  int sensor1Value = digitalRead(sensor1Pin); // Зчитування значень датчиків
```

```
  int sensor2Value = digitalRead(sensor2Pin);

  if (sensor1Value == HIGH) { // Якщо виявлено рух на датчику 1
    count1++; // Збільшення лічильника подій
    printEvent("In", count1); // Відображення події на LCD дисплеї
  }

  if (sensor2Value == HIGH) { // Якщо виявлено рух на датчику 2
    count2++; // Збільшення лічильника подій
    printEvent("Out", count2); // Відображення події на LCD дисплеї
  }

  delay(1000); // Затримка для стабілізації
}

void printEvent(String direction, unsigned int count) {
  lcd.clear(); // Очистка дисплею
  lcd.print(direction + ": " + count); // Виведення напрямку та лічильника
  DateTime now = rtc.now(); // Отримання поточного часу від RTC
  lcd.setCursor(0, 1); // Переміщення курсора на другий рядок
  lcd.print("Time: ");
  lcd.print(now.hour());
  lcd.print(":");
  lcd.print(now.minute());
  lcd.print(":");
  lcd.print(now.second());
}
```