Міністерство освіти і науки України

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»

Кафедра інформатики факультету інформатики

Розробка проекту "Розумне люстерко"

Текстова частина до курсової роботи

за спеціальністю „Комп'ютерні науки та інформаційні технології"

Керівник курсової роботи

Доцент

Ющенко Ю.О

_____

(підпис)

"____" _____ 2020 р.

Виконав студент

Черниш К.О

"____" _____ 2020 р.

Київ 2020

Міністерство освіти і науки України
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «КИЄВО-МОГИЛЯНСЬКА АКАДЕМІЯ»
Кафедра інформатики факультету інформатики

ЗАТВЕРДЖУЮ
Викладач кафедри інформатики ,
канд. фіз-мат. наук, доц. _____ Гороховський С.С.
(підпис)
„____”_____2020р.

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ
на курсову роботу
студенту Чернишу Кірілу Олександровичу
факультету інформатики 4 курсу бакалаврської програми
ТЕМА: **Розробка проекту "Розумне люстерко"**

Зміст ТЧ до курсової роботи:
- Індивідуальне завдання
- Вступ
- Огляд існуючих рішень та опис реалізації проекту
- Висновки

Список літератури
Список посилань

Дата видачі „___” _____ 2020 р. Керівник_____
(підпис)
Завдання отримав _____
(підпис)

Тема: Розробка проекту "Розумне люстерко"

Календарний план виконання роботи:

№	Назва етапу	Термін виконання	Примітка
1.	Отримання теми курсової роботи	07.11.2019	
2.	Пошук тематичної літератури	09.01.2020	
3.	Ознайомлення з літературою	11.01.2020	
4.	Вивчення методів аналізу даних	11.01.2020	
5.	Огляд технологій для проведення кореляційного аналізу	15.01.2020	
6.	Написання умов до програмної частини проекту	20.02.2020	
8.	Написання умов до клієнтської частини проекту	25.02.2020	
10.	Написання першої частини курсової роботи	10.03.2020	
11.	Написання другої частини курсової роботи	21.03.2020	
12.	Перегляд змісту роботи з керівником	10.04.2020	
13.	Написання третьої частини курсової роботи	12.04.2020	
14.	Перегляд змісту роботи з керівником	12.04.2020	
15.	Внесення змін до курсової роботи відповідно до зауважень наукового керівника	17.04.2020	
16.	Створення презентації	18.04.2020	
17.	Захист роботи	До 24.04.2020	

Зміст

Вступ

Однією з новітніх парадигм сучасного світу є Інтернет речей (Internet of Things). **Інтернет речей (IoT)** — це програмно-технічна система взаємопов'язаних у мережу електронних пристроїв, які керуються програмами через Інтернет задля визволення людей та допомоги їм або при вчинені ними при розв'язку задач як у промисловості, побуті так і у дозвіллі. Ця парадигма швидко набирає популярності в системі сучасних бездротових телекомунікацій з очікуваним зростанням від 25 до 50 мільярдів підключених пристроїв до кінця 2020 року. Ця тенденція стала можливою завдяки постійній мініатюризації та підвищенню обчислювальної потужності електронних пристроїв та розвитку засобів програмування, зокрема засобів штучного інтелекту.

IoT розповсюджується по всіх галузях, створюючи нові можливості для інновацій, трансформацій та вдосконалення ланцюга вартості. Щорічно IoT набуває все більше рис конкурентоспроможної зброї на ринку основної вигоди. Такі галузі як транспорт, розумне місто, розумний дім, електронне здоров'я, електронне врядування, соціальний захист, електронна освіта, роздрібна торгівля, логістика, сільське господарство, промислове виробництво, щорічно отримують значні переваги від IoT. В центрі IoT знаходяться розумні пристрої. **Розумний пристрій** — це електронний пристрій, що усвідомлює контекст та здатний виконувати автономні обчислення й підключатися до інших пристроїв дротом або бездротовим пристроєм для обміну даними. З цього визначення випливають **три основні характеристики розумного пристрою**, а саме:

1) усвідомлення контексту;

2) автономні обчислення;
3) підключення.

Усвідомлення контексту

Усвідомлення контексту — це здатність системи чи компонента системи в будь-який момент збирати інформацію про своє оточення та відповідно адаптувати поведінку. Камери, мікрофони та приймачі глобального супутникового позиціонування (GPS), радіолокатори та різноманітні датчики — усі потенційні джерела даних для обчислювальних ситуацій. Система, що усвідомлює контекст, може збирати дані через ці та інші джерела та реагувати за встановленими правилами або за допомогою обчислювальної розвідки.

Автономні обчислення

Основним аспектом автономних обчислень є пристрій або кілька пристроїв, які виконують завдання автономно без прямої команди користувача. Наприклад, коли наші смартфони роблять пропозицію на основі геолокації чи погоди. Для виконання цього "простого"; завдання смартфон повинен бути автономним і використовувати контекстні дані для прийняття рішень.

Підключення

Підключення означає здатність розумного пристрою підключатися до мережі передачі даних. Без підключення немає сенсу розумному пристрою бути автономним і мати усвідомлення контексту. Підключення до мережі є важливою особливістю, яка дозволяє пристрою бути частиною Інтернету речей. Підключення до мережі може бути дротовим або бездротовим.

Основним результатом курсової роботи є розробка моделі розумного пристрою — розумного люстерка. Курсова робота складається з трьох розділів. У першому розділі обґрунтовано актуальність задач. У другому розділі проведено аналіз технологій та інструментів, а також обґрунтовано їх вибір для реалізації. Третій розділ розкриває технічні характеристики програмного забезпечення та особливості його реалізації і функцюювання.

Розділ 1. Обґрунтування актуальності

1.1. Постановка задачі

У сучасному світі інформація знаходиться завжди поруч з людиною — у телефоні, комп'ютері, навіть у годиннику. Потреба постійно отримувати інформацію стає важливою у повсякденному житті людини. При великій різноманітності засобів передачі інформації, котрі оточують людину, виникають труднощі в утриманні всіх потоків даних, які спрямовуються на неї.

