ЧЕРКАСЬКИЙ ДЕРЖАВНИЙ БІЗНЕС-КОЛЕДЖ

Кафедра комп’ютерної інженерії та інформаційних технологій

КВАЛІФІКАЦІЙНА РОБОТА

на тему

«Розробка інтегрованої системи керування смарт-будинком на основі мікроконтролера та ігрової симуляції»

Студента Групи КІ-22

_____Попея Н. Я._____

(прізвище та ініціали)

Керівник роботи:

_ доцент, к.т.н.___ _____ _____

_Розломій І. О.

(посада, вчене звання, науковий ступінь,

прізвище та ініціали)

Кількість балів: _____

Оцінка: ECTS _____

Члени комісії:

_доцент КІІТ, к.т.н.

_____Бурмістров С. В.,

_доцент КІІТ, доцент, к.т.н.

_____Захарова М. В.

_доцент КІІТ, доцент, к.т.н.

_____Михайлюта С. Л.

Черкаси, 2024 рік

Зміст

ВСТУП

Комфорт та якість нашого життя значною мірою залежать від нас самих і від того, які можливості для створення комфортного життя нам надає сучасний прогрес. Можливість комплексного вирішення питань автоматизації інженерних систем і звільнення часу, який раніше витрачався на рутинні побутові процеси, суттєво підвищує якість життя та робить його більш організованим. Не дивно, що з кожним роком інтерес до інтелектуальних систем у всьому світі зростає, і актуальність «розумного будинку» вже не потребує підтвердження. Якщо в 70-х роках такі можливості були недосяжними для вітчизняного споживача, то сьогодні автоматизацією житла цікавиться все більше наших сучасників.

У інтелектуальному комплексі управління будинком головну роль відіграє система безпеки «смарт-будинку», яка забезпечує спокій за життя і здоров'я близьких, а також за збереження майна.

Актуальність теми. Популярність домашньої автоматизації значно зросла останніми роками завдяки підвищеній доступності та простоті використання. З можливістю контролювати різні аспекти наших осель і автоматично реагувати на події, такі системи стають все більш популярними та необхідними з точки зору безпеки та ефективності витрат. Таким чином, «розумні будинки» призначені для забезпечення безпеки, захисту від надзвичайних ситуацій та підвищення загального рівня комфорту. Система безпеки «розумного дому» має насамперед виконувати наступні функції:

1. Захист від вторгнення сторонніх осіб, автоматизації дверей, воріт, охоронної сигналізації, запобігання аварійним ситуаціям.
2. У разі будь-якого займання або задимлення спрацює пожежна

сигналізація.

3. Про протікання води чи підвищеному рівні вологості система відразу повідомить господаря та відповідні служби.

Мета і задачі виконання роботи. Метою цієї кваліфікаційної роботи є розробка системи датчиків на базі мікроконтролера, яка буде відстежувати відхилення від встановлених норм і передавати інформацію до ігрової симуляції «розумного будинку».

Для досягнення встановленої мети потрібно вирішити наступні завдання:

1. Виконати аналіз предметної області системи датчиків «смарт-будинку».
2. Провести порівняльний аналіз та огляд існуючих систем управління

   «смарт-будинком».
3. Виконати аналіз та проектування системи датчиків під’єднаної до ПК-додатку у вигляді ігрової симуляції «смарт-будинку».
4. Обрати необхідні технології, що знадобляться при розробці усіх частин проекту кваліфікаційної роботи та виконати розробку з урахуванням усіх аспектів.
5. Провести аналіз та тестування розробленого проекту.

Об’єктом розробки є процес контролю відхилень системою датчиків на базі мікроконтролера.

Предметом розробки є розробка ПК-додатку ігрової симуляції «смарт- будинку», що візуалізує процес реагування та оброблення повідомлень з під’єднаної системи датчиків на базі мікроконтролера.

Практична цінність проекту полягає у наданні користувачу продукту, який дозволить відстежувати показники температури та вологості, вогню, рівня води та зачинених дверей і, обробляючи дані, викликати відповідні оперативні служби та повідомляти власника будинку.

РОЗДІЛ 1 ОГЛЯД ТА ПОРІВНЯЛЬНИЙ АНАЛІЗ СИСТЕМ ДАТЧИКІВ СМАРТ-БУДИНКУ

1.1 Визначення системи «смарт-будинку» та його застосування

Система розумного дому – це автоматичне керування електронними пристроями в будинку. Ці пристрої підключені до Інтернету, що дозволяє дистанційно керувати ними. Наприклад, користувач може налаштувати освітлення за розкладом, щоб воно вимикалося, коли він лягає спати, або запрограмувати кондиціонер на ввімкнення приблизно за годину до повернення з роботи. Домашня автоматизація робить життя більш зручним і може зекономити гроші на опаленні, охолодженні та електроенергії.

Системи «розумних будинків» функціонують через мережу пристроїв, підключених до Інтернету за допомогою різних протоколів зв’язку, таких як Wi-Fi, Bluetooth, ZigBee тощо. Пристроями можна дистанційно керувати через електронні інтерфейси, використовуючи контролери, голосових помічників або додатки. Багато з цих пристроїв оснащені датчиками, які відстежують зміни в русі, температурі та освітленні, що дозволяє користувачу отримувати інформацію про оточення пристрою.

Однією з найбільших переваг домашньої автоматизації є забезпечення спокою власникам будинків. Вона дозволяє їм дистанційно контролювати свої помешкання і запобігати небезпекам, таким як залишена ввімкнена кавоварка, незамкнені вхідні двері або залишена ввімкнена праска. Однак, одночасно існують значні недоліки

використання систем «смарт будинку». Найпоширенішою проблемою є можливість злому пристроїв і недостатній захист даних користувачів. Проте протоколи безпеки постійно оновлюються, і з кожним роком ці технології стають безпечнішими.

1.2 Огляд існуючих аналогів систем «смарт-будинку»

У даному розділі розглянуто існуючі системи смарт- будинків, що займають передові позиції на ринку та завоювали прихильність користувачів своїм співвідношенням ціна – якість.

1.2.1 Система «розумного дому» Ajax

Система «розумного дому» Ajax вітчизняного виробництва, яка за замовчування підтримує український інтерфейс.

Ця система автоматизації будинку повністю справляється відразу з двома важливими завданнями:

- забезпечує комфорт та зручність в управлінні життєзабезпеченням приміщення;
- гарантує безпеку житла повною мірою, контролюючи межі об'єкта на предмет злому, а також електричну, пожежну, газову та інші можливі загрози для будинку.

Обладнання Ajax працює на надійно зашифрованому та захищеному двосторонньому радіозв'язку Jeweller власної розробки, має повну автономність від електромережі завдяки резервному джерелу живлення – хабу, характеризується стильним дизайном усіх своїх пристроїв (рис. 1.1).

Рисунок 1.1 – Система датчиків Ajax

Серед переваг даної системи можна виділити:

- велика зона дії сигналу (до 2000 м);
- наявність захисту від зняття будь-якого датчика (бампера);
- Wi-Fi та GSM-зв'язок;
- встановлення за QR-кодом та управління за допомогою смартфона через зручний додаток (iOS, Android);
- різноманітність способів інформування користувача (дзвінок, SMS, Push-повідомлення);
- додаток для управління має приємний інтерфейс та повний доступ до керування кожним датчиком (рис. 1.2);
- невисока вартість комплекту (від 200 $).

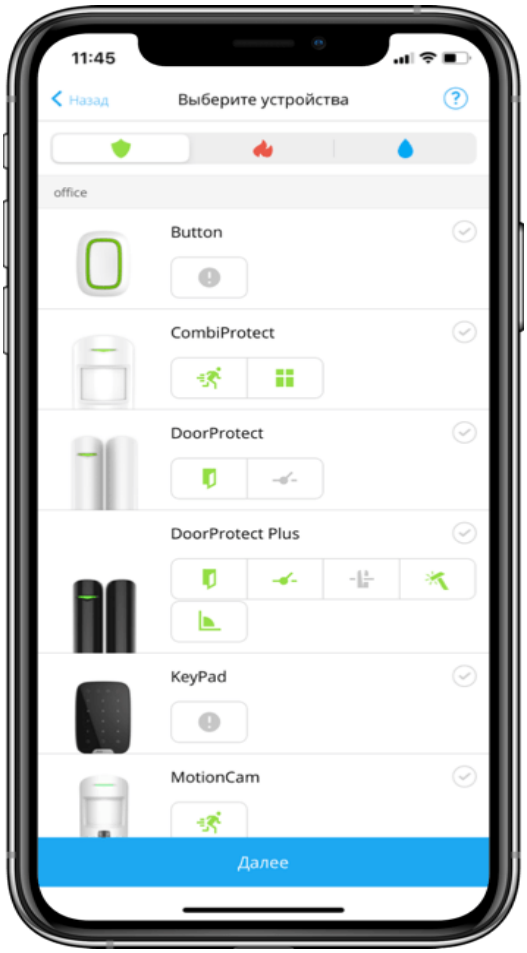


Рисунок 1.2 – Користувацький інтерфейс додатку системи Ajax

Але система має й певні недоліки:

– функціонування лише з роботою центрального контролера (Hub), тобто відсутність автономності датчиків;
– немає власної камери відеоспостереження (але є можливість підключення стороннього обладнання);
– керування лише через телефон, хоча це знімає необхідність встановлення додаткових програм на ПК.

