Форма № 24

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

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

Кафедра програмних засобів
(повне найменування кафедри)

Пояснювальна записка

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

на тему РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СИСТЕМИ З КОДОВИМ ДОТУПОМ
SOFTWARE DEVELOPMENT FOR THE SYSTEM WITH A CODE ACCESS

Виконав(ла): студент(ка) 4 курсу, групи КНТ-130
Спеціальності 121 Інженерія програмного забезпечення
(код і найменування спеціальності)

Освітня програма (спеціалізація)
Інженерія програмного забезпечення

ПРИЩЕПА В.О.
(ПРІЗВИЩЕ та ініціали)
Керівник ГЛАДКОВА О.М.
(ПРІЗВИЩЕ та ініціали)
Рецензент ЗЕЛЕНЬОВА І.Я.
(ПРІЗВИЩЕ та ініціали)

2024

Форма № 25

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет КНТ
Кафедра програмних засобів
Ступінь вищої освіти бакалавр
Спеціальність 121 Інженерія програмного забезпечення
(код і найменування)
Освітня програма (спеціалізація) Інженерія програмного забезпечення
(назва освітньої програми (спеціалізації)

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
_____ Сергій СУББОТІН
“____” _____2024 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТА(КИ)

ПРИЩЕПИ Владислава Олексійовича
(ПРІЗВИЩЕ, ім’я, по батькові)

1. Тема проєкту (роботи) Розробка програмного забезпечення системи з кодовим доступом. Software Development for the System with a Code Access.

керівник проєкту (роботи) к.т.н., ГЛАДКОВА Ольга Миколаївна ,
(науковий ступінь, вчене звання, ПРІЗВИЩЕ, ім’я, по батькові)
затверджені наказом закладу вищої освіти від “ 24 ” квітня 2024 року № 168
2. Строк подання студентом проєкту (роботи) 10 червня 2024 року
3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Інструментарії розробки. 3. Програмні та аппаратні рішення системи з кодовим замком. 4. Інструкції до користування системою.

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

6. Консультанти розділів проєкту (роботи)

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

7. Дата видачі завдання “ 15 ” квітня 2024 року.

КАЛЕНДАРНИЙ ПЛАН

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

Студент(ка) _____ Владислав ПРИЩЕПА
(підпис) (І’мя ПРІЗВИЩЕ)

Керівник проєкту (роботи) _____ Ольга ГЛАДКОВА
(підпис) (І’мя ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до дипломної кваліфікаційної роботи бакалавра: 65 с., 2 табл., 19 рис., 2 дод., 16 джерел.

СИСТЕМИ З КОДОВИМ ДОСТУПОМ, ПРОТОТИП, ПІДСИСТЕМИ РОЗУМНОГО БУДИНКУ, ARDUINO, TINKERCAD.

Об’єкт дослідження – підсистеми розумного будинку.

Предмет дослідження – системи з кодовим доступом.

Мета роботи – створення програмної системи з кодовим доступом для автоматизації та підвищення безпеки доступу.

Матеріали, методи та технічні засоби: високорівневе та об’єктно-орієнтоване програмування, програмно-апаратна платфомра Arduino, мови програмування C++, симулотор Tinkercad, програмне забезпечення Fritzing, персональний комп’ютер з будь-яким браузером та доступом в Інтернет.

Результати. У рамках даної дипломної роботи було розроблено систему замка з кодовим доступом з використанням мікроконтролера Arduino Uno. Система забезпечує введення та перевірку коду доступу, керування замком та індикацію стану системи, інформує користувача про стан системи за допомогою візуальних та звукових сигналів.

Висновки. Розроблено програмну систему з кодовим доступом. Система призначена для автоматизації а також підвищення безпеки доступу.

Галузь використання – системи автоматизації дома та офіса.

ABSTRACT

Explanatory note to the diploma qualifying work of the bachelor: 65 pages, 2 tables, 19 figures, 2 appendixes, 16 sources.

SYSTEMS WITH CODE ACCESS, PROTOTYPE, SMART HOME SUBSYSTEMS, ARDUINO, TINKERCAD.

Object of study is a smart home subsystems.

Subject of study are systems with code access.

The purpose of the work is to create a software system with code access for automation and enhanced access security.

Materials, methods, and tools: high-level and object-oriented programming, Arduino software and hardware platform, C++ programming languages, Tinkercad simulator, Fritzing software, personal computer with any browser, and Internet access.

Results. As part of this diploma project, a code access lock system was developed using the Arduino Uno microcontroller. The system provides code input and verification, lock control, and system status indication, informing the user about the system's status through visual and audio signals.

Conclusions A software system with code access has been developed. The system is designed for automation and enhanced access security.

The field of use is a home and office automation systems.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

IDE – Integrated Development Environment,

IoT – Internet of Things;

I2C – Inter-Integrated Circuit;

LCD – liquid-crystal display;

LED – light-emitting diode.

ВСТУП

Сучасний розвиток науки і техніки дозволяє автоматизувати і покращити життєдіяльність багатьох сфер людської діяльності, в тому числі і в побуті. Сьогодні багато пристроїв економлять час і роблять життя людей вдома і на роботі комфортнішим, зручнішим і дешевшим.

На початку другого десятиліття 21 століття настав час зробити розумні будинки реальністю для регулярного використання. Досліджуються різні частини розумного дому, але до застосовної системи з використанням сучасних технологій ще є відстані.

Технології розвинулись за останні два десятиліття такими швидкими темпами, що, безумовно, перешагнули всі досягнення, яких людство здобуло протягом останніх століть. Технології захопили нинішню епоху з усіма її корисностями та перевагами. За допомогою нинішніх технологій виснажливу роботу, на виконання якої звичайній людині може знадобитися кілька днів, тепер можна легко виконати за кілька секунд.

Сьогодні багато пристроїв економлять час і роблять життя людей вдома і на роботі комфортнішим, зручнішим і безпечнішим. Однією з таких технологій є розумний будинок, який об'єднує різноманітні автоматизовані системи для управління різними аспектами житла.

Розумний будинок включає в себе різні підсистеми, серед яких системи з кодовим доступом займають важливе місце. Такі системи забезпечують безпеку житла, дозволяючи контролювати доступ до приміщення за допомогою введення спеціального коду. Вони можуть бути використані як в житлових, так і в офісних приміщеннях, забезпечуючи зручний та надійний спосіб управління доступом.

Вибір платформи Arduino для розробки системи з кодовим доступом був обумовлений її простотою використання, доступністю та підтримкою великої кількості додаткових модулів. Arduino дозволяє швидко створити прототип системи і протестувати його функціональність. Симулятор

Tinkercad також був використаний для моделювання і тестування системи без необхідності фізичного збирання, що значно знижує витрати на розробку.

У даній роботі розглянуто процес розробки системи замка з кодовим доступом з використанням мікроконтролера Arduino Uno. Система забезпечує введення та перевірку коду доступу, керування замком та індикацію стану системи, інформує користувача про стан системи за допомогою візуальних та звукових сигналів. Метою роботи є створення надійної і ефективної системи, яка підвищить безпеку доступу та автоматизує цей процес.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Розумний будинок як складова людського життя

Сьогодні кожен носить смартфон. Усі елементи керування та складні функції у руках. Жодне цифрове чи технологічне завдання неможливо виконати за допомогою смартфона. Ось чому смартфон є найкориснішим і найбільш використовуваним винаходом у цьому цифровому світі.

Не тільки телефони, все стає розумнішим день у день. Є смарт-телевізор, смарт-годинники, розумні колонки і т.і. І не тільки це, лише за це десятиліття ми познайомилися з перевагами розумних будинків. Уявіть собі, що можна керувати всіма гаджетами, технікою та домом повністю, і можна керувати ними лише голосом або жестами [1].

Далі наведено п’ять шляхів, за допомогою яких розумні будинки можуть покращити спосіб життя в найближчому майбутньому.

Розумний дім складається з різноманітних розумних гаджетів і приладів, таких як розумний холодильник, розумні двері, розумні вентилятори тощо. Усе це створює екосистему один з одним, надаючи контроль з боку власника, а загальний термін, наданий екосистемі, буде «Розумний дім». Давайте тепер обговоримо, як поява розумних будинків допоможе нам покращити наш спосіб життя, а отже, покращити наше життя.

Відсутність додаткової складності фізичного контролю. З розумними будинками людина звільняється від додаткового тягаря фізичного контролю над усіма гаджетами у домі. Все під командою голосу, а також можна контролювати дистанційно [1].

Більше часу для сім'ї та роботи. Не потрібно заздалегідь витрачати час на будь-яке завдання, яке необхідно виконати вдома. Просто внесіть в прогруму дома, і завдання буде зроблено. Не потрібно йти додому, вмикати чайник і чекати, поки вода закипить. Просто задайте завадання в телефоні на шляху додому та заощаджуйте додатковий час.

Автоматичне освітлення в туалетах. Санвузли - це такі місця в будинку, де позапланово виникає потреба в освітленні. І іноді можна тримати їх увімкненими протягом тривалого часу без користі. У розумному домі кімната аналізує присутність людей і відповідно керує освітленням. Таким чином буде оптимальне споживання електроенергії.

Маючи точне спостереження за споживанням енергії. У звичайних рахунках за лічильник або електроенергію, що постачається у вашому домі, ви просто отримуєте стандартний рахунок із детальною інформацією про споживання електроенергії та тариф на електроенергію, і ви не можете підрахувати втрату або надмірне споживання електроенергії. Розумними будинками керує програма на смартфоні. І таким чином представити вам детальний рахунок спожитої енергії. Ви знатимете, який прилад використовувався скільки часу та який внесок у ваш рахунок. Таким чином ви можете контролювати використання та витрати [1].

Жодних складних відносин із ключами. Інколи важко зберігати ключі від дверей у безпеці та в розпорядженні наших потреб. Але в розумних будинках двері будинку керуються голосом і паролями, і таким чином ви економите додатковий час. Крім того, вони досить безпечні. Про будь-які спроби незаконно потрапити до оселі буде повідомлено власнику і місцевій поліції.

Це кілька способів, за допомогою яких розумні будинки можуть покращити життя. У найближчі роки ці будинки будуть скрізь і допоможуть організувати життя кращим чином.

1.2 Аналіз підсистем розумного будинку

Перетворення вашого житлового простору на розумний дім — це більше, ніж просто оновлення. Йдеться про підвищення зручності та ефективності. Розуміючи ключові функції розумного дому та забезпечуючи

сумісність пристроїв, можна створити дім, який буде не лише технологічно просунутим, але також повністю інтегрованим і зручним для користувача.

Перш ніж поринути у світ розумних домашніх пристроїв, важливо точно встановити цілі та завдання які необхідно достігти за допомогою розумного будинку, наприклад [2]:

– спростити щоденні завдання;
– скоротити рахунки за електроенергію;
– підвищити безпеку свого дому.

Однією з основних причин, чому багато хто вирішує будувати розумні будинки, є зручніть і комфорт, які вони пропонують. Щоб регулювати тепло в кімнаті, не встаючи з дивана або готова кави, коли тільки прокинетеся. Такі пристрої, як розумні світильники, розумні термостати та такі інструменти, як голосові помічники, можуть допомогти виконувати повсякденні завдання, роблячи життя комфортнішим.

Розумний будинок також може бути екологічно чистим будинком. За допомогою правильних пристроїв можна використовувати значно менше енергії. Наприклад, розумні термостати запам’ятовують звички людини та встановлюють нагрівання чи охолодження в потрібний момент, тому електроенергія не витрачається даремно. Подібним чином розумні розетки можуть вимкнути те, що не використовуєте, а розумне освітлення регулюється відповідно до наявності природного освітлення. Це може допомогти значно заощадити на рахунках з часом [3].

Почуття безпеки у домі є ключовою частиною зменшення стресу та тривоги у житті, а розумні будинки дають все більше способів залишатися в безпеці. Розумні камери дозволяють бачити дім у будь-який час, навіть якщо вас там немає. Камери дверного дзвінка показують, хто знаходиться зовні, а розумні замки дозволяють відкривати або замикати двері з будь-якого місця. Крім того, розумні детектори диму та чадного газу можуть підключатися до Wi-Fi у домі, тобто вони можуть сповіщати про небезпеку в режимі реального часу, навіть якщо нікого немає вдома [2].