Протягом дня людина періодично опиняється в ситуації, коли не має можливості подивитися в екран комп'ютера або навіть у мобільний телефон аби перевірити останні оновлення. Проте залишається потреба в отриманні інформації. Саме для цього потрібен монітор, в який можна подивитися аби дізнатися необхідну інформацію. При цьому естетика так само важлива, як й відображення інформації. Розміщувати додатковий комп'ютер, наприклад, у ванній кімнаті не є зручним та естетичним рішенням. Елегантний дисплей, простий для пересічного споживача є новою необхідністю сьогодення.

1.2. Шляхи вирішення

Дзеркало зазвичай сприймається як традиційний предмет побуту в нашій повсякденній рутині. Однак насправді це один із продуктів, який може мати високу функціональність завдяки застосуванню сучасних інформаційних технологій. Дзеркало знаходиться у фокусі уваги людини майже кожного дня, при цьому фізичний контакт між цим предметом та людиною відбувається дуже рідко. Ця особливість робить дзеркало надзвичайно простим та зручним персональним асистентом та центром для відображення важливої інформації.

Завдяки поєднанню традиційного дзеркала із сучасними технологіями з’явився новий розумний пристрій — розумне дзеркало.

1.3. Визначення розумного дзеркала

Розумне дзеркало — це двостороннє дзеркало з електронним дисплеєм за склом. Його мета — за допомогою вбудованого штучного інтелекту поєднати повсякденні заняття людини, наприклад, перегляд новин, перевірку прогнозу погоди із часом, коли людина дивиться на себе у дзеркало, та відобразити всі необхідні їй дані відповідно до контексту та голосових команд.

Розумне дзеркало — це черговий крок в автоматизації життєдіяльності людини. На дисплеї розумного дзеркала може відображатися все, що завгодно, наприклад, поточний час, прогноз погоди, стрічка новин, майбутні зустрічі тощо. Розумне дзеркало можна легко налаштувати, аби воно містило ту інформацію, яку потребує його власник. Розумні дзеркала можуть бути представлені у різних форматах: від маленьких люстерок для приліжкових тумб, до великих дзеркал для ванних кімнат. Розумні дзеркала іноді називають «магічними» дзеркалами, але обидва терміни мають один той самий функціонал. Окрім помешкання людини, розумні дзеркала можна використовувати в різних сферах бізнесу, таких як: магазини одягу, салонах краси, перукарнях, кіосках та інших місцях. Але важливо, що це не просто інтерактивний пристрій, який комунікує з людиною, а штучний інтелект, який доносить їй інформацію в залежності від місця, де людина знаходиться та голосових команд людини.

1.4. Основні принципи роботи розумного дзеркала

Будь-яке розумне дзеркало містить три основних компоненти: двостороннє дзеркало, монітор та комп'ютерний пристрій. Розглянемо їх.

Двостороннє дзеркало

Звичайне дзеркало містить плівку за склом, яка відбиває 100% надходить світла, що надходить. Це означає, що дивлячись у дзеркало, людина бачить своє відображення. Двостороннє дзеркало відбиває світло з одного напрямку, але при цьому дозволяє пропускати світло з іншого. У кіно можна нерідко побачити як двосторонні дзеркала використовуються поліцейськими в кімнатах для допитів. Розумні дзеркала використовують двосторонні дзеркала для того, щоб світло з дисплея проходило через дзеркало. Таким чином людина бачить у дзеркалі не тільки своє відображення, але й приглушене зображення з монітора, що знаходиться за дзеркалом.

Монітор

За двостороннім дзеркалом зазвичай знаходиться екран монітора, або телевізор, або планшет. Вони використовуються для відображення будь-якої інформації на поверхню розумного дзеркала. Дисплей може бути такого ж розміру, як дзеркало, або він може бути меншим, ніж дзеркало.

Комп'ютер

Для відображення інформації на моніторі використовуються різноманітні комп'ютерні пристрої. Тип комп'ютерного пристрою залежить від задач, що покладаються на розумне дзеркало.

1.5. Аналіз існуючих аналогів

В останні роки конкуренція на ринку розумних віртуальних асистентів (intelligent virtual assistant (IVA) or intelligent personal assistant (IPA)) загострилась і всі великі гравці пропонують свої рішення. Компанія Apple першою вивела на широкий ринок голосового асистента Siri, який є програмним компонентом смартфонів цієї компанії. Натомість компанія Amazon була піонером у сфері окремих пристроїв категорії “домашній асистент” з серією Amazon Echo та голосовим помічником Alexa. Такі гіганти як Google та Xiaomi також пропонують рішення в цій сфері. Жоден з великих гравців на ринку домашньої автоматизації не пропонує рішення типу “розумне дзеркало”. Пропозиції в ніші розумних дзеркал в основному обмежуються наборами для самостійної зборки (DIY sets) та пропозиціями декількох невеликих компаній, таких як Electric Mirror (https://www.electricmirror.com/smart-mirror-savvy/), Keonjinn (https://www.amazon.com/stores/page/75CFE068-B0AA-4749-B87E-1C598152A3CE) та інших, в основному китайських виробників.

Розділ 2. Огляд підходів та засобів розробки Робота над моделлю розумного люстерка складалася з розробки вимог до технічного рішення, а також аналізу системи побудови розумного дерева (її апаратної та програмної складових).

2.1. Визначення вимог до проєкту

В процесі роботи над проєктом було розроблено три види вимог до технічного рішення: функціональні, нефункціональні та конструктивні обмеження. Розглянемо їх.

Функціональні вимоги

1. Система повинна бути здатною відображати інформацію на екрані.
2. Система повинна реагувати на голосові команди користувача.
3. Система повинна підключатися до мережі інтернет для отримання вхідних даних.
4. Система повинна мати модульну структуру.
5. Система повинна мати можливість легко розширювати функціональні можливості.

Нефункціональні вимоги

1. Система повинна мати більш простий інтерфейс користувача, ніж комп'ютер.
2. Система повинна мати високі показники енергоефективності.
3. Система повинна мати компактні розміри.
4. Система повинна мати низький рівень шуму при роботі.
5. Користувач повинен мати змогу користуватись системою як звичайним дзеркалом.

