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
_____Юрій КРАВЧЕНКО
«_____»_____2022 року

КВАЛІФІКАЦІЙНА РОБОТА БАКАЛАВРА

галузі знань 17 «Електроніка та телекомунікації»
за спеціальністю 172 «Телекомунікації та радіотехніка»
освітньо-професійна програма «Мережеві та інтернет технологій»

на тему:

РОЗРОБКА SPA-ДОДАТКУ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ

Виконав: студент групи МІТ -41
Максим РАДЬКО _____
(ім'я та ПРІЗВИЩЕ) (підпис)

Керівник: завідувач кафедри мережевих та інтернет технологій
(посада)

д.н.т., проф. Юрій КРАВЧЕНКО _____
(науковий ступень, вчене звання, ім'я та ПРІЗВИЩЕ) (підпис)

Київ 2022

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
_____ Юрій КРАВЧЕНКО

«_____»_____2021 року

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Здобувачу вищої освіти Радьку Максиму Олеговичу
(прізвище, ім'я, по батькові)

1. Тема роботи:
Розробка SPA-додатку із голосовим керуванням
затверджена на засіданні кафедри МІТ «24» грудня 2021 р. протокол №8
2. Термін здачі закінченої роботи «30» травня 2022 р.
3. Вихідні дані до проекту (роботи)
Сучасні технології створення додатків
4. Зміст пояснювальної записки (перелік питань, що їх потрібно розробити, обсяг – 35-40 стор.)
Вступ
1. Дослідження методів та технологій побудови додатків із голосовим керуванням. Постановка задачі
1.1 Огляд і аналіз методів побудови додатків із голосовим керуванням
Постановка задачі
2. Розробка SPA-додатку із голосовим керуванням
3. Дослідження розробленого SPA-додатку із голосовим керуванням
3.1. Модель дослідження додатку
3.2. Оцінка якості додатку
3.3. Рекомендації до підвищення ефективності додатку
Висновки
5. Перелік графічного матеріалу 8-10 слайдів
Дата видачі завдання
Керівник роботи Юрій КРАВЧЕНКО
(підпис) (ім'я, ПРІЗВИЩЕ)
Завдання прийняв до виконання Максим РАДЬКО
(підпис) (ім'я, ПРІЗВИЩЕ)

КАЛЕНДАРНИЙ ПЛАН ВИКОНАННЯ РОБОТИ

Номер	Назва етапів роботи	Термін виконання етапів роботи	Примітка
1	Підготовчий	20.01.2022	
2	Розділ 1	15.02.2022	
3	Розділ 2	15.03.2022	
4	Розділ 3	15.04.2022	
5	Доповідь та слайди	25.05.2022	
6	Пояснювальна записка	30.05.2022	

Здобувач вищої освіти _____ Максим РАДЬКО
(підпис)

Керівник _____ Юрій КРАВЧЕНКО
(підпис)

РЕФЕРАТ

Пояснювальна записка: 61 с., 11 рис., 5 табл., 25 джерел.

Мета роботи: розробка SPA-додатку із голосовим керуванням на основі платформи Speechly API для моніторингу фінансового стану споживача.

Об’єкт дослідження: побудова додатків із голосовим керуванням.

Предмет дослідження: SPA-додаток із голосовим керуванням на основі платформи Speechly API.

Методи дослідження: були використані теоретичні, емпіричні та комплексні методи дослідження. Методом аналізу було розділено предмет дослідження на складові частини для їх поглибленого вивчення. Далі методом індукції та моделювання, розділені раніше виділені елементи, було об’єднано у єдине ціле – модель майбутнього додатку. Після чого, на основі порівнянь, вимірювань та спостережень було розроблено додаток із голосовим управлінням.

У ході даної дипломної роботи було досліджено методи та технології побудови додатків із голосовим керуванням. Також було визначено найбільш ефективну технологію, вибір якої було обґрунтовано через найсучаснішу точність та наявність кроссплатформенності, що надає їй більшу перевагу у порівнянні з іншими аналогами. На базі даної технології було розроблено SPA-додаток із голосовим керуванням для моніторингу фінансового стану споживача.

Після створення додатку було проведено оцінку його якості та окреслено рекомендації щодо її підвищення.

З практичної точки зору, створений додаток може бути корисним для людей із обмеженими можливостями, і також слугувати прототипом для створення додатків із голосовим керуванням для інших розробників.

Наукова новизна дослідження полягає у евристичній концепції створення додатків із голосовим керуванням та в більш сучасних рекомендаціях щодо покращення ефективності роботи подібних додатків, що стають все більш затребуваними у сучасному світі.

Галузь використання: сучасна розробка SPA додатків із голосовим керуванням.

Ключові слова:

SPA, JAVASCRIPT, ГОЛОСОВЕ КЕРУВАННЯ, REACT, ВЕБ-ДОДАТОК, CSS, UI, COMPONENT, API, AJAX, SPEECHLY, HTML, GITHUB

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ 6
ВСТУП 7
1 ДОСЛІДЖЕННЯ МЕТОДІВ ТА АЛГОРИТМІВ ПОБУДОВИ ДОДАТКІВ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ. ПОСТАНОВКА ЗАДАЧІ 10
1.1 Огляд і аналіз методів побудови додатків із голосовим керуванням 10
1.2 Аналіз існуючих додатків із голосовим керуванням 12
1.3 Аналіз існуючих технологій автоматичного розпізнавання мови 15
1.4 Постановка завдання 23
2 РОЗРОБКА SPA-ДОДАТКУ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ 24
2.1 Вибір технології програмування 24
2.2 Розробка інтерфейсу користувача 29
2.3 Розробка голосового керування 31
3 ДОСЛІДЖЕННЯ РОЗРОБЛЕНОГО SPA-ДОДАТКУ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ 43
3.1 Модель дослідження додатку 43
3.2 Оцінка якості додатку 50
3.3 Рекомендації до підвищення ефективності додатку 56
ВИСНОВКИ 59
ПЕРЕЛІК ПОСИЛАНЬ 61

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

ДП – Дипломний Проєкт

AI – Artificial intelligence

AJAX – Asynchronous Javascript and XML

ASR – Automatic Speech Recognition

API – Application Programming Interface

CSS – Cascading Style Sheets

HTML – HyperText Markup Language

HTTP – HyperText Transfer Protocol

IDE – Integrated Development Environment

ISO – International Organization for Standardization

JS – JavaScript

JSON – JavaScript Object Notation

MUI – Material UI

NLU – Natural-language understanding

SPA – Simple Page Application

URL – Uniform Resource Locator

W3C – World Wide Web Consortium

ВСТУП

Голосові пристрої та додатки – це важливий прогрес у сфері штучного інтелекту, який дозволяє нам користуватися численними послугами в режимі «вільні руки». Функціональні особливості та якісні характеристики голосових помічників вже виросли до такої міри, що вони стали корисними інструментами в особистому та діловому житті.

За даними Techcrunch, «розумні пристрої, такі як Amazon Echo, Google Home і Sonos One, будуть встановлені в більшості (тобто 55 відсотків) домогосподарств у США до 2022 року». Тільки за останній рік кількість користувачів голосових платформ у США зросла на 128,9 відсотків.

Новітні та найкраще розроблені технології голосового інтерфейсу дозволяють розпізнавати вік, стать та настрій людини, щоб дати найбільш релевантну та відповідну відповідь. Вони також можуть бути використані як захід безпеки, оскільки вони можуть розрізняти голосові моделі певної людини.

Голосові помічники можуть бути як корисними, так і розважальними. Їх можна використовувати для отримання відповіді на загальне запитання, для прослуховування новин, для перегляду розкладу на день або для запуску технології розумного дому. Голосові помічники також полегшують покупки, оскільки дозволяють користувачеві розміщувати, відстежувати та скасовувати онлайн-замовлення.

Розробка голосових додатків є досить простою в теорії, але досить складною та трудомісткою на практиці. Перший і найпоширеніший ризик, пов’язаний з голосовими помічниками, полягає в тому, що вони не можуть належним чином зрозуміти та декодувати сказане. Непорозуміння трапляються через те, що технологія автоматичного розпізнавання голосового інтерфейсу часто не сприймає мовні варіації: акценти, діалекти чи соціолекти. Адже одна помилка в мовленні робить фразу нерозбірливою для голосової платформи, що перетворює використання голосового додатка на величезний виклик для людей з порушеннями мовлення.

Інша важлива проблема, пов’язана з голосовими програмами, полягає в природному розумінні мови, яке визначається як постобробка сприйманого тексту. На цьому етапі технологія має витягти основні сутності з заданого контексту (коли, де, що і хто).

І головна турбота користувачів голосових технологій — безпека даних. Якщо голосовий помічник реагує на слово-тригер, то, мабуть, він повинен постійно прислухатися до того, що користувач говорить. Що робити, якщо ці дані записуються, зберігаються та надсилаються комусь або продаються в рекламних цілях? Без сумніву, ніхто не хоче, щоб їхні приватні розмови витікали.

Тому актуальним питанням є покращення функціональності голосових технологій настільки, щоб вони стали безпечними та незамінними. І вони можуть стати такими в тих місцях, де людям потрібен голосовий помічник, оскільки їхні руки зайняті, брудні чи вологі або коли перебої у роботі додатку погано впливають на продуктивність. Голосова платформа може допомагати в різних ситуаціях, наприклад, допомагати водіям не відволікатися від дороги і тримати руки за кермом або дозволяти пенсіонерам робити покупки за допомгою одного речення.

Як і у випадку з будь-яким іншим проектом, першим кроком розробки голосового додатка є обґрунтоване та всебічне дослідження методів та алгоритмів, необхідних для його створення. Другий етап розробки голосових додатків полягає у виборі технології, з якою працюватимуть голосові програми. Після цього розробляються прототипи майбутнього додатка. І тільки після всіх мозкових штурмів і планування починається сама розробка.

Однак на цьому робота не закінчена навіть після того, як додаток буде створено. Щоб зробити голосову службу помітною та покращити її якість, потрібно буде отримати більше 5-зіркових оглядів і обробити якомога більше «невдалих» запитів — ті, які не привели до дії або викликали неправильну відповідь.

Мета цієї роботи – розробити SPA-додаток із голосовим керуванням, який буде успішно опрацьовувати сто відсотків запитів користувачів на прикладі

застосунку для моніторингу фінансового стану споживача. Для досягнення даної мети було обрано платформу голосового користувацього інтерфейсу Speechly API, так як саме дана платформа має найбільшу та найсучаснішу точність обробки звуку та являється кроссплатформенною, що надає їй більшу перевагу у порівнянні іншими аналогами.

Speechly API дозволяє легко інтегрувати вбудовану технологію автоматичного розпізнавання голосового інтерфейсу у будь-який тип додатку – від мобільних до веб-додатків, завдяки своїй кроссплатформенності, що надає більшої переваги перед аналогами. Дана платформа має свій програмний інтерфейс додатку (API), що досить простий для розуміння та застосування, а також має досить гнучку можливість налаштування розпізнавання мови та команд, специфічних для певних додатків.

1 ДОСЛІДЖЕННЯ МЕТОДІВ ТА АЛГОРИТМІВ ПОБУДОВИ ДОДАТКІВ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ. ПОСТАНОВКА ЗАДАЧІ

1.1 Огляд і аналіз методів побудови додатків із голосовим керуванням

Розробка додатків із голосовим керуванням мало чим відрізняється від розробки додатку без нього. Основною відмінністю у методах є наявність додаткових етапів розробки, таких яких вибір готової системи (платформи) розпізнавання мови або її розробка, та створення словників команд та логіки їх опрацювання. Загалом розробка додатків із голосовим керуванням включає в себе наступні етапи:

– побудова діаграми використання;

– аналіз та вибір платформи розпізнавання мови;

– аналіз та вибір типу додатку;

– аналіз та вибір інструментів для розробки (технології, IDE, система контролю версій, хостинг, платформа голосового управління);

– дизайн. У разі голосу – не відображення екранів, а опрацювання діалогів;

– розробка поділяється на дві частини: розробка додатку та інтеграція платформи розпізнавання мови;

– тестування;

– публікація.

Етапи розробки та дизайну специфічні, оскільки на відміну від стандартних додатків, даний додаток матиме вбудовану систему розпізнавання мови. Головна мета – спроектувати взаємодію між користувачем та програмою.

Практично всі відомі методи розпізнавання мовлення мають низку основних загальних властивостей:

– для розпізнавання використовується метод порівняння з еталонами;

– сигнал може бути представлений у вигляді безперервної функції часу, або у вигляді слова в деякому кінцевому алфавіті;

– для зменшення обсягу обчислень використовуються методи динамічного програмування.

Динамічне програмування – метод вирішення завдань шляхом складання послідовності із підзавдань таким чином, що:

– перший елемент послідовності (можливо кілька елементів) має тривіальне рішення;

– останній елемент цієї послідовності – це вихідне завдання;

– кожна задача цієї послідовності може бути вирішена за допомогою рішення підзадач з меншими номерами.

Методи розпізнавання мови можна розділити на дві великі групи: непараметричні – з використанням непараметричних мір близькості до еталонів (до них можна віднести методи на основі формальних граматик і методи на основі метрик на безлічі мовних сигналів) – і параметричні (імовірнісні – на основі методу прихованих моделей Маркова, нейромережеві) [1].