Далі необхідно обрати пристрої, які відповідають потребам. Кожен компонент «розумного дому» виконує свою функцію. Ключові компоненти розумного дому наведено на рисунку 1.1.

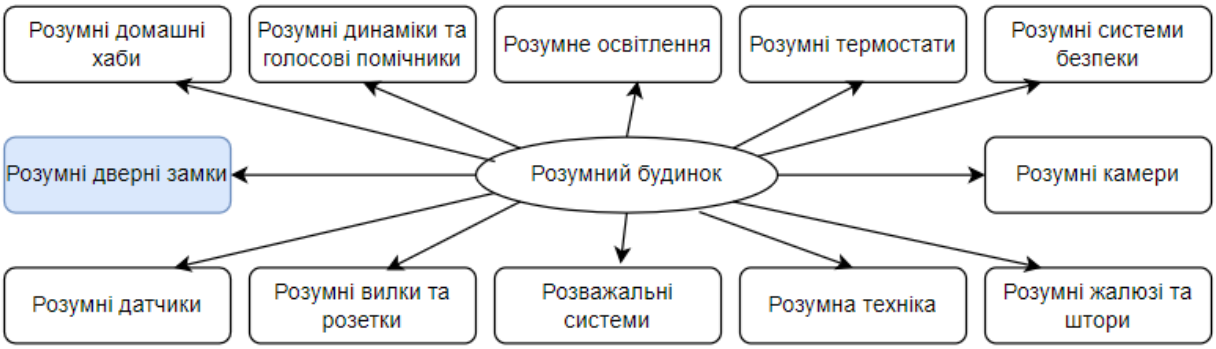


Рисунок 1.1 – Компоненти розумного дому

Розумні домашні хаби. Центральне місце в системі керування розумним будинком займає хаб, пристрій, який з'єднує та керує всіма вашими розумними компонентами. Це мозок, який стежить за тим, щоб усе спілкувалося одне з одним і працювало разом. Можна розглядати його як основну частину розумного дому. Приклади: Amazon Echo Plus, Google Home і Samsung SmartThings Hub [2].

Мережа та підключення. Основою будь-якого розумного будинку є його мережа. Якісні маршрутизатори та мережеві системи, як-от від Netgear або Google Nest Wifi, забезпечують стабільне та далекосяжне з'єднання. Це важливо для того, щоб усі пристрої добре працювали разом.

Розумні динаміки та голосові помічники. Голосове керування змінило спосіб взаємодії з домівками. Такі пристрої, як Amazon Echo (з Alexa), Google Nest Speakers і Apple HomePod, не тільки відтворюють музику, але й виконують роль персональних помічників, відповідаючи на запитання, встановлюючи нагадування та керуючи іншими розумними пристроями [2].

Розумне освітлення. Існують розумні лампочки та вимикачі, які пропонують дистанційне керування, затемнення, зміну кольорів і планування. Philips Hue і LIFX є популярними виборами, оскільки вони

можуть настаюватись відповідіно до бажання користувача та навіть синхронізуватися з музикою чи фільмами.

Розумні термостати. Такі пристрої, як Nest Thermostat або Ecobee, регулюють нагрівання та охолодження відповідно до потреб, забезпечуючи комфорт і заощаджуючи енергію. Вони знають, чи є хтось вдома чи ні, і пристосовуються, щоб забезпечити ефективність.

Розумні системи безпеки. Безпека надзвичайно важлива. У розумних системах безпеки є датчики дверей/вікон, детектори руху та сигналізація. Якщо виникне проблема, вони можуть надіслати сповіщення на телефон. Популярні варіанти включають SimpliSafe і ADT Pulse.

Розумні камери. Незалежно від того, чи це камера дверного дзвінка, як Ring, чи внутрішня камера, як Arlo, ці пристрої дозволяють контролювати дім у режимі реального часу. Вони можуть виявляти рух, записувати кадри та навіть дозволяти говорити як по телефону [2].

Розумні дверні замки. Покращте безпеку за допомогою розумних замків, таких як August або Schlage. Блокуйте або розблокуйте дистанційно, надавати тимчасовий доступ гостям і навіть інтегрувати з іншими пристроями для автоматизованих процедур.

Розумні датчики. Ці розумні пристрої помічають зміни у домі. Вони можуть виявляти рух, вимірювати тепло, перевіряти світло і навіть виявляти витік води. Датчики таких брендів, як Fibaro або Aeotec, можуть надсилати сповіщення або запускати інші пристрої реагувати на зміни.

Розумні вилки та розетки. За допомогою розумних розеток від таких брендів, як TP-Link або Belkin, можна дистанційно керувати, планувати або контролювати споживання енергії будь-яким підключеним пристроєм.

Розважальні системи. Покращуйте розваги за допомогою смарт-телевізорів, звукових систем і потокових пристроїв. Такі системи, як Sonos або Roku, дозволяють відтворювати музику чи відео за допомогою голосового керування, інтегруючись з іншими пристроями розумного дому.

Розумна техніка. Кухня та пральня стають розумнішими. Холодильники, які відстежують продукти, духовки, які можуть почати розігрівати по дорозі додому, або пральні машини, якими можна керувати віддалено, тепер є нормальними. LG і Samsung лідирують у цій революції розумного дому.

Розумні жалюзі та штори. Можна автоматизувати природне освітлення за допомогою розумних жалюзі або штор. Опції таких брендів, як Lutron або IKEA, дозволяють установлювати час, керувати ним дистанційно та регулювати його відповідно до температури [2].

1.3 Архітектура інтернет речей

Архітектура IoT — це дизайн і структура системи IoT (Internet of Things), клей, який забезпечує ефективну та безпечну роботу пристроїв і систем [3]. Вона включає:

– фізичні пристрої або «речі»;

– мережа, яка з’єднує речі та забезпечує їх зв’язок;

– програмне забезпечення, яке керує пристроями, аналізує та зберігає дані та взаємодіє з користувачами.

Архітектура IoT має кілька рівнів, зокрема:

Рівень пристроїв. Його також зазвичай називають «рівнем датчиків». Це місце, де пристрої Інтернету речей, як-от інтелектуальні пристрої та носимі пристрої, підключаються до мережі чи Інтернету.

Рівень підключення: Технічно цей рівень називається «мережевим рівнем». Він має всі технології, необхідні для підключення пристроїв IoT до Інтернету, інших пристроїв або хмари.

Рівень обробки даних і аналітики: цей рівень збирає та обробляє цифрові дані, створені пристроями IoT. Він включає інфраструктуру хмарних обчислень, платформи IoT та інструменти аналізу даних [3].

Рівень безпеки: безпека Інтернету речей забезпечує функціонування всієї системи без зовнішнього втручання, наприклад злому чи пошкодження даних. Цей рівень забезпечує безпеку всього рішення IoT, а також його конфіденційність і цілісність. Він включає в себе методи автентифікації, шифрування та контролю доступу, які запобігають несанкціонованому доступу та забезпечують захист даних [3].

Рівень програми: цей рівень забезпечує інтерфейс для взаємодії з користувачем і керування пристроєм. Він також включає інтерфейси користувача, такі як мобільні або веб-додатки, які дозволяють відстежувати та контролювати пристрої IoT.

Архітектура IoT може бути як централізованою, так і розподіленою, залежно від того, як вона буде використовуватися.

Централізована архітектура збирає дані з пристроїв IoT у центральному місці, як-от центр обробки даних IoT, тоді як розподілена архітектура дозволяє обробляти дані на периферійному рівні або на рівні пристрою.

Обидва підходи мають переваги та недоліки, і вибір залежить від конкретного випадку використання IoT.

Наприклад, коли використовується централізована мережа IoT, усі дані зберігаються в одному місці, тому ними легко керувати. Менше шансів зазнати простою, оскільки доведеться мати справу лише з одним конкретним сервером [4].

Але централізована архітектура системи IoT може бути більш безпечною та вразливішою до атак. Усе, що потрібно зробити хакеру, це отримати доступ до центрального блоку зберігання даних.

Децентралізовані мережі IoT більш безпечні та прозорі. На жаль, їх найбільшим недоліком є те, що їх досить складно масштабувати.

Згадані вище рівні є основними компонентами будь-якого надійного проекту IoT. Залежно від типу IoT-проекту, який планується реалізувати, може знадобитися більше рівнів. Це одна з головних причин, чому багато

рішень IoT мають додаткові рівні, що робить всю систему більш комплексною.

Ось більш глибокий погляд на різні рівні або компоненти архітектури IoT, які роблять процес підключення IoT більш плавним.

Апаратні компоненти IoT (фізичний рівень) — це фізичні пристрої, які взаємодіють із середовищем, збирають дані та надсилають їх до інших компонентів у системі [3].

IoT датчики. Як випливає з назви, датчики — це інтелектуальні пристрої в мережі IoT, які вимірюють все, включаючи вібрацію двигуна літака та пульс, якщо ви носите кардіомонітор. Після того, як датчики вимірюють певні фізичні компоненти, вони перетворюють цю інформацію в дані, які інші компоненти системи IoT можуть легко обробити. Наприклад, коли ми маємо справу з транспортними засобами, у нас є датчики вологості, оптичні датчики та датчики прискорення [3].

Актуатори – ці пристрої можуть керувати фізичними системами, як-от двигунами та перемикачами. Ми можемо використовувати їх, щоб умикати та вимикати розумне світло, відкривати двері чи ворота та керувати іншими фізичними системами [4].

Контролери – ці пристрої не тільки обробляють дані; потім вони приймають рішення на основі даних. Контролери, як правило, програмуються для виконання таких завдань, як керування датчиками або приводами та обмін даними з іншими пристроями в системі IoT [4].

Мережні компоненти IoT. Компоненти мережі IoT з’єднують пристрої в системі та дозволяють їм спілкуватися один з одним.

Ключові мережеві компоненти системи IoT перераховані ніжче.

Протоколи бездротового зв’язку: зв’язок між пристроями IoT зазвичай бездротовий. Вони використовують такі протоколи, як Wi-Fi, Bluetooth, Z-wave. Кожен протокол IoT має різні сильні та слабкі сторони, і ваш вибір має залежати від конкретної програми.

Шлюз Інтернету речей: шлюз Інтернету речей або інтернет-шлюз — це пристрій, який знаходиться між пристроями Інтернету речей і хмарою або центральним сервером. Це допомагає керувати трафіком даних, контролювати безпеку та гарантувати сумісність пристроїв.

Хмарний або центральний сервер: цей компонент пропонує можливості обробки, керування та зберігання даних, а також забезпечує інтерфейс для взаємодії користувачів із системою. Сервер може бути розміщений локально або в хмарі [3].

Безпека: як уже згадувалося, безпека є однією з найбільших проблем у будь-якій системі IoT. Ви повинні застосувати протоколи безпечного зв’язку, автентифікації та шифрування, щоб захистити систему від несанкціонованого доступу та витоку даних.

Програмні компоненти IoT. Оскільки архітектура IoT полягає в підключенні розумних пристроїв через захищену мережу, програмне забезпечення IoT відіграє вирішальну роль.

Архітектура програмного забезпечення IoT є мозком системи. Це програмне забезпечення працює на центральному сервері або шлюзі IoT.

Ключові програмні компоненти системи IoT перераховані нижче.

Збір даних: це програмне забезпечення збирає дані з усіх підключених пристроїв і використовує відповідні протоколи зв’язку для передачі даних на центральний сервер.

Керування даними: центральний сервер зберігає та керує зібраними даними для аналізу, пошуку та зберігання.

Аналіз даних: програмне забезпечення використовує розширені алгоритми для аналізу зібраних даних, виявлення закономірностей і прогнозування. Система може використовувати цей аналіз для оптимізації продуктивності системи для прийняття рішень і надання корисної інформації [4].

Віддалений доступ і контроль: завдяки високому рівню підключення в мережі IoT можна отримати доступ майже до будь-якого пристрою в системі

з будь-якого місця, будь то в сусідній кімнаті чи іншому часовому поясі. Якщо є правильні облікові дані, цей віддалений доступ дозволить виконувати майже будь-яке завдання, включно з усуненням несправностей у разі виникнення проблеми або конфігурації певного пристрою [3].