Конструктивні обмеження

1. Рішення повинно відповідати стандартам популярних інтерфейсів.
2. Рішення повинно бути відкритим.
3. Система повинна бути спроектованою з підтримкою легкого розширювання можливостей.

З огляду на визначені вимоги, було проаналізовано та обрано апаратні та програмні рішення для побудови системи.

2.2. Аналіз системи побудови розумного дзеркала

Система побудови розумного дзеркала розглядається з двох аспектів: апаратного та програмного.

2.2.1. Опис апаратної складової

Апаратна складова розумного дзеркала представлена на рисунку 1.

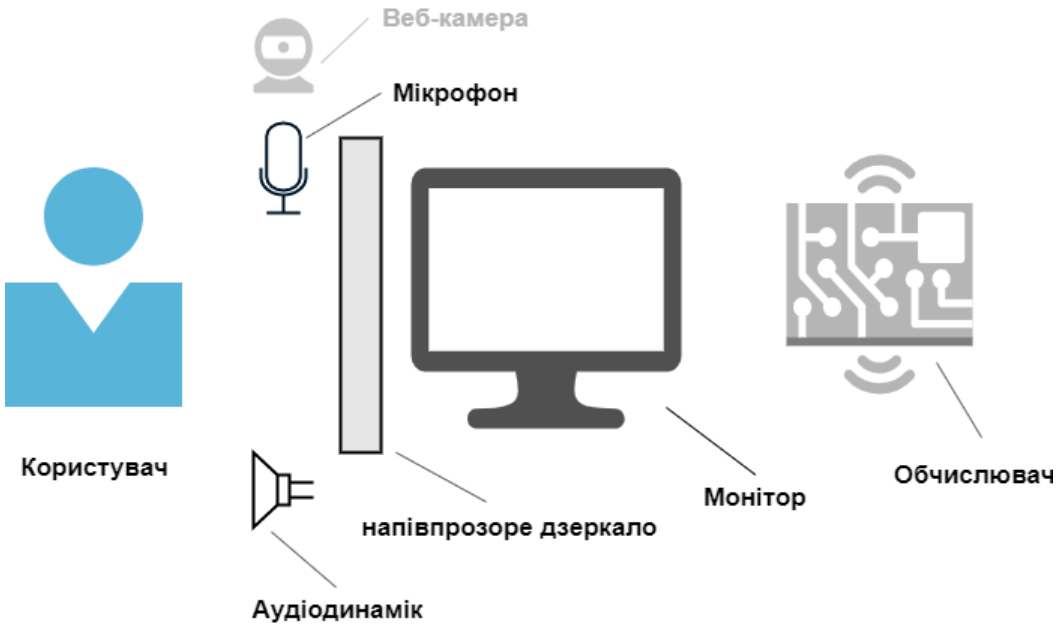

Рис. 1. ***Структура апаратної частини розумного дзеркала***

Користувач отримує інформацію від системи візуально через монітор, встановлений за напівпрозорим дзеркалом та аудіально через динаміки.

Система отримує інформацію від користувача через мікрофон та через веб-камеру (потенційно). Вхідна та вихідна інформація зберігається, обробляється та генерується на обчислювачі. Проаналізуємо складові зазначеної системи.

Дзеркало

Напівпрозоре дзеркало або дзеркало Гезелла представляє собою скло, покрите тонким шаром металу таким чином, що частина падаючого на поверхню скла світла відбивається, а частина проходить наскрізь. При цьому наскрізь світло проходить в обох напрямках. Якщо за таким склом розмістити джерело світла, яке буде випромінювати інтенсивніше світло за світло з передньої частини скла, то таке джерело буде видно через скло. Ті частини скла за якими світло, що випромінюється буде менш яскравими, будуть відбивати світло з передньої частини дзеркала. Такі дзеркала виробляються, в тому числі, і в Україні.

Мікрофон

В якості мікрофона можна використовувати практично будь-який недорогий популярний мікрофон, який можна під’єднати через 3.5мм мініджек роз’єм.

Веб-камера

Для цього проекта веб-камера є додатковим пристроєм, з розрахунком на подальше розширення функцій розумного дзеркала. Веб-камера має під’єднуватись через USB інтерфейс та мати підтримку роботи з боку популярних операційних систем.

Аудіодинамік

Аудіодинаміки мають підключатися через через 3.5мм мініджек роз’єм, мати невеликі розміри та не вимагати додаткового живлення.

Монітор

В якості монітора краще використовувати сучасні LCD монітори з LED підсвіткою через їх достатню яскравість, невисоку вагу та доволі непогану енергоекономічність.

Обчислювач

В якості обчислювача нами розглядалося 3 варіанти:

- персональний компьютер в невеликому корпусі або ноутбук,
- мікрокомп’ютер типу Raspberry Pi,
- система на мікроконтроллері типу Arduino.

Для кожного з них було проаналізовано переваги та недоліки.

Персональний комп’ютер.

Переваги:

- висока обчислювальна потужність,
- простота підключення монітора, мікрофона, динаміків та вебкамери.

Недоліки:

- висока ціна,
- громіздкість,
- високий рівень шуму при роботі через вентилятори,
- відносно висока кількість споживання електроенергії,
- складність підключення додаткових датчиків.

Мікрокомп’ютер типу Raspberry Pi.

Переваги:

- низька ціна,

- компактний розмір,
- достатня обчислювальна потужність,
- простота підключення монітора, мікрофона, динаміків та вебкамери,
- низький рівень споживання електроенергії,
- легкість підключення додаткових датчиків,
- низький рівень шуму.

Недоліки:

- перегрівається при великому обсязі обчислення.

Мікроконтролер типу Arduino.

Переваги:

- низька ціна,
- компактний розмір,
- легкість підключення додаткових датчиків,
- низьких рівень шуму,
- низький рівень споживання електроенергії.

Недоліки:

- недостатня обчислювальна потужність,
- складність підключення монітору, вебкамери.

Головним недоліком мікроконтролерів типу Arduino є недостатня обчислювальна потужність та обмеженість системи графічного виводу інформації. Arduino можна використовувати для виводу інформації на невеликі дисплеї схожі на алфавітно-цифрові дисплеї старих мобільних телефонів, або вбудованих в побутову техніку, але не на комп’ютерні монітори.

