**Пояснювальна записка  
до курсової роботи**

на тему: Голосовий помічник

КПІ.ІП-1315.045490.02.81

Київ – 2023

Зміст

[Перелік умовних позначень 4](#_Toc154859925)

[Вступ 5](#_Toc154859926)

[1 АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 6](#_Toc154859927)

[1.1 Загальні положення 6](#_Toc154859928)

[1.2 Змістовний опис і аналіз предметної області 9](#_Toc154859929)

[1.3 Аналіз існуючих технологій та успішних IT-проєктів 10](#_Toc154859930)

[1.3.1 Аналіз відомих алгоритмічних та технічних рішень 10](#_Toc154859931)

[1.3.2 Аналіз допоміжних програмних засобів та засобів розробки 13](#_Toc154859932)

[1.3.3 Аналіз відомих програмних продуктів 15](#_Toc154859933)

[1.4 Аналіз вимог до програмного забезпечення 17](#_Toc154859934)

[1.4.1 Розроблення функціональних вимог 20](#_Toc154859935)

[1.4.2 Розроблення нефункціональних вимог 24](#_Toc154859936)

[1.5 Постановка задачі 25](#_Toc154859937)

[Висновки до розділу 26](#_Toc154859938)

[2 МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 27](#_Toc154859939)

[2.1 Моделювання та аналіз програмного забезпечення 27](#_Toc154859940)

[2.2 Архітектура програмного забезпечення 30](#_Toc154859941)

[2.3 Конструювання програмного забезпечення 32](#_Toc154859942)

[2.3.1 Оригінальні алгоритми та модифікації 32](#_Toc154859943)

[2.3.2 Структури даних та програмні структури 32](#_Toc154859944)

[2.3.3 Утиліти, бібліотеки та інше ПЗ 33](#_Toc154859945)

[2.4 Аналіз безпеки даних 34](#_Toc154859946)

[Висновки до розділу 34](#_Toc154859947)

[3 АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 36](#_Toc154859948)

[3.1 Аналіз якості ПЗ 36](#_Toc154859949)

[3.2 Опис процесів тестування 37](#_Toc154859950)

[Висновки до розділу 51](#_Toc154859951)

[4 ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ 53](#_Toc154859952)

[4.1 Розгортання програмного забезпечення 53](#_Toc154859953)

[4.2 Підтримка програмного забезпечення 54](#_Toc154859954)

[Висновки до розділу 55](#_Toc154859955)

[ВИСНОВКИ 56](#_Toc154859956)

[СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 58](#_Toc154859957)

Перелік умовних позначень

|  |  |  |
| --- | --- | --- |
| IDE | – | Integrated Development Environment – інтегроване середовище розробки. |
| JDK | – | Java development kit |
| JAR | – | Java archive |
| ПЗ | – | Програмне забезпечення |
| ГП | – | Голосовий помічник |
| ШІ | – | Штучний інтелект |

Вступ

У сучасному цифровому світі голосові помічники (ГП) стають неодмінною складовою повсякденного життя, спрощуючи взаємодію людей з технологіями та надаючи широкі можливості для зручного отримання інформації та виконання різноманітних завдань. Актуальність вивчення та розробки ГП базується на швидкому розвитку галузі штучного інтелекту, зростанні інтересу до голосового, дистанційного інтерфейсу та збільшенні можливостей технологій обробки природної мови[1].

Світові тенденції показують постійне удосконалення ГП у напрямку поліпшення їхньої розпізнавальної точності, розширення функціональності та інтеграції з іншими інформаційними системами. Поява нових технологій, таких як машинне навчання та нейронні мережі, створює нові можливості для розробки більш ефективних та інтелектуальних ГП[2].

Однак в контексті даної курсової роботи акцент робиться на відмінності від наявних ГП. Велика частина наявних рішень є комерційними продуктами з обмеженим доступом до внутрішніх механізмів та алгоритмів. Даний ГП відрізняється тим, що він базується на принципах відкритого програмного забезпечення та вільної ліцензії, надаючи користувачам можливість активної участі в розвитку та вдосконаленні системи.

Метою даної курсової роботи є ретельний аналіз сучасного стану ГП, визначення перспектив розвитку та розробка концепції нового голосового помічника, спрямованого на задоволення потреб користувачів у контексті швидкозмінного технологічного середовища. У роботі буде розглянуто сфери застосування ГП та обґрунтовано його актуальність у різних галузях життя та бізнесу.

# АНАЛІЗ ВИМОГ ДО ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

В умовах сучасного інформаційного середовища розробка програмного забезпечення визнається як ключовий фактор для вирішення широкого спектра завдань та оптимізації бізнес-процесів. Початковий етап цього процесу — аналіз вимог, що визначає основні потреби та очікування від майбутнього продукту.