Інтерфейс користувача: більшість людей зазвичай просто взаємодіють з інтерфейсом користувача, який дозволяє використовувати та отримувати користь від пристроїв у системі. У багатьох випадках інтерфейс користувача – це простий у використанні сенсорний екран.

Розробка архітектури системи IoT є складним завданням, і необхідно враховувати кілька принципів, щоб забезпечити її ефективність і сталість.

Модульність: модульність життєво важлива в архітектурі IoT, щоб забезпечити гнучке масштабування та легке обслуговування. Більшість високорівневих систем IoT розроблено таким чином, що розробники програмного забезпечення можуть або додавати нові пристрої до мережі, або навіть оновлювати частини програмного забезпечення, не обов’язково руйнуючи всю систему [3].

Ефективність: завдяки величезним обсягам даних, які генеруються пристроями Інтернету речей, дуже важливо переконатися, що система може ефективно справлятися з навантаженням. Це передбачає вибір належного обладнання, програмного забезпечення та протоколів зв’язку.

Безпека: хакери завжди шукають найменші лазівки, щоб проникнути та заволодіти критично важливою системою. Тому вкрай важливо включити безпеку в кожен аспект архітектури IoT. Найкращі системи мають численні засоби шифрування та низку безпечних протоколів зв’язку, щоб уберегти систему від хакерів і неавторизованих людей [3].

Масштабованість: кожна система повинна бути здатна витримувати зростання або розширення. Це вимагає проектування системи з урахуванням зростаючої кількості пристроїв і даних, що генеруються ними.

1.4 Замки як складова розумного будинку

У минулому фізичні замки традиційно використовувалися для безпеки та контролю доступу в наших домівках. Із впровадженням нових технологій вони зазнали значних змін. Однак ефективність цих досягнень залежить від різних факторів, включаючи конкретну модель замка та те, наскільки ретельно користувачі захищають свої мережі розумного будинку [4].

Більшість розумних замків, як правило, оснащено бездротовим підключенням, таким як Wi-Fi, Bluetooth тощо. Ці можливості дають їм можливість безперебійно спілкуватися з іншими пристроями розумного дому IoT і величезним простором Інтернету. У результаті, поки замок залишається підключеним, користувачі все ще можуть дистанційно замикати та відмикати двері, перевіряти стан замка та отримувати сповіщення про будь-які спроби несанкціонованого доступу. Однак виникає актуальне питання: наскільки добре ці розумні замки працюють в автономному режимі [4].

Розумні замки розроблені для надійної роботи навіть за відсутності підключення до Інтернету або якщо їхні програми для смартфонів тимчасово недоступні. На рсунку 1.2 зображені різні варіати доступу за допомогою яких можна зробити замки.

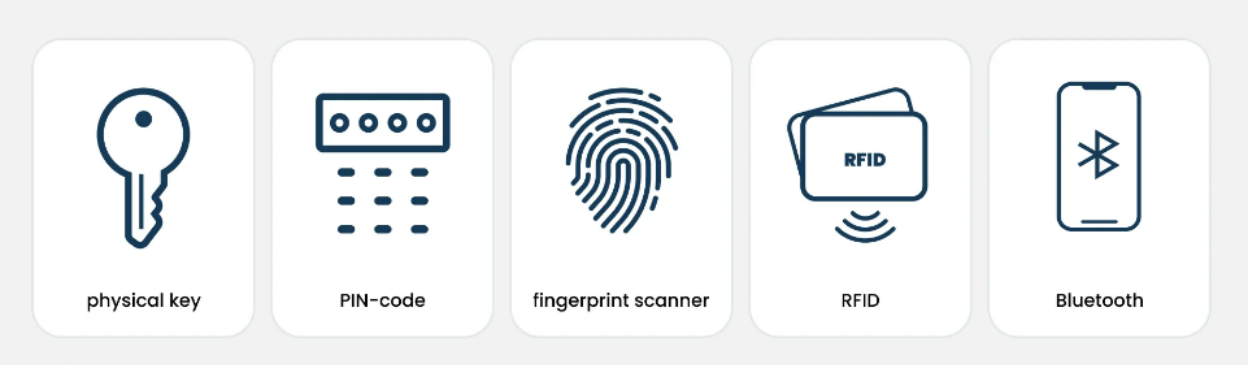


Рисунок 1.2 – Різновиди доступу в замках [4]

Більшість розумних замків все ще мають традиційну замкову щілину, тож користувачі можуть використовувати фізичний ключ, щоб відімкнути двері в разі відключення електроенергії чи мережі [5].

Деякі інші розумні замки мають панель PIN-коду для отримання доступу.

RFID є одним із найпоширеніших методів аутентифікації. Ви можете піднести до замка брелоки на основі RFID або карти доступу для входу. Дізнайтеся більше про технологію RFID у нашій статті.

Багато замків постачаються з уже вбудованим сканером відбитків пальців, який дозволяє ідентифікувати користувача та авторизувати доступ.

Розумні домашні пристрої Bluetooth, такі як розумний замок, мають можливість безпосередньо спілкуватися зі смартфоном, який з’єднано з ним, навіть у ситуаціях, коли підключення до Інтернету недоступне.

Без сумніву, у кожній ситуації, окрім першої, важливо, щоб розумний замок мав ретельно захищене вбудоване автономне джерело живлення, щоб забезпечити бездоганну роботу. Навіть якщо ви спробуєте проникнути в захищене цим замком місце в автономному режимі, це буде надзвичайно складно і повно перешкод. Більшість розумних замків підвищують безпеку своїх даних за допомогою шифрування та зміцнюють захист за допомогою різноманітних функцій безпеки [5].

1.5 Висновки за розділом 1

У цьому розділі було проаналізовано підсистеми розумного будинку такі як: розумні домашні хаби; мережа та підключення; розумні динаміки та голосові помічники; розумне освітлення; розумні термостати; розумні системи безпеки; розумні камери; розумні дверні замки; розумні датчики; розумні вилки та розетки; розважалні системи; розумна техніка; розумні жалюзі та штори.

Розумні замки є одною зі складових підсистем розумного будинку. Як показало дослідження вони можуть бути відчинені зі звичайним фізичним ключем, секретним кодом, по відбітку пальця, за допомогою RFID, або по Bluetooth.

У цій роботі буде розглянуто розробку прототипу автономної системи з кодовим доступом, який є більш надійнишим і безпечнішим ніж зі звичайним фізинм ключом.

2 ІНСТРУМЕНТАРІЇ РОЗРОБКИ

З моменту своєї появи платформи електронного прототипування дозволили творцям, любителям і власникам бізнесу перетворити свої концепції на робочі прототипи.

Багато прототипних платформ зі світу «виробників» і культури «зроби сам» все частіше використовуються для швидкого створення прототипів промислових продуктів. Подібний спосіб має багато переваг, але також має деякі недоліки, які слід враховувати при виборі платформи.

2.1 Аналіз платформ Arduino та Raspberry Pi

З моменту свого запуску в 2005 році добре відома платформа електронного прототипування з відкритим кодом Arduino не створювався як платформа для розробки нових продуктів, і спочатку він був створений для освітніх цілей [6].

Але завдяки своїй гнучкості та популярності він виріс далеко за межі початкового призначення, і багато новаторів використовують його як платформу для створення прототипів.

Фактично, на ринку є багато популярних комерційних продуктів, які почалися як прототип на основі Arduino [6].

Arduino — це бренд кількох електронних плат із відкритим кодом, які містять мікроконтролер. Наприклад, перша плата Arduino Uno, або Duo заснована на мікроконтролерах сімейства AVR. Arduino Due розроблено на основі архітектури Cortex-M3 ARM [7].

Raspberry Pi — це нанокомп’ютер розміром з кредитну картку, призначений для навчання комп’ютерному програмуванню. На ньому можна запускати ОС Linux і сумісне програмне забезпечення [8].

Крім Arduino та Raspberry Pi існують інші платформи для створення прототипів, наприклад BeagleBoard, яка також є міні-комп’ютером з єдиною

електронною платою на базі Texas Instruments AM335x. Це також дає можливість запускати Linux.

2.1.1 Переваги платформ Arduino та Raspberry Pi

Екосистеми Arduino / Raspberry Pi мають спільні переваги.

По-перше, ці дві екосистеми дають змогу швидко й недорого побудувати апаратну архітектуру, зібравши материнську плату з екранами (дочірніми платами) і роз'ємними платами (модулями). Таким чином, можна додати датчики, відеовходи, GPS або комунікаційні модулі, призначені для IoT (LoRa, Sigfox тощо). Крім того, часто це можливо зробити без проектування будь-якої друкованої плати або навіть без зварювання. Нарешті, є пластикові та алюмінієві корпуси для інкапсуляції прототипу [6].

Другою перевагою є те, що програмування на апаратній платформі полегшується завдяки великій спільноті ентузіастів, які безкоштовно діляться кодами та досвідом.

Таким чином, така простота дає змогу швидко продемонструвати концепцію продукту та його функціональні можливості за допомогою демонстратора, розробленого в рекордно короткі терміни. Це роблять, наприклад, компанії-початківці; це також заохочує компанії, які не мають навичок роботи з електронікою, пропонувати продукти на основі цих платформ.

2.1.2 Недоліки платформ Arduino та Raspberry Pi

Коли необхідно перейти від етапу демонстрації до продукту, виникають дві основні проблеми.

Перш ніж вивести продукт на ринок, він повинен відповідати ряду вимог, пов'язаних із маркуванням європейської відповідності; в основному йдеться про електробезпеку та електромагнітну сумісність (EMC) [9].

EMC відноситься до кондуктивних і радіаційних випромінювань і сприйнятливості/сприйнятливості до радіаційних і кондуктивних завад. Тобто, необхідно переконатися, що новий продукт не заважатиме іншим і не заважатиме навколишньому середовищу. Наприклад, треба подбати про те, щоб електронна плата не перезавантажувалася, якщо хтось телефонує поруч зі смартфоном [10].

Прототип, який часто монтували на столі, не обов'язково враховував усі вимоги, пов'язані з його кінцевим використанням. Це температура, вологість, герметичність, стійкість до ударів, падінь, вібрації, тиску тощо.

Наприклад, якщо кінцевий продукт використовується в поїзді, що працює в умовах континентального клімату, розширений температурний діапазон (-20 °C / +60 °C) може призвести до високого відсотка відмов.

Таким чином, те, що було дуже швидким і недорогим для створення прототипу, стає повільнішим і, перш за все, дорожчим для індустріалізації виробництва продукту. Кваліфікаційні тести, наприклад, у лабораторії EMC, можуть навіть продемонструвати, що платформа на основі Arduino або Raspberry PI не підходить. Тоді необхідно повністю перепроектувати плату [9].

2.1.3 Створення прототипів на платформах Arduino та Raspberry Pi

Підсумовуючи, жоден із недоліків не є неприйнятним, вони не ставлять під сумнів переваги платформ Arduino чи Raspberry Pi. Однак необхідно знати про ці обмеження та враховувати їх на етапі створення прототипу [10].

При розробці нового продукту прототип Arduino можна порівняти з грубим ескізом, який дає основне уявлення про остаточний дизайн. Хоча він може бути не таким відшліфованим або вдосконаленим, як остаточний інтегрований дизайн PCB (друкованої плати), це швидкий і простий спосіб

перевірити ідеї, підтвердити їх і створити доказ концепції. Подібно до того, як художники можуть намалювати ескіз перед початком остаточного твору, інженери пристроїв можуть використовувати прототип Arduino, щоб спланувати та вдосконалити свій продукт перед тим, як перейти до більш складного та інтегрованого дизайну друкованої плати [11].

Платформа Arduino є чудовою відправною точкою для створення електронного пристрою. Він зручний, універсальний і надає доступ до різноманітних датчиків, приводів та інших компонентів. Однак прототип Arduino може бути не найкращим рішенням, якщо ви хочете створити продукт, готовий до виробництва.

2.2 Аналіз та вибір симулятору Arduino для створення прототипу

В таблиці 2.1 представлено порівняння використання симулятора чи реального пристрою Arduino для створення початкових прототипів системи [12].

Вартість: Симулятори зазвичай безкоштовні або мають мінімальні витрати, тоді як реальні пристрої вимагають закупівлі фізичних компонентів, що може бути дорогим, особливо для початківців.