Непараметричні методи, засновані на мірах близькості безлічі мовних сигналів. Метод Вінцюка [1, 2], заснований на методі динамічного програмування, дозволив скоротити час обчислення значень функції близькості до еталонних сигналів з експоненціального (від довжини сигналу) до квадратичного. З огляду на те, що основною специфікою методу було нелінійне спотворення тимчасової осі однією з порівнюваних функцій, метод отримав назву «динамічної деформації часу». До переваг відносяться простота його реалізації та навчання. До недоліків можна віднести складність обчислення міри близькості, яка пропорційна квадрату довжини сигналу і великий обсяг пам'яті, необхідний для зберігання еталонів команд – пропорційний довжині сигналу кількості команд у словнику.

Параметричні – це методи, що застосовуються до завдання розпізнавання мови в даний час, були вперше запропоновані рядом американських дослідників, наприклад Дж. Бейкер [3] та Ф. Джелінек [4], у 1970-і роках минулого століття. В них застосовується теорія прихованих моделей Маркова – двічі стохастичні

процеси і ланцюги Маркова [5] з переходів між станами та безлічі стаціонарних процесів у кожному стані ланцюга.

Перевагами методу прихованих моделей Маркова є:

– швидкий спосіб обчислення значень функції відстані (імовірності);

– значно менший обсяг пам'яті, в порівнянні з методом «динамічної деформації часу», який необхідний для зберігання еталонів команд.

Основними недоліками є:

– велика складність його реалізації;

– необхідність використання великих фонетично збалансованих мовних корпусів на навчання параметрів.

Далі розглянемо сучасні додатки з використанням голосового управління.

1.2 Аналіз існуючих додатків із голосовим керуванням

Розпізнавання мовлення є цінним, оскільки воно економить споживачам і компаніям час і гроші.

Суть технології розпізнавання мовлення полягає у використанні природної мови для ініціювання дії. Сучасні мовні технології почалися в 1950-х роках і почали розвиватися протягом десятиліть:

– 1950-ті: Bell laboratories розробили «Audrey», систему, здатну розпізнавати числа 1-9, сказані одним голосом.

– 1960-ті: IBM розробила пристрій під назвою «Shoebox», який міг розпізнавати й розрізняти 16 розмовних англійських слів.

– 1970-ті: була створена система «Гарпія» в Карнегі-Меллоні, яка могла розуміти понад 1000 слів.

– 1990-ті: поява персональних комп'ютерів принесла більш швидкі процесори та відкрила двері для технології диктування. Bell laboratories розробили інтерактивні системи розпізнавання голосу з підключенням.

– 2000-ті: розпізнавання мовлення досягло рівня точності близько 80%, а потім з’явився Google Voice, зробивши технологію доступною для мільйонів користувачів і дозволивши Google збирати цінні дані.

– 2010-ті: Apple запустила Siri, а Amazon – Alexa в спробі конкурувати з Google.

Повільно, але впевнено, розробники рухалися до мети, щоб машини могли розуміти і реагувати на все більшу кількість вербалізованих команд.

Сучасні провідні системи розпізнавання мовлення — Google Assistant, Amazon Alexa та Siri від Apple — не досягли б свого сьогоднішнього рівня без перших розробників, які проклали шлях для розвитку даних тхнологій.

Завдяки інтеграції нових технологій, таких як хмарні, і безперервним удосконаленням завдяки збору мовленнєвих даних, ці мовленнєві системи постійно покращують свою здатність «чути» та розуміти широкий спектр слів, мов та наголосів.

Google Voice Search (компанія Google) [6]. Раніше пошук застосовувався виключно у мобільних пристроях. Зараз голосовий пошук від Google вбудований у браузер Google Chrome, що дозволяє використовувати цей сервіс на різних платформах.

Характеристики:

– підтримка української мови;

– можливість вбудовувати розпізнавання мовлення на веб-ресурси;

– голосові команди, словосполучення;

– для роботи потрібне постійне підключення до мережі internet.

Dragon NaturallySpeaking (компанія Nuance) [7]. Світовий лідер у програмному забезпеченні з розпізнавання людського мовлення. Можливість створювати нові документи, надсилати електронну пошту, керувати популярними браузерами та різноманітними програмами за допомогою голосових команд.

Характеристики:

– точність розпізнавання до 99%.

ViaVoice (компанія «IBM») є програмним продуктом для апаратних реалізацій [8]. Компанія ProVox Technologies на основі цього ядра створила систему для диктування звітів лікарів-радіологів VoxReports.

Характеристики:

– точність розпізнавання сягає 95-98%;

– дикторонезалежність;

словник системи обмежений набором специфічних термінів.

Sphinx – найвідоміший і найпрацездатніший з відкритих програмних продуктів для розпізнавання мови на сьогоднішній день [9]. Розробка ведеться в університеті Карнегі-Меллона, поширюється на умовах ліцензії Berkley Software Distribution і доступна як для комерційного, так і для некомерційного використання.

Характеристики:

– дикторонезалежність;

– розпізнавання злитого мовлення;

– навчання;

– наявність версії для систем, що вбудовуються – Pocket Sphinx.

Найбільші споживачі голосових технологій – електронна комерція, виробники різноманітних пристроїв домашнього застосування, таких як телевізори, відеомагнітофони, мікрохвильові печі, пральні машини та ін.

Мовні технології, що дозволяють розпізнавати команди в умовах шумів, дозволяють доповнити керування в автомобілях такими функціями як керування світлом, радіо, замками тощо. Обсяг ринку в цьому сегменті досягнув 7.5 мільярдів USD у 2020 році. Голосове управління функціями автомобільних аудіо та навігаційних систем вже реалізовано у деяких моделях BMW, Mercedes-Benz, Ford, Toyota та інших. Такі системи допомагають водієві не відволікатися від дороги, проте для того, щоб їх ефективно використовувати, водій повинен знати спеціальні голосові команди, яких, наприклад, у системі Ford SYNC близько десяти тисяч [10].

Також існує потреба у мовленнєвих технологіях у військово-промисловому комплексі: тренажери-імітатори бойової техніки; військова техніка, системи оповіщення (голосового оповіщення оператора про несправності або пошкодження систем, а також про виконані операції/завдання), системи безпеки (наприклад, можливість зупинки бойової техніки при її пошкодженні або пораненні оператора за допомогою голосу), комплекси ППО, радіолокаційні станції та ін.

В освітній сфері затребувані, зокрема, системи навчання мовлення, технологія виділення та вимірювання фонем мовлення відкриває нові можливості для навчання мовлення. Вона вводить у процес навчання мови, крім звукового, візуальний зворотний зв'язок, дозволяє побачити свою та еталонну мову, порівняти їх візуально, побачити помилки вимови та отримати оцінку вимови фонеми, слова та фрази. Візуалізація процесу вимови з виділенням фонем та показом положення артикуляційних органів з аналізу вимови дозволяє створити унікальні системи для навчання вимови для людей з обмеженими можливостями.

1.3 Аналіз існуючих технологій автоматичного розпізнавання мови

Автоматичне розпізнавання мовлення (ASR) – це розділ комп’ютерної лінгвістики, що займається розпізнаванням і перекладом усної мови в текст за допомогою комп’ютерів, процес, у деяких випадках відомий як «мовлення в текст». Системи є поєднанням мов, інформатики та електротехніки. Фраза «розпізнавання мовлення» відноситься до процесу перетворення вимовлених слів у текст загалом; однак такі підполя, як розпізнавання голосу та ідентифікація мовця, спеціалізуються на ідентифікації як мовленнєвого вмісту, так і особистості диктора [11].

За останнє десятиліття сфера експоненціально зросла, а системи ASR з’являються в популярних програмах, які людство використовує щодня, таких як TikTok та Instagram для субтитрів у реальному часі, Spotify для транскрипції підкастів, Zoom для транскрипції зустрічей тощо.

Сьогоднішній ASR – це підмножина машинного навчання (ML), яке само по собі є різновидом штучного інтелекту (AI).

Обробка природної мови (NLP) все частіше включається в більш просунуті версії систем ASR. Ці пристрої записують реальні людські розмови та обробляють їх за допомогою штучного інтелекту. На точність ASR впливають різноманітні параметри, включаючи гучність динаміка, фоновий шум, записуюче обладнання тощо.

Основні характеристики та ознаки, за якими можна класифікувати сучасні системи розпізнавання мови [12]:

– словники розміром у десятки та сотні тисяч слів;
– розпізнавання роздільної чи злитої мови;
– робота у реальному часі;
– дикторозалежність або дикторонезалежність системи;
– надійність роботи 95-98% для граматично правильних текстів;
– призначення.

Класифікація систем розпізнавання мови за складністю [13]:

– системи автоматичного розпізнавання ізольованих слів для розпізнавання команд за словами;
– системи автоматичного розпізнавання злитого мовлення – з можливістю виділяти слова у природному частково злитому потоці людської мови;
– системи розуміння мови – з елементами інтелекту, що дозволяє, по-перше, на основі смислового аналізу більш правильно виділяти слова в потоці мови, а, по-друге, зберігати інформацію в якійсь базі знань, звідки вона може бути легко витягнута для вирішення певних інтелектуальних завдань

Основні компоненти систем розпізнавання мови [12]:

– графічне середовище для розробки, компіляції та оптимізації граматичних та лексичних блоків розпізнавання, перевірки та редагування лексиконів;

– система для протоколювання діалогів з працюючого додатка з метою оцінки якості розпізнавання та підстроювання системи;

– інструмент оцінки якості роботи системи (перевірка відповідності слова, сказаного абонентом, що використовується граматикою);

– система для створення «тренованих» мовних моделей, що підвищують продуктивність та прискорюють процес розпізнавання;

– система для розподілу безлічі паралельних запитів різних типів та прозорої інтеграції різних мовних модулів у мережі.

Повна ілюстрація класифікації систем розпізнавання мови наведена на рис. 1.1. [14].

Умови виникнення проблем систем розпізнавання мови:

– довільний, наївний користувач;

– спонтанне мовлення;

– наявність акустичних перешкод та спотворень, у тому числі мінливих;

– наявність мовних перешкод;

– недостатня капітальна база, що не дає змоги інтенсивно проводити дослідження та розробляти нові інноваційні алгоритми у мовних технологіях.

Умови, на основі яких виявляються такі вимоги та обмеження:

– потрібне попереднє налаштування системи на голос від кількох десятків хвилин до кількох годин попереднього наговорювання текстів;

– деякі перевірки не дають результатів кращих ніж 5% помилок;

– можливість правильного розпізнавання слів не перевищує однієї третини навіть для добре організованих спонтанно вимовлених текстів.

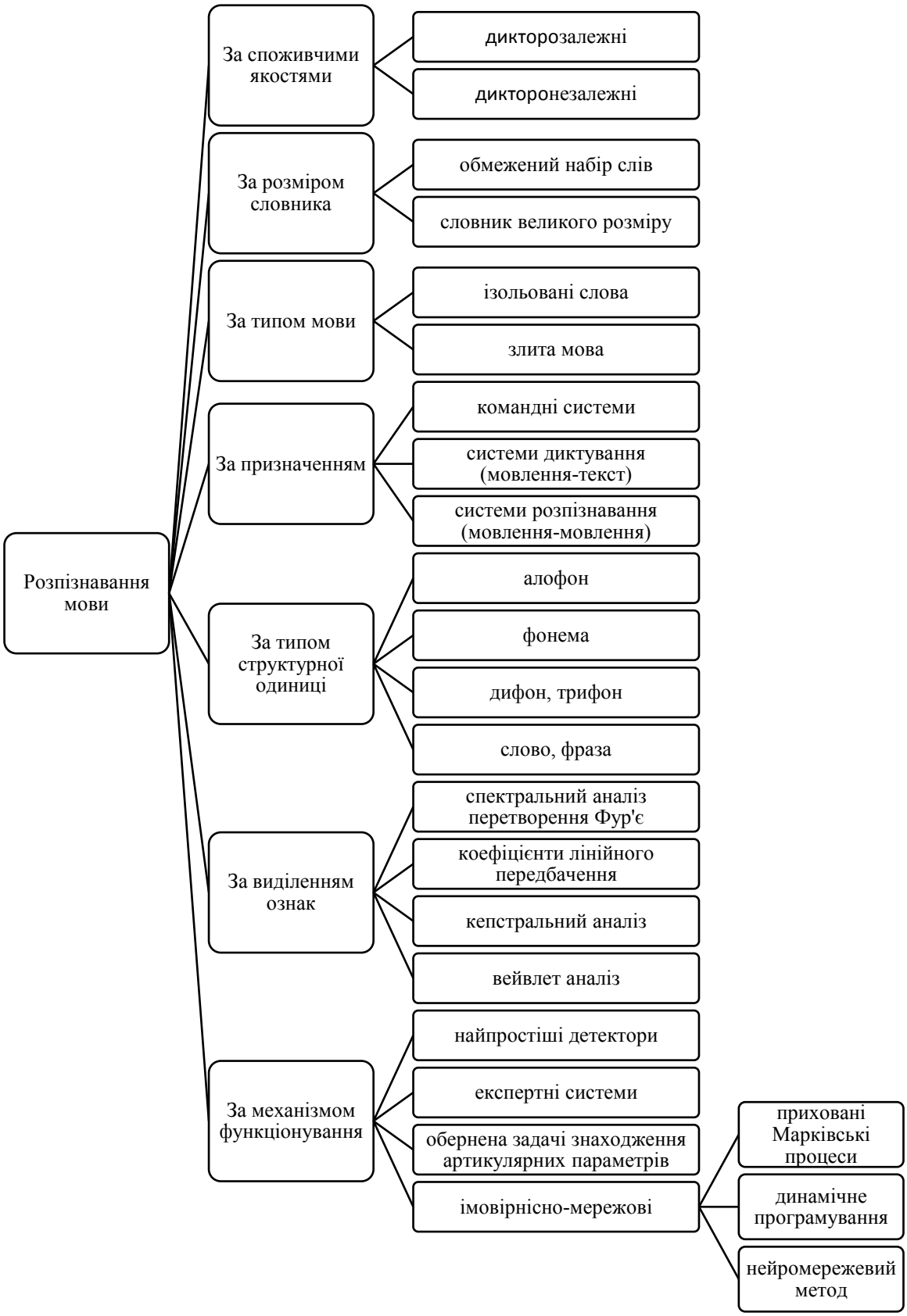

Рисунок 1.1 – Класифікація систем розпізнавання мовлення [14]

У системах розпізнавання мовлення використовуються два типи моделей [15]:

– Акустична модель: файл, що містить статистичні представлення кожного з різних звуків, що утворюють слово, відомий як акустична модель. Фонема – це ярлик, наданий кожному з цих статистичних представлень.

