Міністерство освіти і науки України
Київський національний університет імені Тараса Шевченка

Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ

завідувач кафедри
мережевих та інтернет технологій

_____ Ю.В. Кравченко

«_____»_____2021 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»

на тему:

Розробка інформаційної системи “Голосове управління розумним будинком”

Виконав: студент групи МІТ -41

Ковальчук Данило Сергійович _____
(прізвище ім’я по-батькові) (підпис)

Керівник: доцент кафедри мережевих та інтернет технологій

к.т.н., доцент Дахно Н. Б. _____
(посада, прізвище ім’я по-батькові) (підпис)

Київ 2021

Міністерство освіти і науки України
«Київський Національний університет імені Тараса Шевченка»
Факультет інформаційних технологій
Кафедра мережевих та інтернет технологій

ЗАТВЕРДЖУЮ
завідувач кафедри
мережевих та інтернет технологій
_____ Ю.В. Кравченко
«_____»_____2021 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Ковальчуку Данилі Сергійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи:
Розробка інформаційної системи "Голосове управління розумним будинком"

затверджена на засіданні кафедри МІТ, протокол «11» грудня 2020 р. протокол № 6

2. Термін здачі закінченої роботи «30» травня 2021 р.

3. Вихідні дані до проекту (роботи)

Мова програмування – C, Java, TypeScript

Технології: Angular, Proteus

4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)

Вступ

1. Аналіз і порівняння існуючих систем управління «Розумним будинком», засобів та підходів розпізнавання голосу
   1.1 Огляд сучасних систем розумного будинку.
   1.2 Класифікація систем розпізнавання голосу.
   1.3 Постановка задачі.
   1.4 Аналіз існуючих підходів розпізнавання голосових команд.
2. Аналіз засобів моделювання розумного будинку та можливостей розпізнавання голосових команд
   2.1 Аналіз існуючих протоколів взаємодії між компонентами розумного будинку.
   2.2 Аналіз існуючої інфраструктури хмарних засобів.
3. Реалізація інформаційної системи "Голосове управління розумним будинком"
   3.1. Аналіз вхідної інформації та розробка моделі запропонованої системи.
   3.2. Розробка структури мережі розумного будинку.
   3.3. Вибір апаратних засобів реалізації, стеку технологій, протоколів взаємодії, хмарних засобів, програмних засобів, середовища програмування, Frameworks
   3.4. Реалізація алгоритмів взаємодії між вузлами та хмарою та управління на основі обраного стеку протоколів.
   3.5. Створення інтерфейсу користувача
   3.6. Виконання тестування інформаційної системи

Висновки

4. Перелік графічного матеріалу 8-10 слайдів

Дата видачі завдання

Керівник роботи доц. Дахно Н.Б.
(підпис) (посада, прізвище, ім'я, по батькові)

Завдання прийняв до виконання Ковальчук Д. С.

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	16.03.2021	
2	Розділ 1	26.03.2021	
3	Розділ 2	05.04.2021	
4	Розділ 3	19.04.2021	
5	Доповідь та слайди	25.05.2021	
6	Пояснювальна записка	30.05.2021	

Здобувач вищої освіти _____ Д.С. Ковальчук
(підпис)

Керівник _____ Н.Б. Дахно
(підпис)

РЕФЕРАТ

Пояснювальна записка: 49 с., 21 рис., 3 додатки, 18 джерел.

Об’єкт дослідження: процес управління розумним будинком за допомогою голосу.

Предмет дослідження: системи голосового управління розумним будинком.

Мета роботи: створення прототипу розумного будинку з голосовим керуванням.

Методи дослідження: сучасні засоби моделювання, методи порівняння.

В роботі проведено аналіз: сучасних систем управління розумним будинком, основних протоколів зв’язку між компонентами розумного будинку, алгоритмів розпізнавання мови та сервісів що реалізують дані алгоритми.

Запропоновано: використання програми для емуляції пристроїв розумного будинку Proteus, мов програмування Java для написання backend веб-додатка та typescript у складі платформи Angular для розробки frontend. Використання JS-бібліотеки annyang для розпізнавання голосу, що користується SpeechRecognition API, який є у більшості браузерів.

Побудовано: схему розумного будинку з компонентами, що відображають системи освітлення та кондиціонування кімнат.

Розроблено: програмний додаток для голосового керування пристроями розумного будинку, код для мікроконтролера що дозволяє йому спілкуватися з програмним додатком та керувати пристроями розумного будинку за допомогою команд, які надходять від саме додатку.

Практичне значення роботи полягає у тому, що алгоритми функціонування системи управління й контролю системами можуть працювати на будь-якому сервері та можуть представляти керування будинком навіть якщо людина, яка керує ним знаходиться у іншому кінці світу. За рахунок того що розпізнавання голосу відбувається у хмарі, значно знижуються вимоги до ядра системи, яке не потребує великих потужностей для проведення розпізнавання мови.

Результати здійснених у дипломному проекті досліджень можуть бути використані для побудови з нуля системи розумного будинку з використанням

будь-якого пристрою, що може виступати в якості сервера для веб-додатку, мікроконтролера з bluetooth модулем та саме пристроїв, керування якими і буде здійснюватись.

Наукова новизна дослідження полягає у поєднанні новітніх технологій розпізнавання голосу із системою керування інформаційною системою “Голосове управління розумним будинком”.

Галузь використання – сучасні системи телекомунікацій України.

Напрямки подальших досліджень: вдосконалення додатку з точки зору зручності для користувача, впровадження підтримки додатком інших протоколів для передачі даних, що може надати можливість керувати пристроями розумного будинку від різних виробників.

Ключові слова: «РОЗУМНИЙ БУДИНОК», JAVA, ANGULAR, СИСТЕМА РОЗПІЗНАВАННЯ ГОЛОСУ, ГОЛОСОВЕ УПРАВЛІННЯ, IoT, PROTEUS, ARDUINO, ІНТЕРНЕТ РЕЧЕЙ

ВСТУП

Прогрес зараз як ніколи не стоїть на місці – розвиваються всі галузі і, мабуть, найшвидшими темпами прогресує ІТ. Дана сфера охоплює майже усе, чим займаються люди, та не на останньому місці йде Internet of Things або як прийнято називати – IoT. Інтернетом намагаються спростити все - починаючи від простого віддаленого контролю світла за допомогою телефону або іншого пристрою та закінчуючи комплексними системами для відслідковування, наприклад, грузових перевезень – тобто моніторинг всіх необхідних показників автомобіля і причепу та відповідна реакція на зміни цих показників або ж просто передача їх до контролюючого відділу. Або ж перевірка заповненності холодильника та автоматичне замовлення нових продуктів. Це вже не наукова фантастика – це реальність.

Дуже популярним в розвинутих країнах є таке застосування IoT як розумний будинок. Що ж таке розумний будинок? З першого погляду це звичайне житло, з однією тільки відмінністю – всі системи під’єднані до Інтернету, або ж якоїсь локальної системи керування. Команди можна віддавати за допомогою телефону, комп’ютеру або ж іншого пристрою, для цього використовують браузер або спеціальний додаток. Останнім часом активно впроваджується в життя керування розумним будинком за допомогою голосових команд. Саме голосове управління є найбільш зручним способом віддавати команди інформаційній системі «Розумний будинок».

Мета роботи розглянути та проаналізувати способи опрацювання голосових команд, а також способи їх передачі на кінцеві пристрої. На основі проведених досліджень спроектувати та реалізувати інформаційну систему «Голосове управління розумним будинком».

1. АНАЛІЗ І ПОРІВНЯННЯ ІСНУЮЧИХ СИСТЕМ УПРАВЛІННЯ «РОЗУМНИМ БУДИНКОМ», ЗАСОБІВ ТА ПІДХОДІВ РОЗПІЗНАВАННЯ ГОЛОСУ

1.1 Огляд сучасних систем розумного будинку

Сучасні системи розумного будинку мають за основу одну і ту ж модель:

- ядро обчислень, яке контролює всю систему;

- мікроконтролери, які отримують команди від ядра та в свою чергу працюють вже на фізичному рівні, регулюючи подачу електроенергії на конкретні пов'язані з ними прилади.

Також частиною системи є спосіб комунікації між ядром, мікроконтролерами та деяким приладом керування – смартфон, комп’ютер, планшет або навіть голосовий помічник (пристрій яким може зручно користуватися людина). В даній роботі буде розглянутий останній спосіб - голосе керування.

Для того щоб дізнатися чим же можуть відрізнятися розумні будинки за своєю реалізацією, буде розглянуто основних представників на ринку:

- Ajax Systems;
- Xiaomi;
- Amazon;
- Google;
- Apple;

1.1.1 Ajax Systems

Рисунок 1.1 — Типовий набір приладів від Ajax Systems

Виробник: Україна, відповідно, за замовчуванням підтримується український і російський інтерфейси. Дана система автоматизації будинку в повній мірі вирішує дві великі задачі:

- забезпечує комфорт і зручність в управлінні пристроями;
- гарантує безпеку житла в повній мірі, захищаючи об’єкти від взлому, а також забезпечуючи електричну, пожежну, газову та інші можливі загрози для будинку.

Для зв'язку з пристроями Ajax Hub використовує фірмовий протокол Jeweller. Це апаратно-програмний комплекс, який включає в себе шифрування, синхронізацію і алгоритм управління потужністю для оптимізації живлення. При розробці хабу нового покоління в Ajax Systems прийняли справедливе рішення відмовитися від дротів. Система за задумом інженерів повинна була бути надійною і легкою в установці. А оскільки існуючі бездротові протоколи мали

численні уразливості, Ajax Systems довелося з нуля створювати власний. Деякі з особливостей Jeweller:

- Дані, якими обмінюються датчики з хабом, упаковуються в зашифровані пакети. Ці пару байт інформації передаються за лічені мілісекунди;
- Пакети даних шифруються за допомогою власного алгоритму jCrypt з плаваючим ключем, який теж розроблявся з чистого аркуша;
- Хаб завжди знає, з яким датчиком він обмінюється інформацією навіть в умовах сильних радіоперешкод. У разі виявлення невідповідності або поганого зв'язку, хаб запускає процес аутентифікації датчика для повторної синхронізації з системою;
- Кожен датчик знає своє місце і час виходу в ефір і пінг хаб з інтервалом від 12 секунд;
- Радіоканал зв'язку захищений за допомогою багаторівневої фільтрації. При виявленні хабом сумнівного сигналу запускається процес глибокої перевірки. Одне невідповідність - і система перестає реагувати на підозрілий джерело;

Хаб вміє перемикатися між частотами, щоб запобігти глушіння. До того ж він визначає датчики, які потрапили під вплив джерела перешкод.

Переваги:

- простий монтаж;
- бездротової канал зв'язку між системними елементами;
- велика зона дії сигналу (до 2000 м);
- наявність захисту від зняття будь-якого з датчиків (бампера);
- можливий доступ інших користувачів (повний або частковий);
- автономна робота хаба від акумулятора (до 16 годин);
- різноманітність способів інформування користувача (дзвінок, SMS, Push-повідомлення);

- розумна розетка показує витрата електроенергії (з урахуванням підключених приладів), автоматично відключається при перепадах напруги;
- встановлення по QR-коду і управління за допомогою смартфона (iOS, Android);
- підключення до 100 пристроїв;
- наявність тривожної кнопки на пульті (брелоку);
- невисока вартість комплекту (від 200 $).