Персональний комп’ютер в якості обчислювача може створювати дискомфорт для користувача через постійний або майже постійний шум вентиляторів системи охолодження та відносно великий фізичний розмір. Відносно велике споживання електроенергії та складність підключення додаткових датчиків для розширення функціональності розумного дзеркала, є негативними факторами при виборі персонального комп’ютера в якості платформи обчислювача.

В якості найбільш оптимального варіанту для обчислювача нами було обрано мікрокомп’ютер Raspberry Pi. Raspberry Pi — це серія невеликих одноплатних комп'ютерів, розроблених у Сполученому Королівстві Фондом Raspberry Pi для сприяння викладанню основних комп'ютерних наук у школах та країнах, що розвиваються. Оригінальна модель стала набагато популярнішою, ніж передбачалося, продаючи поза її цільовим ринком для таких видів використання, як робототехніка. Випущено кілька поколінь Raspberry Pi. Всі моделі обладнані системою Broadcom на мікросхемі (SoC) із вбудованим центральним процесорним процесором (CPU), сумісним з ARM, та графічним процесором графічної обробки (GPU).

Швидкість процесора коливається від 700 МГц до 1,4 ГГц для моделі Pi 3 моделі B + або 1,5 ГГц для Pi 4; вбудована пам'ять становить від 256 Мбіт до 1 Гб пам'яті з випадковим доступом (ОЗП), до 4 ГБ, доступних на Pi 4. Захищені цифрові (SD) карти в форм-факторі MicroSDHC (SDHC на ранніх моделях) використовуються для зберігання операційна система та пам'ять програми. Плати мають один-п’ять портів USB. Для відеовиходу підтримується HDMI та композитне відео, із стандартним 3,5-мм роз'ємним кінцевим рукавом для аудіовиходу. Вихід нижнього рівня забезпечується декількома штифтами GPIO, які підтримують поширені протоколи, такі як I^2C.

Моделі B мають порт Ethernet 8P8C, а Pi 3, Pi 4 і Pi Zero W мають вбудований Wi-Fi 802.11n і Bluetooth.

2.2.2. Опис програмної складової

Програмна складова системи розумного дзеркала побудована на основі мікросервісів з загальною системою менеджменту подій. Мікросервіси — це архітектурний стиль розробки програмного забезпечення, за яким програмний продукт будується як набір невеликих сервісів, кожен з яких виконує свою специфічну задачу і комунікує з рештою мікросервісів за допомогою легких протоколів.

У проєктній моделі мікросервіси не комунікують напряму між собою. Натомість вони використовують загальну систему менеджменту подій і можуть або чекати на подію, або публіковати подію.

Загальна схема програмної складової представлена на рисунку 2:

Рис. 2. ***Структура програмної складової розумного дзеркала***

Загальна система менеджменту подій

Забезпечує отримання подій та даних в каналах від компонентів, що їх публікують, а також зберігання та передачу цієї інформації компонентам, що підписані на відповідні канали.

Модуль розпізнавання голосу

Забезпечує перетворення аудіо інформації, отриманої з мікрофону, в текстову.

Модуль трансляції команд

Забезпечує перетворення текстової інформації з природної форми в набір команд, зрозумілих модулям системи, які мають виконати задачу, визначену користувачем.

Візуальний інтерфейс користувача

Забезпечує генерацію та вивід візуального контента відповідно до контексту та отриманих намірів (intents) користувача.

2.2.3. Обґрунтування засобів реалізації модулів

Нижче представлено обгрунтування вибору засобів реалізації модулів, які були використані нами при розробці моделі розумного дзеркала.

Модуль загального менеджменту подій

В якості загальної системи менеджменту подій у проєкті використовується Redis. Redis це сховище пар ключ-значення, які зберігаються в оперативній пам’яті. Це програмне забезпечення з відкритим

кодом. Для реалізації розумного дзеркала важлива швидкість, стабільність та відсутність високих вимог до ресурсів. Однією з функціональних можливостей Redis є підтримка шаблона проектування Публікація-підписка. Саме ця особливість є найбільш цінною в Redis в контексті даного проекта. Redis забезпечує механізм передачі повідомлень, в якому видавці (publishers) публікують повідомлення не знаючи про те, які підписники (subscribers) їх отримують. Видавці та підписники використовують іменовані логічні канали. Таким чином, забезпечується фільтрація повідомлень.

Модуль розпізнавання мови

В якості модуля розпізнавання мови обрано бібліотеку SpeechRecognition. Це бібліотека написана мовою Python, яка забезпечує отримання аудіо сигналу з мікрофона та перетворення цього сигналу в текст. При цьому може бути вказано декілька можливих результатів кожного перетворення через те, що мова користувача може бути нечітка, можуть бути завади та шуми тощо.

Модуль SpeechRecognition для безпосереднього перетворення аудіо в текст може використовувати декілька провайдерів, таких як:

- Microsoft Bing Speech,
- Google Web Speech API,
- Google Cloud Speech,
- Houndify,
- IBM Speech to Text,
- CMU Spinx,
- Wit.ai.

Наразі тільки Гугл пропонує розпізнавання української мови. Google Cloud Speech має обмеження безкоштовного доступу, тому для реалізації

розумного дзеркала був обраний Google Web Speech API. Є сподівання, що незабаром підтримка української мови буде доступна в проектах, таких як Mozilla deepSpeech, що можуть виконувати перетворення аудіо в текст безпосередньо на обчислювальній платформі, а не через інтернет в корпоративних хмарних середовищах.

Модуль трансляції команд

У ході роботи над проєктом не було знайдено доступних рішень з підтримкою нейролінгвістичного програмування NLP (Neuro-linguistic programming) для української мови.

Тому цей модуль було реалізовано як додаток на JavaScript, що виконується в середовищі Node.js. Додаток отримує через систему загального менеджменту подій варіанти розпізнаного тексту та через пошук ключових слів намагається транслювати цю інформацію в команди, зрозумілі системі для виконання візуальним інтерфейсом. У майбутньому застосунок, що матиме кращі результати у трансляції команд, можна буде легко інтегрувати в систему завдяки модульній побудові.