Аналіз вимог — це важливий крок, на якому встановлюються як функціональні, так і нефункціональні характеристики програмного продукту. Цей етап визначає ключові параметри та особливості продукту, що є визначальними для його успішної розробки та функціонування.

У даній курсовій роботі розглядаються загальні аспекти визначення вимог, їх класифікація та вплив на подальший процес розробки. Аналізується досвід теперішніх голосових помічників, зокрема їх функціональні можливості та нефункціональні характеристики, що стає основою для формулювання вимог до нового голосового помічника з відкритим вихідним кодом. Особливий акцент робиться на забезпеченні прозорості та спільної участі у процесі його розвитку.

## Загальні положення

У цьому розділі представлено всебічний огляд голосових помічників, досліджено загальні положення, визначено ключові поняття та проаналізовано тенденції розвитку цієї галузі.

Поява голосових помічників є важливою віхою на перетині штучного інтелекту (ШІ) та людино-машинної взаємодії. Ці інтелектуальні програми стали повсюдними, вбудованими в різні пристрої - від смартфонів до розумних колонок і натільних пристроїв. З розвитком технологій можливості голосових помічників розвиваються, впливаючи на різні аспекти нашого повсякденного життя[3].

Концепція голосової взаємодії з машинами виникла кілька десятиліть тому, коли ранні експерименти та прототипи проклали шлях до складних голосових помічників, з якими ми стикаємося сьогодні. Простеження історичної траєкторії дає цінну інформацію про еволюційний шлях цих технологій.

Голосовий асистент (помічник) - це інтелектуальне програмне забезпечення, яке реагує на голосові команди, використовуючи штучний інтелект і машинне навчання[4].

Прикладами таких програм є Alexa, Siri, Google Assistant і Bixby, які можна знайти на таких пристроях, як смартфони та смарт-колонки. Вони виконують різні завдання - від дзвінків до керування пристроями розумного дому[5].

Ключові компоненти ГП включають перетворення мови в текст, тексту в мову, прийняття рішень і базову архітектуру[4].



1. Загальна схема роботи ГП

Голосові асистенти забезпечують зручну та ефективну взаємодію завдяки обробці природної мови.

Їх впровадження може стати значущим кроком у напрямку зручності та ефективності в щоденному житті, наприклад:

* Залишатися в курсі подій - доступ до широкого спектра інформації, доступної в Інтернеті, дізнатися про погоду, отримати останні новини, перевірити інформацію про дорожній рух тощо.
* Керування музикою - наказуємо їм відтворювати музику, яка відповідає нашому поточному настрою.
* Керування пристроями - підключення до інших розумних пристроїв і виконання команд, таких як "увімкнути телевізор".
* Фінансові оновлення - доступ до інформації про банківський рахунок, перевірка балансу, перегляд транзакцій і навіть запит продуктів, таких як картки.
* Керування розкладом - ефективна організація часу без використання мобільного пристрою, годинника чи щоденника.
* Підтримка доступності - сприяння людям з вадами зору, людям похилого віку та залежним особам за допомогою функції голосового пошуку.
* Посилена безпека - інтеграція домашніх камер безпеки з можливістю віддаленого перегляду прямих трансляцій на наших пристроях.

З іншого боку, голосові асистенти мають свої недоліки, які важливо враховувати при їхньому використанні. До числа ключових труднощів входять:

* Занепокоєння щодо конфіденційності - постійна функціональність означає постійне прослуховування, що підвищує ймовірність запису розмов.
* Доступ до персональних даних - надання доступу до облікових записів електронної пошти, списків контактів та іншої особистої інформації, необхідної для роботи асистентів.
* Знайомство з нашими звичками - вони вловлюють набагато більше, ніж те, що ми говоримо і просимо. Вони також можуть витягувати дані з підключених до них пристроїв і збирати інформацію про те, о котрій годині ми повертаємося додому, та інші дані.
* Ризик крадіжки особистих даних - попри те, що вони розпізнають лише наш голос, ризик проникнення хакерів та отримання ними доступу до персональних даних залишається.

## Змістовний опис і аналіз предметної області

На сучасному етапі розвитку ІТ-технологій голосові помічники є важливою складовою нашого цифрового середовища. Однак існують певні недоліки та виклики, які потребують уваги та подальшого вдосконалення, серед них[6]:

* Низька точність розпізнавання мови - деякі голосові помічники можуть некоректно розпізнавати акценти чи нестандартне мовлення, що призводить до помилок у виконанні команд.
* Обмежені можливості розуміння контексту - голосові асистенти іноді не можуть ефективно узгоджувати різні запитання та відповіді у більшому контексті, що призводить до неповних чи неправильних відповідей.
* Питання конфіденційності та безпеки - збір та обробка великого обсягу особистих даних створює загрозу конфіденційності та можливість несанкціонованого доступу до цих даних.