Недоліки:

- функціонування тільки з роботою центрального контролера (Hub), тобто відсутність автономності датчиків;
- немає власної камери відеоспостереження (але є можливість підключення стороннього обладнання);
- управління тільки через телефон, хоча це знімає необхідність встановлювати будь-які додаткові програми на ПК.

На сьогодні обладнання Ajax - це одна з найкращих систем розумного будинку, яка може скласти конкуренцію навіть таким гігантам як Amazon. Вона багатофункціональна, надійна, зручна, компактна. Має якісний захист від зломів, чудовий дизайн і зрозумілий інтерфейс. Установка і налаштування такого комплексу спрощені до мінімуму і цілком доступна навіть для технічно непідготовлених користувачів. Важливою перевагою є і досить демократична ціна на девайс, беручи до уваги його широкий функціонал.

1.1.2 Xiaomi

Рисунок 1.2 — Система розумного будинку від Xiaomi.

Різноманіттям базових наборів та окремих пристроїв для систем "Розумний будинок" може похвалитися китайська компанія Xiaomi. Популярність продукції експерти пояснюють адекватною вартістю і гарною якістю. В каталозі бренду можна знайти датчики освітленості, диму, вологості, аналізатори грунту і води, "розумні" світильники та вимикачі. Так як компанія випускає телевізори, смартфони, лічильники, замки, то всі ці гаджети не важко об'єднати в одну систему. Керувати нею допомагає фірмовий додаток. Голосове управління здійснюється китайською мовою. Привабливим фактором для користувачів стає цінова доступність і широке поле для фантазії. А ось пристрої інших компаній не завжди можуть працювати з системами Xiaomi.

Переваги:

- доступна ціна;
- великий асортимент;
- гарна якість;
- зрозумілий додаток.

Недоліки:

- не всі чужі пристрої підтримуються;
- голосове управління не підтримує українську, та навіть англійську мови.

1.1.3 Amazon

Рисунок 1.3 — Найпопулярніші прилади для розумного будинку від Amazon

Найбільшим розробником хмарних технологій і електронних пристроїв є американська компанія Amazon. Експертам сподобалися інтелектуальні колонки, які можуть успішно взаємодіяти з розробками інших компаній. Для зручності користування придумані голосові помічники. Завдяки жорсткій конкуренції з Google виробнику вдається створювати нові інноваційні продукти. Amazon на даний момент є найкращою системою голосового управління розумним будинком. Вони

відзначають зручність в застосуванні і багатий вибір інноваційних пристроїв.

Переваги:

- великий асортимент;
- інноваційні розробки;
- сумісність з продуктами інших компаній;
- зручність в роботі.

Недоліки:

- голосові помічники не говорять українською мовою.

1.1.4 Google

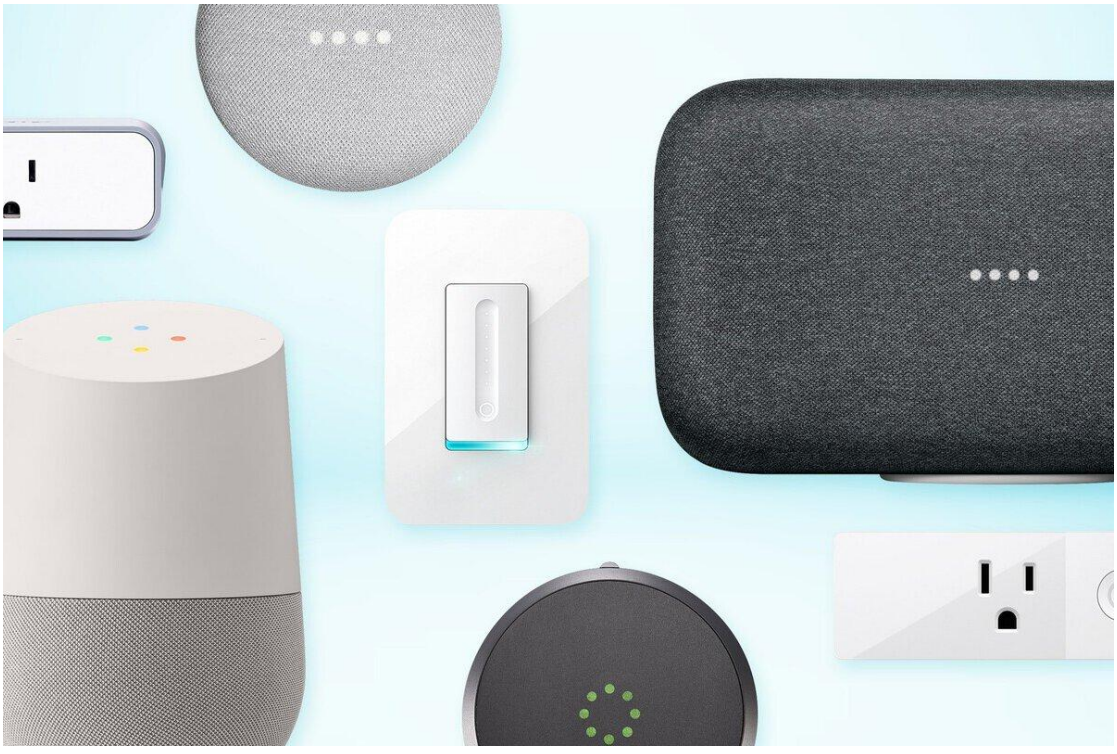

Рисунок 1.4 — Пристрої розумного будинку від Google

Не відстає в розробці і виробництві компонентів для систем "Розумний будинок" відома американська корпорація Google. Багатьом

інтернет-користувачам довелося стикатися з активацією голосового помічника за допомогою знаменитої фрази "Окей, Гугл". На виробничих майданчиках корпорації випускаються інтелектуальні колонки. Всі інші елементи "Розумного будинку" доведеться купувати у інших виробників. Експерти високо оцінили пристрій Home Hub з екраном, яке є серцем системи. Прив'язати до хабу можна як інтелектуальний дверний замок, так і розумну розетку. Вітчизняні користувачі вважають розробки Google ефективними і зручними помічниками. Однак придбати деякі моделі можна тільки в Америці.

Переваги:

- великий асортимент;
- різноманіття кольорів;
- сумісність з розробками інших фірм;
- зручні голосові помічники.

Недоліки:

- не всі системи доступні в Україні.

1.1.5 Apple

Рисунок 1.5 — Apple HomeKit - рішення від APPLE для систем розумного будинку

Є власні системи розумного будинку і у APPLE. Виробники APPLE прив'язують свої розробки до смартфонам на платформі IOS. Експерти відзначають високу якість кожного елемента, надійність і довговічність. Найвідомішою розробкою є Apple HomeKit. Продукт оснащений російськомовним голосовим помічником, що стає важливою перевагою перед конкурентами, хоча українська мова і не підтримується. Система сумісна з багатьма датчиками і пристроями інших виробників. Власники "розумних систем" APPLE хвалять розробки за стильний пульт управління та сумісність з іншими елементами. Мінусом вважається обмежене число пристроїв, що працюють від мережі 220 В.

Переваги:

- голосова підтримка російською мовою;
- зручний додаток;

- сумісність з датчиками і пристроями інших фірм;
- стильний дизайн.

Недоліки:

- деякі системи не працюють від мережі 220 В.

1.2 Класифікація систем розпізнавання голосу

Основою голосового керування є розпізнавання мови – тобто правильна інтерпретація мовного сигналу у цифрову інформацію. Розпізнавання мови є альтернативою традиційним методам взаємодії з комп'ютером, наприклад, введення тексту через клавіатуру. Ефективна система може замінити повністю або частково роботу стандартної клавіатури та миші.

Система розпізнавання мовлення складається з таких компонентів:

- мікрофон;
- програмне забезпечення для розпізнавання мови;
- ядро розумного будинку, або хмарний сервіс на якому працює ПЗ для розпізнавання мови
- якісна звукова карта для введення та / або виведення

На рисунку 1.6 представлена класична функціональна схема системи розпізнавання мови, що складається з наступних компонентів:

- мікрофон;
- блок обробки;
- блок аналізу;

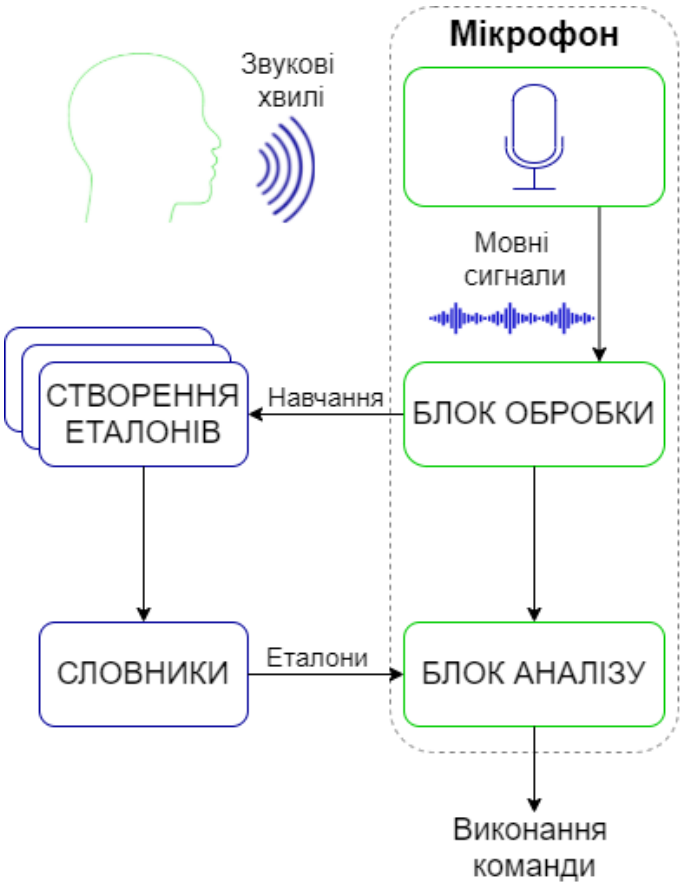

Рисунок 1.6 — Класична функціональна схема системи розпізнавання мови

Захоплення голосу при усуненні сторонньої інформації, що залежить від мовця, такої як висота голосу, тембр і так далі – ключова вимога до точного розпізнавання мови.

Системи розпізнавання мови класифікуються за такими параметрами:

- за розміром словника – обмежений набір слів, словник великого розміру;
- по залежності від диктора – дикторозалежні та дикторонезалежні системи;
- за типом мовлення – разом або окремо;
- за призначенням – системи диктування, командні системи;
- по використовуваному алгоритму - нейронні мережі, приховані марковські моделі, динамічне програмування;

- за типом структурної одиниці – фрази, слова, фонеми, діфони, алофони;
- за принципом виділення структурних одиниць розпізнавання за шаблоном, виділення лексичних елементів.