– Мовна модель: щоб розрізняти слова, які звучать подібно, звуки поєднуються з послідовністю слів. Припускається, що звуковий зразок граматично та семантично правильний, навіть якщо він граматично не досконалий або містить пропущені слова. В результаті включення мовної моделі в декодування може підвищити точність ASR.

Процес розпізнавання мовлення представлений на рис.1.2.

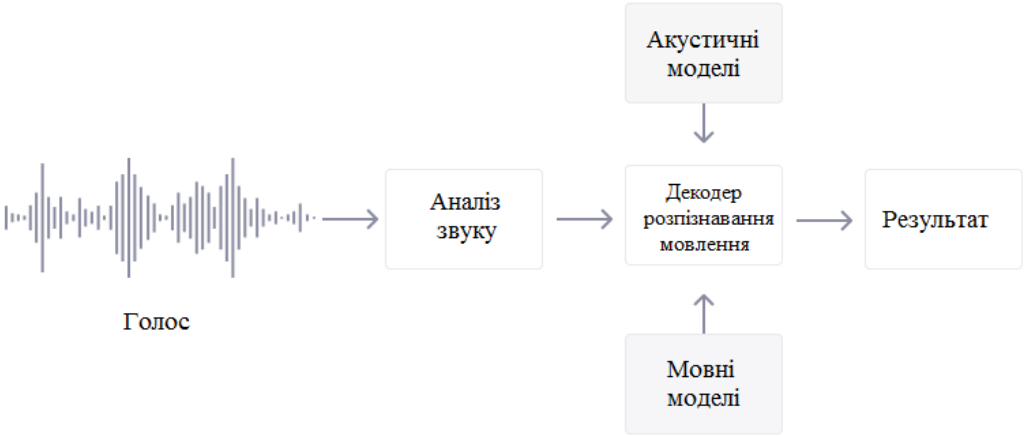

Рисунок 1.2 – Процес розпізнавання мовлення

Етапи процесу розпізнавання мовлення [13]:

– Аналогово-цифрове перетворення: у більшості випадків мова записується і доступна в аналоговому форматі. Для перетворення аналогового голосу в цифровий з використанням методів дискретизації та квантування доступні стандартні методи дискретизації або пристрої. Одновимірний вектор голосових зразків, кожен з яких є цілим числом, зазвичай використовується для представлення цифрового мовлення.

– Попередня обробка мовлення: фоновий шум і тривалі періоди тиші є звичайними для записаної розмови. Ідентифікація та видалення безшумних

кадрів, а також методи обробки сигналів для зменшення/усунення шуму є частиною попередньої обробки мови. Після попередньої обробки мова розбивається на 20-секундні кадри для наступних етапів вилучення ознак.

– Вилучення ознак: це перетворення мовних кадрів у вектор ознак, який визначає, яку фонему або склад вимовляють.

– Вибір слів: послідовність фонем/особливостей перекладається на розмовне слово за допомогою мовної моделі/моделі ймовірності.

Використання програмного забезпечення для розпізнавання мовлення має ряд переваг, серед яких:

– Спілкування між машиною і людиною. Технологія дозволяє електронним пристроям спілкуватися з людьми природною мовою або розмовною мовою.

– Легко доступний. Це програмне забезпечення часто встановлюється на комп’ютерах і мобільних пристроях, що робить його доступним.

– Простий у використанні. Добре розроблене програмне забезпечення просте в експлуатації і часто працює у фоновому режимі.

– Постійне автоматичне вдосконалення. Системи розпізнавання мовлення, які включають AI, з часом стають більш ефективними та простішими у використанні. Оскільки системи вирішують завдання розпізнавання мовлення, вони генерують більше даних про людську мову та покращують свою роботу.

Технологія розпізнавання мовлення, хоча й корисна, все ж має кілька недоліків, які потрібно усунути. Нижче наведено деякі обмеження [13]:

– Непослідовне виконання. Системи можуть бути не в змозі точно вловити слова через різницю у вимові, відсутність підтримки деяких мов і неможливість сортувати фоновий шум. Навколишній шум може бути особливо складним. Акустичні тренування можуть допомогти відфільтрувати це, але ці програми не ідеальні. Іноді неможливо відокремити людський голос.

– Швидкість. Деякі програми розпізнавання мовлення потребують часу для розгортання та освоєння. Обробка мовлення може бути відносно повільною.

– Проблеми з вихідним файлом. Успіх розпізнавання мовлення залежить від використовуваного записуючого обладнання, а не лише від програмного забезпечення.

Системи розпізнавання мовлення мають широкий спектр використання, найпопулярніші з яких [15]:

– автоматичні субтитри з розпізнаванням мовлення;
– мобільний телефон, включаючи мобільну електронну пошту;
– люди з обмеженими можливостями;
– домашня автоматизація;
– віртуальний помічник.

Різновиди програмного забезпечення для розпізнавання голосу [16]:

1. Програмне забезпечення, що залежить від динаміка.

Програмне забезпечення, що залежать від динаміка, обслуговує особа, яка буде керувати системами. Ці системи можуть розпізнавати команду та слова з точністю до 95%. Спочатку вони розуміють і оцінюють характеристики голосу своїх користувачів, оскільки вони сильно залежать від своїх дикторів. Але як тільки вони звикнуть до голосу свого диктора, їх можна розглядати як дуже ефективне програмне забезпечення для розпізнавання мовлення.

2. Програмне забезпечення для розпізнавання голосу незалежно від динаміка.

Незалежна від динаміка система розпізнавання голосу широко використовується для публічного використання. Ця система повинна реагувати на широкий спектр слів. Тому що в громадських місцях колонки неможливо полагодити. Ця система мотивована реагувати на слова незалежно від того, хто говорить. Ефективність цієї системи очікувано нижча, ніж системи, що залежить від динаміків.

3. Програмне забезпечення для розпізнавання голосу та керування ним.

Ця система використовується для керування інструментами за допомогою голосових команд. Такі завдання, як запуск програм, перегляд веб-сайтів та інші

процеси, можна виконати без зусиль. Ця система запускається у віддалене місце через Wi-Fi.

4. Програмне забезпечення для розпізнавання голосу з дискретним введенням.

Ця система забезпечує точність ідентифікації слів. Іноді вона може ідентифікувати слова з точністю до ста відсотків. Але це вимагає паузи після кожної фрази. Система розпізнавання голосу з дискретним введенням має граматику за замовчуванням з ключовими та неключовими словами. Ця інновація розглядається в системах розпізнавання голосу з дискретним введенням з граматикою.

Розширена граматика залишається ефективною, доки оператор не зробить новий вибір. Ця мережа пропонує гнучку пропорцію між граматикою невеликої довжини за замовчуванням і величезною функціональністю, яка сприяє підтримці граматики великого розміру, щоб тим самим зменшити можливість хибного та помилкового розпізнавання [12].

5. Програмне забезпечення для безперервного введення голосу.

Дуже невелика кількість програмного забезпечення забезпечує безперервне розпізнавання голосового введення. Компанія, що займається розробкою програмного забезпечення, розробила програмне забезпечення для безперервного потоку слів і також проаналізувала ці слова.

Основна відмінність від інших програм для розпізнавання голосу, полягає в тому, що вони оцінюють менше слів і забезпечують найкращі результати за лічені хвилини. Програмне забезпечення для безперервного введення голосу в основному використовується в медицині та охороні здоров’я.

Наприклад, Apple Dictation (безкоштовна програма для пристроїв Apple). Dragon Medical One – одна з найпопулярніших програм для розпізнавання голосу, яка використовується в охороні здоров’я.

6. Програмне забезпечення для розпізнавання голосу при введенні природного мовлення.

Цей тип програмного забезпечення є більш просунутим, ніж будь-який інший тип програмного забезпечення. Оскільки цей тип програмного забезпечення може розуміти вільну та швидку розмову. Він може оцінити більше 160 слів за хвилину. Нині використовується у всіх сучасних пристроях [16].

Розпізнавання мовлення – це технологія, що розвивається. Це один із багатьох способів спілкування з комп'ютерами. Різноманітні бізнес-додатки на основі комунікацій використовують зручність і швидкість розмовного спілкування, які забезпечує ця технологія.

За 60 років розвитку програми розпізнавання мовлення значно просунулися вперед. Вони все ще вдосконалюються, зокрема завдяки AI.

1.4 Постановка завдання

Розробити SPA-додаток із голосовим керуванням на основі платформи Speechly API для моніторингу фінансового стану споживача.

Додаток має виконувати наступні функції:

- Записувати та зберігати транзакції користувача (тип (надходження, витрати); категорія; сума; дата);
- Можливість видалення транзакцій зі списку;
- Голосове керування;
- Виведення підказки для голосового керування;
- Виведення загального балансу;
- Візуалізація надходжень та витрат у вигляді діаграм за категоріями.

2 РОЗРОБКА SPA-ДОДАТКУ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ

2.1 Вибір технології програмування

React — це бібліотека розробки інтерфейсу користувача на основі JavaScript. Хоча React є бібліотекою, а не мовою, він широко використовується у веб-розробці. Бібліотека вперше з’явилася у травні 2013 року і зараз є однією з найбільш часто використовуваних інтерфейсних бібліотек для веб-розробки [17].

React пропонує різні розширення для всієї архітектурної підтримки додатків, таких як Flux і React Native, за межами простого інтерфейсу.

Сьогодні популярність React перевершила популярність усіх інших фреймворків розробки інтерфейсу.

Основні переваги [18]:

– Легке створення динамічних програм: React полегшує створення динамічних веб-додатків, оскільки вимагає менше кодування та пропонує більше функціональних можливостей, на відміну від JavaScript, де кодування часто стає складним.

– Покращена продуктивність: React використовує Virtual DOM, тим самим створюючи веб-додатки швидше. Virtual DOM порівнює попередні стани компонентів і оновлює лише ті елементи в Real DOM, які були змінені, замість того, щоб знову оновлювати всі компоненти, як це роблять звичайні веб-додатки.

– Компоненти для багаторазового використання: Компоненти є будівельними блоками будь-якої програми React, і одна програма зазвичай складається з кількох компонентів. Ці компоненти мають свою логіку та елементи керування, і їх можна повторно використовувати в додатку, що, у свою чергу, значно скорочує час розробки програми.

– Односпрямований потік даних: React слідує за односпрямованим потоком даних. Це означає, що при розробці програми React розробники часто вкладають дочірні компоненти в батьківські компоненти. Оскільки дані надходять

в одному напрямку, стає легше налагоджувати помилки та знати, де в програмі виникає проблема в даний момент.

– Невелика крива навчання: React легко вивчається, оскільки він переважно поєднує базові концепції HTML і JavaScript з деякими корисними доповненнями. Проте, як і у випадку з іншими інструментами та фреймворками, доведеться витратити деякий час, щоб отримати належне розуміння бібліотеки React.

– React можна використовувати як для розробки веб-додатків, так і для мобільних додатків. Існує фреймворк під назвою React Native, похідний від самого React, який дуже популярний і використовується для створення красивих мобільних додатків. Таким чином, насправді React можна використовувати для створення як веб-, так і мобільних додатків.

– Спеціальні інструменти для легкого налагодження: Facebook випустив розширення Chrome, яке можна використовувати для налагодження програм React. Це робить процес налагодження веб-додатків React швидшим і простішим.

Перераховані вище причини описують популярність бібліотеки React і те, чому вона використовується великою кількістю організацій і підприємств.

React пропонує деякі видатні функції, які роблять його найбільш поширеною бібліотекою для розробки інтерфейсних програм. Особливості React [19]:

– JSX.

JSX – це синтаксичне розширення JavaScript. Це термін, який використовується в React для опису того, як має виглядати інтерфейс користувача. Можливо писати структури HTML у той самий файл, що й код JavaScript, використовуючи JSX.

```
const name = 'Simplilearn';
const greet = <h1>Привіт, {name}</h1>;
```

Наведений вище код показує, як JSX реалізовано в React. Це не рядок і не HTML. Замість цього він вбудовує HTML у код JavaScript.

Базовий шаблон HTML може виглядати як приклад коду нижче.

```
<!DOCTYPE html>
```

```
<html lang="en">
 <head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <meta http-equiv="X-UA-Compatible" content="ie=edge">
  <title>Hello World!</title>
  <link rel="stylesheet" href="style.css">
 </head>
 <body>
  <h1>Title</h1>
  <p>Description</p>
 </body>
</html>
```

React популярний, оскільки він швидкий, ефективний і простий у використанні. У нього також є велика спільнота розробників, які можуть допомогти у вирішенні проблем, з якими можете зіткнутися розробник.