Швидкість розробки: Використання симуляторів дозволяє швидко створювати та тестувати проекти без необхідності збирати та налаштовувати апаратні компоненти, що значно економить час.

Безпека: Симулятори забезпечують безпечне середовище для тестування проектів, що важливо для новачків, які можуть допускати помилки при роботі з реальними електронними компонентами.

Простота виправлення помилок: У симуляторах легко вносити зміни та тестувати їх у реальному часі, тоді як робота з реальними пристроями вимагає фізичних змін, що може бути трудомістким.

Таблиця 2.1 – Порівняння використання симулятора Arduino чи реального пристрою

Критерій	Симулятори Arduino	Реальні пристрої Arduino
Вартість	Безкоштовні або дуже низькі витрати	Вимагають закупівлі обладнання, що э дорогим.
Швидкість розробки	Можливість швидкого створення та тестування проектів	Потрібен час на збирання і налаштування апаратури
Доступність	Доступні онлайн з будь-якого місця	Потрібно мати доступ до обладнання та інструментів
Безпека	Безпечне середовище для тестування	Можливі ризики при роботі з електронними компонентами
Виправлення помилок	Легко вносити зміни та тестувати	Вимагає фізичних змін, що може бути трудомістким
Реалістичність	Може не повністю відтворювати всі фізичні властивості	Відображає реальні умови роботи пристроїв
Гнучкість	Легко моделювати та змінювати компоненти	Обмежена можливостями наявного обладнання
Тестування складних сценаріїв	Легко тестувати різні сценарії та умови	Вимагає створення фізичних умов для тестування
Екологічність	Не створює фізичних відходів	Можуть виникати електронні відходи при оновленнях або несправностях

Доступність: Симулятори доступні онлайн і можуть бути використані з будь-якого місця з доступом до Інтернету, що забезпечує гнучкість та мобільність у навчанні та розробці проектів.

Навчання: Симулятори забезпечують широкий спектр інтерактивних навчальних матеріалів і дозволяють легко демонструвати та пояснювати принципи роботи електронних компонентів та схем [12].

Проєктування прототипу може спричинити багато накладних витрат, але це можна обійти. Перш ніж починати реальне проєктування, можна почати експериментувати з симулятором Arduino. Хороший симулятор повинен дозволити цифрово відтворити кілька аспектів процесу [13]:

- створити власні компоненти та схеми (або імпортувати з бібліотеки);
- створити програми (скетчі) або імпортувати з Arduino IDE;
- моделювати взаємодію між Arduino, інтерфейсами вводу-виводу та програмою;
- конструкторські дошки та схеми (не обов’язково);
- експортні плати та схеми для виробництва друкованих плат (необов’язково).

2.2.1 Онлайн симулятори

У Microsoft Maker Code створення симуляцій за допомогою різних плат, у тому числі моделей Arduino, виконується за допомогою візуальних блоків, що робить його доступним навіть для тих, хто не має попереднього досвіду програмування. Можна також вибрати програмування на Python або JavaScript. Усе в цьому середовищі дуже інтуїтивно зрозуміле, і як тільки ви визначите основні поняття, ви зможете створювати чудові симуляції [14].

Незважаючи на те, що це онлайн-платформа, також можна підключати фізичні пристрої. Є доступ до різноманітних розширень для датчиків та інших компонентів, які стануть чудовою підмогою для симуляції. На додаток до найпростіших функцій, є розширення, які можна знайти нижче на панелі, щоб надати додаткові функції, такі як джойстик, датчики або навіть команди для підтримки флеш-накопичувача USB і флеш-пам’яті [13].

Інтерфейс користувача досить спрощений, він відображає ілюстративну панель з анімацією ліворуч і панель програмування блоків праворуч. Деякі команди виведення відображають результати навіть без підключення Arduino або компонента, наприклад звуку. Можна роздрукувати електронну схему підключення, щоб зробити фізичний монтаж.

Розробник: Microsoft. Відкритий код: Ні. Найкраще для: від початківців до користувачів середнього рівня. Спеціальні функції: Тестуйте різні віртуальні компоненти, зручно для початківців. Сумісність з: усіма системами (на базі Інтернету). Ціна: безкоштовно.

Tinkercad Circuits — це безкоштовний онлайн-сервіс від Autodesk, який був запущений у 2017 році та є, мабуть, найзручнішим симулятором Arduino. Ви можете легко розробити власні схеми, створити програму в блочному або текстовому форматі, а потім налагодити її [15].

Симуляція плат Arduino та інтерфейсів вводу-виводу та взаємодія з кодом працюють як шарм, і код можна завантажити та поділитися з іншими виробниками.

Є обмеження, звичайно. Tinkercad дозволяє використовувати будь-який елемент зі своєї бібліотеки, але не дозволяє додавати нові параметри компонентів (модулі Arduino, різні моделі плат Arduino, датчики Arduino) до бібліотеки, і ви не можете змінювати елементи, які наявні в бібліотеці. Деякі базові компоненти, такі як резистори, можна параметризувати, але це не можливо для мікроконтролерів, а для Arduino існує лише модель Arduino Uno [13].

Щоб спробувати Tinkercad, потрібно приєднатися, вибравши обліковий запис викладача, студента чи особистий обліковий запис.

Розробник: Autodesk. Відкритий код: Ні. Найкраще для: від початківців до користувачів середнього рівня. Особливості: експортні плати та схеми для виробництва друкованих плат. Сумісність із: MacOS, Windows, Linux тощо (веб-інтерфейс). Ціна: безкоштовно.

Wokwi базується на AVR8js, реалізації JavaScript 8-розрядної архітектури AVR. На сторінці GitHub можна знайти Wokwi-elements і Wokwi-playgrounds. Тут можна досліджувати приклади та симулювати їх, а також змінювати ескіз і зовнішній вигляд схеми (за допомогою файлу diagram.json) [16].

Якщо необхідно зберегти (зробити копію) приклад, потрібно буде зареєструватися в Google або GitHub. Після входу з’явиться меню (у верхньому правому куті екрана) з такими варіантами: сервер Discord, мої проекти, клуб і вихід.

Це не інтерфейс з перетягуванням елементів, тому потрібно буде вивчити наявні приклади, скопіювати їх, змінити та перевірити результати самостійно. Зробивши це, можна створити власну симуляцію. Щоб створити власну схему, потрібно змінити файл diagram.json. Для отримання додаткової інформації дивіться документи [13].

Досвідчені користувачі можуть створювати або додавати власні частини та компоненти, а також додавати бібліотеки Arduino (клацніть маленьку стрілку біля списку файлів, виберіть «Новий файл…» і додайте файли .h або .cpp). Єдине обмеження полягає в тому, що неможливо експортувати плати та схеми для виробництва друкованих плат.

Розробник і спільнота дуже активні, і Wokwi швидко розвивався. У каналі Discord можна задати питання та отримати підтримку, в тому числі від розробника [13].

А якщо безкоштовна версія не відповідає вашим потребам, можна приєднатися до клубу Wokwi за ~$7/місяць, що, окрім можливості налаштування, також дає змогу зберігати конфіденційність власних проєктів.

Розробник: Uri Shaked. Відкритий код: Так. Найкраще для: середніх та досвідчених користувачів. Особливості: чат Discord (онлайн довідка), репозиторій GitHub. Сумісність із: MacOS, Windows, Linux тощо (веб-інтерфейс). Ціна: безкоштовно (необов'язкова щомісячна пожертва)

2.3 Висновки до розділу 2

У цьому розділі було проведено детальний аналіз інструментів для розробки прототипів, зокрема платформ Arduino та Raspberry Pi, а також симуляторів для створення віртуальних прототипів. Встановлено, що кожна з платформ має свої переваги та недоліки, що визначає їх використання залежно від потреб проекту. Arduino відзначається простотою використання та низькою вартістю, тоді як Raspberry Pi пропонує більшу обчислювальну потужність та гнучкість.

Використання симуляторів, таких як Microsoft Maker Code, Tinkercad Circuits та Wokwi, дозволяє знижувати витрати на розробку та прискорювати процес створення прототипів, надаючи можливість тестування та вдосконалення без необхідності фізичного збирання. Таким чином, вибір платформи та інструментів для розробки залежить від конкретних вимог проекту, досвіду розробника та ресурсів, доступних для реалізації проекту.

3 ПРОГРАМНІ ТА АПАРАТНІ РІШЕННЯ СИСТЕМИ З КОДОВИМ ДОСТУПОМ

3.1 Апаратні компоненти для створення системи

Для реалізації системи замка з кодовим доступом було обрано наступні апаратні компоненти:

Arduino Uno як основний мікроконтролер, який керує всіма периферійними пристроями та обробляє введення/виведення даних. Обрано Arduino Uno через його широку популярність, простоту використання та підтримку різноманітних додаткових модулів та бібліотек.

LED світлодіоди для індикації стану замка. LED легко керувати з Arduino та вони дозволяють візуалізувати різні стани системи.

П'єзоелектричний динамік для відтворення звукових сигналів. Він забезпечує можливість відтворювати звукові сигнали для сповіщення про різні події.

Servomotor (сервопривід) для механічного відкривання та закривання замка. Сервопривід забезпечує точний та контрольований рух.

Рідкокристалічний дисплей LCD 16x2 використовується для відображення інформації користувачу, такої як привітальні повідомлення та повідомлення про стан замка.

Клавіатура 4x4 використовується для введення коду доступу користувачем. Модуль 4x4 забезпечує можливість зручного введення чисел та символів.

3.2 Архітекутра системи

Архітектура системи з кодовим доступом базується на використанні мікроконтролера Arduino Uno як основного управляючого пристрою. Крім того, до системи входять ряд периферійних пристроїв, таких як світлодіоди

(LED), п'єзоелектричний динамік (piezo), сервопривід для замка, рідкокристалічний дисплей (LCD) 16x2 та клавіатура 4x4.

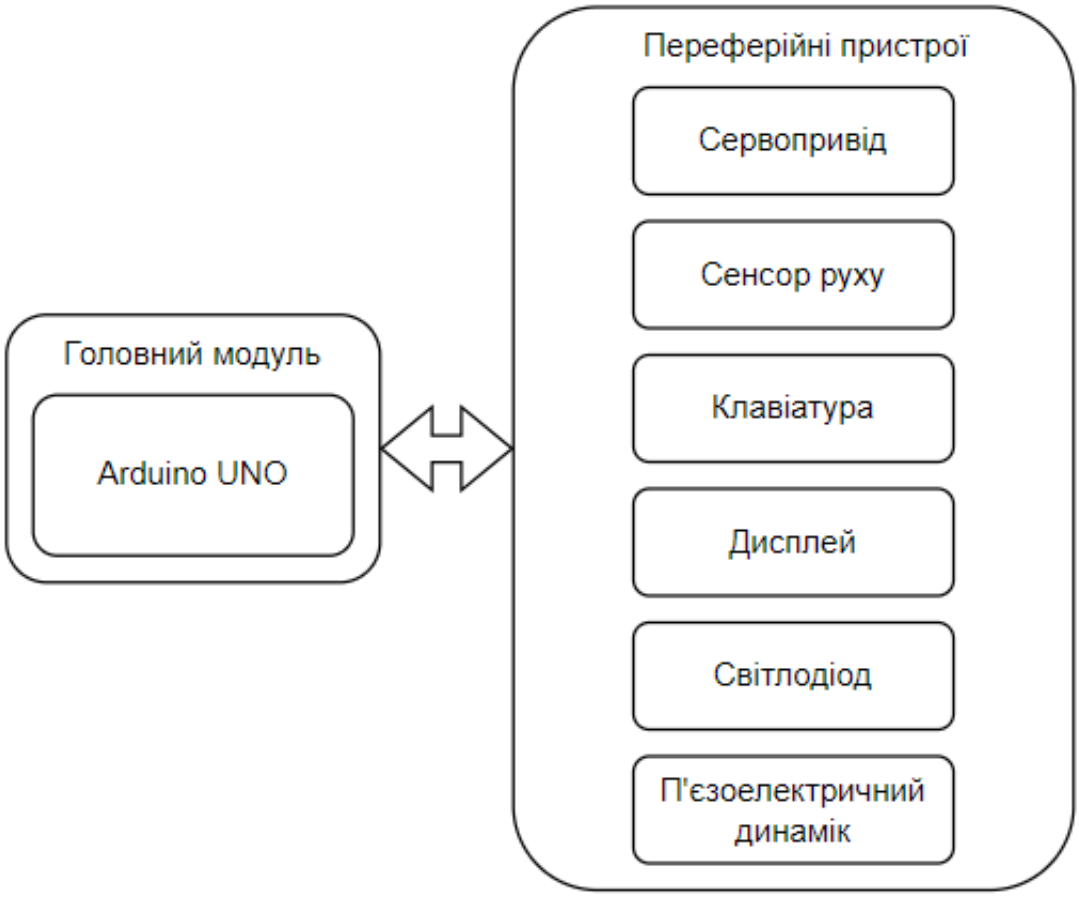


Рисунок 3.1 – Архітектура системи з кодовим досутпом

Архітектура системи відображає взаємозв'язок між всіма компонентами. Мікроконтролер Arduino Uno виступає як центральний вузол, до якого підключені всі інші пристрої через відповідні порти та піни. Наприклад, LED підключаються до цифрових виходів для індикації стану замка, п'єзоелектричний динамік для аудіо сигналізації, сервопривід для відкривання/закривання замка, LCD дисплей для відображення інформації та клавіатура для введення коду доступу.