Архітектура системи розпізнавання включає в себе такі компоненти:

- модуль виділення необхідного сигналу, відокремленого від шуму;
- акустична модель – відповідність заздалегідь заданої статистичної моделі, що описує використання деякого звуку у вимові до отриманого звуку;
- мовна модель – дозволяє визначити найбільш вірогідні послідовності слів. Складність побудови мовної моделі залежить від конкретної мови, яка розпізнається. Для високо флективних мов (мов, в яких існує багато форм одного і того ж слова), мовні моделі, побудовані тільки з використанням статистики, вже не дають такого ефекту - занадто багато потрібно даних, щоб достовірно оцінити статистичні зв'язки між словами. Тому застосовують гібридні мовні моделі, які використовують правила мови, інформацію про частини мови, форми слова і класичну статистичну модель.
- Декодер - програмний компонент системи розпізнавання, Який поєднує дані, отримані в ході розпізнавання від акустичних і мовних моделей, і на підставі їх об'єднання, визначає найбільш вірогідну послідовність слів, яка і є кінцевим результатом розпізнавання мовлення.

1.3 Постановка задачі

Результатом виконання дипломної роботи має стати модель прототипа розумного будинку з голосовим управлінням. Система має включати в себе:

Веб-додаток;

Пристрої.

Веб додаток повинен виконуватися на сервері для керування пристроями розумного будинку. Методи контролера на сервері повинні виконувати керування відповідними мікроконтролерами.

Для безпеки керування пристроями розумного будинку необхідно реалізувати механізми авторизації та аутентифікації.

Систему потрібно змоделювати віртуально та провести тестування.

Для досягнення поставленої мети необхідно провести:

аналіз засобів моделювання розумного будинку;

аналіз можливостей розпізнавання голосових команд;

аналіз існуючих протоколів взаємодії між компонентами розумного будинку.

На основі проведеного аналізу виробити оптимальне рішення.

1.4 Аналіз існуючих підходів розпізнавання голосових команд

Підхід сам по собі це деякий спосіб вирішення певної задачі. В даному випадку будуть представлені основні алгоритми для розпізнавання голосу – способи обробки голосової інформації. Виділяють такі основні методи:

- алгоритм динамічного спотворення часу;
- метод прихованих марковських моделей;
- нейронні мережі;
- згорткові нейронні мережі;

1.4.1 Підхід з використанням алгоритму динамічного спотворення часу

Динамічне спотворення часу – це підхід, який історично використовувався для розпізнавання мови, але тепер в значній мірі витіснений більш успішним способом, заснованим на прихованих марківських моделях.

Динамічне спотворення часу це алгоритм вимірювання подібності між двома послідовностями, які можуть відрізнятися за часом чи швидкістю. Наприклад, схожість в моделях руху буде виявлено, навіть якщо на одному відео людина йшла повільно, а на іншому швидше, або навіть якщо в ході одного спостереження були прискорення і уповільнення. DTW був застосований до відео, аудіо та графіку - дійсно, будь-які дані, які можна перетворити в лінійне уявлення, можна проаналізувати за допомогою DTW.

Добре відомим додатком було автоматичне розпізнавання мови, щоб впоратися з різною швидкістю мови. Загалом це метод, який дозволяє комп'ютеру знайти оптимальну відповідність між двома заданими послідовностями (наприклад, тимчасовими рядами) з певними обмеженнями. Тобто послідовності нелінійно «спотворюються», щоб відповідати один одному. Цей метод вирівнювання послідовностей часто використовується в контексті прихованих марковських моделей.

1.4.2 Підхід з використанням методу прихованих марковських моделей

Сучасні системи розпізнавання побудовані на методі прихованих марковських моделей. Це статистичні моделі, які виводять послідовність символів або величин. Приховані марковські моделі (HMM) використовуються в розпізнаванні мови, тому що мовний сигнал можна розглядати як кусочно-стаціонарний сигнал або як короткочасний стаціонарний сигнал. У короткому часовому масштабі (наприклад, 10 мілісекунд) мова може бути наближена до стаціонарного процесу. Мову можна розглядати як марковскую модель для багатьох стохастичних цілей.

Ще одна причина популярності HMM полягає в тому, що їх можна навчати автоматично, вони прості і доступні з обчислювальної точки зору.

1.4.3 Метод розпізнавання мови за допомогою нейронних мереж

Нейронні мережі стали привабливим методом акустичного моделювання в ASR в кінці 1980-х років. З тих пір нейронні мережі використовувалися в багатьох аспектах розпізнавання мови, таких як класифікація фонем, класифікація фонем за допомогою багатоцільових еволюційних алгоритмів, розпізнавання окремих слів.

Нейронні мережі роблять менше явних припущень про статистичні властивості ознак, ніж HMM, і мають ряд якостей, які роблять їх привабливими моделями розпізнавання для розпізнавання мови. При використанні для оцінки ймовірностей сегмента мовних характеристик нейронні мережі дозволяють природним і ефективним чином розрізняти навчання. Однак, незважаючи на їх ефективність в класифікації короткочасних одиниць, таких як окремі фонеми і окремі слова, ранні нейронні мережі рідко були успішними для задач безперервного розпізнавання через їх обмеженою здатності моделювати тимчасові залежності.

Один з підходів до цього обмеження полягав у використанні нейронних мереж в якості попередньої обробки, перетворення ознак або зменшення розмірності, крок перед розпізнаванням на основі HMM. Однак зовсім недавно LSTM і пов'язані рекурентні нейронні мережі (RNN) і нейронні мережі з тимчасовою затримкою (TDNN) продемонстрували покращену продуктивність в цій галузі.

1.4.4 Метод розпізнавання мови з використання згорткових нейронних мереж

CNN – це популярний варіант глибокого навчання, який широко застосовуються в системах ASR. CNN мають багато привабливих особливостей, таких як розподіл ваги, згорткові фільтри та об'єднання. Таким чином, CNN досягли вражаючих показників у ASR. CNN складаються з безлічі згорткових шарів.

Глибокі CNN досягли точності приблизно людського рівня завдяки вдосконаленій архітектурі та оптимізованому навчанню. CNN використовують нелінійну функцію для безпосередньої обробки даних низького рівня. CNN здатні вивчати особливості високого рівня з високою складністю та абстракцією.

Агрегування – ущільнення карти ознак, при цьому група пікселів (зазвичай розміру 2 × 2) ущільнюється до одного пікселя, проходячи нелінійне перетворення. Найбільш споживані при цьому функція максимуму. Перетворення зачіпають непересічні прямокутники або квадрати, кожен з яких скорочується в один піксель, при цьому вибирається піксель, що має максимальне значення. Коротко принцип агрегації можна описати так: якщо на попередній операції згортки вже були виявлені деякі ознаки, то для подальшої обробки настільки докладне зображення вже не потрібно, і воно ущільнюється до менш докладного. Шар агрегації, як правило, вставляється після шару згортки перед шаром наступної згортки. Принцип роботи даного типу нейронної мережі можна бачити на рисунку 1.7.

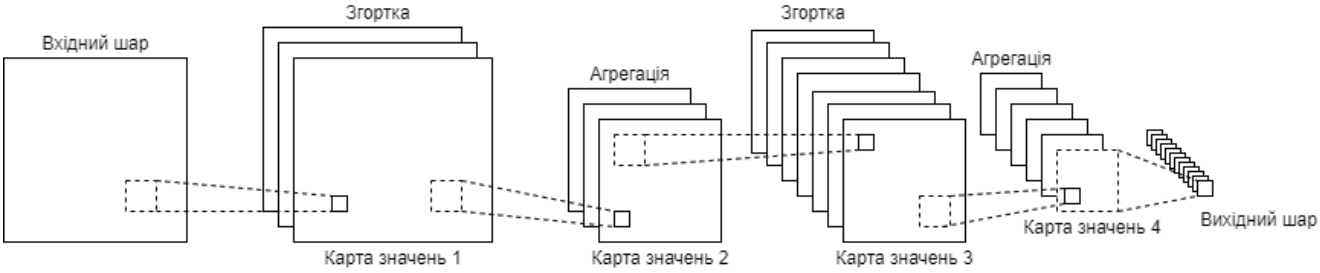

Рис 1.7 — Схема роботи згорткової нейронної мережі

Одна з найбільш точних систем розпізнавання голосу заснована на CNN та HMM, дану схему показано на рисунку 1.8. Ця схема включає в себе попередню обробку мовленнєвого сигналу за допомогою згорткової нейронної мережі та подальше декодування обробленої інформації за допомогою прихованих марковських моделей, а саме пошук тих самих слів, які і були зашифровані у голосі.

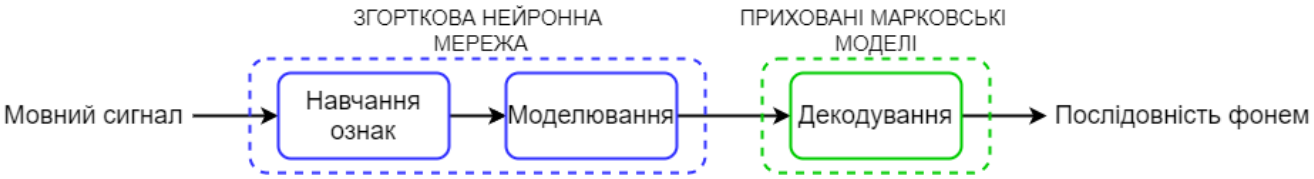

Рис 1.8 — Система розпізнавання мови основана на згортковій нейронній мережі

2. АНАЛІЗ ЗАСОБІВ МОДЕЛЮВАННЯ РОЗУМНОГО БУДИНКУ ТА МОЖЛИВОСТЕЙ РОЗПІЗНАВАННЯ ГОЛОСОВИХ КОМАНД

2.1 Аналіз існуючих протоколів взаємодії між компонентами розумного будинку

2.1.1 Протокол 1-Wire

Мабуть, найпростіший спосіб зв'язати всі пристрої це використати стандарт 1-Wire. Справа в тому, що основною магістраллю для передачі даних тут виступає двунаправленная шина, яка в найпростішому випадку виглядає як двожильний провід. Іншими словами, при створенні мережі можна обійтися навіть дешевим телефонним кабелем. Один провід в даному випадку використовується для живлення і передачі даних, інший для заземлення. Топологія мережі – загальна шина, це означає що пристрої під’єднані до одного кабелю. Звичайно, мережа 1-Wire не зобов'язана виглядати незграбно: можна взяти за основу якісну «виту пару» або FireWire-кабель, а компоненти підключити через RJ-розетки. Власне, чим добротнее організована мережа, тим більшу протяжність вона може мати: в ідеальних умовах це значення досягає 300 метрів. Проте, головні переваги даного стандартну - дешевизна і невибагливість. Є у нього і негативна характеристика - низька відмовостійкість.

Пристрої 1-Wire, як правило, підключається до комп'ютера. Для цього використовуються спеціальні адаптери. Рідше управління мережа віддається на керування мікроконтролеру. Зазвичай список використовуваних компонентів 1-Wire обмежується датчиками температури, освітленості, вологості і протікання, а також електричними розетками, здатними включатися і виключатися по команді. У деяких випадках це рішення також використовується для управління освітленням. В цілому, 1-Wire підійде економним користувачам, які не пред'являють серйозних вимог до функціонала розумного будинку.