Було обрано рішення використовувати Node.js через те, що це крос-платформне середовище виконання JavaScript, і є можливість працювати з інтерфейсом користувача за допомогою API. Також Node.js має такі переваги:

- висока масштабованість,
- швидкість,
- відсутність буферизації,
- однопотоковість,
- асинхронність.

Візуальний інтерфейс користувача

Візуальний інтерфейс реалізовано у вигляді веб сервера та додатку, що виконується в веб браузері. Веб сервер реалізовано як застосунок на JavaScript, що виконується в середовищі Node.js. Використовується пакет Express, який є де факто стандартом при організації веб сервера на Node.Js.

Оновлена інформація для виконання передається в додаток, в веб браузер з веб сервера використовуючи Push API.

Зазвичай ініціація запиту відбувається клієнтом. Push API це стандарт, що дозволяє серверу ініціювати надсилання повідомлення на клієнт.

Для побудови веб клієнта, який безпосередньо виконує трансльовані команди від користувача та відображає інформацію було розглянуто 3 фреймворка:

- React,
- Angular,
- Vue.

Для кожного з них нами було проаналізовано переваги на недоліки.

Angular

Переваги:

- Нові функції, такі як поліпшений RXJS, прискорена компіляція і новий лаунчер HttpClient.
- Детальна документація, яка дозволяє кожному розробнику отримати всю необхідну інформацію без звернення за допомогою до колег.
- Двостороння прив'язка даних, яка забезпечує чудову поведінку додатка, що мінімізує ризики можливих помилок.

- MVVM (Model-View-ViewModel), яка дозволяє розробникам окремо працювати в одному розділі програми з використанням одного і того ж набору даних.
- Впровадження залежностей функцій пов'язаних з компонентами з модулями і модульності в цілому.

Недоліки:

- Складний синтаксис.
- Проблеми з міграцією, які можуть виникнути при переході від старої версії до нової.

React

Переваги:

- Легкий у вивченні.
- Високий рівень гнучкості і максимальна чуйність.
- Віртуальна DOM (document object model), яка дозволяє впорядковувати документи форматів HTML, XHTML або XML в дерево, яке найкраще підходить веб-браузерів для аналізу різних елементів веб-додатки.
- У поєднанні з ES6 / 7 ReactJS може легко працювати при високих навантаженнях.
- Зв'язування даних від великих до менших.
- 100% JavaScript-бібліотека з відкритим вихідним кодом, яка отримує безліч щоденних оновлень і поліпшень відповідно до відгуків розробників по всьому світу.
- Неймовірно легка вага, так як дані, які виконуються на стороні користувача, можуть легко бути представлені на стороні сервера в той же самий час.
- Міграція між версіями, як правило, дуже проста.

Недоліки:

- Брак офіційної документації.
- React не має чіткої мети. Це означає, що розробники, іноді, мають занадто великий вибір.
- Довгий час для освоєння.

Vue

Переваги:

- Посилений HTML.
- Детальна документація.
- Адаптивність.
- Чудова інтеграція.
- Велике масштабування. Vue.js допомагає розробляти досить великі шаблони для багаторазового використання, які можна розробити без витрачання величезної кількості часу на увазі простий структури.
- Крихітний розмір.

Недоліки:

- Брак ресурсів. Vue.js як і раніше має досить невелику частку ринку в порівнянні з React або Angular. Це означає, що обмін знаннями в рамках фреймворка все ще формується.
- Ризик надмірної гнучкості. Іноді у Vue.js можуть виникати проблеми при інтеграції в величезні проекти, а досвіду про можливі рішення до сих пір немає.
- Відсутність повної англомовної документації.

Найкращим варіантом для візуального інтерфейсу користувача ми вбачаємо React. React — це декларативна, ефективна та гнучка бібліотека JavaScript для створення інтерфейсів користувача. Він дозволяє складати складні інтерфейси користувача з невеликих та ізольованих фрагментів коду під назвою «компоненти».

Розділ 3. Реалізація моделі розумного дзеркала

Схема моделі розумного дерева, що була розроблена та реалізована нами, представлена на рисунку 4:

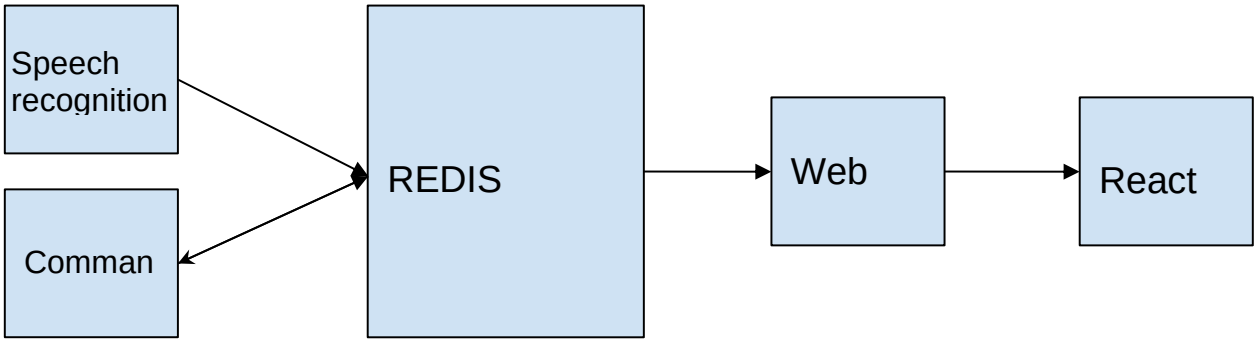

Рис. 4. Схема проєкту

Опис модуля Speech Recognition

Задачі модуля: розпізнавання голосу з мікрофону і перетворення голосу в текст. Модуль використовує бібліотеку SpeechRecognition написану на мові Python https://pypi.org/project/SpeechRecognition/.