На сьогодні обладнання Ajax – це одна з найкращих систем «Розумний дім». Вона багатофункціональна, надійна, зручна, компактна. Має якісний захист від зломів, чудовий дизайн та зрозумілий інтерфейс.

Встановлення та налаштування такого комплексу спрощено до мінімуму і цілком доступне навіть для технічно непідготовлених користувачів.

1.2.2 Система «смарт-будинку» Fibaro

Розумний будинок Fibaro відноситься до професійного обладнання для забезпечення автоматизації та безпеки будинку з найширшим функціоналом. Однак, на відміну від багатьох подібних систем, потребує встановлення та налаштування своєї апаратури досвідченими фахівцями (рис. 1.3).

Рисунок 1.3 – Комплект системи «смарт-будинку» Fibaro

У комплект входять особливо важливі пристрої: розумна розетка Wall Plug, датчик затоплення Fibaro Flood Sensor, задимлення Smoke Sensor, рух Motion Sensor, відкриття вікна, дверей FGK-101, контролер Home Center Lite.

До системи можна підключити понад двісті пристроїв. Безпека та зручність у будинку забезпечуються даним типом обладнання. Устаткування має дистанційне керування за допомогою віддаленого доступу, через програму в мобільному пристрої на базі iOS та Android (рис. 1.4).

Рисунок 1.4 – Інтерфейс додатку «смарт-будинку» Fibaro

Датчик задимлення визначає та повідомляє про підвищену температуру та дим. Пристрій має записуючий пристрій, колірну та звукову індикацію. Додатково можна підключити функції відкриття дверей, вентиляції та інших пристроїв автоматизації. Датчик підтоплення повідомить мобільний пристрій про наявність води на підлозі. Додатково можна підключити перекриття водопроводу.

Переваги системи Fibaro:

- наповнення системи великою кількістю датчиків та пристроїв;
- наявність камери відеоспостереження;
- розсилання повідомлень одразу на кілька телефонів;
- робота на базі протоколу Z-Wave, що дає змогу успішно взаємодіяти з іншим подібним обладнанням;
- розумна розетка відображає рівень енергоспоживання підключених пристроїв, а також вимикається при стрибках

напруги;

- зручний додаток для керування, що має голосове управління через сервіси Google.

Недоліки:

- висока вартість обладнання (від 600 $);
- налаштування і монтаж здійснюється лише працівниками;
- обов'язкове підключення центрального контролера Fibaro Home Center до Інтернету через LAN-кабель;
- відсутність резервного живлення хаба;
- затримка Push-повідомлень;
- необхідність обов'язкової установки програмного забезпечення на ПК, а також урізаний мобільний додаток.

1.2.3 Система «смарт-будинку» Xiaomi

Виробництво даної системи «розумного дому» здійснюється у Китаї. Відсутність українського та російського інтерфейсу за замовчуванням, лише англійська та китайська, що накладає свої складнощі в процесі встановлення та налаштування.

Розумний будинок від Xiaomi відноситься до бюджетного класу обладнання, яке дозволяє зробити управління різними пристроями та побутовою технікою в будинку максимально простим та зручним.

Комплект від популярного китайського бренду Xiaomi вигідно відрізняється від конкурентів невеликою вартістю та якістю. Компанія випускає понад чотири десятки електричних приладів, які контролюються системою Smart Home Suite. До таких приладів відносяться: потери, кондиціонери, мультиварки, лампочки тощо (рис. 1.5).

Рисунок 1.5 – Комплект датчиків системи «розумного дому»

Xiaomi при відкритті дверей або вікна спрацьовує датчик та вмикається очищувач повітря. Датчик підключається до очисника за допомогою смартфона. При відкритті дверей у спальню вмикається будильник із гарним рингтоном. Датчик за допомогою приєднання через смартфон до розумної розетки виконує охоронне завдання. У режимі охорони буде здійснюватися відеозапис та оповіщення на гаджет.

Переваги розумного будинку від Xiaomi:

- повна автономність пристроїв;
- можливість підключення багатьох пристроїв, тобто масштабування;
- наявність власної камери відеоспостереження;
- бездротовий протокол ZigBee;
- низька вартість базового комплекту (всього 90 $);
- зручне керування за допомогою додатку для смартфона через Wi- Fi (рис. 1.6).

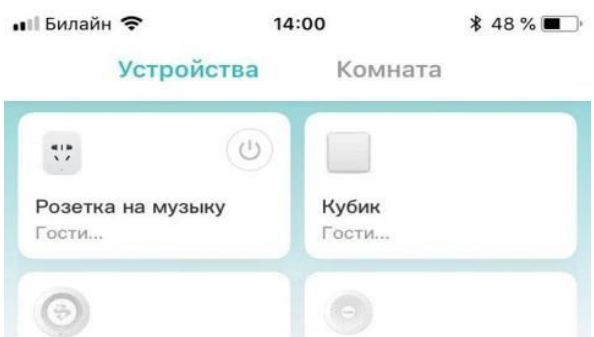


Рисунок 1.6 – Додаток для керування будинком від Xiaomi

Недоліки:

- дуже невелика зона впливу сигналу (до 10 м);
- скромний набір сенсорів та виконавчих пристроїв у базовому наборі;
- відсутність резервного живлення хаба.

Комплект Xiaomi є чудовою стартовою платформою для підключення інших сенсорів та пристроїв, у тому числі сторонніх виробників. Її елементи працюють як самостійно, так і в якості єдиного цілого. З їхньою допомогою можна створити досить функціональну систему контролю житлового простору, включаючи забезпечення безпеки. Дана система, враховуючи її вартість, підходить для ознайомлення із системою розумного будинку.

1.3 Обґрунтування вибору інструментів розробки

Для розробки апаратної частини проекту потрібно, щоб контролер підтримував підключення та управління усіх необхідних датчиків та з нього можна було легко передавати дані цих датчиків далі. Було прийнято рішення використовувати плату Arduino Uno. Серед основних причин можна виділити наступні:

- комплектація наборів плати Arduino усіма потрібними датчиками та додатковими елементами для підключення;
- гарна документація та наявність багатьох ресурсів з процедурою підключення датчиків;
- сумісність плати Arduino з ігровими двигунами, що має важливу роль для розробки симуляції.

Для розробки ігрової симуляції було прийнято рішення використовувати програмне середовище Unity. Це легкий у використанні, гарно документований та кросплатформовий ігровий двигун. Серед додаткових причин можна виділити:

- сумісність та наявність засобів підключення до плати Arduino та обмін інформацією з програмною частиною контролера;
- особистий досвід розробки на базі даного двигуна протягом 2-х років;
- розробка у даному середовищі здійснюється за допомогою мови програмування C#.

1.4 Постановка задачі для розробки

В результаті проведення огляду та порівняльного аналізу систем датчиків «смарт-будинку» було розроблено та сформовано таблицю (Таблиця 1.1), що включає в себе порівняння головних переваг та

недоліків систем датчиків, тобто їх порівняльну характеристику. Для порівняння аналогів були зібрані системи, що найкраще зарекомендували себе на ринку при багаторічному досвіді їх використання, а також зібрані ті системи, що найкраще відповідають параметрам:

- масштабування (тобто додавання обладнання розширення функціоналу);
- спосіб керування (через ПК, смартфон, панель керування);
- вартість системи.

Використовуючи ці дані, було проаналізовано та виділено основні недоліки кожної системи, а також виділено важливі переваги.

На основі зібраних даних можна сформувати основні задачі для розробки проекту:

- спроектувати та розробити систему датчиків (датчик вогню та диму, датчик тепла та вологості, датчик рівня води, датчик зачинених дверей) для моніторингу безпеки будинку;
- розробити ігрову симуляцію будинку, що буде приймати та оброблювати дані від датчиків системи безпеки;
- розробити користувацький інтерфейс для керування системою «смарт-будинку»;
- забезпечити інформування користувача про спрацювання датчиків за допомогою Push-повідомлень або інформування через e-mail.

Таблиця 1.1 – Порівняльна характеристика систем-аналогів

Назва системи	Переваги	Недоліки
Ajax	встановлення за QR-кодом та управління за допомогою смартфона через зручний додаток (iOS, Android); різноманітність способів інформування користувача (дзвінок, SMS, Push-повідомлення).	функціонування лише з роботою центрального контролера (Hub), тобто відсутність автономності датчиків; керування лише через телефон.
Fibaro	наповнення системи великою кількістю датчиків та пристроїв; зручний додаток для керування, що має голосове управління через сервіси Google.	висока вартість обладнання (від 600 $); налаштування і монтаж здійснюється лише працівниками.
Xiaomi	можливість підключення багатьох пристроїв, тобто масштабування; зручне керування за допомогою додатку для смартфона через Wi-Fi.	відсутність резервного живлення хаба.