2.1.2 Протокол X10

X10 – стандарт, розроблений ще в 1975 році, проте він як і раніше активно використовується при проектуванні розумних будинків. Секрет подібної живучості – надзвичайна універсальність при відносно низькій вартості. На відміну від 1-Wire, X10 не вимагає прокладки спеціального кабелю: для передачі сигналу використовується електропроводка будівлі. Також передбачена можливість використання трансиверів, які ловлять радіосигнал від бездротових пристроїв, перетворюють його в потрібний формат і передають в електричну мережу. Призначений він для взаємодії з датчиками і пультами дистанційного керування.

X10 може похвалитися великою кількістю всіляких виконавчих модулів. При їх належній кількості, автоматика здатна управляти електроприладами, освітленням, опаленням, вентиляцією і охоронними системами. Можна навіть налаштувати ваш будинок самостійно здійснювати полив рослин і регулювати ступінь відкриття віконних штор на підставі різних факторів. Для коректної роботи всього цього необхідні спеціальні контролери. Як правило, їх можна один раз запрограмувати за допомогою комп'ютера, і надалі вони будуть виступати в якості незалежних пристроїв.

Серйозним недоліком X10 є низька швидкість передачі даних, через яку відгук на будь-яку дію відбувається з секундною затримкою. Ну, а оскільки команди передаються послідовно, то, скажімо, організувати складне динамічне освітлення за допомогою даного протоколу практично неможливо. Крім того, для використання X10 може знадобитися незначна переробка електропроводки.

2.1.3 Протокол KNX

KNX – дорогий варіант, який користується популярністю в Європі. Стандарт характеризується великою кількістю закладених в нього функцій, а

також складністю проектування і монтажу. Як середовище для передачі даних протокол KNX може використовувати шину (виту пару), електричну мережу або радіоканал. Найчастіше застосовується перший варіант, нерідко необхідні дроти прокладаються сумісно з силовими кабелями ще на етапі будівництва. Стандарт передбачає різні варіанти топології мережі. Підсумкова система повинна мати власне джерело живлення, але при цьому в ній може бути відсутнім центральний контролер, тобто KNX дозволяє створювати децентралізовані рішення, в яких сенсори і виконавчі модулі взаємодіють безпосередньо. Протокол підходить для автоматизації великих будівель, в одну мережу можна об'єднати до 58000 пристроїв. Пристрої в даному випадку відрізняються різноманітністю і дозволяють обладнати будинок нетривіальним функціоналом.

З програмно-апаратної точки зору KNX це серйозний стандарт для вирішення серйозних завдань. Ось тільки він погано підходить для ентузіастів-одинаків: обладнання коштує дорого і зв'язати його воєдино непросто. Ну а якщо довірити проектування і монтаж KNX-рішення фахівцям, то підсумкова вартість проекту буде занадто високою.

2.1.4 Протокол Wi-Fi

Wi-Fi використовується практично в кожному розумному будинку. Зазвичай цей стандарт зв'язку застосовується для того, щоб подружити смартфон або планшет з уже готовою автоматизованою системою. Мобільний пристрій завжди під рукою, а тому управляти будинком з його допомогою, часом, зручніше, ніж використовувати для цієї мети комп'ютер, настінну сенсорну панель, пульт дистанційного керування або інтерпретатор мови. Особливо це актуально в тих випадках, коли для взаємодії з системою передбачено спеціальний додаток, а не тільки веб-інтерфейс.

Іноді Wi-Fi застосовується для зв'язку з пристроями, які можуть функціонувати автономно, без допомоги «розумної» мережі. Приклад такого

гаджета – лампочка LIFX. Останнім часом подібні пристрої набирають популярність, адже вони дозволяють долучитися до технологій розумного будинку, не витрачаючи час і сили на установку допоміжного обладнання. Для складних систем автоматизації Wi-Fi не підходить: модулі зв'язку цього стандарту дорогі, а демонстровані ними швидкості в рамках розумного будинку просто не потрібні.

2.1.5 Протокол ZigBee

ZigBee як протокол зв'язку по радіоканалу, чудово вписується в концепцію розумного будинку. По-перше, стандарт дозволяє створювати датчики з низьким енергоспоживанням і чудовою чутливістю: більшу частину часу їх бездротові модулі знаходяться в сплячому режимі, але на пробудження останніх витрачається всього 15 мілісекунд. По-друге, ZigBee підтримує сітчасту топологію мережі, при якій окремі компоненти можуть виступати в якості посередника, що передає сигнал від одного пристрою до іншого. Подібна структура здатна до самоорганізації і самовідновлення, вихід з ладу одного-двох елементів, як правило, не призводить до серйозних наслідків. Сітчаста мережа також дозволяє істотно збільшити зону покриття бездротової мережі, так що при грамотному підході ZigBee можна використовувати для автоматизації не тільки житлових будинків, але і великих робочих приміщень.

Компоненти мережі поділяються на три типи: координатори, маршрутизатори і кінцеві пристрої. Перші беруть на себе функції управління мережею. Це означає, що у системі обов'язково повинен бути присутнім хоча б один координатор. У ролі маршрутизаторів виступають пристрої-посередники. Вони зобов'язані бути активними постійно, і тому мають потребу в живленні від електромережі. Кінцеві пристрої це різноманітні датчики і контролери виконавчих пристроїв. Вони можуть працювати від комплекту батарей місяцями і навіть роками.

В цілому, ZigBee підходить для вирішення всіх типових завдань, пов'язаних з проектуванням розумного будинку. При цьому вартість обладнання можна назвати помірною, та й процес монтажу відносно простий. Втім, є у стандарту і невеликий мінус: ZigBee-пристрої різних виробників нерідко виявляються несумісними.

2.1.6 Протокол Z-Wave

Z-Wave - протокол бездротового зв'язку, який багато в чому схожий з ZigBee - тут і низьке енергоспоживання, і підтримка сітчастої топології мережі. Ці стандарти також можна порівняти за вартістю обладнання. Зрозуміло, при детальному вивченні протоколів можна виявити деякі відмінності в технічній частині, однак куди важливішим є той факт, що їх розробники по-різному ставляться до питання стандартизації. У Z-Wave з цим строго: всі пристрої, незалежно від фірми-виробника, базуються на бездротових модулях Sigma Designs (саме ця компанія розробила протокол). Як наслідок, всі вони сумісні один з одним. Також для даного стандарту випущено менше програмного забезпечення, ніж для ZigBee, але зате воно гарантовано працює з будь обладнанням Z-Wave.

Таким чином, Z-Wave - це конструктор LEGO від світу розумних будинків. Так, це не найдешевший і не найнадійніший варіант, але для тих, хто хоче самостійно автоматизувати своє житло, це один з найкращих варіантів.

2.1.7 Протокол Insteon

Insteon користується великою популярністю в США, проте в Європу він прийшов зовсім недавно. Фактором, що стримує поширення, стала несумісність початкової версії протоколу з європейськими електромережами: як і X10, Insteon використовує проводку будівлі для передачі сигналів. Однак, на відміну від

застарілого конкурента, новий стандарт також підтримує зв'язок по радіоканалу, причому провідна і бездротова мережа функціонують одночасно, доповнюючи один одного і істотно підвищуючи надійність автоматизованої системи. Крім того, у Insteon немає проблем з чуйністю, властивих X10.

До плюсів Insteon також можна віднести сітчасту топологію мережі і сумісність з пристроями X10: є можливість поступово перейти зі старого стандарту на новий. Цікава особливість - можливість організувати працездатну мережу без використання центрального контролера. Звичайно, в цьому випадку функціонал розумного будинку буде сильно обмежений.

З точки зору проектування, американський протокол схожий з Z-Wave: все жорстко стандартизовано, значна частина обладнання випускається під брендом Insteon самими творцями протоколу - фірмою Smartlabs. Систему для розумного будинку на основі нового стандарту можна збирати поступово, докуповуючи необхідні компоненти по ходу.

2.1.8 Протокол Bluetooth

Типова архітектура IoT складається з апаратного забезпечення, способу комунікації та програмного забезпеченняа. Bluetooth - це один із способів комунікації. Принцип дії заснований на використанні радіохвиль. Радіозв'язок Bluetooth здійснюється в ISM-діапазоні, який використовується в різних побутових приладах і бездротових мережах. Частоти Bluetooth: 2.402-2.48 ГГц (в мегагерцах 2402-2480 МГц). Bluetooth або BLE є частиною рівня передачі даних, який підключає або датчик до датчика, або датчик до шлюзу.

Порівняно з іншими протоколами зв'язку, такими як RFID, NFC, WLAN, LoRa WAN, LTE-A, WiFi-Direct, технологія Bluetooth є менш затратною для розгортання. Він забезпечує бездротове підключення та створює миттєву персональну мережу (PAN), де бездротова інфраструктура недоступна. Мало того,

типовий діапазон Bluetooth від 0 до 30 метрів можна збільшити, посиливши потужність, що перевищує один міліват (мВт).

Системи IoT із підтримкою Bluetooth все частіше використовуються в різних галузях, включаючи управління активами, безпеку та охорону праці, охорону здоров'я, автомобільну промисловість, побутові та розважальні програми та безпеку. Оскільки ринок IoT, за оцінками, зросте з 157,05 млрд. Доларів США у 2016 році до 661,74 млрд. Доларів США до 2021 року, IoT, що працює від Bluetooth, може стати великим диференціатором для компаній, які прагнуть підвищити ефективність.

2.2 Аналіз існуючої інфраструктури хмарних засобів

2.2.1 Google Voice Recognition

Система розпізнавання голосу від світового гіганту Google вигідно виглядає на сучасному ринку такого типу технологій. Вона має ряд функцій, що дозволяють налаштувати систему як вам буде потрібно.

Налаштування системи для розпізнавання специфічних для домену термінів та рідкісних слів, надаючи підказки та підвищуючи точність транскрипції певних слів або фраз. Можливо за допомогою класів автоматично перетворювати розмовні числа в адреси, роки, валюти тощо.

Можливість вибору з набору навчених моделей для голосового управління, телефонної розмови та транскрипції відео, оптимізованих для вимог до якості для конкретного домену. Наприклад вдосконалена модель телефонних дзвінків налаштована на звук, що походить від телефонії, наприклад, телефонні дзвінки, записані з частотою дискретизації 8 кГц.

API дозволяє отримувати результати розпізнавання мовлення в режимі реального часу, за допомогою хмарного серверу Google.

2.2.2 Microsoft Azure Speech to Text

Microsoft Azure Speech to Text – одна з найдосконаліших платформ розпізнавання голосу. Як частина асортименту продуктів Microsoft Cognitive Speech Services, вона використовує алгоритми глибокого навчання для подолання низької якості звуку і може адаптуватися до різних стилів мовлення для точної транскрипції звуку.

Варто зазначити, що Microsoft Azure Speech to Text це орієнтована на розробників платформа, призначена для допомоги підприємствам у створенні, тестуванні та управлінні власними продуктами.