В межах виконання курсової роботи обрано напрямок забезпечення безпеки та конфіденційності за допомогою відкритого вихідного коду, відмови від зберігання в базі даних інформації про користувачів та постійного контролю за прослуховуванням. Основний акцент робиться на забезпеченні користувачів контролем над процесом та відсутності комерційних аспектів[7].

Заходи забезпечення безпеки та конфіденційності включають:

* Використання відкритого вихідного коду - активне використання відкритого вихідного коду надає можливість експертам і громаді перевіряти код на наявність можливих вразливостей та документувати, як сам голосовий помічник обробляє дані користувачів.
* Відмова від зберігання в базі даних - інформація про користувачів не зберігається в базі даних, що мінімізує ризик витоку конфіденційної інформації через можливі порушення безпеки бази даних.
* Користувач-центричний підхід - користувачі мають повний контроль над процесом прослуховування. Вони можуть визначати момент активації голосового помічника та його зупинки.
* Відсутність комерційних цілей - проєкт фокусується на вирішенні завдань без наміру отримання прибутку, що убезпечує від використання даних користувачів у комерційних цілях.

## Аналіз існуючих технологій та успішних IT-проєктів

Проаналізуємо відоме на сьогодні алгоритмічне забезпечення у даній області та технічні рішення, що допоможуть у реалізації голосового помічнику. Далі будуть розглянуті допоміжні програмні засоби, засоби розробки та готові програмні рішення.

### Аналіз відомих алгоритмічних та технічних рішень

У світі голосових помічників визначені ключові алгоритми для розпізнавання мовлення та обробки голосових команд. Розглянемо основні алгоритми, які використовуються у відомих голосових помічниках.

* Google Speech-to-Text - Використовує глибоке навчання для розпізнавання мовлення.

Забезпечує високу точність та широку підтримку мов, це надійний інструмент транскрипції для глобальної аудиторії з підтримкою понад 125 мов[8].

Він пропонує розпізнавання потокового мовлення в реальному часі, адаптивність до специфічних термінів і завадостійкість. Користувачі можуть керувати інфраструктурою за допомогою локальної опції та вибирати з моделей для конкретного домену.

Функції включають багатоканальне розпізнавання, фільтрацію контенту, автоматичну пунктуацію (бета-версія) та діалогізацію мовлення (бета-версія). Ціноутворення базується на кількості обробленого аудіо щомісяця, що вимірюється з кроком в одну секунду, забезпечуючи прозору та гнучку структуру витрат[9].

* CMUSphinx - Робить акцент на локальному розпізнаванні мовлення, що може бути корисним для деяких сценаріїв.

Забезпечує можливість роботи без підключення до Інтернету. CMUSphinx є провідним рішенням для розпізнавання мови з численними перевагами.

Завдяки найсучаснішим алгоритмам, він вирізняється ефективним розпізнаванням мовлення, пристосованим до платформ з обмеженими ресурсами. Його гнучкий дизайн робить акцент на розробці практичних додатків, а не на чистих дослідженнях, що робить його цінним інструментом для різноманітних проектів.

CMUSphinx підтримує кілька мов, зокрема англійську, англійську для США, англійську для Великобританії, французьку, китайську, німецьку, голландську тощо, з можливістю побудови моделей для інших мов. Платформа працює за ліцензією, що дозволяє комерційне розповсюдження, і пропонує комерційну підтримку. Завдяки активній розробці та графіку випусків, активній спільноті з понад 400 користувачів у групі LinkedIn CMUSphinx, а також ряду інструментів для різних цілей розпізнавання мови, таких як визначення ключових слів, вирівнювання та оцінка вимови, CMUSphinx виділяється як надійне та універсальне рішення для потреб розпізнавання мови[10].

* DeepSpeech (Mozilla) - Використовує нейронні мережі для глибокого розпізнавання мовлення.

Open-source проект, що надає гнучкість для модифікацій та удосконалень.

Має на меті створити доступну та відкриту систему розпізнавання мовлення, призначену для роботи на різних платформах і не потребуючу висококласного обладнання.

Випущений під ліцензією Mozilla Public License, він використовує рекурентну нейронну мережу для обробки мовних спектрограм і генерування транскрипцій англійського тексту.

Завдяки наявності попередньо навчених моделей і можливості користувачам навчати свої власні, DeepSpeech підтримує різні мови і платформи. Документація охоплює створення бінарних файлів, декодування, підрахунок балів та методи оптимізації, що робить його універсальним і зручним у використанні рішенням. API надаються для C, .NET Framework, Java, JavaScript (NodeJS/ElectronJS) та Python, з прикладами використання для кожної мови, що сприяє досягненню мети DeepSpeech - широкому використанню[11].