Основний компонент HTML може виглядати як приклад коду:

```
import "./styles.css";
const App = () => {
 return (
  <div className="App">
   <h1>Title</h1>
   <h2>I am React component</h2>
   <p>And I'm using HTML</p>
  </div>
 );
}
export default App
```

HTML використовується для надання веб-сайту структури. React використовується для створення цілих веб-сайтів, або для створення інтерфейсів користувача.

Коли проект розробляється в React, не обов’язково потрібен HTML. React – це бібліотека JavaScript, яка дозволяє створювати інтерфейси користувача. Він працює з «компонентами», які є невеликими частинами багаторазового коду, який можна створити один раз і використовувати де завгодно.

Отже, якщо потрібно створити простий компонент React, HTML не потрібен. Однак, якщо потрібно створити більш складний компонент React, може знадобитися використовувати HTML для визначення структури компонента.

– Об'єктна модель віртуального документа (DOM).

Virtual DOM – це полегшена версія Real DOM від React. Маніпулювання в Real DOM значно повільніші, ніж маніпуляції у Virtual DOM. Коли стан об’єкта змінюється, Virtual DOM оновлює лише цей об’єкт у Real DOM, а не всі (рис.2.1).

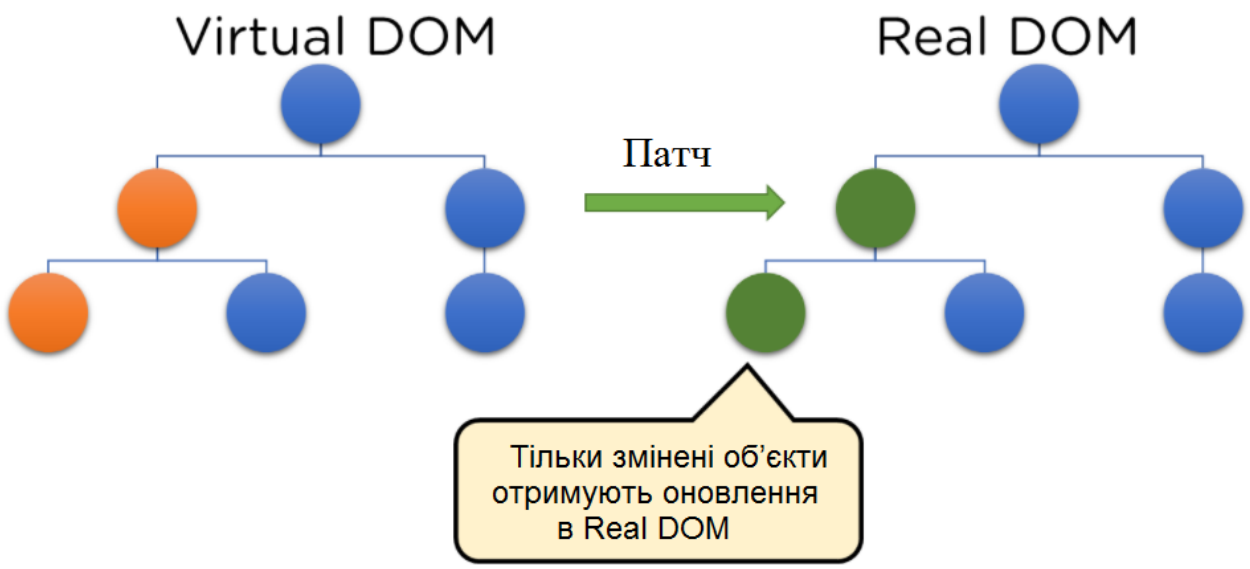

Рисунок 2.1 – Зміни в об'єктній моделі віртуального документа [19]

DOM (Document Object Model) розглядає документ XML або HTML як структуру дерева, в якій кожен вузол є об’єктом, що представляє частину документа.

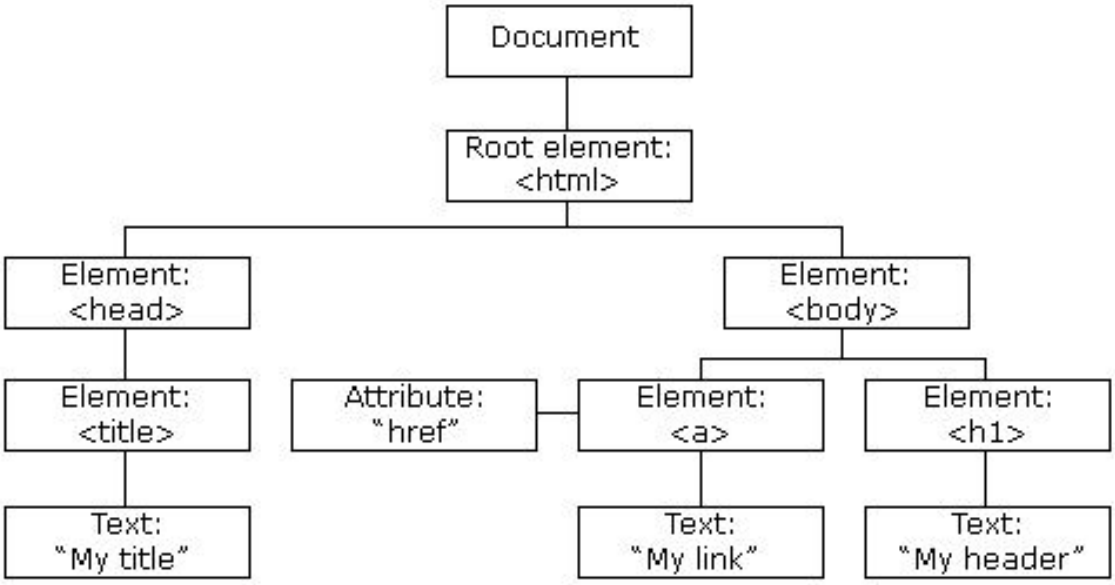

Рисунок 2.2 – DOM веб-сторінки

– Архітектура.

MVC — це архітектурний шаблон, який розділяє прикладний рівень на модель, перегляд і контролер. Модель стосується всієї логіки, пов'язаної з даними; подання використовується для логіки інтерфейсу користувача програми, а контролер є інтерфейсом між моделлю та представленням.

– Розширення.

React виходить за межі рамки інтерфейсу користувача; він містить багато розширень, які охоплюють всю архітектуру програми. React допомагає створювати мобільні додатки та забезпечує рендеринг на стороні сервера. Flux і Redux, серед іншого, можуть розширювати React.

– Прив’язка даних.

Оскільки React використовує одностороннє прив’язування даних, усі дії залишаються модульними та швидкими. Крім того, односпрямований потік даних означає, що під час розробки проекту React дочірні компоненти зазвичай вкладаються в батьківські компоненти.

– Налагодження.

Оскільки існує широка спільнота розробників, програми React прості та легко тестуються.

Отже, для розробки додатку була використана бібліотека React.

2.2 Розробка інтерфейсу користувача

Material Design – це мова дизайну, розроблена в 2014 році Google і дуже популярна для веб- та мобільних додатків.

Material UI (MUI) містить компоненти React. Існує багато компонентів інтерфейсу користувача, які відповідають потребам бізнесу. Крім того, настроювані компоненти Material UI можна легко змінити відповідно до вимог. Це бібліотека, яка дозволяє імпортувати та використовувати різні компоненти для створення інтерфейсу користувача в програмах реагування. Це значно економить час, оскільки розробникам не доводиться писати все з нуля [20].

Material UI – це найпотужніший та найефективніший інструмент для створення програми шляхом додавання дизайнів та анімацій з технічними та науковими інноваціями. Він використовує фреймворки та бібліотеки JavaScript, такі як AngularJS та ReactJS, щоб зробити програму більш привабливою. Крім того, дозволяє користувачам додавати дизайн, анімацію, сітку, тіні та ефекти блискавки, що дозволяє швидше та легше стилізувати веб-розробку.

Material Design є ключовим підходом до платформи Android як для UI, так і для UX. Його початкова мета полягала в розробці додатків як для настільних комп’ютерів, так і для мобільних платформ.

Він створений для надання зручності користувачам і надає додаткові компоненти CSS Baselines. Він має багато тем за замовчуванням, тому є можливість легко налаштувати теми та компоненти.

Material UI підтримує останні стабільні випуски всіх основних браузерів і платформ. Він також підтримує Internet Explorer 11.

Material UI надає низькорівневі допоміжні функції, так звані «функції стилю», для створення потужних систем проектування.

Особливості Material UI [20]:

– послідовність інтерфейсу;

– адаптивний стиль без зусиль;

– робота з будь-яким об’єктом теми;

– архів менше ніж КБ;

– досить швидкий для виконання.

Необхідні передумови використання Material UI:

– редактор коду;

– має бути встановлений NodeJS.

Причини, чому розробники вважають за краще інтегрувати MUI у свої програми:

– Попередньо розроблені компоненти інтерфейсу користувача. MUI надає кілька попередньо розроблених компонентів, завдяки яким легко знайти підхід до програми, що розроблюється, та приєднати її, забезпечуючи привабливий, простий у використанні та візуально привабливий дизайн та швидку реалізацію.

– Material Design. Material Design — це добре продумана система проектування, яка описує цінність і варіанти використання компонентів. У Material Design, наприклад, є можливість використовувати значки матеріалів, тобто вибрати піктограми, які відповідають системі дизайну.

– Регульована тема. MUI забезпечує прості теми встановлення та налаштування для використання та глобальної реалізації для всіх доступних компонентів, що робить інтерфейс надзвичайно функціональним і динамічним. Таким чином можна налаштувати колір теми компонента, інформацію про палітру, властивості поверхні та деякі інші стилі, що означає, що всі компоненти можуть бути узгоджені.

– Добре структурована документація. MUI має чітко зрозумілу та добре структуровану документацію, яка включає посібники з прикладами коду для практики.

– Широка підтримка. Завдяки своїй бібліотеці, яка постійно оновлюється, MUI має чудову підтримку для виправлення помилок і проблем. У

невеликих випусках для всіх знайдених проблем команда надає виправлення. Команда щороку надсилає опитування всім розробникам бібліотеки, щоб виправити будь-які проблеми та зробити інтерфейс Material UI зручнішим.

– Спільнота. Є можливість знайти основні посилання з підтримкою та приклади використання MUI.

Завдяки компонентам з бібліотеки Material UI були використані елементи Material Design у веб-додатку React.

2.3 Розробка голосового керування

Speechly API – це інструмент розробника для створення мультимодальних голосових інтерфейсів користувача в режимі реального часу. Це дозволяє покращити поточний сенсорний інтерфейс користувача за допомогою голосових функцій для кращого користування [21].

Основні характеристики Speechly [21]:

– повністю потоковий API;

– мультимодальний з нуля;

– легко налаштувати для будь-якого випадку використання;

– швидко інтегрується в будь-яку програму з сенсорним екраном;

– підтримує природні корекції;

– візуальний зворотній зв’язок у реальному часі.

Speechly – розробник програмного забезпечення для голосових команд, призначеного для перекладу вимовлених слів у практичні завдання. Програмне забезпечення компанії використовує обробку звуку, розпізнавання мовлення, обробку природної мови та виявлення голосової активності для розпізнавання команд. API, розроблений Speechly, — це потоковий API для розуміння розмовної мови, який працює для розуміння складних завдань із мультимодальним інтерфейсом, доступним на різних платформах для використання голосу в

електронній комерції, віртуальній реальності, іграх, цифрових програмах та в професійній роботі.

Speechly швидко вивчає лексику для певної області та налаштовується на акустичні умови.

Команда провідних дослідників Speechly власноруч створила запатентовану найсучаснішу технологію машинного навчання.

Порівняння точності Google та Speechly представлено на рис.2.3.

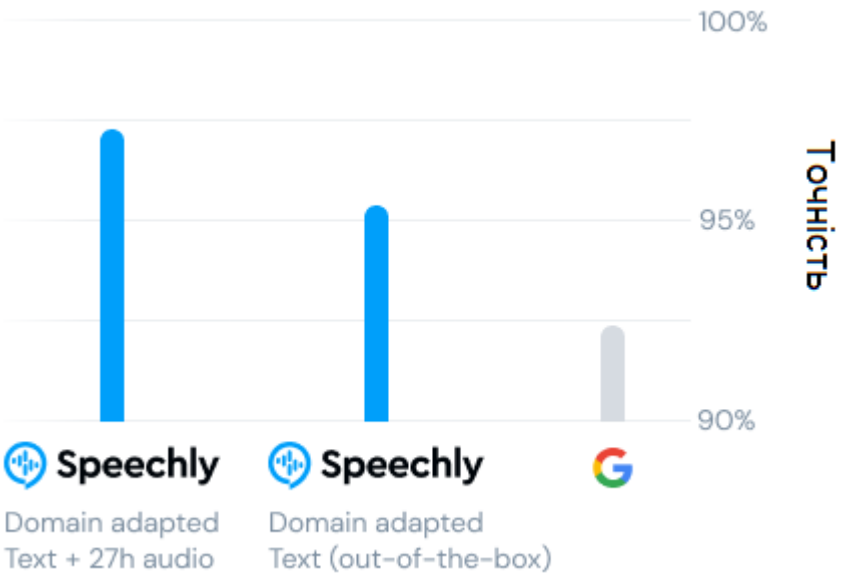

Рисунок 2.3 – Порівняння точності Google та Speechly (набір даних ATIS3 грудень 1993 року) [21]

Speechly API призначений для того, щоб допомогти іншим розробникам розробляти програми з голосовим керуванням.