1.5 Висновок до розділу

При роботі над даним розділом було проведено огляд

найпопулярніших систем датчиків «смарт-будинку». Було проаналізовано їх переваги та недоліки та складено таблицю з порівняльною характеристикою. Це допомогло виділити серед них важливі функції, що знадобляться при проектуванні та розробці проекту, а саме:

- система повинна мати датчики вогню та диму, температури та вологості, рівня води, зачинених дверей. Цей базовий набір є головним для забезпечення безпеки будинку;
- система повинна надавати зручний інтерфейс для управління та налаштування датчиків;
- система має забезпечити швидке інформування користувача за допомогою Push-повідомлень або e-mail інформування.

РОЗДІЛ 2 ОПИС І ОБГРУНТУВАННЯ ПРОЕКТНИХ РІШЕНЬ СИСТЕМИ СИМУЛЯЦІЇ СМАРТ-БУДИНКУ

2.1 Аналіз та виявлення вимог функціонування системи

Провівши огляд та порівняльний аналіз систем датчиків смарт-будинку, можна описати наступні пункти функціонування системи:

1. Система повинна мати модуль для реєстрації та авторизації користувача, для збереження даних і налаштувань датчиків розумного дому.
2. Система повинна бути підв'язана до сервісу, що забезпечить збереження базових даних, таких як електронна пошта та пароль, а також модель даних про налаштування, наприклад у вигляді JSON файлу.
3. Система повинна надати користувачу можливість відновлення паролю, у випадку якщо той забув його. Така процедура повинна включати в себе підтвердження операції відновлення за допомогою електронної пошти.
4. Система повинна мати зручний та інтуїтивно зрозумілий інтерфейс для налаштувань. Серед таких обов'язково повинно бути окреме налаштування кожного датчика, налаштування інформувань та особистого профілю.
5. Серед способів інформування повинні бути Push-повідомлення та повідомлення надіслані на електронну пошту.
6. Датчики для моніторингу повинні бути підключені до системи та інформувати користувача моментально у випадку відхилення показників від встановлених норм.
7. Серед датчиків системи повинні бути базові пристрої для контролю за рівнем води, температури та вологості та датчик вогню.

8. Для демонстрації роботи системи необхідно розробити симуляцію будинку, що відповідно до налаштувань та сигналу від датчиків, буде демонструвати надзвичайні ситуації, такі як пожежа, затоплення чи проникнення в будинок. У випадку кожної ситуації буде викликатися відповідна служба для усунення проблеми.
9. Розроблений проект буде працювати на базі ОС Windows 10 для забезпечення зручності демонстації та швидкого обміну даних з мікроконтролером з датчиками.

Відповідно до поставлених вимог можна виділити елементи пристрою, необхідні для функціонування системи:

1. Пристрій на базі мікроконтролеру, що забезпечить просте підключення датчиків та дозволить встановити зв’язок із ігровою симуляцією.
2. Датчик вогню.
3. Датчик температури та вологості.
4. Датчик рівня води.
5. Датчик для фіксування проникнення у будинок. Для демонстації роботи системи планується використати більш простий елемент, наприклад кнопку, підключену до мікроконтролеру.

Також можна скласти перелік основних сутностей, що будуть враховані під час моделювання та реалізації програмної частини проекту:

1. Користувач (пароль, електронна пошта, збережені налаштування).
2. Налаштування користувача (метод інформування, датчики).
3. Датчик (тип, налаштування).
4. Будинок (стан).
5. Ситуація (тип, анімація, служба).
6. Служба (тип, анімація).

2.1.1 Опис функціонування системи за допомогою діаграми прецедентів

В уніфікованій мові моделювання (UML) діаграма варіантів використання може узагальнити деталі користувачів системи (також відомих як акторів) та їхню взаємодію з системою. Щоб створити її, потрібно використовувати набір спеціалізованих символів і сполучників.

Акторами називають користувачів, які взаємодіють із системою або зовнішні системи, які взаємодіють з програмою. Актор є зовнішнім об'єктом, який виробляє або споживає дані.

Першим актором у даній ситуації є користувач, який взаємодіє з додатком та з системою датчиків, а другим актором є сервіс авторизації користувача. На створеній діаграмі (рис. 2.1) продемонстровано взаємодії визначених акторів із системою. Для побудови діаграми були використані відношення: узагальнення, включення та розширення.

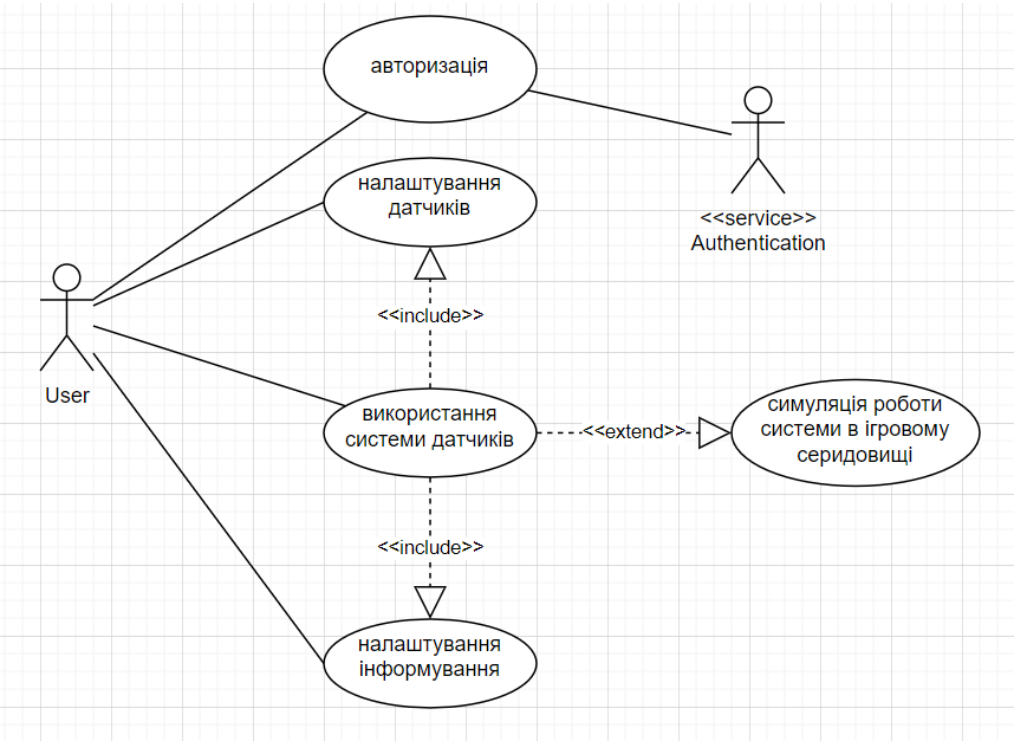


Рисунок 2.1 – Діаграма прецеденті в системи

2.2 Структурна схема системи

Система практичної частини роботи буде складатися з двох компонентів: пристрій та програмна частина для обробки та візуалізації даних з пристрою. Коли в системі спрацьовує певний датчик, його дані одразу передаються на контролер. Контролер обробляє інформацію за допомогою програмного коду, який повинен бути прописаний у “прошивці”. Після обробки дані надходять у додаток, тобто симуляцію “смарт-будинку”. Симуляція враховує налаштування користувача та візуалізує певну надзвичайну ситуацію, наприклад пожежу, затоплення будинку чи підвищений рівень вологості. Крім цього, відбувається інформування користувача відповідно до встановлених налаштувань (рис. 2.2).