* Wit.ai - Використовується для розпізнавання не тільки тексту, але і сутностей та інтентів у мовленні.

Частина компанії Meta, пропонує зручну платформу для створення та навчання додатків для розпізнавання мови. Користувачі можуть навчати програму на прикладах, пов'язуючи наміри з висловлюваннями, щоб керувати процесом навчання. Система вдосконалюється з більшою кількістю прикладів і дозволяє безперервну перевірку.

Користувачі можуть запитувати свій додаток через Wit.ai API, налаштовуючи рівні достовірності намірів для точного налаштування.

Підтримуючи сутності для вилучення конкретної інформації, Wit.ai робить акцент на практичних кроках, таких як створення намірів та сутностей[12].

Платформа полегшує безперешкодну розробку та навчання додатків для розпізнавання мови. Екосистема Wit.ai заохочує співпрацю спільноти, надаючи доступ до знань, ресурсів та інструментів для прискорення проектів з розпізнавання природної мови.

В якості технічного рішення було обрано CMUSphinx через його акцент на локальному розпізнаванні мовлення та можливості працювати без підключення до Інтернету. Такі особливості роблять CMUSphinx оптимальним вибором для проектів, де важливо забезпечити ефективне розпізнавання мовлення в умовах обмежених ресурсів та забезпечити незалежність від мережі для нормальної роботи. Крім того, гнучкий дизайн та активна розробка роблять CMUSphinx надійним та універсальним рішенням для різноманітних проектів.

### Аналіз допоміжних програмних засобів та засобів розробки

В даному пункті розглядається низка допоміжних програмних засобів, які були вивчені та проаналізовані для використання у розробці проекту. До цих засобів включаються мови програмування, інтегровані середовища розробки (IDE) та бібліотеки для виконання завдань проекту.

#### Мови програмування

* Java

Обрана через свою платформенну незалежність, високу надійність, та велику спільноту розробників. Вище інших мов за стабільністю та широким спектром застосувань[13].

* Python

Зручна для роботи з різноманітними бібліотеками та широким спектром задач, включаючи обробку даних та штучний інтелект. Гнучка та зручна, але може вимагати додаткових оптимізацій для великих проектів[14].

#### Інтегровані середовища розробки (IDE)

* IntelliJ IDEA

Можливості автоматизації, висока продуктивність та зручний інтерфейс. Зручна та потужна, надає багато інструментів для комфортної розробки[15].

* Eclipse

Відкрите програмне забезпечення, широко використовується для розробки різноманітних застосунків. Має велику спільноту користувачів, але може бути менш продуктивною порівняно з IntelliJ IDEA[16].

#### Бібліотеки

* Sphinx4

Бібліотека для розпізнавання мови та обробки голосу. Висока точність розпізнавання та широкий функціонал[17].

* Google Speech Recognition API

Хмарна платформа для розпізнавання мови в реальному часі. Може вигідно використовуватися для великих обсягів даних та високої швидкості розпізнавання[18].

* Swing

Бібліотека для створення графічного інтерфейсу користувача. Проста у використанні та добре інтегрується з Java, але існують більш сучасні альтернативи[19].

* JavaFX

Сучасна бібліотека для створення графічного інтерфейсу та багатофункціональних додатків. Забезпечує більш сучасний та гнучкий інтерфейс, але може вимагати більше зусиль для вивчення[20].

* FreeTTS

Бібліотека для генерації голосу. Використовується для створення текст-на-мову та імітації голосу. Додає можливість генерації голосу, що може бути корисним для реалізації голосового інтерфейсу та взаємодії з користувачем[21].

Після ретельного аналізу визначено, що оптимальним вибором для даного проекту є використання мови програмування Java, інтегрованої середовищі розробки IntelliJ IDEA, бібліотек Sphinx4 для розпізнавання мовлення, бібліотеки Swing для створення графічного інтерфейсу користувача, та бібліотеки FreeTTS для генерації голосу. Цей вибір базується на їхній високій продуктивності, надійності та специфічних функціях, необхідних для успішної реалізації голосового інтерфейсу та взаємодії з користувачем.

### Аналіз відомих програмних продуктів

Розглянувши наявні успішні IT-проекти, складемо таблиці порівняння.

#### Amazon Alexa

Amazon Alexa - це голосовий асистент, розроблений компанією Amazon. Він взаємодіє з користувачем через голосові команди, надаючи інформацію, виконуючи завдання та контролюючи підключені пристрої[22].