Speechly API розроблено, щоб допомогти розробникам створювати природні голосові враження з візуальним зворотним зв'язком. API розроблено, щоб зробити використання голосу під час пошуку більш природним, з функцією «натисни і говори», щоб було зрозуміло, коли користувачі говорять і коли вони закінчили, без затримки. Speechly API також працює, щоб підтримувати виправлення користувачів, щоб дозволити більш природну манеру говорити та відновлюватися після можливих помилок пошуку.

Speechly також допомагає розробникам більше зосередитися на інтеграції API та варіантів його використання в продукт, а не на вбудовуванні функції голосового пошуку в свої програми. Крім того, компанія пропонує безліч інструментів, заснованих на передбачуваній інтеграції, щоб допомогти розробникам інтегрувати Speechly у свої програми.

Speechly API був розроблений, щоб повертати намір користувача разом із стенограмою, бути потужним і простим у використанні, запропонувати послідовну модель, доступну для всіх платформ, і продемонструвати низьку затримку для швидшого та простішого користування.

Speechly працює для інтеграції з усіма основними платформами, які включають сенсорні екрани, включаючи веб-браузери, React, iOS, Android і React Native.

Отже, за допомогою Speechly можна вдвічі скоротити час, витрачений на повторювані, трудомісткі завдання введення даних, таких як заповнення форм і пошук даних.

Speechly API приймає аудіо потік в якості вхідних даних та повертає розшифровку мовлення користувачів разом із ідентифікованими значеннями та сутностями до програми. Speechly API досягає цього, застосовуючи машинне навчання. Однак для моделей машинного навчання потрібні приклади висловлювань, анотовані інформацією, характерною для програми, що розроблюється. Під час розгортання ці дані використовуються як для адаптації моделі мовлення до словникового запасу, присутнього в навчальних прикладах, так і для навчання моделей розпізнавання природної мови (NLU) для виявлення конкретних намірів і сутностей програми [21].

За допомогою Speechly конфігурація виконує дві важливі цілі:

– Навчання системі розпізнавання мовлення словникового запасу, який є актуальним у програмі, що розроблюється. У додатку може знадобитися використання незвичайних слів (наприклад, незрозумілі назви брендів або спеціалізований жаргон), які повинні бути чітко закріплені в моделі розпізнавання мовлення.

– Визначення інформації (намірів та сутностей), яку слід витягти з висловлювань користувачів. Важко надати готові конфігурації, які б в достатній мірі відповідали різноманітним варіантам використання. Набір намірів і сутностей тісно пов’язаний з роботою кожної конкретної програми.

Конфігурація Speechly містить навчальні дані для моделей машинного навчання. Speechly описує низку прикладів висловлювань, які можуть говорити користувачі, і з яких слід розібрати намір і, можливо, кількість сутностей. Приклади висловлювань записуються у синтаксисі [21]:

```
*search do you have [blue](color) [jackets](product)
```

У наведеному вище прикладі визначається висловлювання користувача «чи є у вас сині куртки», призначається пошук за намірами та визначаються дві сутності, названі кольором і продуктом, зі значеннями blue та jackets відповідно. Намір і об’єкти повертаються до програми, і на їх основі голосовий інтерфейс може виконувати дії, запитані користувачем. У цьому випадку інтерфейс користувача має оновити перегляд результатів пошуку, щоб показувати лише сині куртки.

Конфігурація повинна містити принаймні кілька прикладів висловлювань для кожної функціональності конкретного голосового інтерфейсу. Отже, чим більше прикладів надається, тим краще.

Конфігурації Speechly можна записати як компактні шаблони, які потім розширюються у великий набір прикладів висловлювань під час навчання моделі. Наприклад, конфігурація [21]:

```
product = [t shirts | hoodies | jackets | jeans | slacks | shorts | sneakers | sandals]
color = [black | white | blue | red | green | yellow | purple | brown | gray]
*search do you have $color(color) $product(product)
```

Оголошує дві змінні, продукт і колір, і призначає обом список відповідних значень. Третій рядок визначає шаблон, який генерує 72 приклади висловлювань, кожне з яких починається з «do you have», за яким слідує колір та сутність продукту, значення яких взято з відповідних списків [21]:

```
*search do you have [black](color) [t shirts](product)
```

```
*search do you have [white](color) [t shirts](product)
*search do you have [blue](color) [t shirts](product)
...
*search do you have [gray](color) [sandals](product)
```

Усі ці 72 приклади висловлювання компактно визначені лише трьома рядками «коду» вище.

Підготовку прикладів висловів корисно розглядати як завдання «програмування» генератора даних.

Синтаксис мовних анотацій Speechly (SAL) складається з Синтаксису анотації та Шаблону позначень.

Синтаксис анотації використовується для анотування намірів і сутностей.

Наміри визначаються шляхом додавання до прикладу *intent_name. Речення, що залишилося після частини *intent_name, буде розпізнано як таке, що має намір intent_name.

Наприклад [21]:

```
*show_products show all products
```

Висловлення кінцевого користувача «Показати всі продукти» поверне намір show_products

Сутності визначаються за допомогою `[entity value](entity name)` позначення.

Наприклад [21]:

```
*show_products show [jeans](category)
```

Висловлення кінцевого користувача «Show jeans» («Показати джинси») поверне значення `jeans` для імені сутності `category`.

Шаблон позначення використовується для визначення шаблонів, які під час розгортання розширюються на приклади висловлювань.

Списки визначаються за допомогою `[exp1 | exp2 | ... | exp_N]`, де exp1, exp2 … є довільними виразами SAL.

Коли шаблон зі списком розгортається, в останньому прикладі висловлювання використовується лише один із елементів списку [21]. Наприклад, шаблон:

```
*show_products [show | view | i want to see] products
```

Еквівалентно написанню:

```
*show_products show products
*show_products view products
*show_products i want to see products
```

Підрядок прикладу висловлювання може бути оголошено необов'язковим, укладаючи його в фігурні дужки {цей підрядок необов'язковий}. Додаткова частина може бути довільним виразом SAL.

Необов'язкові частини прикладу висловлювання можуть існувати або не існувати. Шаблон [21]:

```
*show_products {show} products {please}
```

Еквівалентно написанню:

```
*show_products show products please
*show_products show products
*show_products products please
*show_products products
```

Змінні оголошуються з синтаксисом `variable_name = довільний-SAL-вираз`, а доступ до їх значення здійснюється за допомогою $variable_name. Можливо призначити будь-який довільний вираз SAL змінній.

Наприклад, звичайним випадком використання змінних є списки різних значень сутності [21]:

```
categories = [jeans | shoes | shirts | accessories]
*show_products show $categories(category)
```

В коді, що вказаний вище $categories(category) є скороченням від [$categories](category). Коли значення сутності шукається зі змінної, дужки не потрібні.

Змінні також можна використовувати для складання складних фраз із простих компонентів

```
digit = [one | two | three | four | five | six | seven | eight | nine | zero]
symbol = [hash | slash | dash]
```

```
product_code = $digit $digit $symbol $digit $digit $digit $digit
```

В коді, що вказаний вище product_code визначає шаблон, який розгортається до всіх можливих висловів, які починаються з двох цифр, за якими слідує один із символів, за яким слідують чотири цифри, наприклад «six four dash nine nine zero four» («шість чотири тире дев'ять дев'ять нуль чотири») або «one two hash three four five six» («один два хеш три чотири п'ять шість») [21].

Кожна змінна x має бути оголошена у конфігурації, перш ніж її можна буде використовувати з позначенням $x. Правильний варіант:

```
x = [hello | hi | greetings]
*greet $x
```

Не правильний варіант:

```
*greet $x
x = [hello | hi | greetings]
```

Крім того, оголошення змінної не може бути рекурсивним.

Перестановка генерує всі можливі перестановки заданого списку виразів. Вона визначається за допомогою синтаксису `![exp1 | exp2 | ... | exp_N]`, де exp1, exp2, … можуть бути довільними виразами SAL.

Наприклад [21]:

```
*book Book a ticket ![from [New York](from) | to [London](to) | for [two](num_passengers)]
```

Еквівалентно написанню:

```
*book Book a ticket from [New York](from) to [London](to) for [two](num_passengers)
*book Book a ticket from [New York](from) for [two](num_passengers)] to [London](to)
*book Book a ticket to [London](to) from [New York](from) for [two](num_passengers)
*book Book a ticket to [London](to) for [two](num_passengers) from [New York](from)
*book Book a ticket for [two](num_passengers) from [New York](from) to [London](to)
*book Book a ticket for [two](num_passengers) to [London](to) from [New York](from)
```

Конфігурація SAL складається з виразів SAL. Вираз SAL є або

- приклад висловлювання;
- шаблон;
- частковий шаблон;

– визначення змінної.

Кожен рядок у конфігурації SAL повинен визначати приклад висловлювання, шаблон або визначення змінної. Частковий шаблон може відображатися лише як частина шаблону, а не сам по собі. Крім того, визначення змінної має з'являтися перед використанням або посиланням на відповідну змінну [21].

Вираз SAL є прикладом висловлювання, якщо він не містить жодних позначень шаблону. Тобто його можна розширити лише на себе. Усі наступні вирази SAL є прикладами висловів:

```
*search show [red](color) [pants](product)
*book book a flight from [New York](depart_city) to [London](arrival_city)
*greeting hello
```

Усі вирази SAL, які є прикладами висловлювань, мають починатися з визначення наміру.

Вираз SAL є шаблоном, якщо він містить позначення шаблону, тобто принаймні одне з: список, необов'язкова частина, посилання на змінну або перестановку, і його можна розширити до приклад висловлювання. Усі ці вирази SAL є шаблонами [21]:

```
*search show {[red | green | blue](color)} [pants | shirts | shoes](product)
*book book a flight ![from $city(depart_city) | to $city(arrival_city)]
*greeting $all_greeting_phrases
```

Шаблон починається з визначення наміру. Однак, якщо визначення змінної розширюється до шаблону, то посилання на цю змінну також є шаблоном. Правильний варіант [21]:

```
all_my_intents = [
  *buy buy $company(stock_name) for $amount(value) dollars
  *sell sell $company(stock_name) for $amount(value) dollars
]
$all_my_intents
```

Не правильний варіант:

```
all_my_intents = [
```

```
  buy $company(stock_name) for $amount(value) dollars
  sell $company(stock_name) for $amount(value) dollars
]
$all_my_intents
```

Це пов’язано з тим, що вирази в списку all_my_intents не розширюються до дійсних прикладів висловлювання, оскільки в них відсутнє визначення наміру.

Частковий шаблон подібний до шаблону, але він не розширюється до дійсного прикладу висловлювання. Тобто в ньому відсутнє визначення наміру. Приклади часткових шаблонів [21]:

```
hello
[New York | London | Paris | Berlin | Tokyo]
{can i have} a [large](size) [coffee](product)
```

Частковий шаблон має значення лише як

– елемент списку;

– додаткова частина;

– елемент перестановки;

– права частина визначення змінної.

Зокрема, частковий шаблон не можна використовувати як такий, але він завжди має відображатися як частина повного шаблону.

Вираз SAL є визначенням змінної, якщо він має формат

```
LHS = RHS
```

де LHS – ім'я змінної, а RHS – це або приклад висловлювання, або шаблон або частковий шаблон.

Під час навчання шаблони у конфігурації SAL випадковим чином розширюються до прикладів висловлювань. Система навчання не вичерпно розширює всі можливі висловлювання з шаблонів, але випадковим чином генерує достатню кількість прикладів висловлювань.

Шаблон розгортається, обробляючи його зліва направо. Щоразу, коли зустрічається позначення шаблону, алгоритм розширення розширює відповідну частину відповідно до свого правила розширення. Алгоритм застосовується

рекурсивно, якщо застосування правила розширення призводить до чогось, що може бути додатково розширено.

Правила розширення

– Списки. Елемент списку вибирається рівномірно випадковим чином з усіх елементів списку.

– Необов’язкові частини. Вираз, що міститься в додатковій частині, розгортається з імовірністю 0,5 і опускається з імовірністю 0,5.

– Змінні. Посилання на змінну замінюється значенням змінної і звідти починається алгоритм розширення.

– Перестановки. Вирази в списку перестановок упорядковуються в результуючому прикладі висловлювання таким чином, що кожна розстановка має однакову ймовірність. Тобто з ймовірністю 1/N!, якщо в перестановці є N виразів.

Стандартні змінні полегшують використання загальних виразів у конфігурації без необхідності писати все з нуля самостійно.

Стандартні змінні — це будівельні блоки, які полегшують підтримку певних поширених, але дещо складних виразів у конфігурації. Стандартні змінні виглядають і використовуються як звичайні змінні, але не потрібно визначати їх у конфігурації. Стандартні змінні можна ідентифікувати за їх іменами, які починаються з SPEECHLY [21].

Стандартну змінну, яка використовується в прикладі висловлювання нижче, дозволяє розпізнавати різні способи вираження дат програмою:

```
*book book a flight for $SPEECHLY.DATE(departure)
```

Хоча стандартні змінні можна використовувати як такі, що відображаються як значення сутності, також рекомендується призначити відповідній сутності відповідний тип даних.

Нижче в стовпці прикладів окремо написані літери (наприклад, h t t p) вказують на те, що висловлювання користувача може містити написання.