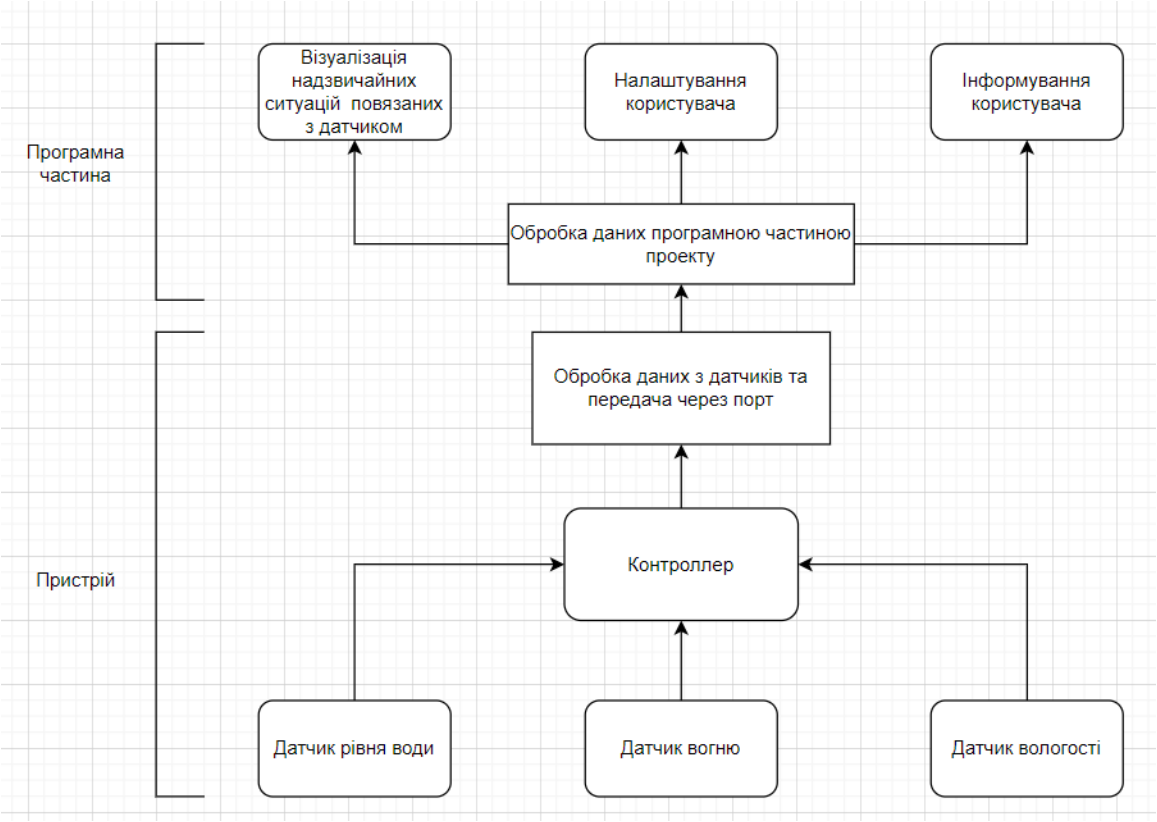


Рисунок 2.2 – Структурна схема системи симуляції розумного дому

2.3 Функціональні схеми компонентів пристрою

Практична частина даної кваліфікаційною роботи включає в себе створення пристрою, що повинен складатися з датчиків, підключених до мікроконтролеру. У якості управляючого елементу пристрою пропонується використати плату Arduino Uno R3.

2.3.1 Датчик рівня води

Датчик рівня води має 3 контакти:

– S (Signal) pin: це аналоговий вихід, який повинен бути підключений до одного з аналогових входів Arduino;
– + (VCC) pin: живлення для датчика. Рекомендується живлення датчика від 3,3 В до 5 В;
– - (GND) pin: це заземлення (рис. 2.3).

Вихідний сигнал(показання датчика) залежить від ступеня занурення датчика в рідину і параметрів, що впливають на коефіцієнт передачі напруги, наприклад, провідність рідини.

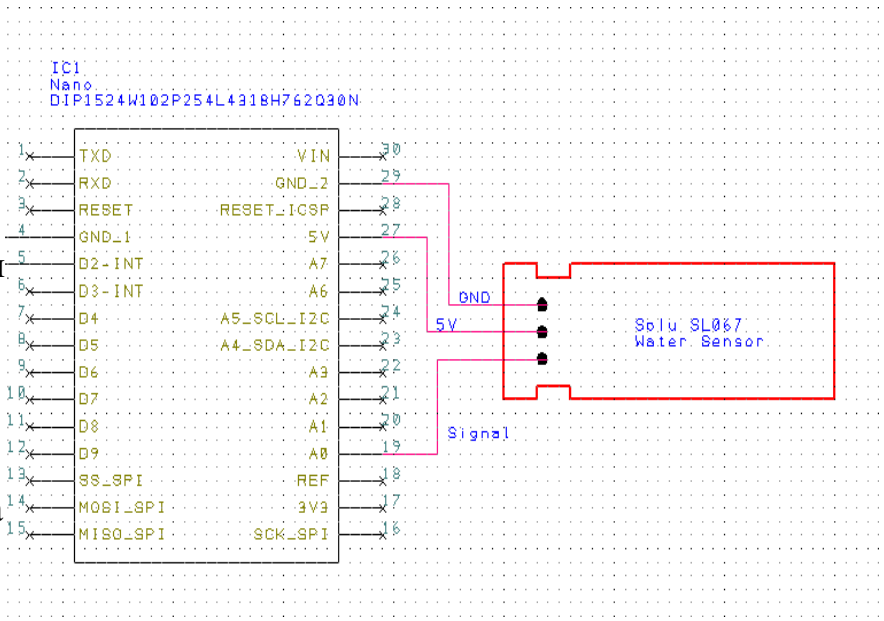


Рисунок 2.3 – Схема підключення датчику рівня води до плати Arduino

Сам датчик має серію з десяти відкритих мідних слідів. П'ять з них для живлення, а п'ять — вхідних даних. Ці сліди переплетені паралельно, так що між кожними двома силовими є один для сприйняття. Ці сліди не

з’єднуються, якщо вони не перекриваються водою під час занурення.

2.3.2 Датчик вогню

Датчик вогню зазвичай має 3 контакти:

– Dout pin: це цифровий вихід, який повинен бути підключений до одного з цифрових входів Arduino;
– + (VCC) pin: живлення для датчика. Рекомендується живлення датчика від 3,3 В до 5 В;
– - (GND) pin: це заземлення (рис. 2.4).

Коли вогонь горить, він випромінює невелику кількість інфрачервоного світла, це світло може прийматися фотодіодом (ІЧ-приймачем) на модулі датчика. Можна використати операційний підсилювач, щоб перевірити зміну напруги на ІЧ-приймачі. Таким чином, якщо буде виявлено пожежу, вихідний контакт (DO) дасть 0В, а у звичайному стані, вихідний контакт буде 5 В [8].

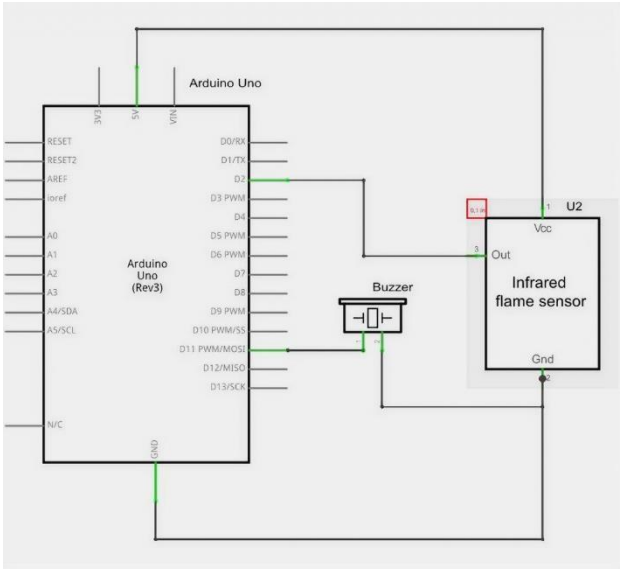


Рисунок 2.4 – Схема підключення датчику вогню до плати Arduino

2.3.3 Датчик температури та вологості

Серед датчиків температури та вологості для плати Arduino зазвичай використовують DHT11 і DHT22. Другий дає більше можливостей для вимірювання, але для дослідницьких цілей достатньо буде DHT11.

Датчик температури та вологості має 3 контакти:

– Dout pin: цифровий вихід з даними з датчику, який повинен

бути підключений до одного з цифрових входів Arduino;

– + (VCC) pin: живлення для датчика. Рекомендується живлення датчика від 3,3 В до 5 В;
– - (GND) pin: це заземлення (рис. 2.5) [9].

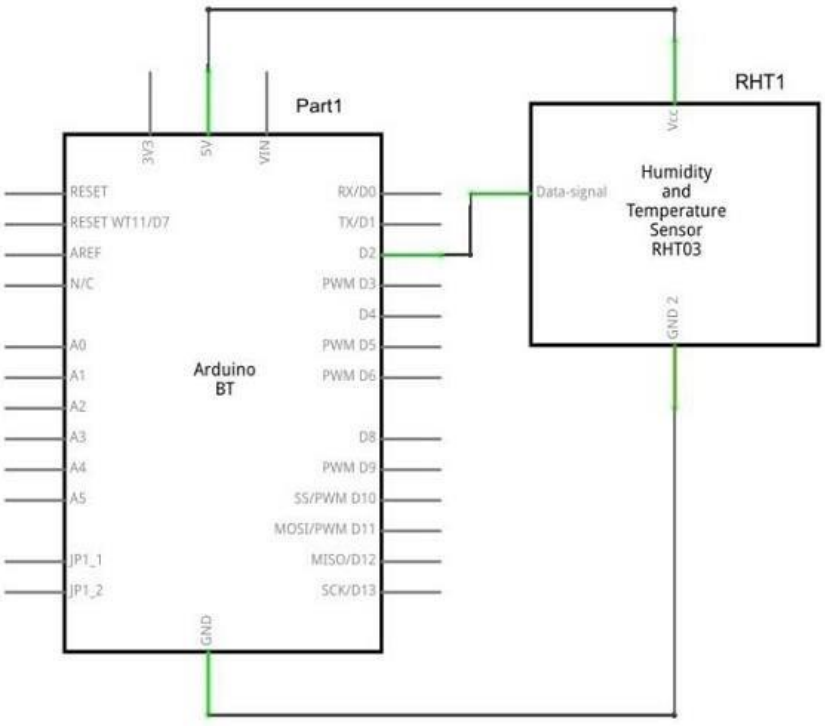


Рисунок 2.5 – Схема підключення датчику температури та вологості до плати Arduino

DHT11 — це цифровий датчик температури та вологості. Він використовує ємнісний датчик вологості та термістор для вимірювання навколишнього повітря та видає цифровий сигнал з даними. Він досить простий у використанні, але вимагає ретельного визначення часу для отримання даних. Недолік цього датчика полягає в тому, що він може видавати нові дані лише раз кожні 2 секунди, але цього буде досить для дослідницьких цілей.