1. Порівняння з аналогом (Alexa)

|  |  |  |  |
| --- | --- | --- | --- |
| Функціонал | Vocalia | Alexa | Пояснення |
| Розпізнавання голосових команд | Так | Так | Обидва продукти мають можливість розпізнавання голосу та виконання команд. |
| Забезпечення інформацією | Так | Так | Обидва можуть надавати інформацію користувачам з різних джерел. |
| Управління підключеними пристроями | Ні | Так | Alexa може керувати підключеними пристроями у домашній мережі, в той час як Vocalia - ні. |

#### Siri

Siri - голосовий асистент, розроблений Apple для операційних систем (ОС) iOS, iPadOS, macOS та watchOS. Вона використовується для виконання завдань та отримання інформації через голосові команди[23].

1. Порівняння з аналогом (Siri)

|  |  |  |  |
| --- | --- | --- | --- |
| Функціонал | Vocalia | Siri | Пояснення |
| Розпізнавання голосових команд | Так | Так | Обидва продукти мають можливість розпізнавання голосу та виконання команд. |
| Інтеграція з Apple-продуктами | Ні | Так | Siri може взаємодіяти з іншими пристроями в екосистемі Apple, в той час як Vocalia - ні. |
| Особистий асистент | Так | Так | Обидва продукти можуть служити особистими асистентами для користувача. |

#### Google Assistant

Google Assistant - голосовий асистент, розроблений Google. Він взаємодіє з користувачем через голосові команди, надаючи інформацію, керуючи пристроями та виконуючи завдання[24].

1. Порівняння з аналогом (Google Assistant)

|  |  |  |  |
| --- | --- | --- | --- |
| Функціонал | Vocalia | Google Assistant | Пояснення |
| Розпізнавання голосових команд | Так | Так | Обидва продукти мають можливість розпізнавання голосу та виконання команд. |
| Інтеграція з Google-продуктами | Частково | Так | Google Assistant може інтегруватись з іншими послугами та пристроями від Google, Vocalia також інтегрується з деякими Google-продуктами. |
| Пошук інформації онлайн | Так | Так | Обидва можуть шукати та надавати інформацію з Інтернету. |

#### Bixby

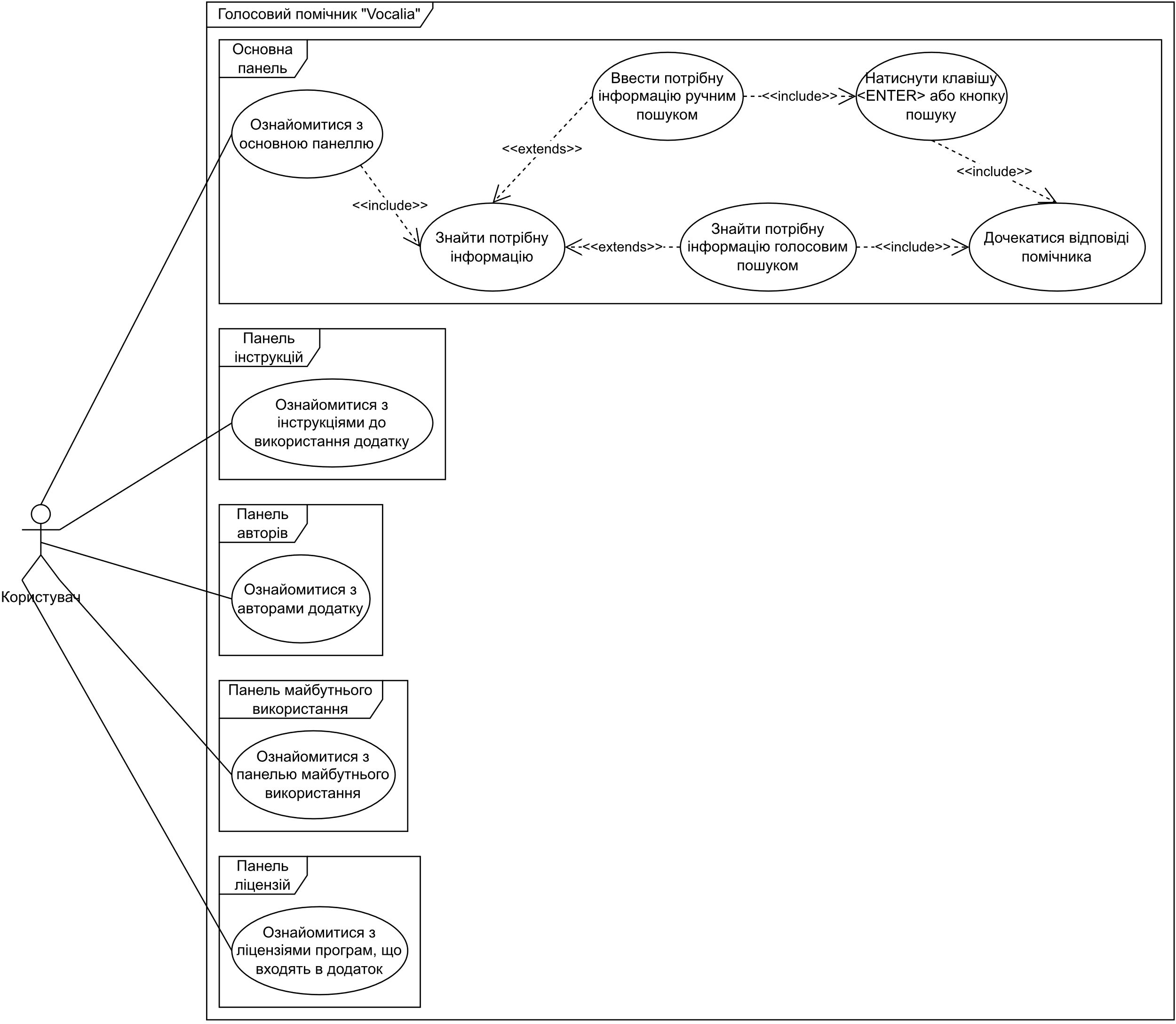
1. Порівняння з аналогом (Bixby)[25]