Для встановлення модуля в ОС Ubuntu треба зробити наступне:

```
pip install SpeechRecognition
```

Також необхідно встановити додаткові застосунки для отримання аудіо сигналу з мікрофону:

```
sudo apt-get install python-pyaudio python3-pyaudio
```

Код модуля:

```
import speech_recognition as sr
import redis
import json

r = redis.Redis(host="localhost", port=6379, db=0)

def recognize_speech_from_mic(recognizer, microphone):
```

```
    if not isinstance(recognizer, sr.Recognizer):
        raise TypeError("`recognizer` must be `Recognizer` instance")

    if not isinstance(microphone, sr.Microphone):
        raise TypeError("`microphone` must be `Microphone` instance")

    with microphone as source:
        recognizer.adjust_for_ambient_noise(source)
        audio = recognizer.listen(source)

    response = {
        "success": True,
        "error": None,
        "transcription": None
    }

    try:
        response["transcription"] =
recognizer.recognize_google(audio, show_all=True, language="uk-UA")
    except sr.RequestError:
        response["success"] = False
        response["error"] = "API unavailable"
    except sr.UnknownValueError:
        response["error"] = "Unable to recognize speech"

    return response

if __name__ == "__main__":

    recognizer = sr.Recognizer()
    microphone = sr.Microphone()
    while True :
        guess = recognize_speech_from_mic(recognizer, microphone)
        r.publish("lyusterko_speech_recognition",json.dumps(guess))
        print("published to redis")
```

SpeechRecognition підтримує різні провайдери перетворення голосу в текст. У роботі ми використовуємо Гугл в якості провайдера (а саме Google Web Speech Recognition), оскільки в Гугл (як було зазначено у другому розділі) є підтримка української мови.

При використанні Google Web Speech Recognition є можливість отримати список можливих результатів перетворення. Скориставшись цією опцією ми отримуємо на виході модуля розпізнавання мови об’єкт у форматі JSON подібний до наступного:

```
{
  alternative: [
    { confidence: 0.68273622, transcript: 'надобраніч' },
    { transcript: 'На добраніч' },
    { transcript: 'добраніч' }
  ],
  final: true
}
```

Аналізуючи всі можливі результати перетворення ми отримуємо можливість підвищити якість роботи системи.

Об’єкт з результатами розпізнавання серіалізується в строку та передається в канал загальної системи менеджменту подій.

Опис модуля Command Interpreter

Модуль перетворення команд очікує на повідомлення в каналі загальної системи менеджменту подій, читає повідомлення та намагається перетворити інформацію цього повідомлення в команду для модуля відображення інформації.

Нижче представлений код для розпізнавання команди відображення календаря подій

Код модуля:

```
const redis = require( "redis" );

const subscriber = redis.createClient(6379,"192.168.0.250");
```

```
const publisher = redis.createClient(6379,"192.168.0.250");

let findWord = ( word, phrases ) => {

    for ( let element of phrases )
    {
        if ( element.indexOf( word ) > -1 )
        {
            return true
        }
    }

    return false;
}

subscriber.on( "message", (channel, message) => {
    let sr = JSON.parse( message );
    console.log( sr.transcription );

    if ( sr.transcription && sr.transcription.alternative )
    {
        let phrases = sr.transcription.alternative.map( a =>
a.transcript.toLowerCase() );
        let command;
        let payload = {}
        if ( findWord( "подія", phrases ) || findWord( "події",
phrases ) )
        {
            command = "show_events";
        }
        else
        {
            command = "not_recognized";
            payload = phrases;
        }

        publisher.publish( "lyusterko_commands", JSON.stringify( {
command: command, payload: payload } ) );
    }
})

subscriber.subscribe( "lyusterko_speech_recognition" );
```

Якщо розпізнавання команди відбулось успішно, відповідне повідомлення записується в інший канал загальної системи менеджменту повідомлень. На повідомлення в цьому каналі очікує система відображення інформації.

Опис модуля WebAPI server

Цей модуль очікує повідомлення від модуля трансформації команд. Як було зазначено вище та, за допомоги технології WebPush передає команду в код, що виконується в веб браузері.

```
const redis = require( "redis" );
const express = require( "express" );
const webpush = require( "web-push" );

const app = express();
const subscriber = redis.createClient(6379,"192.168.0.250");

const publicVapidKey = "BBrmMAY9bsvBSPFyH2-bgh5ptD0DP-
8q0fUo_CFr7igTrBvTa_WvnNuFs8ieMSEcLbbo7NjuytdePl3KtChWMV0";
const privateVapidKey =
"s0UjlPYJlAzLmDbMJCnt4W0XgSBOQyUP2Q24ttpW2og";

subscriber.subscribe( "lyusterko_commands" );

webpush.setVapidDetails( "mailto:kiril.chernysh@gmail.com",
publicVapidKey, privateVapidKey );

app.use( require( "body-parser" ).json() );

app.post( "/subscribe", ( request, responce ) => {
    const subscription = request.body;
    responce.status( 201 ).json( {} );
    const payload = JSON.stringify( {title: "test" } );
    console.log( subscription );

    webpush.sendNotification( subscription, payload ).catch( error =>
{
        console.error( error.stack );
    } );
```

```
    subscriber.on( "message", (channel, message) => {
        console.log( JSON.parse(message) );
        webpush.sendNotification( subscription, message ).catch(
error => {
            console.error( error.stack );
        } );
    })
} );

app.listen( 3000, () => console.log( "API server listening on 3000" )
);
```

Опис реалізації візуального інтерфейсу користувача

Інтерфейс календаря (рис.5) розроблено на React.js та стилізовано за допомогою css. Для побудови календаря було застосовано npm пакет "*react-big-calendar*". React-big-calendar — це компонент календаря подій,побудований для React та створений для сучасних браузерів. і