2.4 Висновок до розділу

При роботі над другим розділом було проаналізовано та виявлено вимоги функціонування системи. Відповідно до цих вимог було

складено список елементів пристрою, необхідних для функціонування системи. Крім цього було представлено перелік основних сутностей, що будуть враховані під час моделювання та реалізації програмної частини проекту. Також представлено діаграму прецедентів, яка включає в себе, як програмну, так і апаратну частину, а також структурну схему систему проекту, що пізніше знадобиться для вибору технологій та інструментів розробки.

Отримавши список елементів системи та безпосередньо пристрою, було проаналізовано та складено функціональні схеми підключення конкретних датчиків, які дозволять збирати та аналізувати інформацію і передавати її у ігрову симуляцію «смарт-будинку» для візуалізації процесу роботи системи.

РОЗДІЛ 3 ПРАКТИЧНА РЕАЛІЗАЦІЯ РОБОТИ

3.1 Розробка та опис алгоритмів функціонування системи

Для зручності під час тестування та перевірки системи, проектна частина даної роботи повинна розроблятися послідовно, спершу програмна частина потім апаратна. Перед тим, як приступити до

написання коду, пропонується обрати технології, які допоможуть покращити та прискорити розробку.

3.1.1 Вибір технології для авторизації користувача

Програмна частина додатку включає в себе процес налаштування датчиків користувачем. Для збереження цих даних необхідно, щоб додаток був підключений до бази даних (БД). Така БД повинна включати в себе можливість запису даних авторизації та певної структури для збереження налаштувань датчиків. Крім цього, необхідно встановити зв'язок БД з середовищем Unity.

Так як процес створення БД та її інтеграція є доволі громіздким процесом, було прийнято рішення скористатися платформою PlayFab. PlayFab — це повноцінна серверна платформа для онлайн ігор із керованими ігровими сервісами та аналітикою в реальному часі.

Традиційно, при створенні подібних додатків, довелось би розробляти серверну частину самостійно. Вам доведеться створити власні бази даних і, можливо, налаштувати сервери та створити весь бекенд-код, який може взаємодіяти з грою, тобто з клієнтом гри. Але в цьому випадку PlayFab вже має все налаштоване. Тут вже є автентифікація гравця та пов’язані облікові записи. Можна створювати акаунти гравців з іменем користувача, паролем, електронною поштою. Також тут присутні додаткові налаштування, за допомогою яких можна увійти у свій акаунт, використовуючи Facebook, Google та інші сервіси.

Отже з корисних переваг PlayFab для даного проекту можна виділити:

- легкий процес інтеграції в середовище Unity;
- вже реалізована вся бекенд-частина необхідна для авторизації користувача;
- безкоштовне використання у невеликих масштабах;
- добре документований код.

3.1.2 Вибір шаблону проектування для створення та управління UI

Під час розробки завжди є гарною практикою написання чистого і структурованого коду для проектів. Організація коду відповідно до шаблону проектування допомагає підтримувати програмне забезпечення. Знаючи всі важливі логічні частини програми, легше додавати та видаляти функції програми. Крім того, шаблони проектування також гарантують, що всі коди будуть охоплені модульним тестуванням без втручання інших класів.

Model-View-ViewModel (MVVM) — це шаблон проектування програмного забезпечення, структурований для розділення логіки програми та елементів керування інтерфейсом користувача. Як і багато інших шаблонів проектування, MVVM допомагає організувати код і розбити програми на модулі, щоб зробити розробку, оновлення та повторне використання коду простішими та швидшими [11].

Поділ коду в MVVM виконується на View, ViewModel і Model (рис. 3.1):

– View — це набір видимих елементів, який отримує введення користувача. Це включає в себе інтерфейси користувача (UI), анімацію та текст. Вміст View не взаємодіє безпосередньо з тим, щоб змінити те, що представлено;

– ViewModel розташована між View і Model. Тут розміщуються елементи керування для взаємодії з View, тоді як прив’язування використовується для з’єднання елементів інтерфейсу користувача у View з елементами керування ViewModel;

– Model відповідає за абстракцію джерел даних. Модель і ViewModel працюють разом, щоб отримати та зберегти дані.

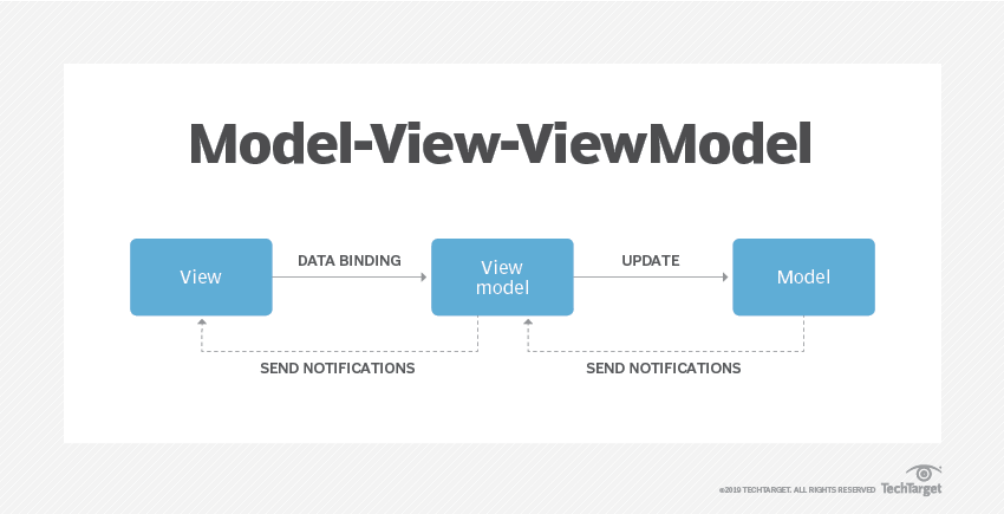


Рисунок 3.1 – Структура шаблону проектування MVVM

3.1.3 Вибір шаблону проектування для обробки даних з датчиків

Перед тим як обрати паттерн для обробки даних з різних датчиків, необхідно визначити задачі, які повинні покриватися даним шаблоном. Серед них можуть бути:

- ініціалізація, що в даному випадку виконується отриманням даних з контролера, здійснюється в одному місці;
- при такій ініціалізації потрібно зафіксувати певні дані, що потім будуть передаватися далі;
- проініціалізовані дані можуть знадобитися в багатьох класах, тож потрібно, щоб такі класимали можливість підписатися на отримання інформації.

Проаналізувавши задачі, поставлені вище, можна зробити висновок, що для подібних цілей добре підходить шаблон Publisher – Subscriber.

У шаблоні обміну повідомленнями Publisher – Subscriber видавці не надсилають повідомлення безпосередньо всім підписникам; натомість повідомлення надсилаються через «брокерів». Видавці не знають, хто є підписниками або на які події (якщо є) вони підписані. Це означає, що операції видавця та підписника можуть працювати незалежно один від одного. Це ще називають слабким з’єднанням і воно усуває залежності об’єктів, які інакше були б у традиційних шаблонах обміну повідомленнями.

Publisher – Subscriber відрізняється від стандартних моделей запитів/відповідей, у яких видавці перевіряють, чи доступні нові дані. Це робить метод Publisher – Subscriber центральним для ефективної потокової передачі даних у режимі реального часу. Шаблон Publisher – Subscriber дозволяє створювати надзвичайно динамічні мережі в масштабі, не перевантажуючи видавничі компоненти та не викликаючи зайвих витрат.

Типове використання даного паттерну включає обмін повідомленнями про події, миттєві повідомлення та потокове передавання даних. Publisher – Subscriber також використовується для балансування робочого навантаження та з асинхронними робочими процесами (рис. 3.2).