|  |  |  |  |
| --- | --- | --- | --- |
| Функціонал | Vocalia | Bixby | Пояснення |
| Розпізнавання голосових команд | Так | Так | Як і дипломний проєкт, Bixby може розпізнавати голосові команди користувача. |
| Інтеграція з Samsung-продуктами | Ні | Так | Bixby спеціально розроблений для взаємодії зі смартфонами, телевізорами та іншими пристроями від Samsung, в той час як Vocalia – десктопний додаток на персональний комп’ютер. |

Власно-розроблений голосовий асистент враховує найкращі практики та характеристики успішних аналогів, прагнучі створити продукт, що задовольнить стандарти якості та ефективності в галузі голосових технологій, і одночасно буде з відкритим вихідним кодом. Це стратегічне рішення відкриває можливості для широкого співробітництва та сприяє взаємодії з розробниками з усього світу.

## Аналіз вимог до програмного забезпечення

Головною функцією програмного забезпечення є вдосконалення ефективності взаємодії з користувачем та автоматизація повсякденних завдань, більше функцій можна побачити на рисунку 1.2.



1. Діаграма варіантів використання

В таблицях 1.5 – 1.11 наведені варіанти використання програмного забезпечення.

1. Варіант використання UC-01

|  |  |
| --- | --- |
| Use case name | Вхід користувача на головну сторінку |
| Use case ID | UC-01 |
| Goals | Успішний вхід користувача на головну сторінку |
| Actors | Користувач |
| Trigger | Користувач бажає увійти на головну сторінку |
| Pre-conditions | У користувача встановлений додаток “Vocalia” |
| Flow of Events | Користувач відкриває додаток “Vocalia”, після чого відкривається головна сторінка та користувач опиняється на неї. |
| Extension | Відсутні. |
| Post-Condition | Завантаження голосового асистенту та інших модулів |

1. Варіант використання UC-02

|  |  |
| --- | --- |
| Use case name | Вхід користувача на сторінку інструкцій |
| Use case ID | UC-02 |
| Goals | Успішний вхід користувача на сторінку інструкцій |
| Actors | Користувач |
| Trigger | Користувач бажає увійти на сторінку інструкцій |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться не на цій сторінці. |
| Flow of Events | Користувач переходить на сторінку інструкцій. |
| Extension | Відсутні. |
| Post-Condition | Перехід на сторінку інструкцій |

1. Варіант використання UC-03

|  |  |
| --- | --- |
| Use case name | Вхід користувача на сторінку авторів |
| Use case ID | UC-03 |
| Goals | Успішний вхід користувача на сторінку авторів |
| Actors | Користувач |
| Trigger | Користувач бажає увійти на сторінку авторів |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться не на цій сторінці. |
| Flow of Events | Користувач переходить на сторінку авторів. |
| Extension | Відсутні. |
| Post-Condition | Перехід на сторінку авторів |

1. Варіант використання UC-04

|  |  |
| --- | --- |
| Use case name | Вхід користувача на сторінку майбутнього використання |
| Use case ID | UC-04 |
| Goals | Успішний вхід користувача на сторінку майбутнього використання |
| Actors | Користувач |
| Trigger | Користувач бажає увійти на сторінку майбутнього використання |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться не на цій сторінці. |
| Flow of Events | Користувач переходить на сторінку майбутнього використання. |
| Extension | Відсутні. |
| Post-Condition | Перехід на сторінку майбутнього використання |