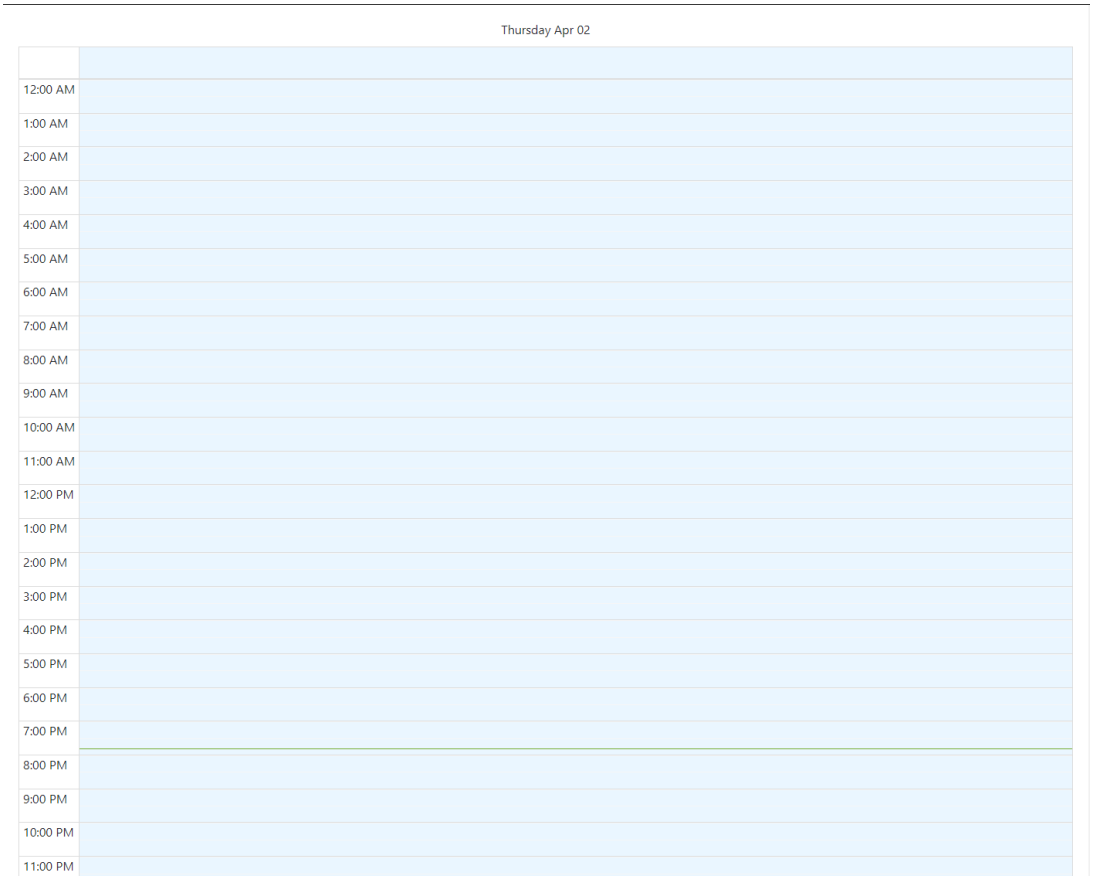

Рис. 5. Інтерфейс календаря

Код реалізації:

```
import React from "react";
import { Calendar, momentLocalizer } from "react-big-calendar";
import moment from "moment";
import Modal from "react-bootstrap/Modal";
import CalendarForm from "./CalendarForm";
import { observer } from "mobx-react";
import { getCalendar } from "./requests";
const localizer = momentLocalizer(moment);
var CALDATE = "04/02/2020";
const calendar_configuration = {
  api_key: `AIzaSyDwfeilq4hI7CWcCWPI0H9_CVSvkQcdTJw`,
  calendars: [
    {
      name: 'demo', // whatever you want to name it
      url: 'kiril.chernysh@gmail.com' // your calendar URL
    }
  ],
  dailyRecurrence: 700,
  weeklyRecurrence: 500,
  monthlyRecurrence: 20
}
function HomePage({ calendarStore }) {
  const [showAddModal, setShowAddModal] = React.useState(false);
  const [showEditModal, setShowEditModal] = React.useState(false);
  const [calendarEvent, setCalendarEvent] = React.useState({});
  const [initialized, setInitialized] = React.useState(false);
  const hideModals = () => {
    setShowAddModal(false);
    setShowEditModal(false);
  };
  const getCalendarEvents = async () => {
    const response = await getCalendar();
    const evs = response.data.map(d => {
      return {
        ...d,
        start: new Date(d.start),
        end: new Date(d.end)
      };
    });
    calendarStore.setCalendarEvents(evs);
    setInitialized(true);
  };
```

```
const handleSelect = (event, e) => {
  const { start, end } = event;
  const data = { title: "", start, end, allDay: false };
  setShowAddModal(true);
  setShowEditModal(false);
  setCalendarEvent(data);
};
const handleSelectEvent = (event, e) => {
  setShowAddModal(false);
  setShowEditModal(true);
  let { id, title, start, end, allDay } = event;
  start = new Date(start);
  end = new Date(end);
  const data = { id, title, start, end, allDay };
  setCalendarEvent(data);
};
React.useEffect(() => {
  if (!initialized) {
    getCalendarEvents();
  }
});
return (
  <div className="page">
    <Modal show={showAddModal} onHide={hideModals}>
      <Modal.Header closeButton>
        <Modal.Title>Add Calendar Event</Modal.Title>
      </Modal.Header>
      <Modal.Body>
        <CalendarForm
          calendarStore={calendarStore}
          calendarEvent={calendarEvent}
          onCancel={hideModals.bind(this)}
          edit={false}
        />
      </Modal.Body>
    </Modal>
    <Modal show={showEditModal} onHide={hideModals}>
      <Modal.Header closeButton>
        <Modal.Title>Edit Calendar Event</Modal.Title>
      </Modal.Header>
      <Modal.Body>
        <CalendarForm
          calendarStore={calendarStore}
```

```
            calendarEvent={calendarEvent}
            onCancel={hideModals.bind(this)}
            edit={true}
          />
        </Modal.Body>
      </Modal>
      <Calendar
        localizer={localizer}
        defaultDate={CALDATE}
        events={calendarStore.calendarEvents}
        defaultView="day"
        startAccessor="start"
        endAccessor="end"
        selectable={true}
        style={{ height: "100vh" }}
        onSelectSlot={handleSelect}
        onSelectEvent={handleSelectEvent}
        config = {calendar_configuration}
      />
    </div>
  );
}
export default observer(HomePage);
```

Висновки

Особливості рішення

Розумні дзеркала мають великий потенціал для покращення досвіду користувачів у доступі та взаємодії з інформацією. Вони не лише дозволяють користувачам бачити відповідну інформацію без особливих зусиль, але вони також можуть бути інтегровані в більшу систему, наприклад, систему домашньої автоматизації. Для смарт-дзеркальних платформ досить часто використовувати веб-браузери як основний метод відображення, оскільки веб-браузери пропонують вбудовану підтримку різних форматів медіа. Однак такі платформи, як правило, мають обмеження, накладені з міркувань безпеки, які створюються веб-браузерами.