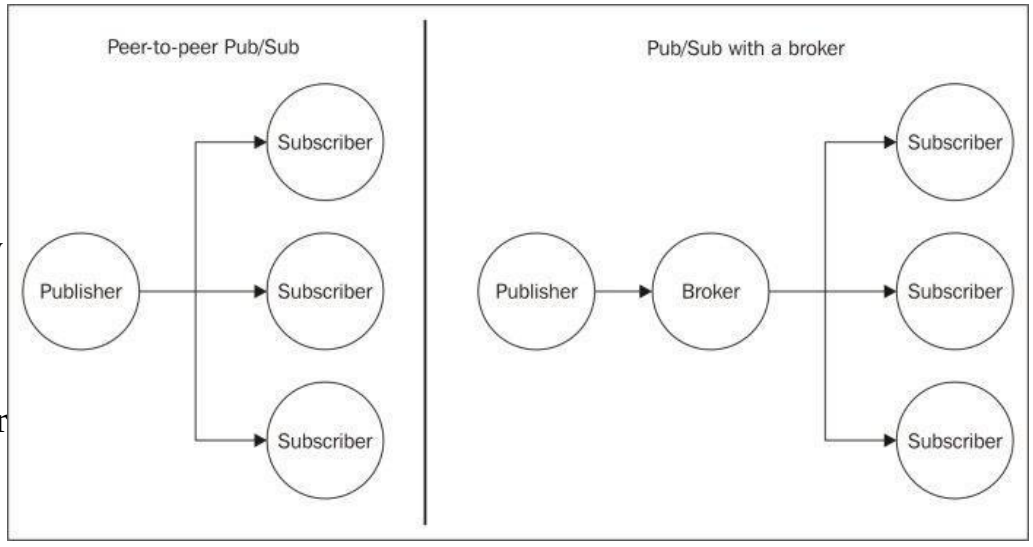


Рисунок 3.2 – Модель шаблону програмування Publisher – Subscriber

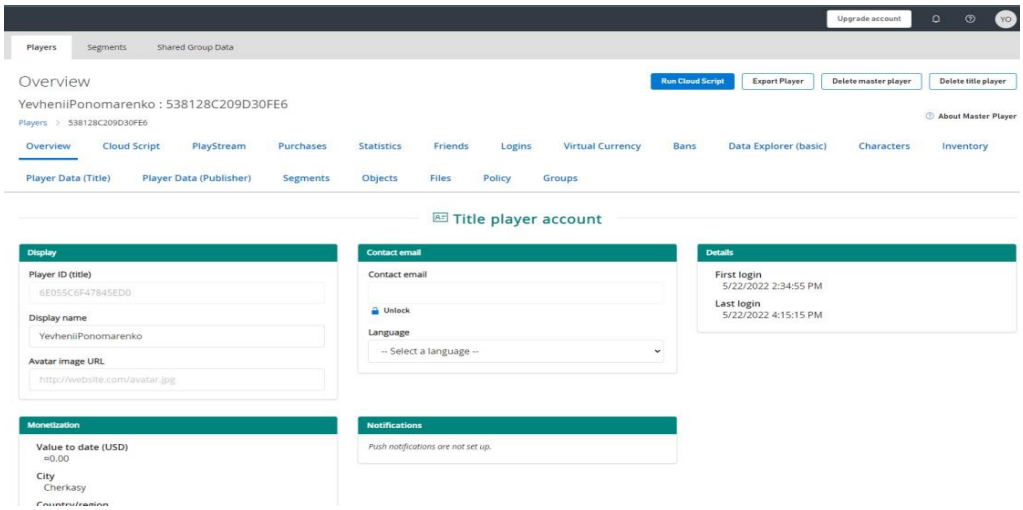
3.1.4 Реалізація підключення серверної частини

Перше, що знадобиться для реалізації, це обліковий запис PlayFab. Потрібно перейти на сайт даного сервісу і зареєструватися там. Після створення облікового запису необхідно створити проект і перейти на його сторінку. На даному етапі це все, що може знадобитися на стороні PlayFab. PlayFab включає в себе ще дуже багато сервісів які можна налаштовувати — це гравці, економіка, таблиця лідерів та керування контентом. Ці послуги охоплюють облікові записи, торгівлю/валюти, таблиці лідерів та такі речі, як інвентар (гравець,

NPC, магазин тощо).

Враховуючи цілі даної кваліфікаційної роботи, на даному етапі необхідно лише дозволити користувачам входити або створювати нові облікові записи. Також для того, щоб створити проект Unity та інтегрувати туди цю платформу, ще знадобляться дві речі: пакет PlayFab Unity SDK і пакет розширень PlayFab Editor.

Також для перевірки правильності налаштування PlayFab, можна перейти у розділ гравців та створити там нового гравця. Одразу після створення його дані будуть занесені до БД створеного проекту. Дані цього гравця можна буде отримати у будь-який момент виконавши



запит по його автентифікаційним даним (рис. 3.3).

Рисунок 3.3 – Детальна інформація про створеного гравця PlayFab

Після виконання необхідних налаштувань проекту Unity, можна приступити до написання коду для запитів до сервісу PlayFab. Для таких запитів потрібно створити окремий клас, що служитиме у ролі API. Методи авторизації повинні приймати на вхід дані про email, пароль та нікнейм користувача і створювати спеціальний request у вигляді моделі. У даному випадку для реєстрації користувачів існує

RegisterPlayFabUserRequest модель, а для входу під вже створеним обліковим записом існує LoginWithEmailAddressRequest.

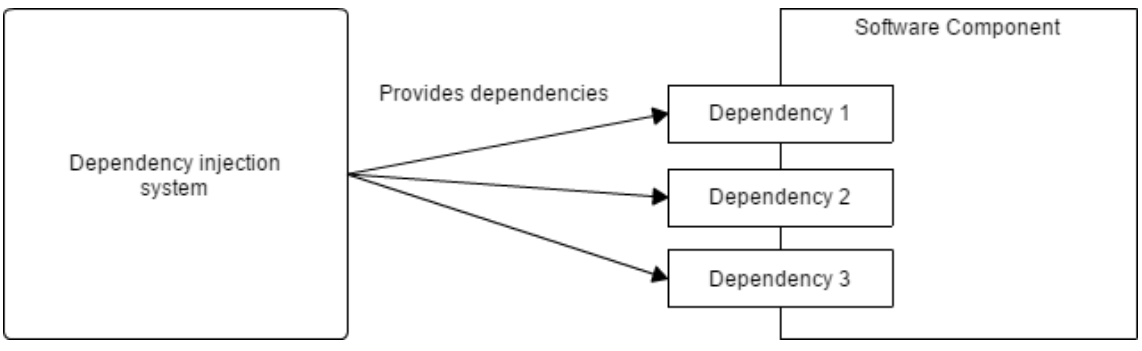
Коли модель запиту створена, можна передати його за допомогою PlayFabClientAPI. Цей виклик API очікує (і вимагає), щоб ми отримали зворотний виклик для успіху і один для помилки. Ми можемо створити два окремі методи для цього, але замість цього прийнято використовувати вбудовані лямбда-вирази для обробки відповіді успіху/помилки прямо у методі для запиту. У разі, якщо спрацював зворотний виклик і запит успішно обробився, ми можемо вже далі працювати з отриманими даними. Лістинг коду для демонстрації роботи запитів PlayFab показаний у додатку А.

3.1.5 Реалізація класів програмної частини

Реалізація програмної частини у середовищі Unity починається з імплементації базового класу DependentMonoBehaviour, що реалізує Dependency Injection паттерн. Це є дуже важливою частиною проекту. На етапі створення базових модулів програми виникає необхідність зв’язати ці модулі не порушуючи принципи SOLID програмування.

Наприклад, є програмний компонент. Він оголошує свої потреби в системі впровадження залежностей. Це ті дані, від яких він залежить. Це його посилання на інші компоненти. Також можна сказати, що таким чином він з’єднується із зовнішнім світом. Компонент не повинен піклуватися про те, звідки беруться його залежності, лише те, що вони якимось чином для нього передбачені. Це суть даного шаблону програмування (рис. 3.4).

Рисунок 3.4 –



Структура шаблону Dependencies Injection

Від описаного вище базового класу створюються дочірні класи UIController та UIPresentationModel<T>. Вони є реалізацією шаблону проектування MVVM. В свою чергу від цих класів створюються необхідні нам класи для управління датчиками вогню, води, дверей та температури і волості. Крім цього, від даних класів наслідуються і модулі для процесу авторизації користувачів. Такими класами є RegisterForm та LoginForm У свою чергу, кожен з контроллерів для датчиків реалізує інтерфейс ISensorSignalReceiver, який містить у собі метод з типом датчику. Цей метод викликається у SensorsSignalReceiver. Такий процес реалізує шаблон програмування Publisher – Subscriber.

Також для роботи процесу авторизації існує клас LoginAPI, який містить у собі методи для обробки запитів з сервісом PlayFab. Цей клас застосовується з LoginForm та RegisterForm, які отримують дані користувача з полів, що обробляються у View частині. Повна діаграма вище описаних класів представлена на рис 3.5.