В системі використовується шина I2C (Inter-Integrated Circuit) для забезпечення комунікації між мікроконтролером Arduino Uno та деякими периферійними пристроями, такими як LCD дисплей та клавіатура.

I2C дозволяє передавати дані між пристроями за допомогою двох дротів (SCL - Serial Clock та SDA - Serial Data), що спрощує підключення та зменшує кількість потрібних портів на мікроконтролері.

Кожен пристрій на шині I2C має свій унікальний адрес, за допомогою якого мікроконтролер визначає, з яким пристроєм він спілкується.

Завдяки використанню I2C, до Arduino Uno можна підключити декілька пристроїв за допомогою одного набору дротів, що дозволяє ефективно використовувати обмежену кількість доступних портів.

3.3 Опис діаграми послідовності роботи системи

Алгоритм роботи системи розумного кодового замка представлений у вигляді діаграми послідовності дій на рисунку 3.2. Він включає в себе кроки перераховані нижче.

Ініціалізація системи (initializeComponents):

– під час включення системи виконується ініціалізація всіх необхідних компонентів;

– ініціалізуються LCD дисплей, серводвигун, п'єзоелектричний динамік та клавіатура;

– встановлюються відповідні піни для LED, сенсора та інших компонентів.

Відображення привітального повідомлення (displayWelcomeMessage):

– на LCD дисплеї відображається привітальне повідомлення "Enter pswrd: \n";

– встановлюється початковий стан системи, включаючи LED та серводвигун.

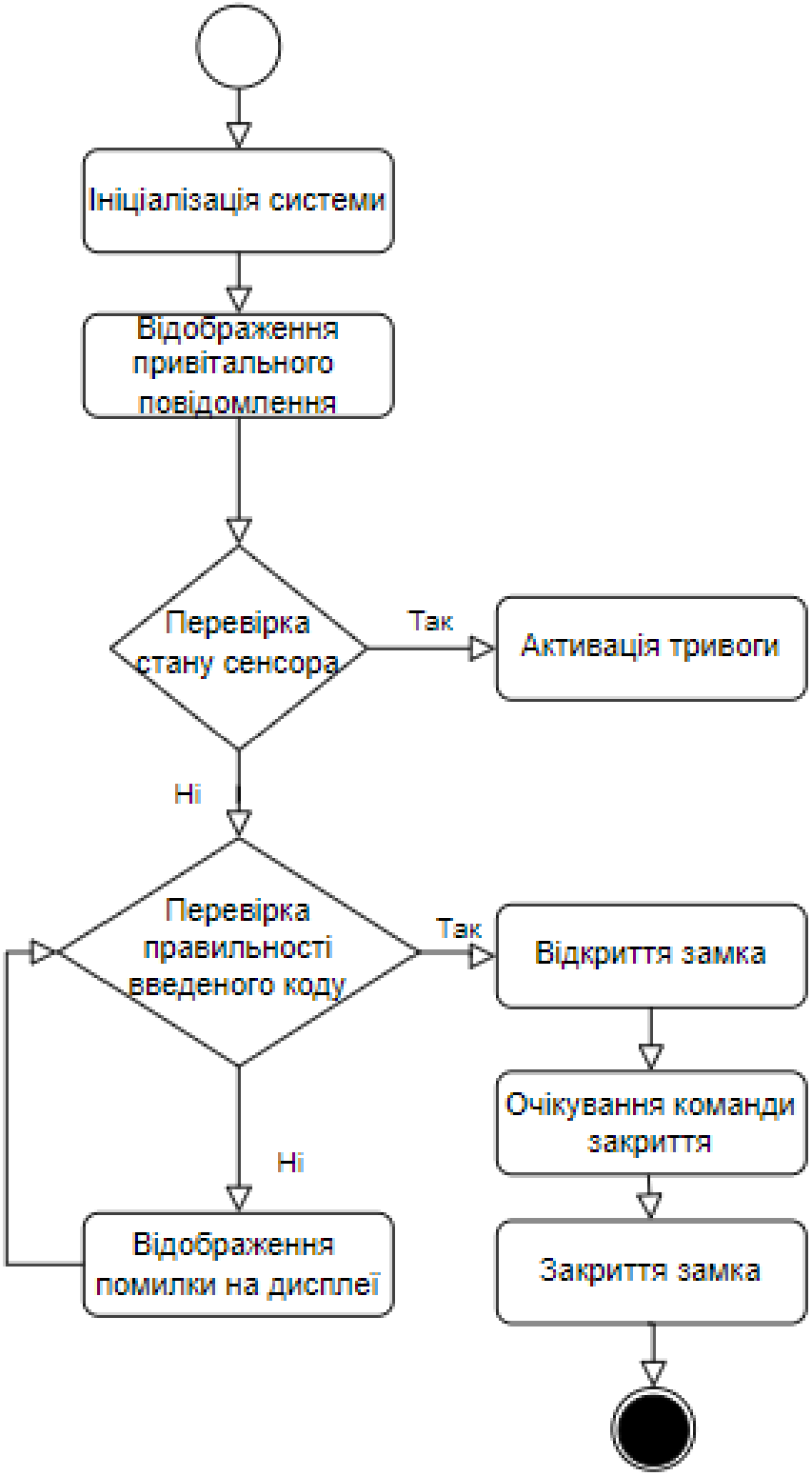


Рисунок 3.2 – Діаграма послідовності роботи системи

Перевірка правильності введеного коду (validatePassword):

– система чекає на введення чотиризначного коду користувачем через клавіатуру;

– введені символи зберігаються в масиві enterPassword;

– після введення чотиризначного коду система порівнює його з заздалегідь встановленим паролем.

Перевірка стану сенсора (checkSensorState):

– під час введення коду система постійно перевіряє стан сенсора на предмет несанкціонованого доступу;

– якщо сенсор виявляє несанкціонований доступ, викликається функція triggerAlarm.

Активація тривоги (triggerAlarm):

– у випадку несанкціонованого доступу активується звуковий сигнал через п'єзоелектричний динамік;

– на LCD дисплеї відображається повідомлення "Alarm! \n";

– тривога триває до введення спеціальної команди для її відключення.

Порівняння введеного коду (comparePassword):

– введений користувачем код порівнюється з заданим паролем;

– якщо код правильний, викликається функція displayCorrectPasswordMessage, яка відображає повідомлення про успішний доступ на LCD дисплеї;

– якщо код неправильний, викликається функція displayIncorrectPasswordMessage, яка відображає повідомлення про неправильний код на LCD дисплеї.

Відкриття замка (openLock):

– при правильному введенні коду серводвигун активується для відкриття замка;

– на LCD дисплеї відображається повідомлення "Door is opened. \n";

– LED змінює свій стан для індикації успішного доступу.

Очікування команди на закриття замка (waitForCloseCommand):

– після відкриття замка система чекає на введення команди для закриття замка;

– користувач повинен ввести спеціальну команду для закриття замка через клавіатуру.

Закриття замка (closeLock):

– при отриманні команди на закриття серводвигун активується для закриття замка;

– на LCD дисплеї відображається повідомлення "Door is closed. \n";

– LED змінює свій стан для індикації закритого замка.

Цей алгоритм забезпечує надійний доступ до приміщення через введення коду, а також має зручну індикацію та звукову сигналізацію для інформування користувача про стан системи та потенційні загрози.

3.4 Схема електрична принципова

Схема електрична принципова системи з кодовим доступом зображена на рисунку 3.3 та демонструє підключення та взаємодію всіх компонентів системи між собою та з мікроконтролером Arduino Uno. Кожен компонент має відповідні з'єднання з Arduino Uno та іншими пристроями, щоб забезпечити правильне функціонування системи.

Компоненти підключаються до Arduino наступним чином:

– LED: підключені до цифрових виходів (наприклад, пін 13) для індикації стану системи;

– Piezo: підключений до аналогового виходу (наприклад, пін A0) для відтворення звукових сигналів;

– Servo: підключений до цифрового виходу (наприклад, пін 10) для керування механічним замком;

– LCD 16x2: підключений через I2C інтерфейс (пін SDA до A4, пін SCL до A5) для відображення інформації;

– Keyboard 4x4: підключений до цифрових виходів (наприклад, пін 2-9) для введення коду доступу користувачем.

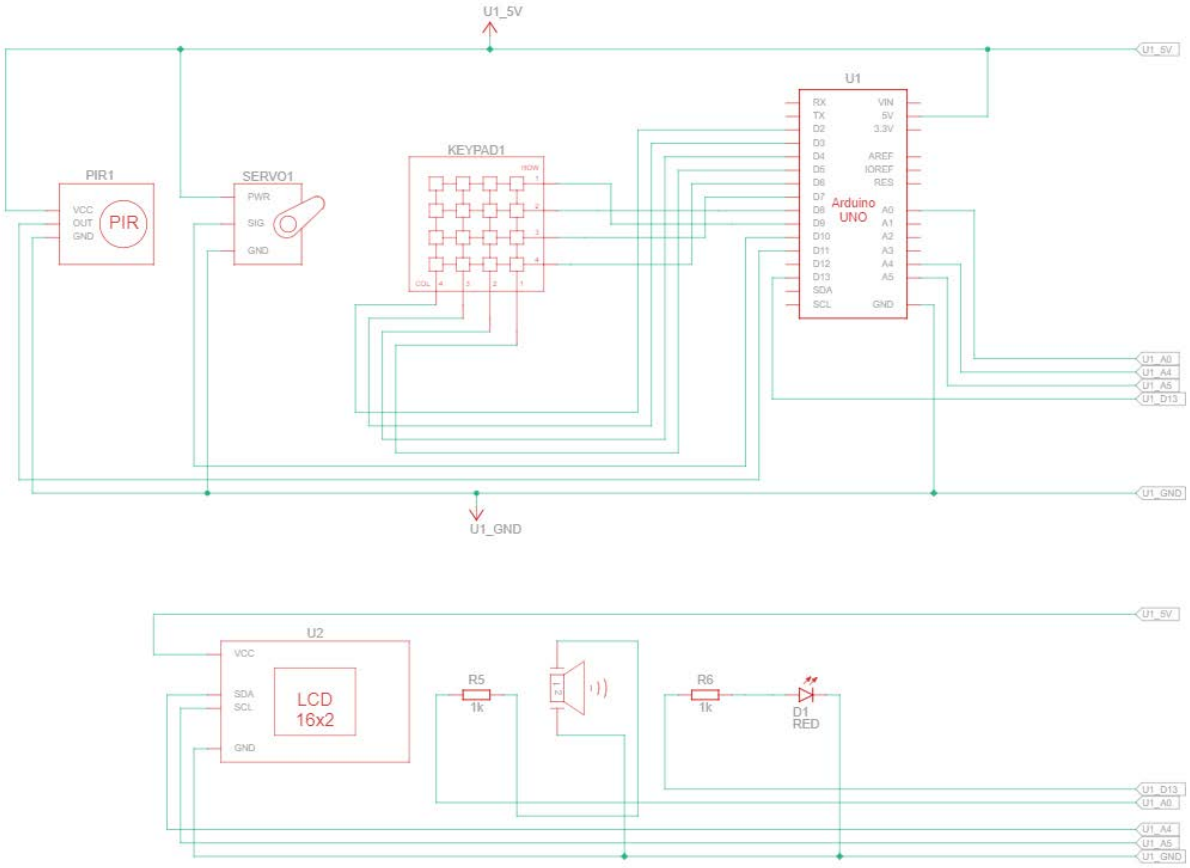


Рисунок 3.3 – Схема електрична принципова

3.5 Прототип системи

Прототип системи був створений на основі компонування всіх вищезгаданих елементів і зображений на рисунку 3.4. Він включає Arduino Uno як центральний контролер, до якого підключені всі периферійні пристрої. Система була протестована на функціональність та надійність, щоб забезпечити безперебійну роботу в різних умовах експлуатації.