Ключовою функцією Azure Speech to Text є доступ, який вона надає до потужної системи обробки мов Microsoft. За останні кілька років мовний інтелект від Microsoft досягнув нового рівня. Це означає, що тепер він може виконувати завдання, які раніше були неможливими для систем розпізнавання мови, наприклад, точну транскрипцію перехресних розмов під час невеликих групових бесід.

Azure працює з десятками мов та діалектів, і його можна навчити, використовуючи власні моделі розпізнавання мови, щоб краще адаптувати до стилю мовлення користувача, фонового шуму та словникового запасу.

Microsoft Azure для розробників, а не для споживачів. Це означає, що його налаштування є дещо складною процедурою, яку найкраще залишити комусь із значним технічним досвідом.

Найшвидший спосіб налаштування Azure це використання Azure Speech SDK у мові програмування, як Java або C ++. Для цього потрібно буде зареєструватися для безкоштовного облікового запису Azure і створити порожній проект у своєму середовищі розробки. Потім потрібно буде використовувати Microsoft Visual Studio і написати коротку програму для ініціалізації об’єкта SpeechRecognizer від Microsoft.

Як і інші платформи масової транскрипції, Microsoft Azure Speech to Text призначений для запуску як інтерфейсу програмування програм (API), додавання

до програм Office 365 або інтеграції в нові платформи та служби. Через це немає єдиного інтерфейсу Azure Speech to Text. Те, що побачить кінцевий користувач, залежить від того, як було інтегровано Azure Speech to Text.

2.2.3 Amazon Transcribe

Amazon Transcribe надає рішення своїм клієнтам, додаючи функцію перетворення мови в текст в свої програмах. Він використовує глибокі нейронні мережі, які займаються автоматичним розпізнаванням мови, що дозволяє миттєво перетворити її на текст, який майже гарантовано буде точним. Постачальник пропонує серед своїх функцій легкі для читання транскрипції, користувальницький словник, фільтрацію словника, ідентифікацію каналів та автоматичне редагування вмісту.

Функції Amazon Transcribe

Основними особливостями Amazon Transcribe є:

- транскрипції;
- потокова передача транскрипцій;
- створення мітки часу;
- словник;
- словникова фільтрація;
- розпізнавання декілька мовців;
- ідентифікація каналу

3. РЕАЛІЗАЦІЯ ІНФОРМАЦІЙНОЇ СИСТЕМИ "ГОЛОСОВЕ УПРАВЛІННЯ РОЗУМНИМ БУДИНКОМ"

3.1 Аналіз вхідної інформації та розробка моделі запропонованої системи

Проаналізувавши усі доступні засоби зв’язку, прийшли до висновку, що доцільно використовувати Bluetooth, як протокол, що активно підтримується та має високу швидкість передачі даних. Сервер обробки показників датчиків та подачі команд буде знаходитися в самому будинку, що означає майже відсутність затримки передачі даних. Кінцеві пристрої будуть приєднані до мікроконтролерів моделі Arduino UNO R3 та Bluetooth модулем HC5 які отримуватимуть сигнали від сервера.

3.2 Розробка структури мережі розумного будинку

Мережа розумного будинку буде складатися з деякого ядра, розташованого в самому будинку, яке буде контролювати всі прилади, які під’єднані до нього. Керування користувачем буде відбуватися за допомогою веб-додатку з функцією голосового розпізнавання. Представлення даної системи ви можете бачити на рисунку 3.1.

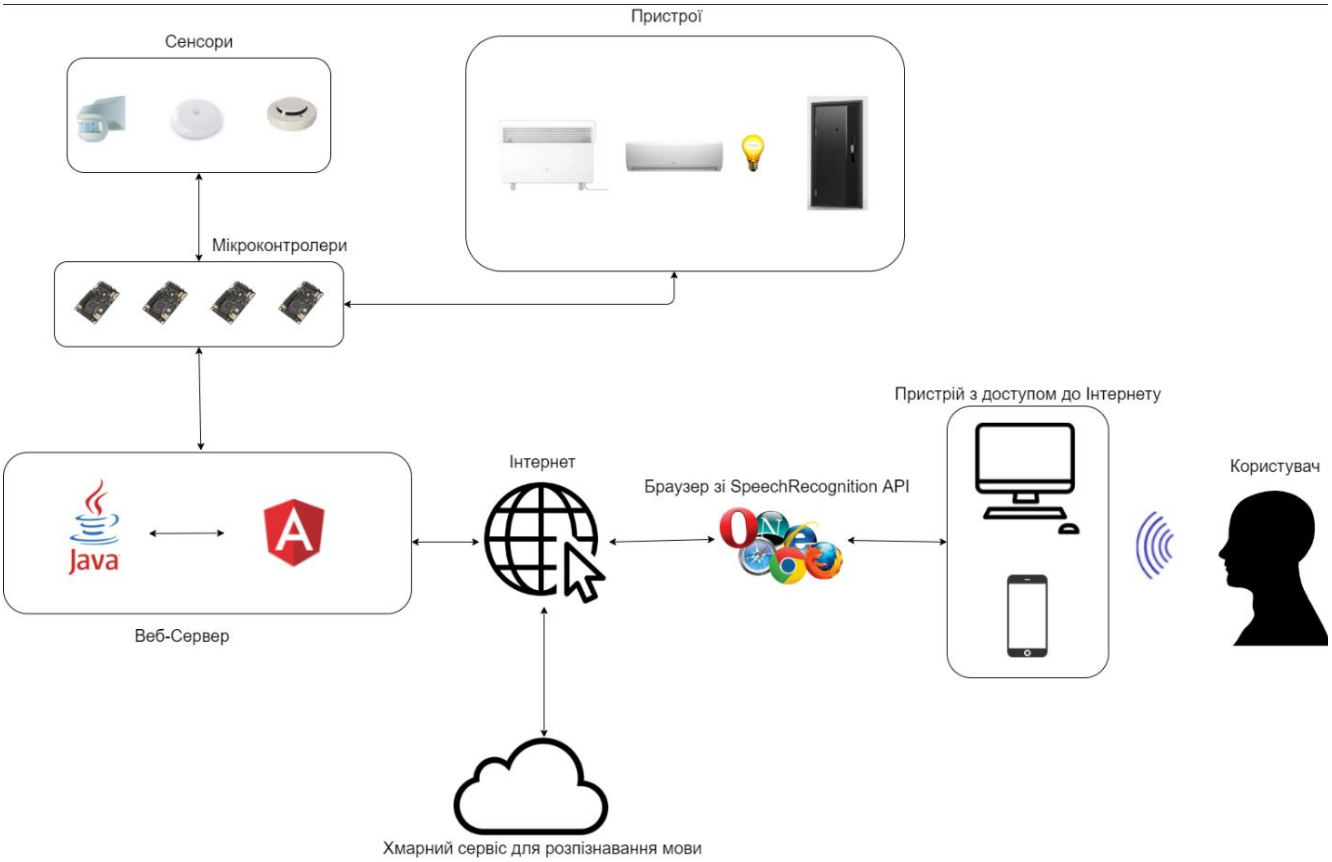

Рисунок 3.1 — Система розумного будинку з голосовим управлінням

3.3 Вибір апаратних засобів реалізації, стеку технологій, протоколів взаємодії, хмарних засобів, програмних засобів, середовища програмування, Frameworks

Оскільки реалізація повної моделі розумного будинку зараз неможлива, необхідно знайти альтернативу у програмному середовищі - тобто замінити реальний прототип на симуляцію розумного будинку, для цього був обраний такий стек технологій:

Java - back-end server

Angular - front-end

Proteus - симуляція пристроїв розумного будинку, під’єднана до сервера за допомогою віртуального послідовного порта.

Для веб-серверу будемо використовувати фреймворк Java Spring зі зручною документацією та відносно простою реалізацією. Також одним з найбільших переваг самої Java є платформонезалежність. Як IDE ми використали Intellij Idea від JetBrains, тому що ця IDE перевірена часом та зарекомендувала себе, як одна з найкращих.

Angular був обраний як сучасна front-end платформа з детальною документацією, надзвичайно великою кількістю реалізованих функцій, що дає можливість створювати дуже швидкі та надійні інтерфейси користувача. Для написання коду використовується IDE дуже високого рівня WebStorm від компанії JetBrains з усім необхідним функціоналом для зручного програмування.

Для розпізнавання мови була використана JS бібліотека annyang. Ця бібліотека використовує SpeechRecognition API вмонтоване в браузер. Таким чином, вказана бібліотека використовує засоби браузера для розпізнавання мови і виділення команди управління із потоку. Оскільки розпізнавання проводиться на віддаленому сервері з великими потужностями, точність розпізнавання мови може доходити до 99%, що робить керування голосом надзвичайно зручним та надійним.

Для симуляції пристроїв розумного будинку розглядали два основних рішення Cisco Packet Tracer і Proteus. Cisco Packet Tracer є дуже зручним для моделювання і навіть презентації реальної роботи систем на базі інтернету речей. Proteus – більш спеціалізована програма для радіотехніки, що дозволяє будувати керування пристроями на рівні мікроконтролерів, датчиків. В даному випадку вибір впав на Proteus, оскільки він має зручний спосіб зв’язку з реальним світом за допомогою віртуального послідовного інтерфейсу, чого на жаль немає у Cisco Packet Tracer. В Cisco Packet Tracer тільки є простий веб-сервер, який не має потрібного функціоналу.

3.4 Реалізація алгоритмів взаємодії між вузлами та хмарою та управління на основі обраного стеку протоколів

Для взаємодії з клієнтом використовується інтерфейс, який відправляє команди користувача у вигляді запитів до сервера, на якому контролери їх обробляють, та відправляють команду через віртуальний послідовний порт на мікроконтролер Arduino UNO R3 в програмі Proteus, який у свою чергу обробляє дану команду так виконує відповідні дії з кінцевими пристроями. На рисунку 3.2 можна бачити блок коду, що реалізує контроллер, який обробляє запити від фронтенду та відправляє команди керування на мікроконтролер Arduino UNO R3, що створений в симуляції Proteus.

```
@PostMapping
private Map<String, String> SendCommand(@RequestBody String command) throws SerialPortException {
    System.out.println(command);
    serialPort.writeBytes(command.getBytes());
    return Collections.singletonMap("success", "true");
}
```

Рисунок 3.2 — Контроллер для обробки запитів від користувача та відправки команд на мікроконтролер

На рисунку 3.3 знаходиться блок коду, що описує відповідності між деякою командою та дією. В даному випадку - відправкою post-запиту до контролюючого систему розумного будинку сервера, при надходженні голосового сигналу у вигляді слова, фрази або речення. Дану команду буде порівняно з елементами масиву значень строкового типу даних, що надходять від хмарного сервера, після відправлення на нього цього мовного сигналу за допомогою SpeechRecognition API, вбудованого в браузер, та подальшої обробки ним цього самого сигналу за допомогою системи розпізнавання мови.

```
const commands = {
  'turn on light': () => {
    this.light = 'ON';
    this.http.post( url: 'http://localhost:8080', body: 'turn on light').subscribe( next: result => {console.log(result); }); },
  'turn off light': () => {
    this.light = 'OFF';
    this.http.post( url: 'http://localhost:8080', body: 'turn off light').subscribe( next: result => {console.log(result); }); },
  'turn on fan': () => {
    this.fan = 'ON';
    this.http.post( url: 'http://localhost:8080', body: 'turn on fan').subscribe( next: result => {console.log(result); }); },
  'turn off fan': () => {
    this.fan = 'OFF';
    this.http.post( url: 'http://localhost:8080', body: 'turn off fan').subscribe( next: result => {console.log(result); }); },
  'turn on night light': () => {
    this.nightLight = 'ON';
    this.http.post( url: 'http://localhost:8080', body: 'turn on night light').subscribe( next: result => {console.log(result); }); },
  'turn off night light': () => {
    this.nightLight = 'OFF';
    this.http.post( url: 'http://localhost:8080', body: 'turn off night light').subscribe( next: result => {console.log(result); }); }
};
```