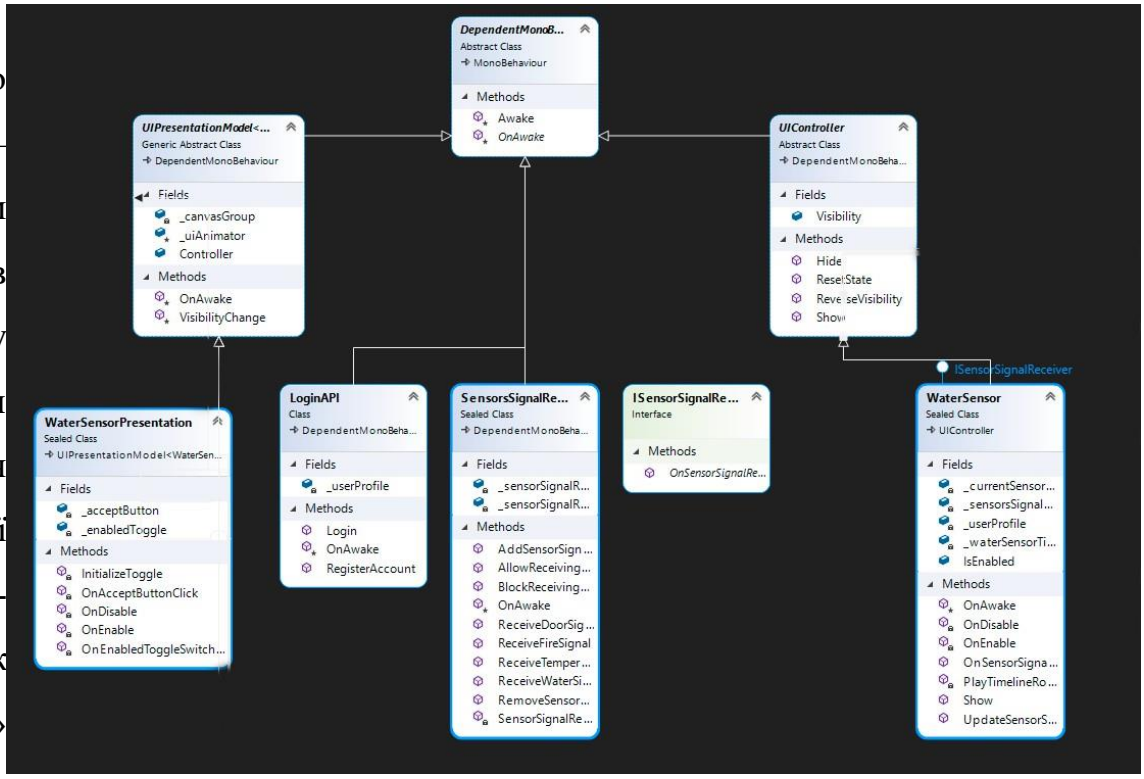


Рисунок 3.5 – Діаграма класів додатку для симуляції «смарт-будинку»

3.1.6 Розробка апаратної частини проекту

Розробка та складання апаратної частини здійснювалася на базі плати Arduino. Для того, щоб зібрати пристрій підготуємо мати наступні компоненти:

- макетна плата;
- датчик рівня води (з вбудованим резистором);
- датчик температури та вологості (з вбудованим резистором);
- датчик вогню;
- кнопка;
- резистори для кнопки та датчику вогню;
- перемички;
- плата Arduino.

Розпочнемо з підключення макетної плати. Необхідно під'єднати вихід 3,3V або 5V, залежно від вимог перемичкою до входу «плюс». Аналогічно потрібно з'єднати виходи для заземлення, тобто GND. Таким чином, ми забезпечимо живлення усієї плати.

Наступним кроком підключаємо датчики води та температури та вологості. Їх підключення є подібним. Вони обидва мають три виходи: заземлення, живлення та сигнал. З'єднуємо необхідні контакти з лінією

«плюс» та «мінус» на макетній платі, а сигнал виводимо на аналогові входи на платі Arduino. Оскільки обидва датчика вже мають вбудовані резистори, додаткове підключення такого не знадобиться.

Для кнопки і датчику вогню знадобляться два резистори. Встановлюємо датчик вогню на плату. До плюсу (тобто до довшої ніжки) підключаємо одну частину резистору та перемичку для аналогового сигналу. На іншому боці резистору підключаємо живлення.

Від коротшої ніжки датчику виводимо заземлення. У випадку кнопки до нижньої лівої ніжки підводимо резистор, а до правої заземлення. Інший кінець резистору підключаємо по «плюсу» на макетній платі. Нижче від лівої ніжки кнопки також виводимо перемичку для аналогового сигналу.

Загалом отримуємо макетну плату, що має чотири зайняті роз'єми для аналогового сигналу та ще дві перемички для живлення та заземлення. Фото отриманого зібраного пристрою можна побачити на рис. 3.6.

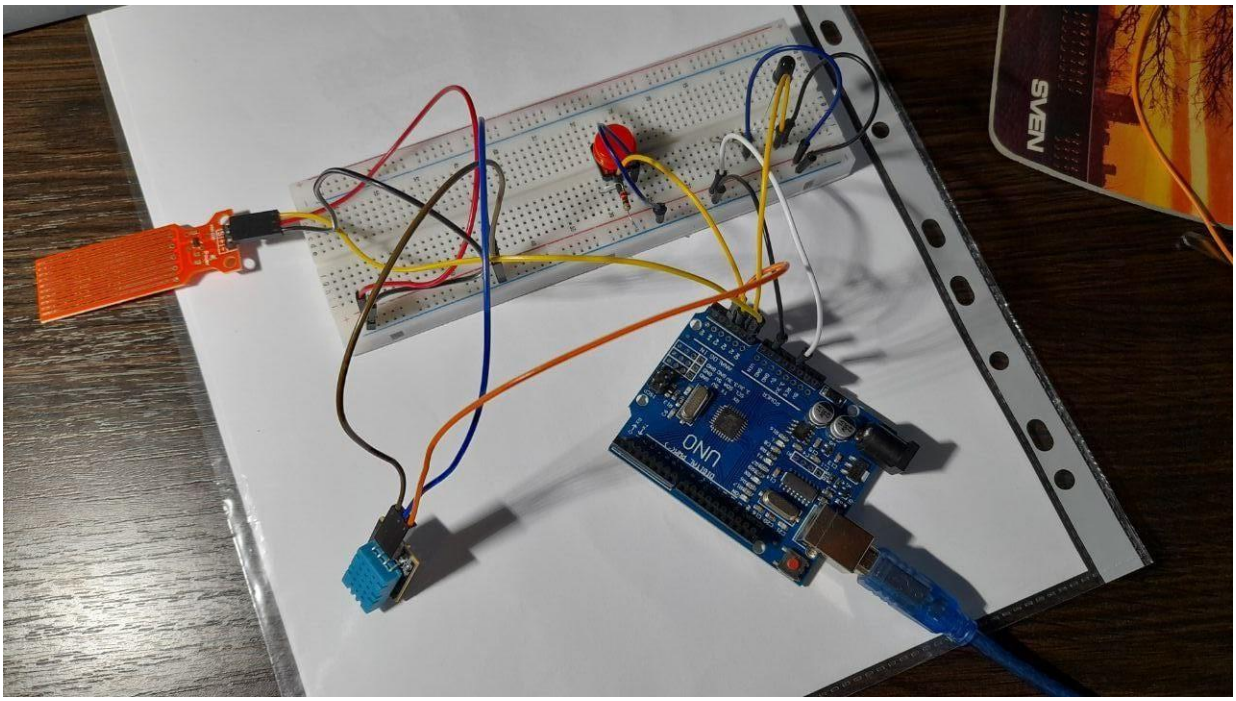


Рисунок 3.6 – Зібраний пристрій на базі плати Arduino

3.2 Механізми захисту програмного забезпечення

У даному підрозділі пропонується оцінити ступінь захисту ігор на основі Unity та додатково розглянути розроблені методи захисту.

Unity — це кросплатформний ігровий двигун, який дозволяє розробникам створювати ігри для Android та iOS, використовуючи

єдину кодову базу. Він використовує C# як мову сценаріїв, яка не підтримується ні на Android, ні на iOS. Код C# перетворюється на код C++ компілятором Unity Il2Cpp, який потім використовується для створення проектів Xcode та Android Studio.

На додаток до коду C++, Il2cpp генерує метадані, які зберігаються як зовнішній файл під назвою «global-metadata.dat». Файл зберігається всередині згенерованої програми і завантажується середовищем виконання Il2Cpp під час ініціалізації програми. Він містить інформацію про всі об’єкти середовища виконання, які використовуються в програмі, включаючи імена та типи всіх класів, методів і властивостей і зв’язки між ними, а також усі рядкові літерали, такі як ключі API, що використовуються в коді C#. Метадані посилаються зі згенерованого коду C++ і служать для зв’язування методів з їх реалізацією [13].

Усі метадані, які зберігаються у файлі global-metadata.dat, легко отримати і надають зловмисникам інформацію, яку вони можуть використовувати для зламування ігор.

Оскільки в програмній частині проекту використовується збереження даних авторизації користувача для підстановки, було прийнято рішення використати алгоритм шифрування AES.

AES є шифром, здатним обробляти 128-бітові блоки, використовуючи ключі розміром 128, 192 і 256 біт. Серед його переваг можна виділити безпеку, просту реалізацію та легку обробку, що не потребує багато ресурсів.

У межах даного проекту цей алгоритм застосовується для шифрування паролю, програмну реалізацію якого можна переглянути у додатку Б.