Таблиця 2.1 – Підтримувані стандартні змінні [21]

Стандарт	Змінна	Розгортається до прикладів
$SPEECHLY.DATE	Вирази дати	tomorrow, next Friday, January fifth twenty twenty
$SPEECHLY.TIME	Вирази часу	three thirty pm, quarter past eleven, fifteen twenty five
$SPEECHLY.NUMBER	Довільні числа	five million five hundred twenty-eight thousand eight point twelve, minus zero point zero five, eleven thousand
$SPEECHLY.CARDINAL_NUMBER	Довільні кардинали	five million five hundred twenty-eight thousand eight, minus three, eleven hundred eleven
$SPEECHLY.SMALL_NUMBER	Малі числа	seventeen point five, minus five
$SPEECHLY.SMALL_CARDINAL_NUMBER	Малі кардинальні числа	seventeen, minus five, ninety-five
$SPEECHLY.FOUR_DIGIT_NUMBER	Чотиризначні числа	five six four nine, one nine eight four
$SPEECHLY.POSITIVE_NUMBER	Додатні числа	eleven hundred eleven, seventeen, five million five hundred
$SPEECHLY.NEGATIVE_NUMBER	Від’ємні числа	minus five, negative twenty-four
$SPEECHLY.SMALL_ORDINAL_NUMBER	Порядкові числа	first, second, thirty-first
$SPEECHLY.IDENTIFIER_SHORT	Ідентифікатор із 1-4 символів	zero zero seven x, alpha dash one, two seven
$SPEECHLY.IDENTIFIER_MEDIUM	5-8 ідентифікатор символів	a b one two dash nine x, delta foxtrot five seven dash two
$SPEECHLY.IDENTIFIER_LONG	Ідентифікатор із 9-12 символів	one two seven dot zero dot zero dot one slash x y
$SPEECHLY.IDENTIFIER	1-12 ідентифікатор символів	two seven, one two seven dot zero dot zero dot one slash x y
$SPEECHLY.PHONE_NUMBER	Номери телефонів	plus three five eight four zero one two three four five six, one two three four five six
$SPEECHLY.PERSO	Імена людей:	amelia m earhart, john smith, _c o n a n

N_NAME		o’brien
$SPEECHLY.EMAIL_ADDRESS	Адреси електронної пошти:	hello at speechly dot com, john dot smith at company dot com
$SPEECHLY.WEB_ADDRESS	Адреси веб-сайтів	w w w dot speechly dot com, h t t p s colon slash slash docs dot speechly dot com
$SPEECHLY.STREET_ADDRESS	(стиль США) Вулиця адреса	one twenty three michigan avenue, sixty four east twenty second street

3 ДОСЛІДЖЕННЯ РОЗРОБЛЕНОГО SPA-ДОДАТКУ ІЗ ГОЛОСОВИМ КЕРУВАННЯМ

3.1 Модель дослідження додатку

Відстеження витрат часто є першим кроком до впорядкування фінансів. Розуміючи, на що витрачаються гроші і скільки витрачається, користувач додатку зможе точно відстежити, на що витрачаються кошти та сфери, де можна скоротити витрати.

Завдяки SPA-додатку для відстеження витрат, це легко зробити частиною повсякденного життя. Ця програма, безумовно, збігається з програмами для складання бюджету, але хоча останні надають широке уявлення про фінанси, розроблений SPA-додаток робить більший акцент на витратах користувача. Розроблений SPA-додаток класифікує витрати і допомагає отримати уявлення про купівельну поведінку користувача.

SPA-додаток – це односторінковий веб-додаток, в якому є три форми: надходження, витрати та створення транзакцій (рис.3.1)

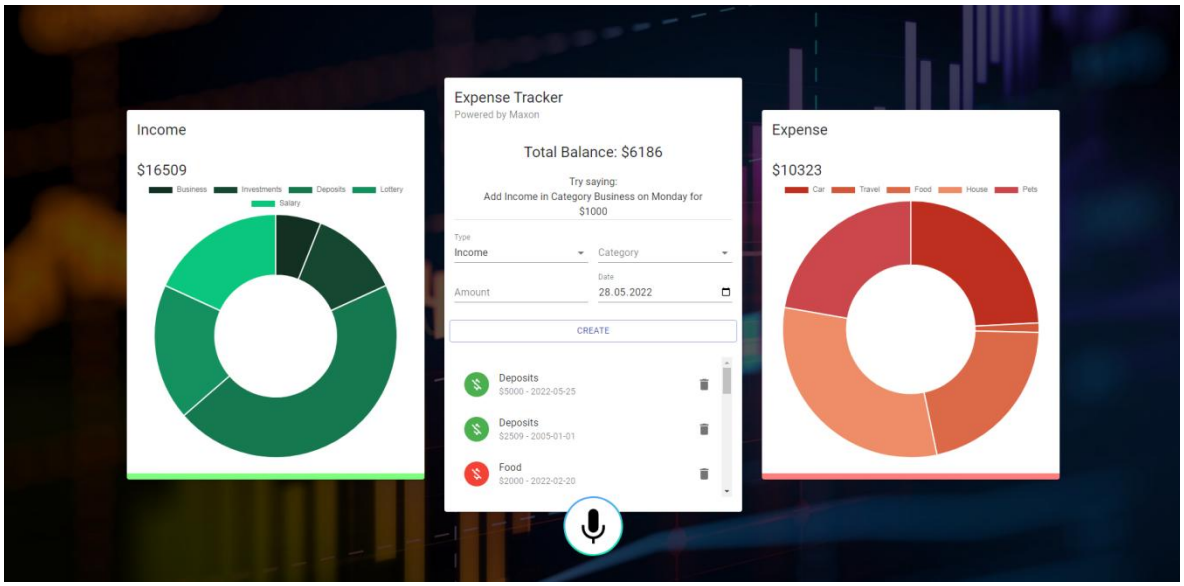

Рисунок 3.1 – Інтерфейс SPA-додатку

Створення транзакцій містить в собі наступну інформацію (рис.3.2):

– Загальний баланс;

– Підказка для голосового керування;
– Тип (надходження, витрати);
– Категорія;
– Сума;
– Дата;
– Кнопка «Створити» для запису данних;
– Список введених доходів та витрат;
– Кнопка голосового керування.

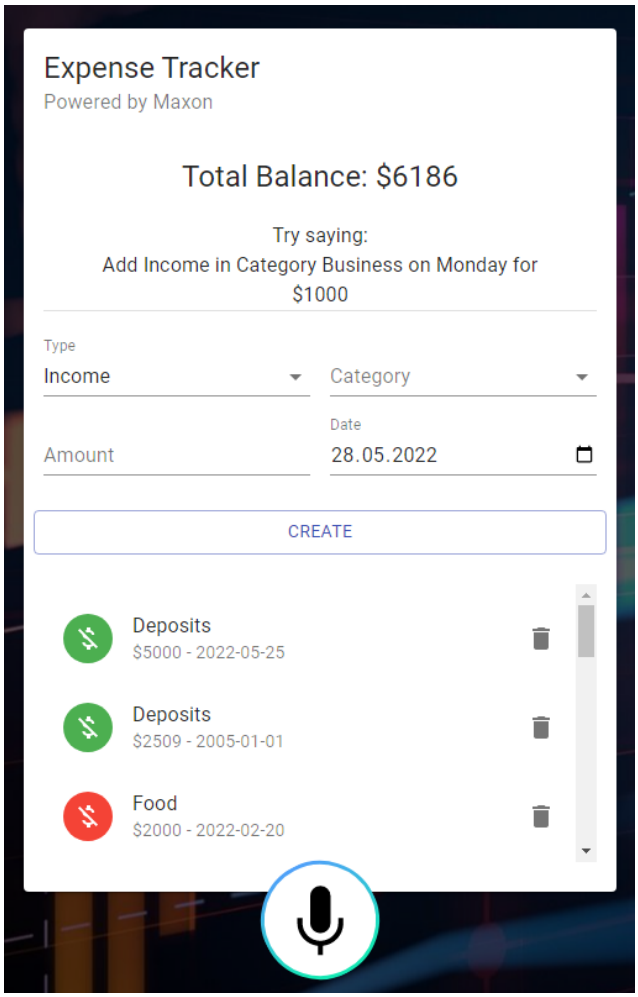

Рисунок 3.2 – Створення транзакцій

Тип, категорію, суму та дату є можливість обрати вручну (рис.3.3) або за допомогою голосового керування.

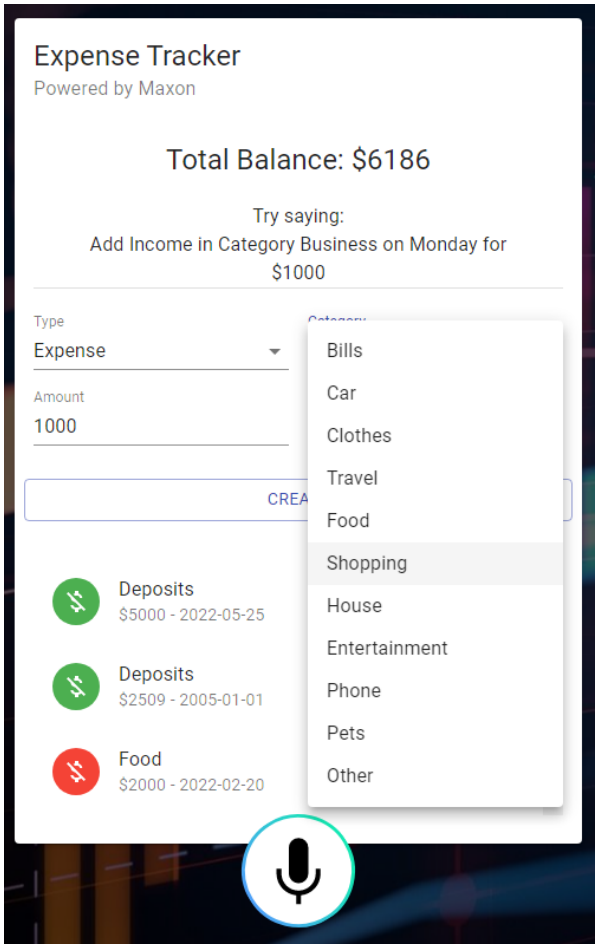

Рисунок 3.3 – Вибір даних вручну

Для того, щоб скористатись голосовим керуванням потрібно натиснути кнопку голосового керування і тримати, при цьому кнопка візуально підсвічується, щоб користувач бачив коли саме зчитуються команди (рис.3.4).

```
const App = () => {
  const classes = useStyles();
  const {speechState} = useSpeechContext();
  const main = useRef(null);
  const executeScroll = () => main.current.scrollIntoView({behavior: "smooth"});

  useEffect(() => {
    if(speechState === SpeechState.Recording){
      executeScroll();
    }
```

```
  }, [speechState]);
  return (
    <div>
      <Grid className={classes.grid} container spacing={0} alignItems='center'
justifyContent='center' style={{height: '100vh'}}>
        <Grid item xs={12} sm={3} className={classes.mobile}>
          <Details title="Income"/>
        </Grid>
        <Grid ref={main} item xs={12} sm={3} className={classes.main}
style={{margin:'16px'}}>
          <Main />
        </Grid>
        <Grid item xs={12} sm={3} className={classes.desktop}>
          <Details title="Income"/>
        </Grid>
        <Grid item xs={12} sm={3} className={classes.last}>
          <Details title="Expense"/>
        </Grid>
      </Grid>
      <PushToTalkButtonContainer>
        <PushToTalkButton/>
        <ErrorPanel />
      </PushToTalkButtonContainer>
    </div>
  )
}
```

Після створення транзакцій кругові діаграми створюються в розділі доходів або витрат на основі типу транзакції в режимі реального часу, показуючи розподіл доходів і витрат відповідно.

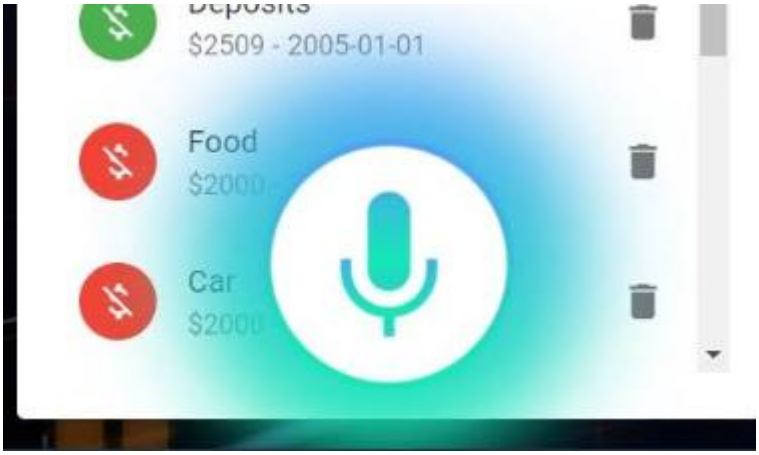

Рисунок 3.4 – Кнопка голосового керування в процесі зчитування даних

Також є можливість видалення зі списку збереженних надходжень та витрат:

```
export const Provider = ({ children }) => {
   const [transactions, dispatch] = useReducer(contextReducer, initalState);
   //Action Creators
   const deleteTransaction = (id) => dispatch({ type: 'DELETE_TRANSACTION', payload: id });
   const addTransaction = (transaction) => dispatch({ type: 'ADD_TRANSACTION', payload:
transaction });
   const balance = transactions.reduce((acc, curVal) => {
      return (curVal.type === 'Expense' ? acc - curVal.amount: acc + curVal.amount)
   }, 0);
   return (
      <ExpenseTrackerContext.Provider value={{
         deleteTransaction:deleteTransaction,
         addTransaction:addTransaction,
         transactions,
         balance
      }}>
         {children}
      </ExpenseTrackerContext.Provider>
   )
}
```

Категорії надходжень:

– Бізнес;
– Інвестиції;
– Депозити;
– Лотерея;
– Заробітна плата.