Для створення прототипу використовувався симулятор Tinkercad Circuits, який дозволив моделювати роботу всіх компонентів без фізичного збирання системи. Це забезпечило можливість виявлення і виправлення потенційних помилок на ранніх етапах розробки.

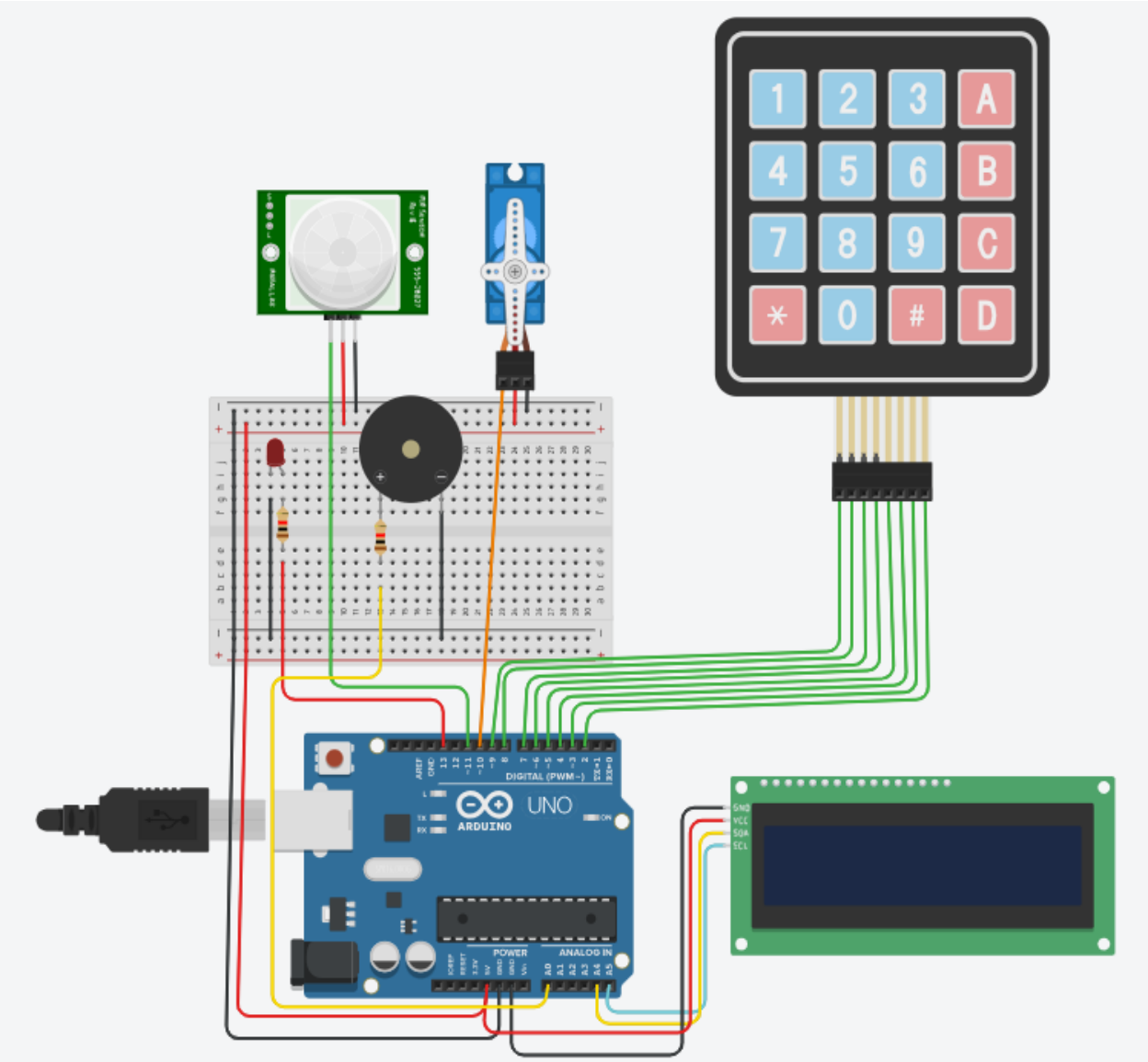


Рисунок 3.4 – Прототип системи побудований в середовищі Tinkercad

3.6 Опис основних функцій системи

Основні функції системи з кодового доступом можна поділити на такі модулі:

– безпечний доступ (система забезпечує доступ до приміщення лише за умови введення правильного коду);

– індикація стану (візуальна та звукова індикація для повідомлення користувача про успішний або невдалий доступ);

– автоматичне керування замком (відкриття та закриття замка за допомогою серводвигуна);

– відображення інформації (LCD дисплей для відображення поточного стану та інструкцій для користувача).

Вбудовані функції системи розумного кодового замка перераховані нижче.

Ініціалізація компонентів системи (setup): Ця функція відповідає за ініціалізацію всіх необхідних компонентів системи. Вона налаштовує пін контроллера, ініціалізує LCD дисплей, серводвигун, п'єзоелектричний динамік та клавіатуру.

Основний цикл роботи системи (loop): Ця функція виконується безперервно і відповідає за основну логіку роботи системи. Вона включає процес введення коду, перевірку правильності введеного коду та керування станом замка.

Функції введення та перевірки коду: Основна логіка введення та перевірки коду реалізована в основному циклі loop(). Користувач вводить код, який зберігається в масиві enterPassword. Потім введений код порівнюється з заданим паролем.

Керування серводвигуном (servo): Серводвигун використовується для відкриття та закриття замка. Функція servo_9.write() встановлює кут повороту серводвигуна для відкриття або закриття замка.

Виведення повідомлень на LCD дисплей: Функція lcd.print() використовується для виведення текстових повідомлень на LCD дисплей. Вона інформує користувача про стан системи (наприклад, успішний доступ, помилку при введенні коду, стан замка).

Аудіо сигналізація (tone, noTone): Функції tone() та noTone() використовуються для відтворення звукових сигналів через п'єзоелектричний динамік, щоб повідомляти користувача про різні події (наприклад, тривогу або помилку введення).

Опис основних розроблених функцій наведено у таблиці 4.1.

Таблиця 4.1 – Опис розроблених функцій

Функція	Опис
initializeComponents	Ініціалізація всіх необхідних компонентів системи, таких як LCD дисплей, серводвигун, п'єзоелектричний динамік та клавіатура.
displayWelcomeMessage	Відображення привітального повідомлення на LCD дисплеї та встановлення початкового стану системи.
validatePassword	Функція для введення та перевірки правильності введеного коду користувачем.
checkSensorState	Перевірка стану сенсора для виявлення несанкціонованого доступу.
triggerAlarm	Активація тривоги у випадку несанкціонованого доступу.
comparePassword	Порівняння введеного коду з заданим паролем.
displayCorrectPasswordMessage	Відображення повідомлення про успішне введення пароля.
displayIncorrectPasswordMessage	Відображення повідомлення про неправильне введення пароля.
openLock	Активація серводвигуна для відкриття замка.
waitForCloseCommand	Очікування команди на закриття замка.
closeLock	Деактивація серводвигуна для закриття замка.

3.7 Висновки за розділом

Цей розділ надає детальний огляд архітектури, структурної схеми, алгоритму роботи та принципової схеми системи розумного кодового замку, що допоможе зрозуміти принципи його роботи та взаємодії компонентів.

У цьому розділі було детально описано апаратні та програмні компоненти системи розумного кодового замка, архітектуру системи, алгоритм її роботи, а також принципову електричну схему. Це допоможе глибше зрозуміти принципи функціонування системи та взаємодію її компонентів.

4 ІНСТРУКЦІЇ ДО КОРИСТУВАННЯ СИСТЕМОЮ

4.1 Алгоритм запуску та роботи з системою

Нижче наведений покроковий текстовий алгоритм запуску та роботи з системою.

Крок 1. Ініціалізація системи:

- підключіть живлення до мікроконтролера Arduino Uno;
- переконайтеся, що всі компоненти підключені належним чином;
- відображення привітального повідомлення на LCD дисплеї.

Крок 2. Введення коду доступу:

- введіть чотиризначний код через клавіатуру;
- переконайтеся, що кожен символ правильно відображається на LCD дисплеї.

Крок 3. Перевірка коду:

- система автоматично перевіряє введений код;
- якщо код правильний, замок відкривається, і на LCD дисплеї відображається повідомлення про успішний доступ;
- якщо код неправильний, на LCD дисплеї відображається повідомлення про помилку, і прозвучить звуковий сигнал.

Крок 4. Закриття замка:

- для закриття замка повторно введіть код і натисніть відповідну кнопку;
- після введення правильного коду, система закриє замок і відобразить відповідне повідомлення на LCD дисплеї.

Крок 5. Резервне живлення та обслуговування:

- система обладнана резервним живленням, яке забезпечує роботу у випадку відключення основного живлення;
- регулярно перевіряйте стан батарей резервного живлення та замінюйте їх у разі потреби.

4.2 Вхідні дані

Система розумного кодового замка включає різноманітні вхідні та вихідні дані, які забезпечують її функціонування та взаємодію з користувачем. У цьому підрозділі розглянемо детальніше ці дані та повідомлення, які виводяться користувачу.

Вхідні дані наведені нижче.

Чотиризначний код доступу, введений через клавіатуру:

– користувач вводить чотиризначний код за допомогою клавіатури 4x4, кожне натискання клавіші передається на мікроконтролер для обробки;

– введений код зберігається в масиві `enterPassword` для подальшої перевірки.

Команди користувача для відкриття та закриття замка:

– після введення коду користувач підтверджує введення спеціальною клавішею, наприклад, клавішею `D`;

– команда для закриття замка може бути введена, наприклад, клавішею `C`.

Сигнали від датчика несанкціонованого доступу. Датчик, підключений до мікроконтролера, постійно моніторить стан приміщення. Якщо датчик виявляє несанкціонований доступ, він передає сигнал на мікроконтролер. Його можна вимкнути за допомогою клавіщі `A`.

4.3 Вихідні дані

Стан замка (відкритий або закритий): стан замка керується серводвигуном, який відкриває або закриває замок залежно від введеного коду.

Візуальні повідомлення на LCD дисплеї: система використовує LCD дисплей для відображення повідомлень користувачу про стан системи та введення коду.

Аудіо сигналізація про успішний або невдалий доступ: п'єзоелектричний динамік відтворює звукові сигнали для інформування користувача про успішний або невдалий доступ, а також про тривогу у випадку несанкціонованого доступу.

4.4 Повідомлення користувачу

Нижче перераховані існуючі повідомлення.

Привітальне повідомлення. Після запуску системи на LCD дисплеї відображається повідомлення: "Hola".

Запит на введення коду. Система запитує користувача на введення коду: "Enter pswrd: " (рис. 4.1).

Інформація про введення коду. Під час введення коду система відображає символи або маскує їх (наприклад, зірочками): "Enter pswrd: \n" "****" (рис. 4.1).

Рисунок 4.1 – Запит на введення коду та інфо про введення кода

Повідомлення про правильний код. Якщо введений код правильний, система відображає: "Correct password \n".

Повідомлення про неправильний код. Якщо введений код неправильний, система відображає: "Incorrect passw", "Try again" (рис. 4.2).

Рисунок 4.2 – Повідомлення про неправильний код

Повідомлення про відкриття замка. Після успішного введення коду та відкриття замка система відображає: "Door is opened. \n".

Повідомлення про закриття замка. Після введення команди на закриття замка система відображає: "Door is closed. \n" (рис. 4.3).