3.3 Висновок до розділу

При роботі над третім розділом було розглянуто технології та алгоритми, які використовувалися під час розробки проекту. Було обґрунтовано вибір сервісу для авторизації користувачів. На основі обраного сервісу було реалізовано підключення до серверної частини.

Крім цього, було обрано шаблони програмування для зручності та правильності написання коду. Серед них можна виділити шаблон для управління UI частиною, шаблон для забезпечення обміну даних з датчиків у проекті між отримувачем даних та підписниками. Вагому роль також в проекті відіграє шаблон Dependencies Injection, що лужить для зменшення зв’язності коду. На основі обраних технологій було розроблено програмний код, основні модулі якого було винесено на діаграму класів.

Використовуючи інформацію та схеми з другого розділу, було складено апаратну частину проекту на базі плати Arduino, з під’єднаними датчиками води, вогню, кнопки і датчику температури та вологості.

Ще однією важливою частиною даного розділу став реалізований алгоритм шифрування даних авторизації користувача, що дає змогу підвищити безпеку розробленого продукту.

РОЗДІЛ 4 ВПРОВАДЖЕННЯ СИСТЕМИ В ПРОМИСЛОВУ ЕКСПЛУАТАЦІЮ

4.1 Тестування розробленого програмного продукту

При запуску додатку користувач потрапляє одразу на сцену, в якій присутня модель «смарт-будинку». Крім цього, на даному етапі можна помітити елементи UI у правому та лівому верхніх кутках. Натиснувши на них можна відкрити меню телефону для управління або меню паузи відповідно. Крім цього у лівому нижньому кутку розташовані Debug елементи для прямого запуску сценаріїв, або для емуляції сигналу з датчиків (рис. 4.1).

Рисунок 4.1 – Сцена з моделлю «смарт-будинку»

Натиснувши на кнопку «меню» користувач бачить, як на екрані з’являється телефон, що симулює додаток для управління «смарт-будинку». Даний додаток спершу пропонує пройти процес авторизації. Потрібно створити новий акаунт або зайти під вже існуючим. Крім цього, процес авторизації дозволяє зберігати дані, щоб не вводити їх кожен раз при вході (рис. 4.2). При наступному заході збережені дані автоматично підставляться у поля.

Рисунок 4.2 – Екран авторизації користувача у додатку

Після проходження процесу авторизації з’являється меню для налаштування датчиків. У даному меню зверху відображається ім’я користувача. Крім цього у даному меню присутній кожен з датчиків. Тут можна увімкнути чи вимкнути їх. Після зміни налаштувань нові дані відправляються на сервер та зберігаються. Тож при наступному

заході користувач помітить такі налаштування, які він зберіг до цього. Можна одразу спробувати переключити стан деяких датчиків, щоб побачити вікно збереження (рис. 4.3).

Рисунок 4.3 – Екран з меню для налаштування датчиків

Виконавши налаштування датчиків, можна приступити до тестування сценаріїв. Вони розроблені таким чином, що поки один з них не буде завершений, інший не почнеться. Для початку можна запустити сценарій пожежі. Він починається з ефекту пожежі. Після спалаху будинку приїжджає пожежна служба і починає гасити полум’я. Наступною спрацьовує анімація ефекту захисту будинку і пожежна служба від’їжджає від будинку (рис. 4.4).

Рисунок 4.4 – Сценарій пожежі та реагування пожежної служби Таким же чином працюють і інші сценарії. Відмінність між ними полягає у ефекті для певної надзвичайної ситуації та відповідної служби, що приїжджає для усунення небезпеки. Для того, щоб протестувати решту сценаріїв можна натиснути на кнопки «Water», «Door» та «Humidity».

Крім вище описаних етапів тестування, можна відвідати сайт сервісу PlayFab та перевірити успішність авторизації користувача. Якщо перейти на вкладку «Players» і відкрити зареєстрований раніше акаунт то можна переглянути поля зі своїми збереженими даними. На вкладці «PlayerData (Title)» можна побачити поле UserData, що має збережені налаштування з датчиків у форматі JSON.

4.2 Висновок до розділу

При роботі над четвертим розділом було проведено тестування розробленого продукту. Було розглянуто кожний етап додатку, починаючи від екрану авторизації користувача до моменту запуску і перегляду різних сценаріїв. Перевірка бази даних сервісу реєстрації показала, що користувач реєструється успішно, а усі налаштування датчиків зберігаються у полі UserData на кладці налаштування гравця.

Отже, тестування проекту продемонструвало, що розроблений

додаток працює правильно, що підтверджує й те, що робочий процес не показав явних помилок. Наступним кроком потрібно проаналізувати усі написанні розділи та оформити загальний висновок до виконаної роботи.

ВИСНОВКИ

Під час виконання даної кваліфікаційної роботи було розроблено продукт для моделювання системи датчиків на базі мікроконтролера під'єднаного до ігрової симуляції «смарт-будинку». Даний продукт дає змогу продемонструвати ефективність системи датчиків, що може забезпечити безпеку будинку.

Для досягнення поставлених цілей було виконано такі етапи:

- проведено огляд існуючих популярних систем «смарт-

будинків»;

- поставлено задачі для розробки кваліфікаційної робот;
- виконано аналіз для виявлення вимог функціонування системи;
- побудовано структурну та функціональні схеми пристрою;
- проаналізовано та вибрано технології та шаблони проектування для зручності створення продуту;
- реалізовано кожен з етапів написання програмного забезпечення, визначених на етапі проектування;
- проведено тестування розробленого проекту, що допомогло визначити ступінь виконання роботи.

Апаратна та програмна частина добре функціонують та готові до використання. Інформація з датчиків, під'єднаних до мікроконтролера успішно надходить до ігрової симуляції. Сценарії реагування на порушення безпеки будинку спрацьовують точно та демонструють роботу системи розумного дому. Крім цього, тестування сервісу авторизації користувачів показало, що система зберігає встановленні налаштування датчиків для кожного користувача.

Отже, після аналізу виконаної роботи можна описати можливі покращення на майбутнє:

- забезпечення безпровідного зв'язку мікроконтролеру з датчиками та системою за допомогою Wi-Fi або Bluetooth модулю;
- перенесення додатку на мобільну платформу;
- покращення та оновлення архітектури модулю авторизації користувачів;
- розширення можливостей управління датчиками та

поліпшення інтерактивної частини користувацького інтерфейсу.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Пономаренко Є.О., Розломій І.О. ЗАСТОСУВАННЯ ПЛАТФОРМИ ARDUINO ДЛЯ СИМУЛЯЦІЇ СИСТЕМИ РОЗУМНОГО ДОМУ. Free and
   Open Source Software: матеріали міжнар. наук.-практ. конф., (Харків, 16-18 листопада 2021 р.). Харків, 2021, С. 50-51.
2. Пономаренко Є.О., Розломій І.О. МОДЕЛЮВАННЯ СИСТЕМИ ДАТЧИКІВ НА ПЛАТФОРМІ ARDUINO ДЛЯ

ЗАБЕЗПЕЧЕННЯ
ПОЖЕЖНОЇ БЕЗПЕКИ. Наука про цивільний захист як шлях становлення молодих вчених: матеріали всеукр. наук.-практ. конф., (Черкаси, 13 травня 2022 р.). Черкаси, 2022. С. 141 – 142.

3. Визначення поняття «смарт-будинок» [Електронний ресурс]: https://www.techtarget.com/iotagenda/definition/smart-home-or-building (дата звернення 27.12.2021).
4. Система «Ajax» [Електронний ресурс]: https://ajax.systems/ua/about/ (дата звернення 28.12.2021).
5. Система «Fibaro» [Електронний ресурс]: https://www.fibaro.com/en/ (дата звернення 28.12.2021).
6. Система «Xiaomi» [Електронний ресурс]: https://xiaomi-smarthome.ru/ (дата звернення 29.12.2021).
7. Датчик рівня води для Arduino [Електронний ресурс]: https://arduinogetstarted.com/tutorials/arduino-water-sensor (дата звернення 10.03.2022).
8. Датчик вогню для Arduino [Електронний ресурс]: https://circuitdigest.com/microcontroller-projects/arduino-flame-sensor- interfacing (дата звернення 15.03.2022).
9. Датчик температури та вологості для Arduino [Електронний ресурс]: https://www.tutorialspoint.com/arduino/arduino_humidity_sensor.htm?key=Ar duino+-+Humidity+Sensor (дата звернення 20.03.2022).
10. Сервіс PlayFab [Електронний ресурс]: https://docs.microsoft.com/en- us/gaming/playfab/what-is-playfab (дата звернення 10.04.2022).
11. Шаблон MVVM [Електронний ресурс]: https://www.geeksforgeeks.org/mvvm-model-view-viewmodel-architecture- pattern-in-android/ (дата звернення 20.04.2022).
12. Шаблон Publisher – Subscriber [Електронний

ресурс]: https://docs.microsoft.com/en-us/azure/architecture/patterns/publisher-subscriber (дата звернення 15.05.2022).

13. Компілятор IL2CPP [Електронний ресурс]: https://docs.unity3d.com/Manual/IL2CPP.html (дата звернення 17.05.2022).