Для полегшення цієї проблеми розроблена програмна платформа для розумних дзеркал, яка має кілька переваг. По-перше, наша платформа розроблена з фокусом на легку вагу (lightweight). Вона працює на крихітному комп'ютері Raspberry Pi. По-друге, вона модульна і має можливості для розширювання. Розробники можуть реалізовувати власні плагіни за допомогою будь-яких мов програмування та легко інтегрувати їх у свої розумні дзеркальні системи, побудовані на базі нашої платформи. По-третє, серверний компонент на нашій платформі забезпечує безперервне підключення в режимі реального часу та відображення інформації, що ініційовано зовнішніми подіями. Така конфігурація дозволяє використовувати апаратні можливості, що зазвичай не можуть бути використані у веб браузерах. В той же самий час відтворення медіа (візуальне, та аудіо) покладається на веб-браузер, як на технологію яка чудово справляється з завданнями такого штибу.

Що стосується майбутньої роботи, то наразі на ринку доступні нові розширення, що забезпечують доступ до зовнішнього обладнання. У перспективі пропонується використати різні можливості плагінів для детекторів руху, датчиків температури та світла, розпізнавачів жестів, тощо.

Додатковим напрямом розвитку системи може бути можливість більш легкого розширення функціональності користувачем без глибокого знання в царині програмування та інформаційних технологій. Наприклад, встановлення програмних та апаратних розширень для системи.

Можливості застосування

Важливо продовжувати дослідження можливостей використання розумного дзеркала. Використання дзеркала зазвичай передбачає локальну присутність, включно з рухами та візуальною взаємодією, а також залежить від контексту і розташування дзеркала. У деяких місцях можуть використовуватися більш персоналізовані додатки (наприклад, у приватних помешканнях), а в інших використовуються більш узагальнені функції, орієнтовані на велику базу користувачів. Можна розвивати різноманітні ментальні моделі роботи розумного дзеркала, наприклад, стилі поведінки дзеркала при взаємодії з ним, або підходи до питань конфіденційності та безпеки. Розуміння потреб користувачів та їх очікування від взаємодії з розумним дзеркалом є ключовим для успішного використання цих моделей у повсякденному житті.

Так, у приватному використанні розумні дзеркала можна використовувати для відображення витребуваною людиною інформації, контролю побутових приладів у будинку або наданні емоційної підтримки користувачу. У ванних кімнатах розумні дзеркала можуть виявитись цінним

інструментом для деяких людей. Коли людина готується до свого робочого дня її руки, як правило, зайняті, виконуючи прості когнітивні завдання. У цей час людина могла б за допомогою дзеркала переглядати свої електронні повідомлення, твіти чи повідомлення у Facebook, або знайомитися з оглядом останніх новин.

Розумні дзеркала можуть полегшити спілкування в сім'ї. Таке дзеркало можна розташувати при вході чи у холі. В цьому місці воно може виступати центральним хабом для планування дня членів сім'ї, що підвищить обізнаність про плани кожного всередині сім'ї. Так, наприклад, вбудований у дзеркало цифровий календар можна синхронізувати з календарем кожного користувача на телефоні. Цифрові нотатки, які можна було б залишати на дзеркалі, не відпадали б при відкритті дверей у кімнату або дверцят холодильника.

У публічних місцях програми розумних дзеркал можуть виконувати більш загальні функції. У магазинах та кіосках розумні дзеркала можуть виконувати інформаційну й рекламну функції, а також функцію аварійних сповіщень. Встановлене у примірочних кімнатах розумне дзеркало могло б революціонувати процес покупки одягу: використовуючи доповнену реальність, людина за допомогою розумного дзеркала могла б переглядати себе у різноманітних елементах одягу, які є в наявності у магазині, фізично не вдягаючи його на себе. Така система дозволила б клієнтам швидше переглядати каталог одягу, вдосконалювати свій вибір та приміряти фізично лише той одяг, який викликає комфорт та задоволення. Сама примірочна кімната може забезпечувати безпечне та зручне середовище для обробки доповненої реальності, що може бути доповнене датчиками для визначення правильних розмірів одягу для кожного клієнта.

Перелік використаних джерел

1. Chandel S, Mandwarya A, Ushasukhanya S. (2018). Implementation of Magic Mirror Using Raspberry Pi 3. // https://www.semanticscholar.org/paper/IMPLEMENTATION-OF-MAGIC-MIRROR-USING-RASPBERRY-PI-3-Chandel-Mandwarya/77949c9cd001db37137d85531adc26e2a1c686b5

2. Chen J, Koken M. (2017). SmartMirror: a Glance into the Future. // https://scholarcommons.scu.edu/cgi/viewcontent.cgi?article=1092&context=cseng_senior

3. Gold, D, Sollinger, D. (2016). SmartReflect: A Modular Smart Mirror Application Platform. //

   https://www.researchgate.net/publication/310845534_SmartReflect_A_Modular_Smart_Mirror_Application_Platform

4. Gubbi, J, Buyya, R, Marusic, S, Palaniswami, M. (2013). Internet of things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7), 1645–1660.// http://www.buyya.com/papers/Internet-of-Things-Vision-Future2013.pdf

5. Manual S. What is a smart device? - a conceptualisation within the paradigm of the internet of things. (2018). // https://link.springer.com/article/10.1186/s40327-018-0063-8

6. Manual S. (2019). What is a smart device? — The key concept of the Internet of Things. //

   https://towardsdatascience.com/what-is-a-smart-device-the-key-concept-of-the-internet-of-things-52da69f6f91b

7. Moyer K, Geschickter Chet. Measuring the Strategic Value of the Internet of Things for Industries. (2016). // https://www.gartner.com/en/documents/3299317