Рисунок 4.3 – Повідомлення про закриття замка

Повідомлення про включення тривоги. Якщо датчик виявляє несанкціонований доступ, система відображає та відтворює звуковий сигнал: "Alarm! \n".

Також в системі є звукові сигнали. Сигнал успішного доступу – короткий звуковий сигнал після введення правильного коду.

Сигнал неправильного доступу – серія коротких звукових сигналів після введення неправильного коду.

Сигнал тривоги – протяжний звуковий сигнал у випадку несанкціонованого доступу.

Таким чином, система розумного кодового замка забезпечує зручний і зрозумілий інтерфейс для користувача, інформуючи його про стан системи та результати введених команд за допомогою візуальних та звукових сигналів.

4.5 Висновки до розділу

У цьому розділі було детально описано інструкції до користування системою розумного кодового замка, включаючи алгоритм запуску та роботи з системою, а також опис вхідних та вихідних даних. На основі виконаної роботи можна зробити наступні висновки.

Розроблений алгоритм роботи системи забезпечує інтуїтивно зрозумілий процес введення коду доступу та керування замком. Користувач отримує чіткі інструкції на LCD дисплеї щодо введення коду та стану замка, що спрощує взаємодію з системою.

Система ефективно обробляє вхідні дані, такі як введення коду через клавіатуру та сигнали від датчика несанкціонованого доступу. Вихідні дані представлені у вигляді стану замка, візуальних повідомлень на дисплеї та звукових сигналів. Це забезпечує користувача повною інформацією про стан системи та результати його дій.

Інформаційні повідомлення на LCD дисплеї та звукові сигнали інформують користувача про стан системи, успішність або неуспішність введення коду, а також про наявність тривоги у випадку несанкціонованого доступу. Це сприяє зручності та безпеці використання системи, дозволяючи користувачу швидко реагувати на будь-які зміни у стані системи.

ВИСНОВКИ

У дипломному проєкті було розроблено систему розумного кодового замка на базі мікроконтролера Arduino Uno. Система забезпечує надійний доступ до приміщення через введення коду, а також має зручну індикацію, виводить повідомлення користувачу на дисплей для розуміння поточного стану системи та звукову сигналізацію.

Результатами виконання дипломної роботи є:

– проаналізовано підсистеми розумного будинку та системи з кодовим доступом, що дозволило визначити основні вимоги та функціональні можливості системи;

– обґрунтовано вибір платформи Arduino для реалізації системи замка з кодовим доступом, враховуючи її простоту, доступність та підтримку великої кількості додаткових модулів та бібліотек;

– спроєктовано архітектуру системи, що включає мікроконтролер Arduino Uno, світлодіоди, п'єзоелектричний динамік, серводвигун, LCD дисплей 16x2 та клавіатуру 4x4, а також изначено взаємодію між компонентами та їх підключення;

– розроблено програмне забезпечення для мікроконтролера Arduino Uno, яке забезпечує введення та перевірку коду доступу, керування замком та індикацію стану системи, код було організовано у вигляді підмодулів для покращення читабельності та масштабованості;

– використано симулятор Tinkercad для моделювання системи без необхідності фізичного збирання, що дозволило знизити витрати на розробку та прискорити процес створення прототипу.

Розроблена система з кодовим доступом може бути використана в різних сферах, де потрібен контроль доступу, включаючи житлові приміщення, офіси та інші об'єкти.

В подальшому можливе розширення функціональності системи, зокрема інтеграція з іншими підсистемами розумного будинку та підвищення рівня безпеки за рахунок додаткових методів аутентифікації.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. 5 Ways in Which Smart Homes Can Improve Your Life [Electronic resource]. – Access mode: https://www.linkedin.com/pulse/5-ways-which-smart-homes-can-improve-your-life-laburnum-developers-jqqzc.

2. How to Build a Smart Home: A Step-by-Step Guide 2024 [Electronic resource]. – Access mode: https://www.techopedia.com/how-to/how-to-build-a-smart-home.

3. What Is IoT Architecture [Electronic resource]. – Access mode: https://smartcitystyle.com/iot-architecture.

4. Smart Home IoT Devices: Understanding the Hardware Components [Electronic resource]. – Access mode: https://indeema.com/blog/smart-home-iot-devices--understanding-the-hardware-components.

5. Designing Suitable Access Control for Web-Connected Smart Home Platforms [Electronic resource]. – Access mode: https://www.researchgate.net/publication/325785698_Designing_Suitable_Access_Control_for_Web-Connected_Smart_Home_Platforms.

6. 5 Successful Products that Began as Arduino Prototypes [Electronic resource]. – Access mode: https://predictabledesigns.com/5-successful-products-that-began-as-arduino-prototypes.

7. Arduino [Electronic resource]. – Access mode: https://www.arduino.cc.

8. Raspberri Pi [Electronic resource]. – Access mode: https://www.raspberrypi.org.

9. What are the Differences between Arduino and Raspberry Pi [Electronic resource]. – Access mode: https://www.elprocus.com/difference-between-arduino-and-raspberry-pi.

10. Prototyping an electronic board [Electronic resource]. – Access mode: https://www.elsys-design.com/en/prototyping-electronic-board.

11. You’ve built an Arduino prototype: Where do you go from here?

[Electronic resource]. – Access mode: https://www.ignitec.com/insights/youve-built-an-arduino-prototype-where-do-you-go-from-here.

12. An Arduino Simulator in Classroom– a Case Study [Electronic resource]. – Access mode: https://drops.dagstuhl.de/storage/01oasics/oasics-vol081-icpec2020/OASIcs.ICPEC.2020.12/OASIcs.ICPEC.2020.12.pdf.

13. The Best Arduino Simulators (Online & Offline) [Electronic resource]. – Access mode: https://all3dp.com/2/best-arduino-simulators-online-offline.

14. Make and Code [Electronic resource]. – Access mode: https://www.microsoft.com/en-us/makecode.

15. Tincercad [Electronic resource]. – Access mode: https://www.tinkercad.com/circuits.

16. Wokwi [Electronic resource]. – Access mode: https://wokwi.com.

ДОДАТОК А

Текст програми

```
#include <Keypad.h>
#include <Servo.h>
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x27, 16, 2);
Servo servo_9;

const byte numRows = 4;
const byte numCols = 4;
char keymap[numRows][numCols] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

int led = 13, servo = 10, sensorPin = 11, buzzerPin = A0;
byte rowPins[numRows] = {9, 8, 7, 6};
byte colPins[numCols] = {5, 4, 3, 2};
Keypad myKeypad = Keypad(makeKeymap(keymap), rowPins, colPins, numRows, numCols);

char password[4] = {'1', '2', '3', '4'};
char enterPassword[4];

void setup() {
  initializeComponents();
}
```

```
void loop() {
  displayWelcomeMessage();
  if (validatePassword()) {
    openLock();
    waitForCloseCommand();
  }
}

void initializeComponents() {
  lcd.init();
  lcd.backlight();
  lcd.begin(16, 2);

  pinMode(led, OUTPUT);
  pinMode(sensorPin, INPUT);
  pinMode(buzzerPin, OUTPUT);
  servo_9.attach(servo, 500, 2500);

  Serial.begin(9600);
}

void displayWelcomeMessage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Enter pswrd: \n");
  Serial.print("start \n");
  digitalWrite(led, HIGH);
  servo_9.write(0);
}
```

```
bool validatePassword() {
  bool passwordTrue = false;
  while (!passwordTrue) {
    if (checkSensorState()) {
      triggerAlarm();
    }

    lcd.setCursor(0, 0);
    lcd.print("Enter pswrd: \n");

    char keypressed = myKeypad.getKey();
    if (keypressed != NO_KEY) {
      for (int i = 0; i < 4; i++) {
        enterPassword[i] = keypressed;
        Serial.println(keypressed);
        keypressed = myKeypad.getKey();
      }
      if (comparePassword()) {
        passwordTrue = true;
      } else {
        displayIncorrectPasswordMessage();
      }
    }
  }
  return passwordTrue;
}

bool checkSensorState() {
  int sensorState = digitalRead(sensorPin);
  return sensorState == HIGH;
```

```
}

void triggerAlarm() {
  tone(buzzerPin, 500);
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Alarm! \n");

  while (myKeypad.getKey() != 'A') {
    // wait for 'A' key to stop the alarm
  }
  noTone(buzzerPin);
}

bool comparePassword() {
  for (int i = 0; i < 4; i++) {
    if (password[i] != enterPassword[i]) {
      return false;
    }
  }
  displayCorrectPasswordMessage();
  return true;
}

void displayCorrectPasswordMessage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Correct password \n");
  Serial.print("correct password \n");
  delay(50);
```

```
}

void displayIncorrectPasswordMessage() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Incorrect passw");
  lcd.setCursor(0, 1);
  lcd.print("Try again");
  Serial.print("incorrect password \nTry again \n");
}

void openLock() {
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Door is opened. \n");
  Serial.print("servo \n");
  digitalWrite(led, LOW);
  servo_9.write(180);
}

void waitForCloseCommand() {
  while (myKeypad.getKey() != 'C') {
    // wait for 'C' key to close the lock
  }
  closeLock();
}

void closeLock() {
  lcd.clear();
  lcd.setCursor(0, 0);
```

```
  lcd.print("Door is closed. \n");
  servo_9.write(0);
  delay(1000);
}
```