Рисунок 3.3 — Блок коду, що реалізує відправку команди до сервера, у випадку надходження відповідного голосового сигналу - слова чи фрази

На рисунку 3.4 реалізована схема розумного будинку в програмі Proteus. На схемі можна бачити вентилятор, який зображає умовний кондиціонер, лампочку, що імітує систему освітлення будинку та світлодіод, що представляє LED світильники будинку. Всі пристрої під’єднані до відповідних реле, що контролюють подачу електроенергії до пристроїв від джерел живлення. Ланцюг живлення замикається при подачі сигналу від Arduino UNO R3 на відповідний транзистор, який з’єднаний з реле та діодом. Діод в свою чергу виконує шунтування ЕДС котушки реле, що виникає в момент закриття транзистора, щоб транзистор не вийшов із ладу. Сам Arduino UNO R3 також під’єднано до Bluetooth модуля. За допомогою Bluetooth модуля відбувається керування Arduino UNO R3 через віртуальний послідовний порт.

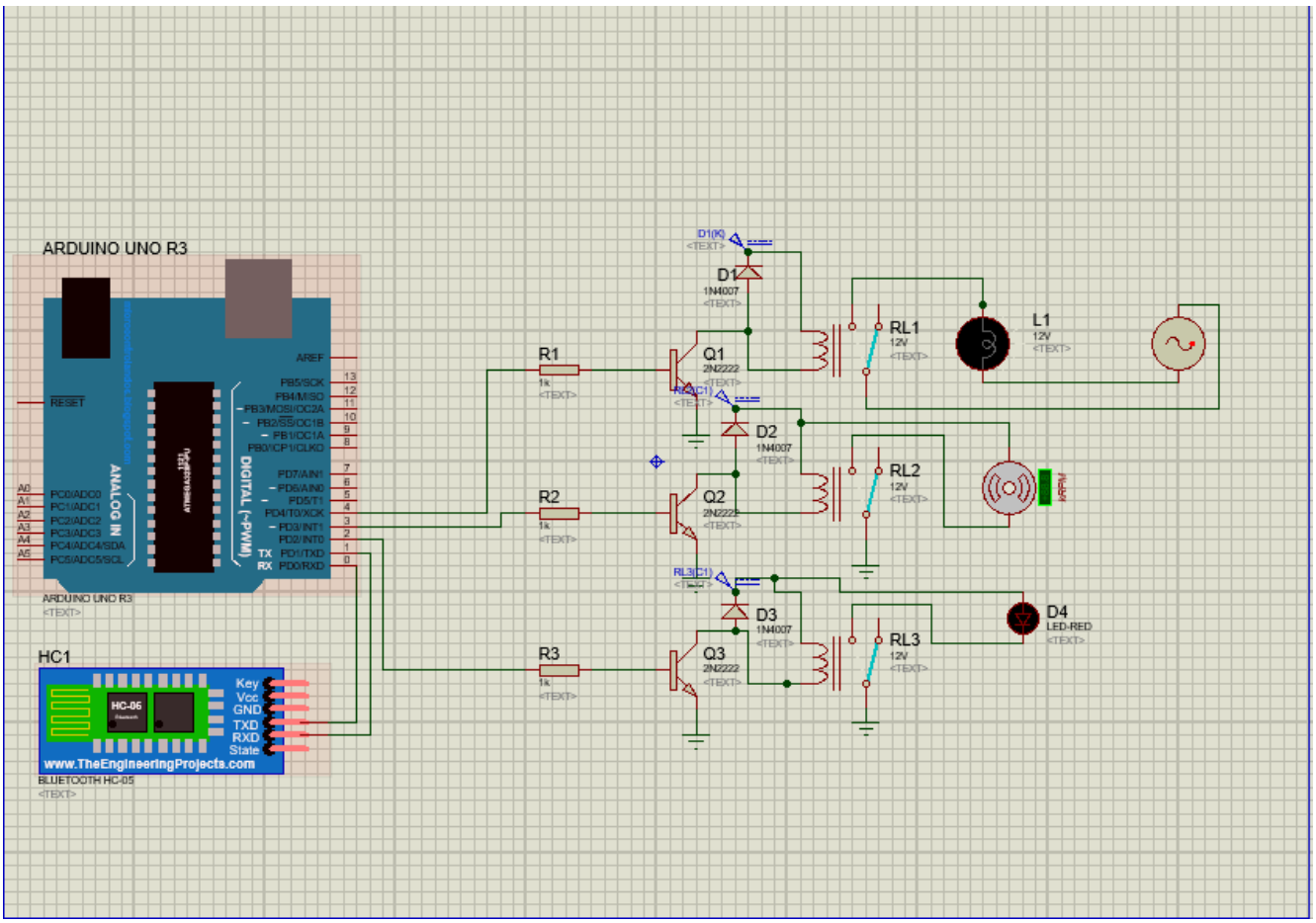

Рисунок 3.4 — Схема в програмі Proteus, що представляє пристрої розумного будинку

На рисунку 3.5 можна бачити блок коду для Arduino UNO R3. На ньому видно умови подачі напруги на відповідний вихід.

```
String command;
void setup()
{
  Serial.begin(9600);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
}

void loop()
{
  if (Serial.available()){
   command = Serial.readStringUntil('\n');
   command.trim();
   if (command.equals("turn on light")){
     digitalWrite(4, HIGH);
   }
   if (command.equals("turn off light")){
     digitalWrite(4, LOW);
   }
   if (command.equals("turn on fan")){
     digitalWrite(3, HIGH);
   }
   if (command.equals("turn off fan")){
     digitalWrite(3, LOW);
   }
   if (command.equals("turn on night light")){
     digitalWrite(2, HIGH);
   }
   if (command.equals("turn off night light")){
     digitalWrite(2, LOW);
   }
  }
}
```

Рисунок 3.5 — Код для Arduino UNO R3, що реалізує на фізичному рівні керування пристроями

3.5 Створення інтерфейсу користувача

Інтерфейс користувача складається з механізму голосового розпізнавання, сторінки зі статусом розумних речей, наприклад станів “світло включене” чи “світло виключене” та авторизації.

На рисунку 3.6 можна бачити форму реєстрації, що потребує введення даних для створення аккаунту користувача.

Рисунок 3.6 — Форма реєстрації

Після реєстрації потрібно увійти до свого аккаунту для перегляду статусу пристроїв та керування ними за допомогою голосу. Цю форму можна побачити на рисунку 3.7.

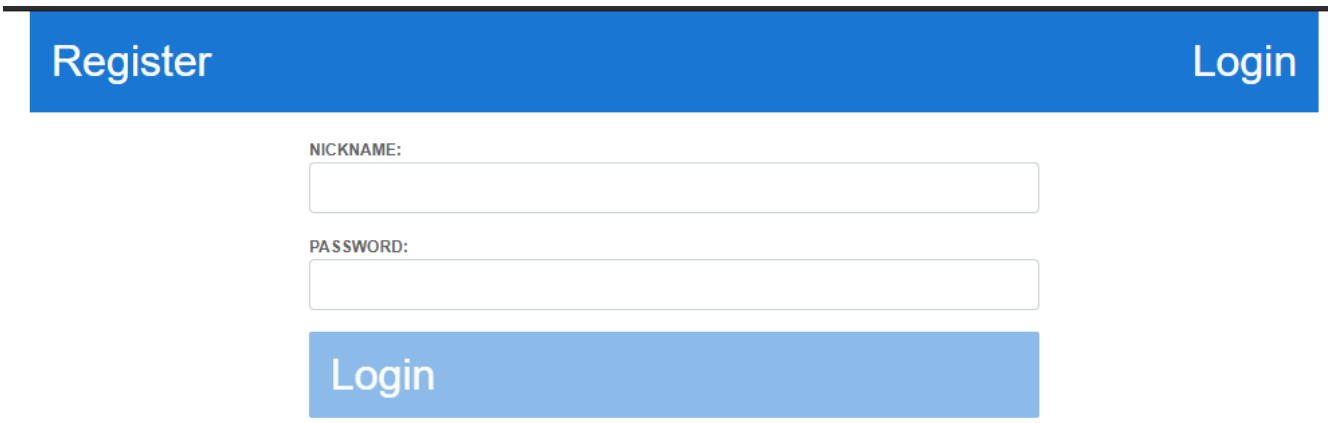

Рисунок 3.7 — Форма авторизації

Після авторизації у користувача з’являється можливість перегляду стану та керування пристроями, дану сторінку можна бачити на рисунку 3.8. Як можна

побачити по червоному колу, йде прослуховування з мікрофона. Воно починається при заході на сторінку зі станом приладів.

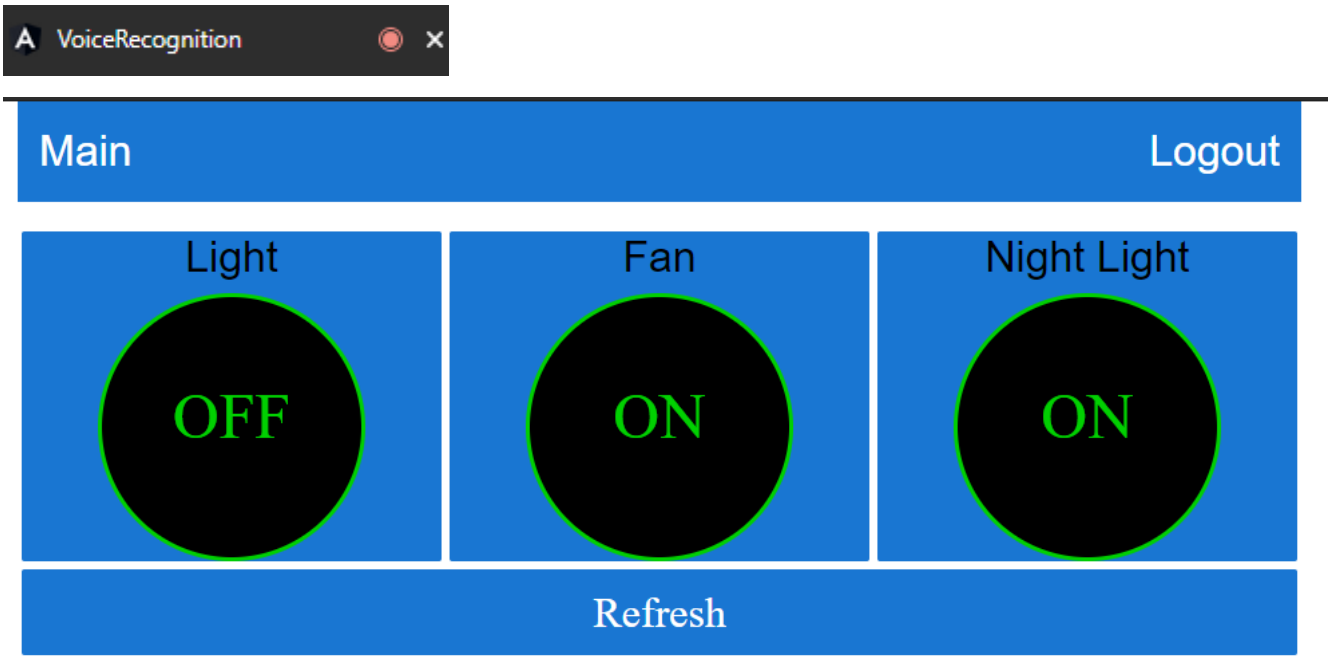

Рисунок 3.8 — Сторінка керування та станів приладів

3.6 Виконання тестування інформаційної системи

Ще одним дуже важливим етапом розробки інформаційної системи в роботі є проведення тестування отриманого програмного продукту.