1. Варіант використання UC-05

|  |  |
| --- | --- |
| Use case name | Вхід користувача на сторінку ліцензій |
| Use case ID | UC-05 |
| Goals | Успішний вхід користувача на сторінку ліцензій |
| Actors | Користувач |
| Trigger | Користувач бажає увійти на сторінку ліцензій |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться не на цій сторінці. |
| Flow of Events | Користувач переходить на сторінку ліцензій. |
| Extension | Відсутні. |
| Post-Condition | Перехід на сторінку ліцензій |

1. Варіант використання UC-06

|  |  |
| --- | --- |
| Use case name | Пошук потрібної інформації ручним пошуком |
| Use case ID | UC-06 |
| Goals | Успішне знаходження потрібної інформації ручним пошуком |
| Actors | Користувач |
| Trigger | Користувач бажає знайти потрібну інформацію ручним пошуком |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться на головній сторінці. |

Продовження Таблиці 1.10.

|  |  |
| --- | --- |
| Flow of Events | Користувач вводить потрібний запит в поле для пошуку. Натискає клавішу <ENTER> або іконку пошуку. Чекає відповіді ГП. |
| Extension | У разі відсутності доступу в інтернет, пошук буде заблоковано та написано відповідне повідомлення про це. У разі натискання клавіші <ENTER> або іконці пошуку, поки ГП озвучує відповідь, пошук буде заблоковано та написано відповідне повідомлення про це. |
| Post-Condition | Створення відповіді на запит користувача. Її озвучення голосом. |

1. Варіант використання UC-07

|  |  |
| --- | --- |
| Use case name | Пошук потрібної інформації голосовим пошуком |
| Use case ID | UC-07 |
| Goals | Успішне знаходження потрібної інформації голосовим пошуком |
| Actors | Користувач |
| Trigger | Користувач бажає знайти потрібну інформацію голосовим пошуком |
| Pre-conditions | Користувач успішно увійшов в додаток та знаходиться на головній сторінці. |
| Flow of Events | Користувач натискає на іконку голосового пошуку. Вводить потрібний запит голосом. Чекає відповіді ГП. |
| Extension | У разі відсутності доступу в інтернет, пошук буде заблоковано та написано відповідне повідомлення про це. У разі натискання іконці голосового пошуку, поки ГП озвучує відповідь, пошук буде заблоковано та написано відповідне повідомлення про це. |
| Post-Condition | Створення відповіді на запит користувача. Її озвучення голосом. |

### Розроблення функціональних вимог

Програмне забезпечення розділене на модулі. Кожен модуль має свій певний набір функцій. На рисунку 1.3 наведено загальну модель вимог, а в таблицях 1.12 – 1.27 наведений опис функціональних вимог до програмного забезпечення. Матрицю трасування вимог можна побачити на рисунку 1.4.



1. Модель вимог у загальному вигляді
2. Функціональна вимога FR-1

|  |  |
| --- | --- |
| Назва | Можливість закриття програми |
| Опис | Система повинна надавати можливість закриття програми. |

1. Функціональна вимога FR-2

|  |  |
| --- | --- |
| Назва | Можливість зміни розміру програми |
| Опис | Система повинна надавати можливість зміни розміру програми. |

1. Функціональна вимога FR-3

|  |  |
| --- | --- |
| Назва | Можливість згортання програми |
| Опис | Система повинна надавати можливість згортання програми. |

1. Функціональна вимога FR-4

|  |  |
| --- | --- |
| Назва | Можливість перетягування вікна програми |
| Опис | Система повинна надавати можливість перетягування вікна програми. |

1. Функціональна вимога FR-5

|  |  |
| --- | --- |
| Назва | Можливість ручного пошуку |
| Опис | Система повинна надавати можливість виконання ручного пошуку. |

1. Функціональна вимога FR-6

|  |  |
| --- | --- |
| Назва | Можливість голосового пошуку |
| Опис | Система повинна надавати можливість голосового пошуку. |

1. Функціональна вимога FR-7

|  |  |
| --- | --- |
| Назва | Відтворення відповідей та інформації |
| Опис | Система повинна надавати можливість відтворення голосом асистента та друкованого тексту у вікні відповіді. |

1. Функціональна вимога FR-8

|  |  |
| --- | --- |
| Назва | Можливість прокручування відповіді ГП |
| Опис | Система повинна надавати можливість прокручування вмісту вікна відповіді ГП. |

1. Функціональна вимога FR-9

|  |  |
| --- | --- |
| Назва | Можливість переходу на основну панель |
| Опис | Система повинна надавати можливість переходу на основну панель. |