ДОДАТОК Б

Слайди презентації

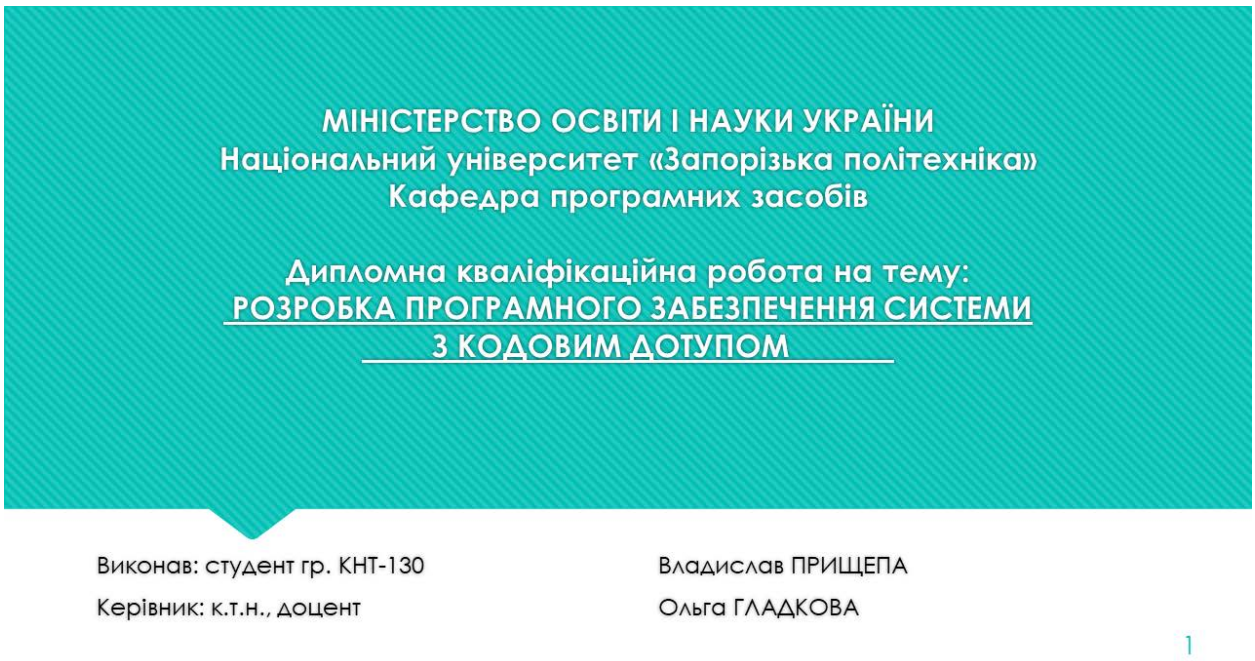


Рисунок Б.1 – Слайд №1

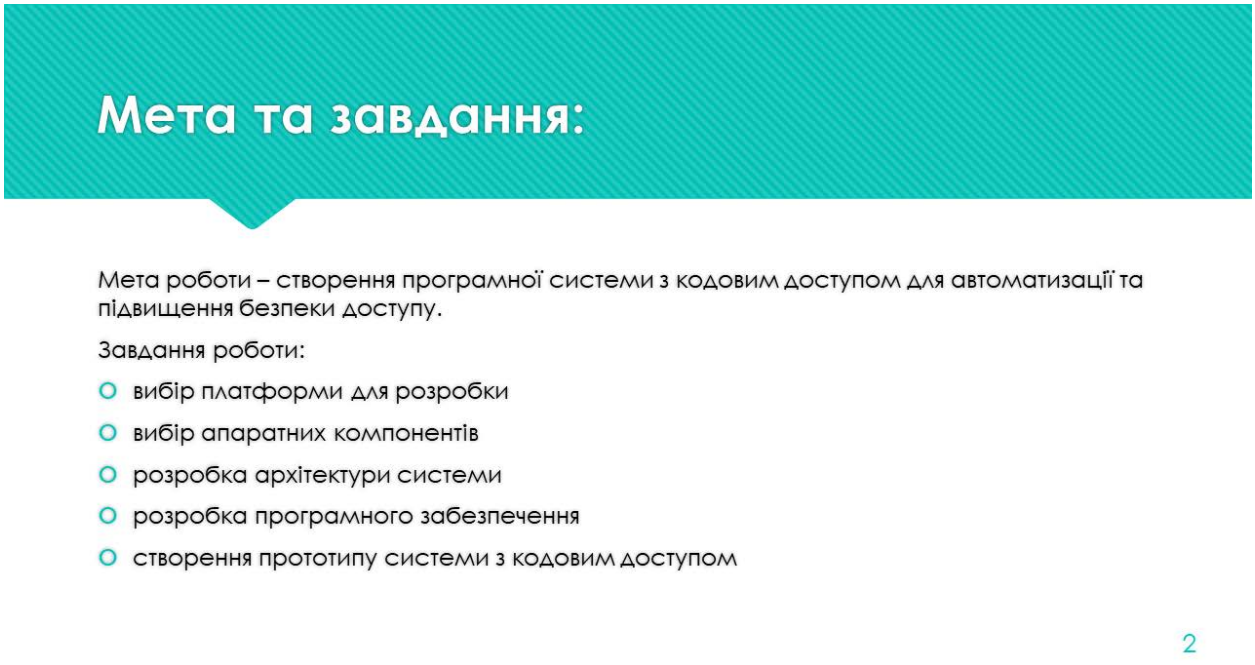
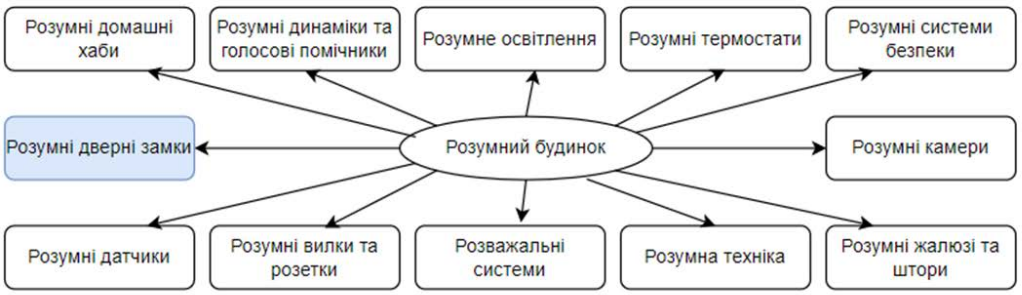


Рисунок Б.2 – Слайд №2

Підсистема з кодовим доступом як складова розумного будинку



3

Рисунок Б.3 – Слайд №3

Аргументація вибору симулятора Arduino

Критерій	Симулятори Arduino	Реальні пристрої Arduino
Вартість	Безкоштовні або дуже низькі витрати	Вимагають закупівлі обладнання, що є дорогим.
Швидкість розробки	Можливість швидкого створення та тестування проектів	Потрібен час на збирання і налаштування апаратури
Доступність	Доступні онлайн з будь-якого місця	Потрібно мати доступ до обладнання та інструментів
Безпека	Безпечне середовище для тестування	Можливі ризики при роботі з електронними компонентами
Виправлення помилок	Легко вносити зміни та тестувати	Вимагає фізичних змін, що може бути трудомістким
Реалістичність	Може не повністю відтворювати всі фізичні властивості	Відображає реальні умови роботи пристроїв
Гнучкість	Легко моделювати та змінювати компоненти	Обмежена можливостями наявного обладнання
Тестування складних сценаріїв	Легко тестувати різні сценарії та умови	Вимагає створення фізичних умов для тестування
Екологічність	Не створює фізичних відходів	Можуть виникати електронні відходи при оновленнях або несправностях

4

Рисунок Б.4 – Слайд №4

Рисунок Б.5 – Слайд №5

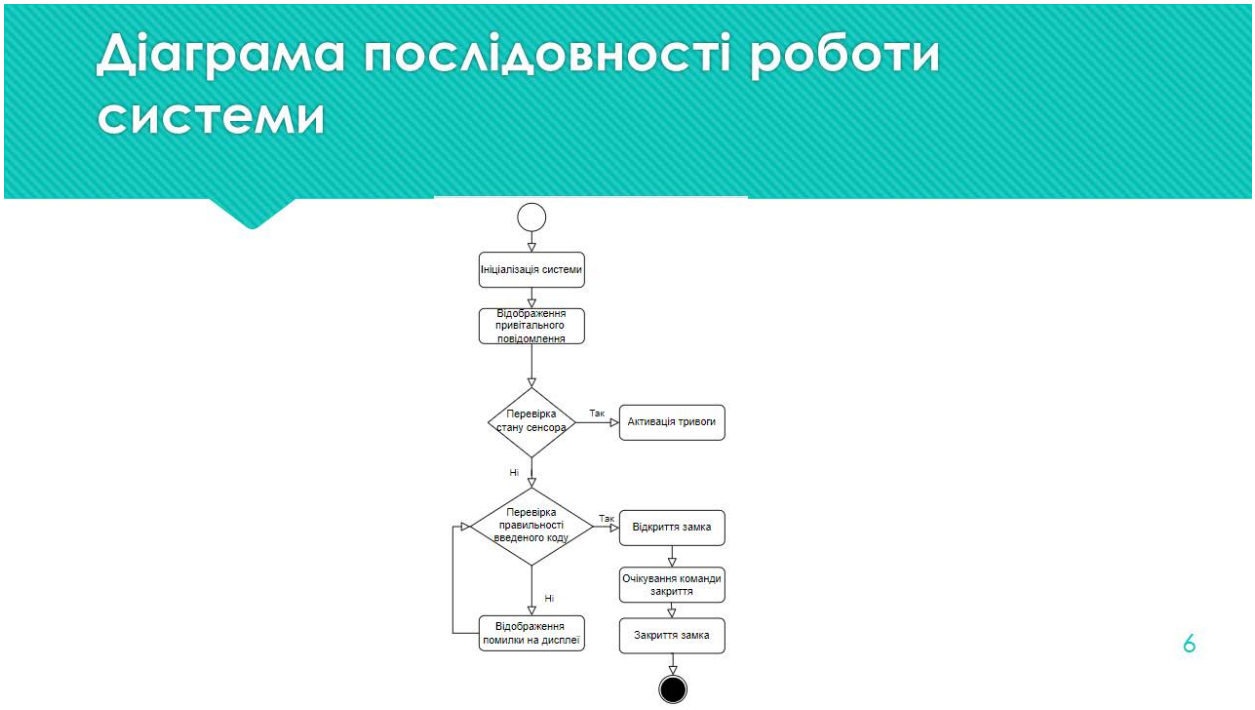


Рисунок Б.6 – Слайд №6

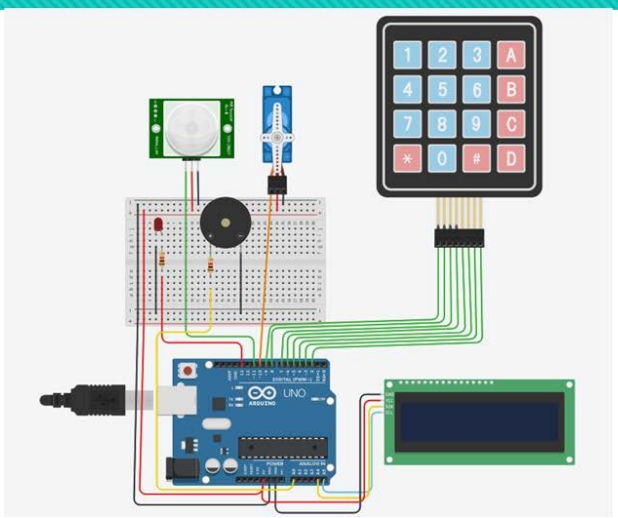
Основні функції

Функція	Опис
initializeComponents	Ініціалізація всіх необхідних компонентів системи, таких як LCD дисплей, серводвигун, п'єзоелектричний динамік та клавіатура.
displayWelcomeMessage	Відображення привітального повідомлення на LCD дисплеї та встановлення початкового стану системи.
validatePassword	Функція для введення та перевірки правильності введеного коду користувачем.
checkSensorState	Перевірка стану сенсора для виявлення несанкціонованого доступу.
triggerAlarm	Активація тривоги у випадку несанкціонованого доступу.
comparePassword	Порівняння введеного коду з заданим паролем.
displayCorrectPasswordMessage	Відображення повідомлення про успішне введення пароля.
displayIncorrectPasswordMessage	Відображення повідомлення про неправильне введення пароля.
openLock	Активація серводвигуна для відкриття замка.
waitForCloseCommand	Очікування команди на закриття замка.
closeLock	Деактивація серводвигуна для закриття замка.

7

Рисунок Б.7 – Слайд №7

Прототип системи



8

Рисунок Б.8 – Слайд №8

Вхідні та вихідні данні, повідомлення користувачу

Вхідні дані:

- чотиризначний код доступу;
- команди користувача для закриття дверей;
- виключення сигналу несанкціонованого доступу.

Вихідні дані:

- стан замка (відкрито чи закрито);
- повідомлення користувача на дисплей;
- аудіо сигнали про успішний та невдалий доступ.

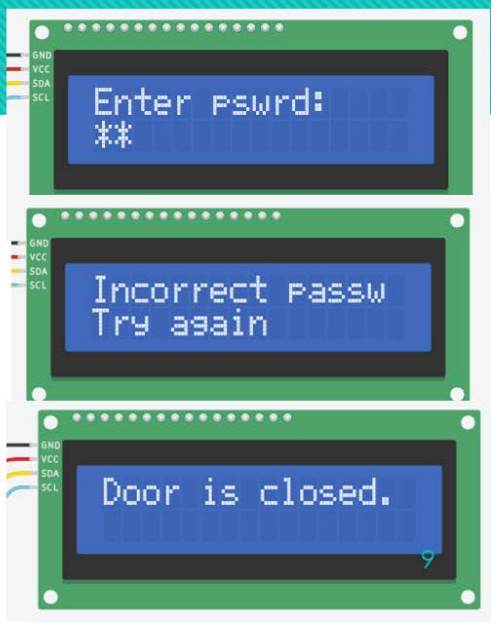


Рисунок Б.9 – Слайд №9

Висновки

Результатами виконання дипломної роботи є:

- проаналізовано підсистеми розумного будинку та системи з кодовим доступом, що дозволило визначити основні вимоги та функціональні можливості системи;
- обґрунтовано вибір платформи Arduino для реалізації системи замка з кодовим доступом, враховуючи її простоту, доступність та підтримку великої кількості додаткових модулів та бібліотек;
- спроєктовано архітектуру системи, що включає мікроконтролер Arduino Uno, світлодіоди, п'єзоелектричний динамік, серводвигун, LCD дисплей 16x2 та клавіатуру 4x4, а також визначено взаємодію між компонентами та їх підключення;
- розроблено програмне забезпечення для мікроконтролера Arduino Uno, яке забезпечує введення та перевірку коду доступу, керування замком та індикацію стану системи, код було організовано у вигляді підмодулів для покращення читабельності та масштабованості;
- використано симулятор Tinkercad для моделювання системи без необхідності фізичного збирання, що дозволило знизити витрати на розробку та прискорити процес створення прототипу.

10

Рисунок Б.10 – Слайд №10