Категорії витрат:

– Автомобіль;
– Подорожі;
– Їжа;
– Дім;
– Домашні тварини.

Надходження візуально зображуються у вигляді кругової діаграми у відтінках зеленого (рис.3.5), а витрати – червоного (рис.3.6) кольору, зображенняALT.

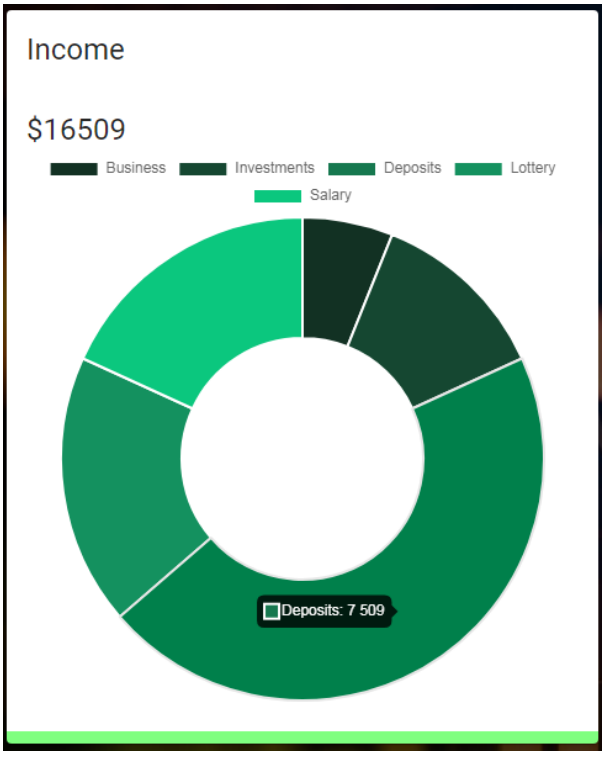

Рисунок 3.5 – Діаграма надходжень

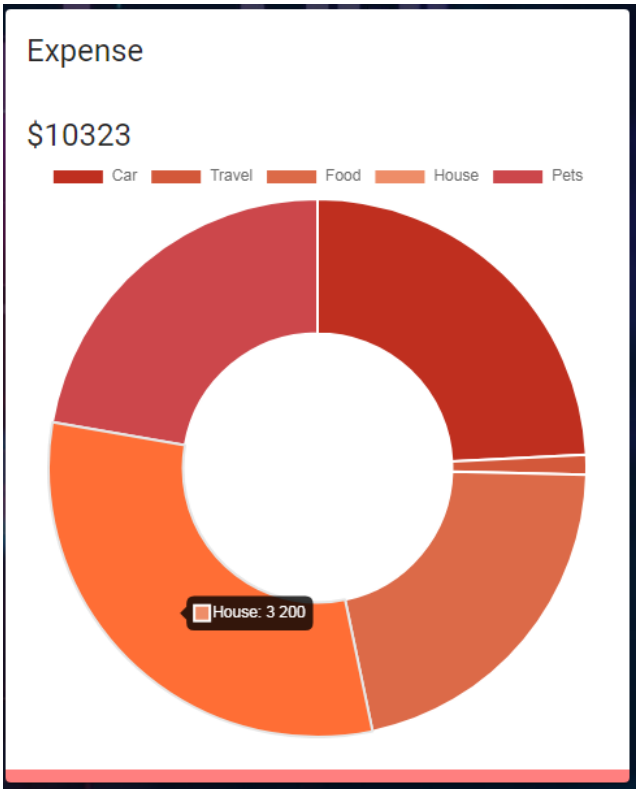

Рисунок 3.6 – Діаграма витрат

```
const incomeColors = ['#123123', '#154731', '#165f40', '#16784f', '#14915f', '#10ac6e',
'#0bc77e', '#04e38d', '#00ff9d'];
const expenseColors = ['#b50d12', '#bf2f1f', '#c9452c', '#d3583a', '#dc6a48', '#e57c58',
'#ee8d68', '#f79d79', '#ffae8a', '#cc474b', '#f55b5f'];

export const incomeCategories = [
  { type: 'Business', amount: 0, color: incomeColors[0] },
  { type: 'Investments', amount: 0, color: incomeColors[1] },
  { type: 'Extra income', amount: 0, color: incomeColors[2] },
  { type: 'Deposits', amount: 0, color: incomeColors[3] },
  { type: 'Lottery', amount: 0, color: incomeColors[4] },
  { type: 'Gifts', amount: 0, color: incomeColors[5] },
  { type: 'Salary', amount: 0, color: incomeColors[6] },
  { type: 'Savings', amount: 0, color: incomeColors[7] },
  { type: 'Rental income', amount: 0, color: incomeColors[8] },
```

```
];
export const expenseCategories = [
  { type: 'Bills', amount: 0, color: expenseColors[0] },
  { type: 'Car', amount: 0, color: expenseColors[1] },
  { type: 'Clothes', amount: 0, color: expenseColors[2] },
  { type: 'Travel', amount: 0, color: expenseColors[3] },
  { type: 'Food', amount: 0, color: expenseColors[4] },
  { type: 'Shopping', amount: 0, color: expenseColors[5] },
  { type: 'House', amount: 0, color: expenseColors[6] },
  { type: 'Entertainment', amount: 0, color: expenseColors[7] },
  { type: 'Phone', amount: 0, color: expenseColors[8] },
  { type: 'Pets', amount: 0, color: expenseColors[9] },
  { type: 'Other', amount: 0, color: expenseColors[10] },
];
export const resetCategories = () => {
  incomeCategories.forEach((c) => c.amount = 0);
  expenseCategories.forEach((c) => c.amount = 0);
};
```

Додаток використовує локальне сховище, яке зберігає дані програми користувача, навіть коли сайт закритий.

Розроблений SPA-додаток простий у використанні, зручний та має привабливий інтерфейс.

3.2 Оцінка якості додатку

Веб-сайт фактично є специфічним програмним забезпеченням, яке визначається в системі атрибутів різних рівнів, які разом утворюють модель якості [22]. Модель якості веб-сайту повинна визначати вимоги до якості, які визначаються набором вимірюваних атрибутів і відповідають очікуванням

користувачів. Для виконання вимірювань у межах однієї моделі необхідно визначити так звані метрики, які є певним методом і шкалою вимірювання.

Метод використання стандартизованих метрик передбачає використання рекомендацій або стандартів деяких організацій, що займаються дослідженнями у сфері розробки програмного забезпечення. Цей метод заснований на емпіричному дослідженні. Його використання обмежено сферою дослідження, рекомендації щодо вибору метрики носять загальний характер. Найпоширенішим методом вибору метрик властивостей програмного забезпечення є «ціль-запитання-метрика» [23]. Цей метод заснований на постановці цілей і запитань при підборі метричного програмного забезпечення для характеристики.

Метод складається з трьох основних етапів:

– сформулювати цілі, яких слід досягти, оцінюючи властивості програмного забезпечення;

– сформулювати питання для кожної мети, на які потрібно відповісти, щоб визначити, чи досягнуті цілі;

– визначити показники, які можна використовувати для відповіді на запитання, які були задані на другому етапі.

Застосування методу «ціль-запитання-метрика» включає процеси планування, визначення цілей, питань і метрик, збору даних та аналізу даних (рис. 3.1). Важливим аспектом використання цього методу є досвід і компетентність експерта. У той же час цей метод не є формалізованим, що дає певну частку ймовірності неадекватного підбору метрик. Крім того, цей метод не дозволяє вибрати метричне програмне забезпечення, яке найкраще підходить для характеристики властивості. Тому для вирішення цієї задачі пропонується формалізувати задачу за допомогою предметно-орієнтованого методу побудови зв’язків між програмними метриками.

Модель оцінки якості програмного забезпечення, орієнтована на користувача [24], дозволяє оцінити якість сайту стосовно оцінки різних груп кінцевих користувачів. Це створило узагальнену оцінку процесу, яку можна застосувати до різних доменів кінцевих користувачів.

Ефективним механізмом контролю якості веб-додатків є використання веб-метрик. Через швидке зростання веб-ресурсів виникає потреба в актуальних показниках, які можуть мати великий вплив на ресурс в цілому та на окремі сайти. Якість сайту (або якісні веб-додатки) можна оцінити з двох точок зору: програмістів і кінцевих користувачів. Аспекти якості веб-сайту зосереджені програмістами на рівні ремонтопридатності, безпеки, функціональності тощо. Хоча кінцевих користувачів цікавлять зручність, ефективність, безпека.

Розширюючи ці поняття, оцінка якості веб-сайту може залежати від:

1. Факторів, пов'язаних з завданням, які впливають на кінцевий результат оцінки якості та змісту.

2. Факторів, пов'язаних з продуктивністю, а саме: ефективність і технологія веб-додатків. Наприклад: час відповіді, вихід транзакції та надійність.

3. Факторів розвитку, які мають вирішальне значення для розробників веб-додатків.

Таким чином, визначено параметри веб-сайту, які можна реалізувати як інструмент для забезпечення всебічних результатів.

Для оцінки якості розробленого SPA-додатку була використана модель якості веб-сайту з використанням методології оцінки якості, рекомендованої Б. Ліберном (рис.3.1). Ієрархічна трирівнева модель дерева, що складається з п'яти атрибутів моделі: естетика, зручність використання, мультимедіа, вміст, репутація.

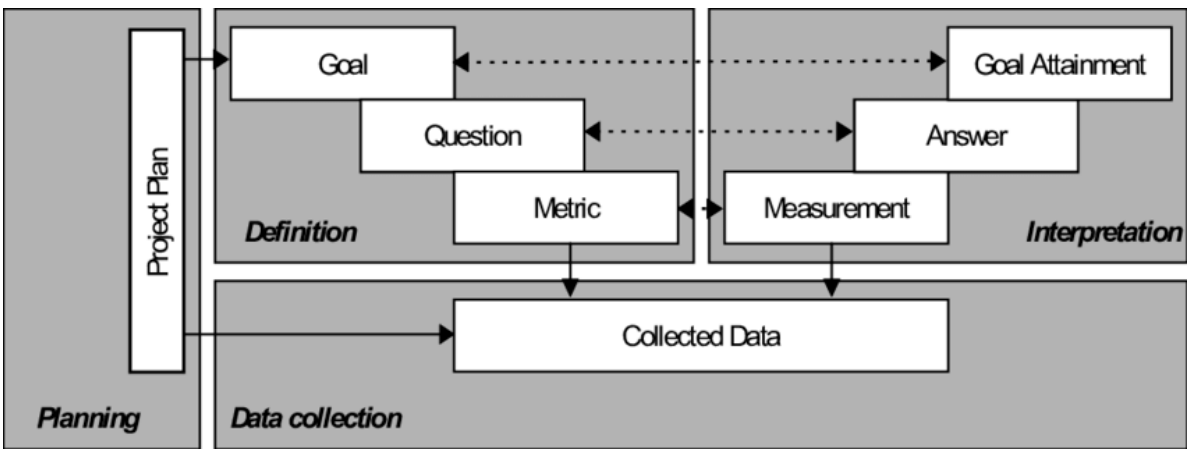

Рисунок 3.1 – Метод «ціль-запитання-метрика»

Кожен атрибут має свою вагу. Атрибути естетики, зручності використання та мультимедіа найбільше впливають на якість сприйняття користувачами, тоді як вміст і репутація здебільшого розглядають з точки зору адміністраторів.

Система оцінки якості забезпечує просту шкалу відповідності. Ця шкала починається з 0% і закінчується на 100%. За цією шкалою 0% показує низький показник якості, а за шкалою 100% — чудовий показник якості. Це QCF (Quality Compliance Framework) для веб-додатків.

Оцінка якості:

$$\text{Оцінка якості} = \frac{\sum \text{Оцінка характеристики QCF}}{\text{Кількість характеристик}}. \quad (3.1)$$

Оцінка відповідності якості для характеристик і підхарактеристик (оцінка QCF):

$$\text{Оцінка характеристик QCF} = \frac{\sum \text{Величина оцінки характеристики QCF}}{\text{Кількість характеристик}}. \quad (3.2)$$

Рейтинг відповідності якості атрибута:

$$\text{Індикатор якості} = \frac{\text{Отриманий бал} * 100\%}{\text{Можлива помилка}}. \quad (3.3)$$

Після вибору атрибутів кожен з них розбивається на ключові характеристики. Метод «ціль-запитання-метрика» вибирає показники, які призначаються кожній ознаці. Потім для кожної характеристики методом «ціль-запитання-метрика» підбирається набір метрик для кожної характеристики. На наступному етапі моделі оцінки якості визначають вагові коефіцієнти для кожної характеристики. Структура моделі оцінки якості сайту, представлена в таблиці 3.1. [25].

Як зазначалося, наведена вище модель спрямована на узагальнення оцінки якості веб-сайту для всіх галузей, чого на практиці неможливо досягти через

певну специфіку конкретної галузі. Зокрема, для оцінки якості розробленого додатку пропонується змінити значення ваги окремих атрибутів.