Тестування програми є процесом дослідження програмного забезпечення з метою отримання інформації про якість продукту і проводиться для забезпечення якості розробленого програмного продукту. В рамках дипломного проєкту передбачається провести тестування розробленого програмного продукту на відповідність вимогам, що пред'являються до нього. Дані вимоги виділяються зі специфікації системи.

Тестовий сценарій складається з тестових випадків. Тестування проводиться з метою виявлення помилок у роботі програмного забезпечення. Це означає, що

результатом перевірки має стати або виявлення та подальше виправлення помилок, або підтвердження повної працездатності програмного продукту.

Критеріями працездатності системи управління розумним будинком будуть:

- відправлення команд, що точно відповідають сказаному людиною, що керує системою за допомогою голосу;
- правильна обробка сервером команд;
- відповідна точна реакція Arduino UNO R3 на команди;

Тестування системи голосового керування буде виконуватися у такий спосіб:

Людина говорить фразу, фраза обробляється бібліотекою annyang.

На рисунку 3.9 можна бачити що SpeechRecognition API повернув масив найбільш можливих значень розпізнаного голосового сигналу.

Рисунок 3.9 — Результат обробки голосового сигналу

Після цього фраза порівнюється зі значеннями елементів заданого масиву, і якщо є співпадіння, то на сервер керування пристроями відправляється відповідна команда. Як можно побачити на рисунках 3.10 та 3.11 команда відправлена успішно.

Рисунок 3.10 — Відповідь сервера

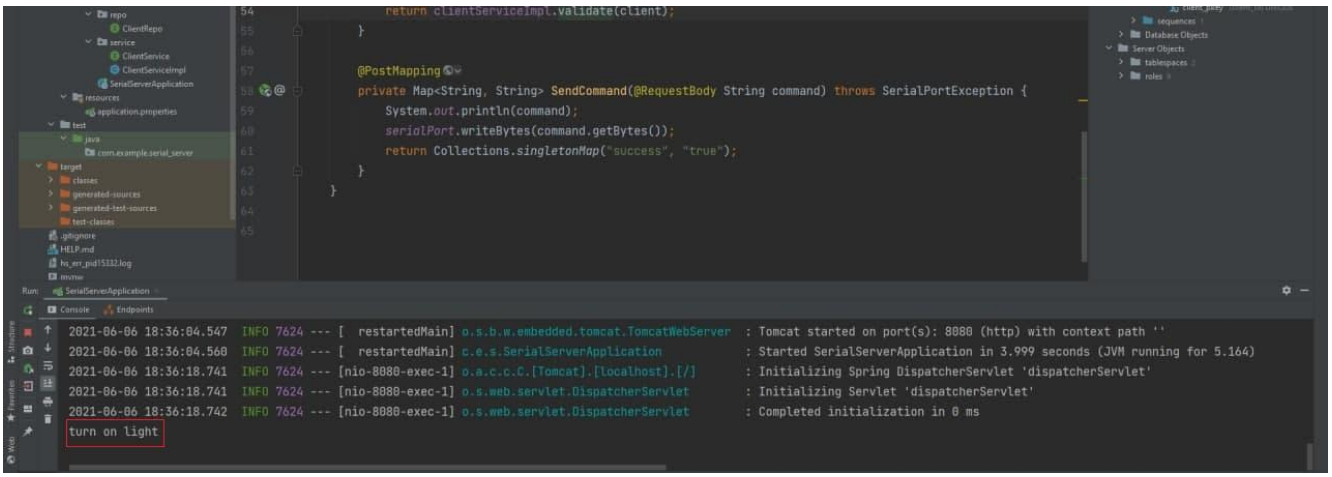

Рисунок 3.11 — Результат роботи контролера

Стан змінився і на сторінці зі статусами приладів. Це можна побачити на рисунку 3.12.

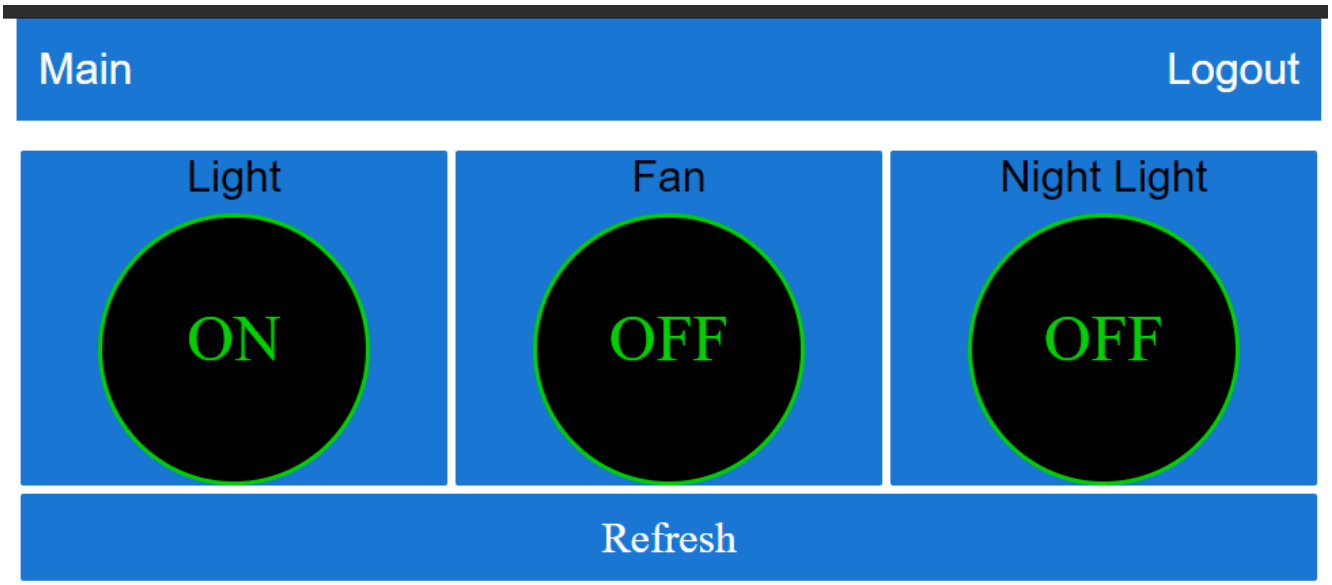

Рисунок 3.12 — Стан приладів після голосової команди

Також відповідні зміни можна побачити в симуляції, що зображено на рисунку 3.13.

Видно, що реле замкнуло ланцюг та лампочка працює, а все інше – ні.

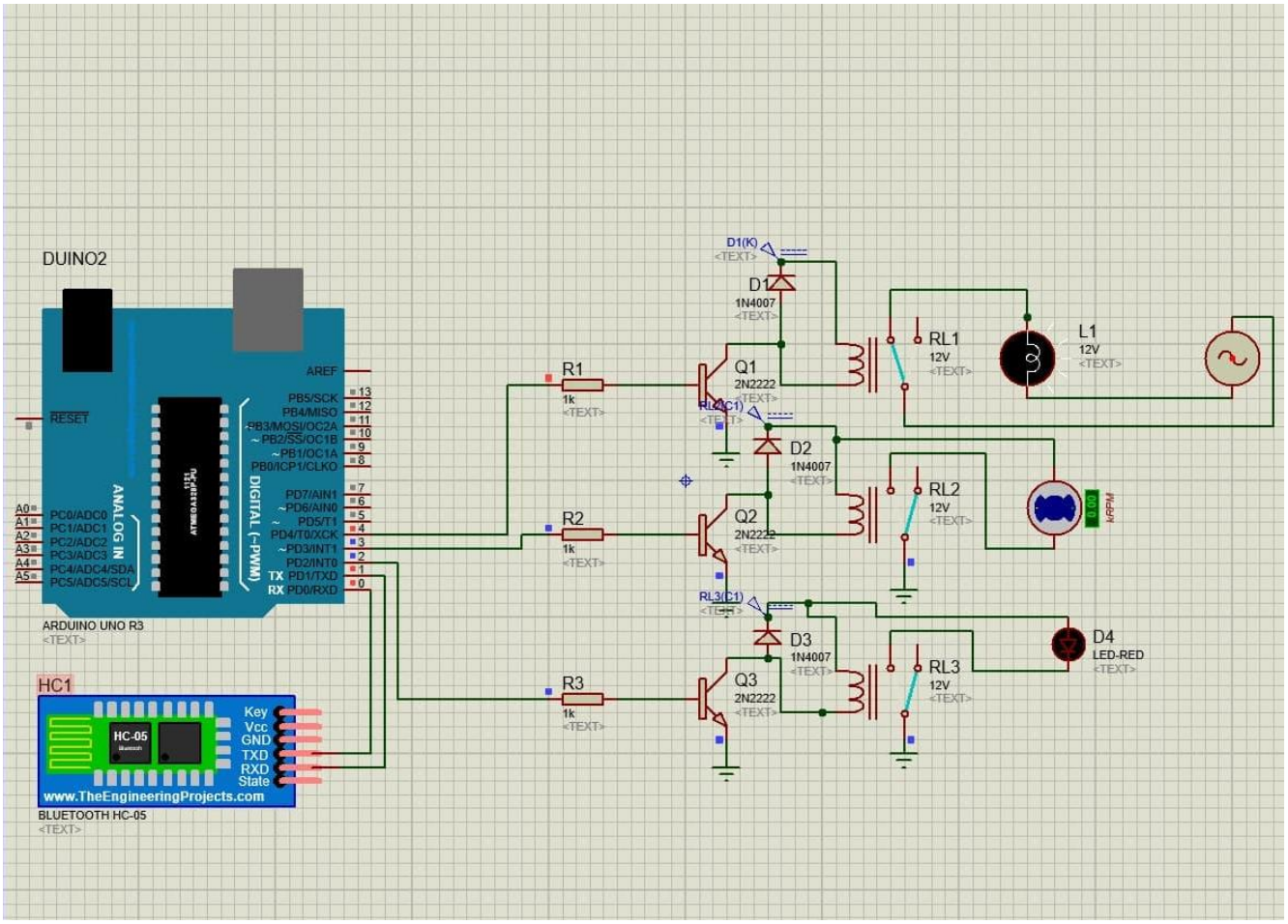

Рисунок 3.13 — Стан схеми після голосової команди.