Таблиця 3.1 – Загальна структура моделі оцінки якості [25]

Метрика	Характеристика	Ваговий коефіцієнт характеристики	Атрибут
1	2	3	4
Розмір зображень	Зображення	0,3	Естетика
Одне велике зображення на одній сторінці			
Зображення ALT			
Зображення-посилання			
Стандартний розмір таблиці	Роздільна здатність сторінки	0,2	
Оптимізація роздільної здатності сторінки			
Використання кількох кольорів	Колір	0,3	
Використання безпечного кольору			
Обмеження кольорів для колірної сліпоти			
Підкреслення тексту	Акцент	0,2	
Атрибути CSS	Послідовність	0,4	Зручність використання
Використання фреймів	Навігація	0,4	
Посилання на домашню сторінку			
Панель навігації			
Мітка посилання, таблиці та форми	Анотація	0,2	
Опис META			
Підтримка плагіна	Підтримка плагіна	0,2	Мультимедіа
Мультимедійні компоненти	Мультимедійні компоненти	0,2	
Один медіа на одній сторінці	Один медіа на одній сторінці	0,3	
Використання ескізів	Використання ескізів	0,3	
Дошка оголошень	Дошка оголошень	0,2	Вміст
Інформаційний довідник	Інформаційний довідник	0,2	
Пошукова система	Пошукова система	0,4	
Уникнення автоматичного оновлення	Уникнення автоматичного оновлення	0,2	
Відгук клієнтів	Відгук клієнтів	0,3	Репутація
Веб-трафік	Веб-трафік	0,3	
Доменне ім’я	Доменне ім’я	0,2	
Публічна інформація	Публічна інформація	0,2	

Зокрема, враховуючи те, що розроблений додаток використовують користувачі для моніторингу власного фінансового стану, то вагові коефіцієнти атрибутів «Естетика» та «Зручність використання» повиннні мати більше значення, ніж ваговий коєфіцієнт атрибута «Репутація» (табл. 3.2). Вагові коефіцієнти інших атрибутів якості пропонують залишити без змін.

Таблиця 3.2 – Порівняння оригінальних та пропонованих вагових коефіцієнтів атрибутів якості

Атрибут	Ваговий коефіцієнт атрибута відповідно до оригінальної моделі	Ваговий коефіцієнт атрибута
Естетика	0,3	0,4
Зручність використання	0,2	0,3
Мультимедіа	0,1	0,1
Вміст	0,1	0,1
Репутація	0,3	0,1

Детальні результати оцінки якості розробленого додатку представлені в таблиці 3.3.

Таблиця 3.3 – Отримані значення показників і якісних характеристик для розробленого додатку

Метрика	Значення метрики	Характеристика	Значення характери-стики
1	2	3	4
Розмір зображень	1	Зображення	0,23
Одне велике зображення на одній сторінці	1		
Зображення ALT	1		
Зображення-посилання	0		
Стандартний розмір таблиці	1	Роздільна здатність сторінки	0,2
Оптимізація роздільної здатності сторінки	1		
Використання кількох кольорів	1	Колір	0,3
Використання безпечного кольору	1		
Обмеження кольорів для колірної сліпоти	1		
Підкреслення тексту	1	Акцент	0,2
Атрибути CSS	1	Послідовність	0,4

Метрика	Значення метрики	Характеристика	Значення характери-стики
Використання фреймів	1	Навігація	0,13
Посилання на домашню сторінку	0		
Панель навігації	0		
Мітка посилання, таблиці та форми	1	Анотація	0,2
Опис META	1		
Підтримка плагіна	1	Підтримка плагіна	0,2
Мультимедійні компоненти	1	Мультимедійні компоненти	0,2
Один медіа на одній сторінці	1	Один медіа на одній сторінці	0,3
Використання ескізів	0	Використання ескізів	0
Дошка оголошень	0	Дошка оголошень	0
Інформаційний довідник	0	Інформаційний довідник	0
Пошукова система	0	Пошукова система	0
Уникнення автоматичного оновлення	1	Уникнення автоматичного оновлення	0,2
Відгук клієнтів	0	Відгук клієнтів	0
Веб-трафік	1	Веб-трафік	0,2
Доменне ім’я	1	Доменне ім’я	0,2
Публічна інформація	0	Публічна інформація	0

Загальні результати оцінки якості розробленого додатку наведені в таблиці 3.4.

Таблиця 3.4 – Остаточна оцінка якості розробленого додатку

Атрибут	Естетика	Зручність використання	Мульти медіа	Вміст	Репутація	Загальна оцінка
Розроблений SPA-додаток	0,93	0,73	0,7	0,2	0,4	0,6

Найгірші показники за атрибутами вмісту та репутації сайту, що призвело до невисокої загальної оцінки.

3.3 Рекомендації до підвищення ефективності додатку

Під час розробки була використана найпоширеніша мовна версія сайту – англійська. Англійська мова зазвичай вважається міжнародною. Вона займає

третє місце у світовому рейтингу після іспанської та китайської за кількістю людей, які є її носіями. Також англійська – друга за значимістю у багатьох країнах, де вона не є основною, її активно вивчають у навчальних закладах та на курсах. Для підвищення ефективності додатку пропонується зробити додаток мультимовним. Мультимовність стане конкурентною перевагою ресурсу та допомаже розкрити його потенціал.

Вихід проекту на міжнародну арену відкриває нові можливості для популяризації бренду та підвищення прибутку за рахунок залучення іноземних клієнтів. Більшість відомих та успішних компаній мають модифіковану під різні регіони платформу.

Можливість ознайомитися з вмістом веб-сторінки рідною мовою – вагомий бонус для користувача. У конкурентному середовищі важливо утримати увагу клієнта, створити умови, щоб він швидко та без зусиль отримав потрібну інформацію, оцінив переваги компанії та став активним.

Сайт, адаптований під користувачів з інших країн, є ефективним інструментом збільшення продажів. Іноземні відвідувачі, які мають можливість прочитати інформацію своєю мовою, почуваються з таким ресурсом комфортно і більше йому довіряють.

Професійно перекладений та оптимізований під інший регіон сайт – це надійний спосіб збільшити трафік ресурсу, розширити цільову аудиторію та випередити конкурентів, зосереджених виключно на локальному ринку. Надання користувачеві якісного перекладу та використання ефективної маркетингової стратегії веде до зростання прибутку з урахуванням нових споживачів.

Створення мультимовного сайту включає комплекс дій і, як правило, вимагає участі кількох компетентних фахівців. Помилки в цьому питанні можуть залишитися непоміченими розробником, однак їх буде виявлено іноземною аудиторією. Щоб одразу скласти про себе сприятливе враження у користувача, слід звернути увагу на важливі аспекти перекладу та налаштування мультимовності ресурсу.

У майбутньому SPA-додаток для відстеження доходів і витрат може бути додатково вдосконалено, щоб включати такі функції:

– Додаток можна розширити, включивши в нього сканування штрих-коду на ціннику, що зменшує зусилля при введенні даних у поля введення.

– Група: окрім ведення особистого журналу, можливо розширити цю систему, щоб включити спільну групу витрат.

– Програма може бути розроблена таким чином, щоб створювати щомісячний аналіз і звіт про доходи та витрати користувача, щоб забезпечити краще розуміння користувача та отримати контроль над його витратами.

– Система сповіщень може бути увімкнена у випадку, коли витрати перевищують доходи, отримані користувачем, щоб попередити його про ситуацію.

ВИСНОВКИ

Щоденне відстеження витрат може допомогти заощадити гроші користувача, але також може допомогти поставити фінансові цілі на майбутнє та працювати над ними. Якщо користувач точно знає, куди спрямовується сума щомісяця, він може легко побачити, де можуть бути зроблені певні скорочення та компроміси. Проект успішно уникає ручного розрахунку, та має голосове керування для заощадження часу користувача. Модулі розроблені ефективно, надійно, а також привабливо.

Розроблено SPA-додаток із голосовим керуванням на основі платформи Speechly API для моніторингу фінансового стану споживача.

Доступ до розробленого SPA-додатку можна отримати з веб-браузера, такого як Google Chrome або Mozilla Firefox, що забезпечує портативне робоче середовище. Додаток містить усі функції цифрового ведення записів із привабливим візуальним зображенням та графікою витрат і навіть усуває потребу в фізичних записах за допомогою голосового керування.

SPA-додаток було розроблено з використанням ReactJS в якості фреймворка. ReactJS — це декларативна, ефективна та гнучка бібліотека інтерфейсу JavaScript для створення компонентів інтерфейсу користувача, які можна повторно використовувати. Це інтерфейсна бібліотека, заснована на компонентах, і є вільно доступною для всіх, хто відповідає лише за зовнішній вигляд програми або рівень перегляду програми. Мета ReactJS полягає в тому, щоб розробники могли легко розробляти інтерфейси користувача (UI), розділяючи їх на різні компоненти, а також розробляти швидкі системи. Він використовує Virtual DOM (об’єкт JavaScript), що підвищує ефективність та продуктивність програми.

При оформленні інтерфейсу було використано Material UI.

У проекті є функція розпізнавання голосу, яка була покращена за допомогою бібліотеки Speechly. Це було використано для того, щоб заощадити час користувачів на введенні тексту.

Програма розділена на три основні компоненти, а саме: надходження, витрати та розділ створення транзакцій. Компонент «Створення транзакцій» керує рештою системи, він дозволяє користувачеві вибирати з двох категорій, будь то витрати чи надходженння, а також вибрати зі списку типів доходів чи витрат, ввести суму та дату та створити запис. Після створення транзакцій кругові діаграми створюються в розділі доходів або витрат на основі типу транзакції в режимі реального часу, показуючи розподіл доходів і витрат відповідно.

Усі ці функції створення та видалення транзакцій також реалізуються за допомогою голосових команд за допомогою голосового механізму Speechly, який є механізмом на основі штучного інтелекту, що забезпечує розпізнавання голосу для впровадження в системі.

Стилізовані компоненти використовуються для стилізації веб-програми та забезпечення адаптивності для всіх пристроїв.

ПЕРЕЛІК ПОСИЛАНЬ

1. Винцюк Т.К. Анализ, распознавание и интерпретация речевых сигналов. – Киев, Наук. думка, 1987. – 264 с.

2. Винцюк Т.К. Распознавание слов устной речи методами динамического программирования // Кибернетика. 1968. № 1. С. 81–88.

3. Baker J.K. Stochastic modeling for automatic speech understanding// Speech Recognition / ed.: D.R. Reddy. New York: Academic Press, 1975. P. 521–542.

4. Джелинек Ф. Распознавание непрерывной речи с помощью статистических методов // ТИИЭР. 1976. Т. 64. No4. С.131–160.

5. Марков А.А. Пример статистического исследования над текстом «Евгения Онегина», иллюстрирующий связь испытаний в цепь// Известия Академии наук. СПб. VI. Т. 7. 1913. No3. С. 153–162.

6. Google Voice Search. URL: https://voice.google.com/.

7. Dragon NaturallySpeaking. URL: https://www.nuance.com/dragon.html.

8. ViaVoice. URL: https://www.ibm.com/common/ssi/cgi-bin/ssialias?infotype=an&subtype=ca&htmlfid=897/ENUS202-174&language=enus&appname=skmwww

9. Sphinx. URL: http://sphinxsearch.com/.

10. Петров А. О. Моделі та методи розпізнавання мови. Сучасний захист інформації. 2012. № 1. С. 25-33.

11. Zhang Guofeng. Analysis of Speech Recognition Methods for Deep Learning under Artificial Intelligence. Electronics and Software Engineering. 2020(11). pp. 176-177.

12. Na Tian. Application and Challenge of Artificial Intelligence Technology in Language Learning. Modern Information Technology. 2019. 3(19). pp. 109-110.

13. Bi Xinwen. Speech Recognition Method Based on Deep Learning. Electronics and Software Engineering. 2019(08). pp. 245.

14. Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In International conference on machine learning. PMLR. 2020. pp. 1597–1607.

15. Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. arXiv preprint arXiv:1807.03748. 2018.

16. Daniel Galvez, Greg Diamos, Juan Manuel Ciro Torres, Keith Achorn, Anjali Gopi, David Kanter, Max Lam, Mark Mazumder, and Vijay Janapa Reddi. The people's speech: A large-scale diverse English speech recognition dataset for commercial usage. 2021.

17. React JS. URL: https://reactjs.org/.

18. Pratik Sharad Maratkar, Pratibha Adkar. React JS - An Emerging Frontend JavaScript Library. Iconic Research And Engineering Journals. Volume 4. Issue 12. 2021. Page 99-102.

19. React Virtual DOM. URL: http://stackoverflow.com/questions/21109361/why-is-Reacts-concept-ofvirtual-dom-said-to-be-more-performant-than-dirty-mode.

20. Material UI. URL: https://material-ui.com.

21. Speechly API. URL: https://www.speechly.com/.

22. Fitzpatrick R. Additional Quality Factors for the World Wide Web. The Second World Congress for Software Quality. Yokohama, Japan: Union of Japanese Scientists and Engineers (JUSE), May 2000.

23. Olsina L., Molina H. How To Measure And Evaluate Web Applications In A Consistent Way. G. Rossi & et al Ed., Web Engineering: Modelling and Implementing Web Applications. London: Springer. 2008, pp.385–420, ch. 8.

24. Nakwichian S. and Sunetnanta. User-Centric Web Quality Assessment Model. 7th National Computer Science and Engineering Conference (NCSEC2003). Burapha University, Chonburi, Thailand, October 2003.

25. Kravchenko, Y., Leshchenko, O., Dakhno, N., Radko, M. Comparative evaluation of a universities websites quality. CEUR Workshop Proceedingsthis link is disabled, 2022, 3132, pp. 166–175.