Зі скріншотів видно, що система успішно виконує поставлені перед нею задачі, всі елементи працюють як задумано.

ВИСНОВКИ

Результатом виконання даної роботи стала повністю функціональна система розумного будинку.

Система складається з веб-застосунку, мікроконтролера Arduino UNO R3, та керованих ним пристроїв.

Веб-застосунок містить в свою чергу frontend та backend.

Frontend написаний на TS та HTML5 із використанням платформи Angular та бібліотеки annyang. Бібліотека annyang використовує вмонтований у браузер SpeechRecognition API для проведення розпізнавання мови на хмарному сервері відповідного браузера та отримання тексту.

Backend реалізовано на мові програмування Java з використанням фреймворку Spring Boot та СУБД Postgresql для збереження користувачів. Веб-додаток реалізовано за парадигмою REST – backend та frontend в даному випадку спілкуються інформацією у форматі JSON. У цьому ПЗ реалізовано процес реєстрації з шифруванням пароля з використанням алгоритмів реалізованих у Spring Security.

Керування Arduino UNO R3 відбувається також за допомогою backend з використанням бібліотеки Java Simple Serial Connector для відправки команд на віртуальний послідовний порт.

Мікроконтролер Arduino UNO R3 із зв’язаними пристроями був реалізований у програмі для створення схем Proteus. Було створено відкриту для масштабування систему, що представляє основні функції розумного будинку із голосовим керуванням. Код для Arduino UNO R3 було написано у середовищі розробки Arduino IDE, що включає в себе керування напругою на виходах, до яких підключено прилади. Зв’язок між сервером та Arduino UNO R3 відбувається через віртуальний послідовний порт, що є дуже зручним при віртуалізації схеми розумного будинку.

Результати, отримані при дипломному проектуванні, повністю відповідають постановці задачі. Таким чином, поставлена мета досягнута.

ПЕРЕЛІК ПОСИЛАНЬ

1. Convolutional Neural Networks for Raw Speech Recognition By Vishal Passricha and Rajesh Kumar Aggarwal [Електронний ресурс]. — Режим доступу: https://www.intechopen.com/books/from-natural-to-artificial-intelligence-algorithms-and-applications/convolutional-neural-networks-for-raw-speech-recognition
2. Bluetooth - A Smart Communication Protocol for the Internet of Things [Електронний ресурс]. — Режим доступу: https://www.cyient.com/blog/communications/bluetooth-a-smart-communication-protocol-for-the-internet-of-things
3. Convolutional Neural Networks [Електронний ресурс]. — Режим доступу: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/
4. Елементи системи розумний будинок, їх призначення та принцип роботи [Електронний ресурс]. — Режим доступу: https://mastery-of-building.org/uk/sostavlyayushhie-elementy-sistemy-umnyj-dom-ix-naznachenie-i-princip-raboty/
5. Audio Deep Learning Made Simple: Automatic Speech Recognition (ASR), How it Works [Електронний ресурс]. — Режим доступу: https://towardsdatascience.com/audio-deep-learning-made-simple-automatic-speech-recognition-asr-how-it-works-716cfce4c706
6. Rabiner L. R. Hidden Markov Models for Speech Recognition — Strengths and Limitations./ L. R. Rabiner, B. H. Juang // Speech Recognition and Understanding. - 1992, pp.3-29. DOI: 10.1007/978-3-642-76626-8_1
7. Mark Gales The Application of Hidden Markov Models in Speech Recognition./ Mark Gales, Steve Young // Foundations and Trends in Signal Processing. — 2007. — Vol. 1, No. 3, pp.195–304. DOI: 10.1561/2000000004
8. 12 лучших систем "Умный дом" [Електронний ресурс]. — Режим доступу: https://simplerule.ru/12-luchshikh-sistem-umnyy-dom/
9. Akhilesh Halageri Speech Recognition using Deep Learning./ Akhilesh Halager, Amrita Bidappa, Arjun C, Madan Mukund Sarathy, Shabana Sultana //

International Journal of Computer Science and Information Technologies. — 2015. — Vol. 6 (3), pp. 3206–3209.

10. Comparing The Best Smart Home Network Protocols in 2019 [Електронний ресурс]. — Режим доступу: https://www.iotforall.com/comparing-best-smart-home-network-protocols-2019
11. Yurika Permanasari Speech recognition using Dynamic Time Warping (DTW)./ Yurika Permanasari, Erwin H. Harahap, Erwin Prayoga Ali // Journal of Physics: Conference Series. — 2019. — Vol.1366, pp.1-6. DOI: 10.1088/1742-6596/1366/1/012091
12. Wouter Gevaert Neural Networks used for Speech Recognition./ Wouter Gevaert, Georgi Tsenov, Valeri Mladenov // Journal of Automatic Control. — 2010. — Vol.20, pp.1-7. DOI:10.2298/JAC1001001G
13. Speech-to-text apps: Microsoft vs Google - which is the best for dictation? [Електронний ресурс]. — Режим доступу: https://www.techradar.com/news/speech-apps-microsoft-vs-Google
14. Записки маковода: обзор охранной системы Ajax [Електронний ресурс]. — Режим доступу: https://gagadget.com/smarthome/24114-zapiski-makovoda-obzor-ohrannoj-sistemyi-ajax/
15. Классификация систем распознавания речи [Електронний ресурс]. — Режим доступу: http://uc.org.ru/node/59
16. Microsoft Azure Speech to Text review [Електронний ресурс]. — Режим доступу: https://www.techradar.com/reviews/microsoft-azure-speech-to-text-review
17. Smart home or building (home automation or domotics) [Електронний ресурс]. — Режим доступу: https://internetofthingsagenda.techtarget.com/definition/smart-home-or-building
18. How Does Speech Recognition Work? Which Algorithm is Used in Speech Recognition? [Електронний ресурс]. — Режим доступу: https://indiantts.com/blog/how-speech-recognition-synthesis-work-which-algorithm-used-voice-recognition/

Додаток А

MainController.java

```
package com.example.serial_server.controller;

import com.example.serial_server.model.Client;
import com.example.serial_server.port.PortReader;
import com.example.serial_server.repo.ClientRepo;
import com.example.serial_server.service.ClientServiceImpl;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.context.event.ApplicationReadyEvent;
import org.springframework.context.event.EventListener;
import org.springframework.web.bind.annotation.*;
import jssc.*;

import java.util.Collections;
import java.util.Map;


@CrossOrigin
@RestController
@RequestMapping("/")
public class MainController {
   static SerialPort serialPort;

   private final ClientServiceImpl clientServiceImpl;
   private final ClientRepo clientRepo;

   @Autowired
   public MainController(ClientServiceImpl clientServiceImpl,
ClientRepo clientRepo) {
       this.clientServiceImpl = clientServiceImpl;
       this.clientRepo = clientRepo;
   }

   @EventListener(ApplicationReadyEvent.class)
   public void OnStartUp() {
       serialPort = new SerialPort("COM4");
       try {
           serialPort.openPort();//Open serial port
           serialPort.setParams(9600, 8, 1, 0);
           PortReader portReader = new PortReader(serialPort);
           serialPort.addEventListener(portReader,
SerialPort.MASK_RXCHAR);
       } catch (SerialPortException ex) {
           System.out.println(ex);
       }

   }
       @PostMapping("register")
       private Client create(@RequestBody Client client) {
```

```
            System.out.println("goose");
            return clientServiceImpl.saveClient(client);
        }

        @PostMapping("login")
        private Map<String, Object> login(@RequestBody Client
client){
            System.out.println("goose");
            return clientServiceImpl.validate(client);
        }

        @PostMapping
        private Map<String, String> SendCommand(@RequestBody String
command) throws SerialPortException {
            System.out.println(command);
            serialPort.writeBytes(command.getBytes());
            return Collections.singletonMap("success", "true");
        }
```

Додаток Б

SmartHouse.ino

```
String command;
void setup()
{
  Serial.begin(9600);
  pinMode(2, OUTPUT);
  pinMode(3, OUTPUT);
  pinMode(4, OUTPUT);
  Serial.println("well done");
}

void loop()
{
  if (Serial.available()){
   command = Serial.readStringUntil('\n');
   command.trim();
   if (command.equals("turn on light")){
     digitalWrite(4, HIGH);
   }
   if (command.equals("turn off light")){
     digitalWrite(4, LOW);
   }
   if (command.equals("turn on fan")){
     digitalWrite(3, HIGH);
   }
   if (command.equals("turn off fan")){
     digitalWrite(3, LOW);
   }
   if (command.equals("turn on night light")){
     digitalWrite(2, HIGH);
   }
   if (command.equals("turn off night light")){
     digitalWrite(2, LOW);
   }
  }
```

Додаток В

devices-page.component.ts

```
import {Component, NgZone, OnDestroy, OnInit} from '@angular/core';
import {HttpClient} from '@angular/common/http';

declare const annyang: any;

@Component({
 selector: 'app-devices-page',
 templateUrl: './devices-page.component.html',
 styleUrls: ['./devices-page.component.css']
})
export class DevicesPageComponent implements OnInit, OnDestroy {

 light: any;
 fan: any;
 nightLight: any;
 constructor(private ngZone: NgZone,
             private http: HttpClient){}

 ngOnInit(): void {
   this.light = localStorage.getItem('light');
   this.fan = localStorage.getItem('fan');
   this.nightLight = localStorage.getItem('nightLight');
   const commands = {
     'turn on light': () => {
       this.light = 'ON';
       this.http.post('http://localhost:8080', 'turn on
light').subscribe(result => {console.log(result); }); },
     'turn off light': () => {
       this.light = 'OFF';
       this.http.post('http://localhost:8080', 'turn off
light').subscribe(result => {console.log(result); }); },
     'turn on fan': () => {
       this.fan = 'ON';
       this.http.post('http://localhost:8080', 'turn on
fan').subscribe(result => {console.log(result); }); },
     'turn off fan': () => {
       this.fan = 'OFF';
       this.http.post('http://localhost:8080', 'turn off
fan').subscribe(result => {console.log(result); }); },
     'turn on night light': () => {
       this.nightLight = 'ON';
       this.http.post('http://localhost:8080', 'turn on night
light').subscribe(result => {console.log(result); }); },
     'turn off night light': () => {
       this.nightLight = 'OFF';
       this.http.post('http://localhost:8080', 'turn off night
light').subscribe(result => {console.log(result); }); }
   };
```

```
    annyang.addCommands(commands);
    annyang.addCallback('resultMatch', (userSaid, commandText,
phrases) => {
      console.log(userSaid);
      console.log(commandText);
      console.log(phrases);
    });

    annyang.start();
  }

  refresh(): any{
      this.light = localStorage.getItem('light');
      this.fan = localStorage.getItem('fan');
      this.nightLight = localStorage.getItem('nightLight');
  }


  ngOnDestroy(): void{
    annyang.abort();
  }
}
```