1. Функціональна вимога FR-10

|  |  |
| --- | --- |
| Назва | Можливість переходу на панель інструкцій |
| Опис | Система повинна надавати можливість переходу на панель інструкцій до використання. |

1. Функціональна вимога FR-11

|  |  |
| --- | --- |
| Назва | Можливість переходу на панель авторів |
| Опис | Система повинна надавати можливість переходу на панель авторів. |

1. Функціональна вимога FR-12

|  |  |
| --- | --- |
| Назва | Можливість переходу на панель майбутнього використання |
| Опис | Система повинна надавати можливість переходу на панель майбутнього використання. |

1. Функціональна вимога FR-13

|  |  |
| --- | --- |
| Назва | Можливість переходу на панель ліцензій |
| Опис | Система повинна надавати можливість переходу на панель ліцензій. |

1. Функціональна вимога FR-14

|  |  |
| --- | --- |
| Назва | Надання текстового та голосового доступу до сервісів |
| Опис | Система повинна надавати можливість текстового та голосового доступу до різних сервісів, таких як пошук, погода, переклад, музика тощо. |

1. Функціональна вимога FR-15

|  |  |
| --- | --- |
| Назва | Забезпечення функції розпізнавання англійської мови |
| Опис | Система повинна надавати можливість розпізнавання англійської мови для забезпечення введення команд клавіатурою та голосом. |

1. Функціональна вимога FR-16

|  |  |
| --- | --- |
| Назва | Гарантування конфіденційності всіх запитів |
| Опис | Система повинна забезпечувати конфіденційності всіх запитів для захисту приватності користувачів. |



1. Матриця трасування вимог

### Розроблення нефункціональних вимог

В таблицях 1.28 – 1.32 наведений опис нефункціональних вимог до програмного забезпечення.

1. Нефункціональна вимога NFR-1

|  |  |
| --- | --- |
| Назва | Надійність |
| Опис | Зміна розміру програми повинна відбуватися без будь-яких суттєвих затримок або відмов. |

1. Нефункціональна вимога NFR-2

|  |  |
| --- | --- |
| Назва | Видимість |
| Опис | Згортання та розгортання програми повинні відбуватися з максимальною відмітністю для користувача. |

1. Нефункціональна вимога NFR-3

|  |  |
| --- | --- |
| Назва | Продуктивність |
| Опис | Час від видачі команди на пошук до відображення результатів повинен бути менше 5 секунд. |

1. Нефункціональна вимога NFR-4

|  |  |
| --- | --- |
| Назва | Масштабованість |
| Опис | Система має оптимізовано працювати на різних розмірах екрану, забезпечуючи оптимальний користувацький досвід. |

1. Нефункціональна вимога NFR-5

|  |  |
| --- | --- |
| Назва | Ергономіка |
| Опис | Користувацький інтерфейс системи повинен бути інтуїтивно зрозумілим та зручним для використання. Результати пошуку мають бути представлені чітко та лаконічно, а користувач повинен мати можливість взаємодіяти з ними без зайвих ускладнень. |

## Постановка задачі

Створення голосового помічника розкриває великий потенціал в застосуванні, здатний успішно інтегруватися в різноманітні сфери. Основною метою є вдосконалення користувацького досвіду та автоматизація щоденних завдань, пов'язаних з використанням технологічних пристроїв та браузерів.

Голосовий помічник спрямований на широкий спектр користувачів, які бажають полегшити взаємодію з технічними пристроями та оптимізувати свої робочі процеси. Беззаперечно, програмне забезпечення має бути доступним та зручним для кінцевих користувачів, які використовують його для ефективного управління та взаємодії з різними технологічними пристроями.

Не менш важливим є виконання всіх визначених функціональних і нефункціональних вимог програмного забезпечення.

## Висновки до розділу

У розділі визначено встановлення завдання та розглянуто наявні аналоги голосових помічників. Представлені альтернативи оцінені як високоякісні, але мають свої недоліки, такі як використання закритого коду або зберігання даних на серверах.

Крім того, в першому розділі здійснено аналіз вимог, складено таблицю для відповідного сценарію використання та кожної функціональної вимоги. Це надає більш детальне уявлення про вимоги до мови програмування та технологій, які слід використовувати.

Після ретельного аналізу було визначено, що оптимальним вибором для даного проєкту є використання мови програмування Java, інтегрованого середовища розробки IntelliJ IDEA, бібліотеки Sphinx4 для розпізнавання мовлення, бібліотеки Swing для створення графічного інтерфейсу користувача, та бібліотеки FreeTTS для генерації голосу. Цей вибір обґрунтовується їх високою продуктивністю, надійністю та специфічними можливостями, необхідними для успішної реалізації голосового інтерфейсу та ефективної взаємодії з користувачем.

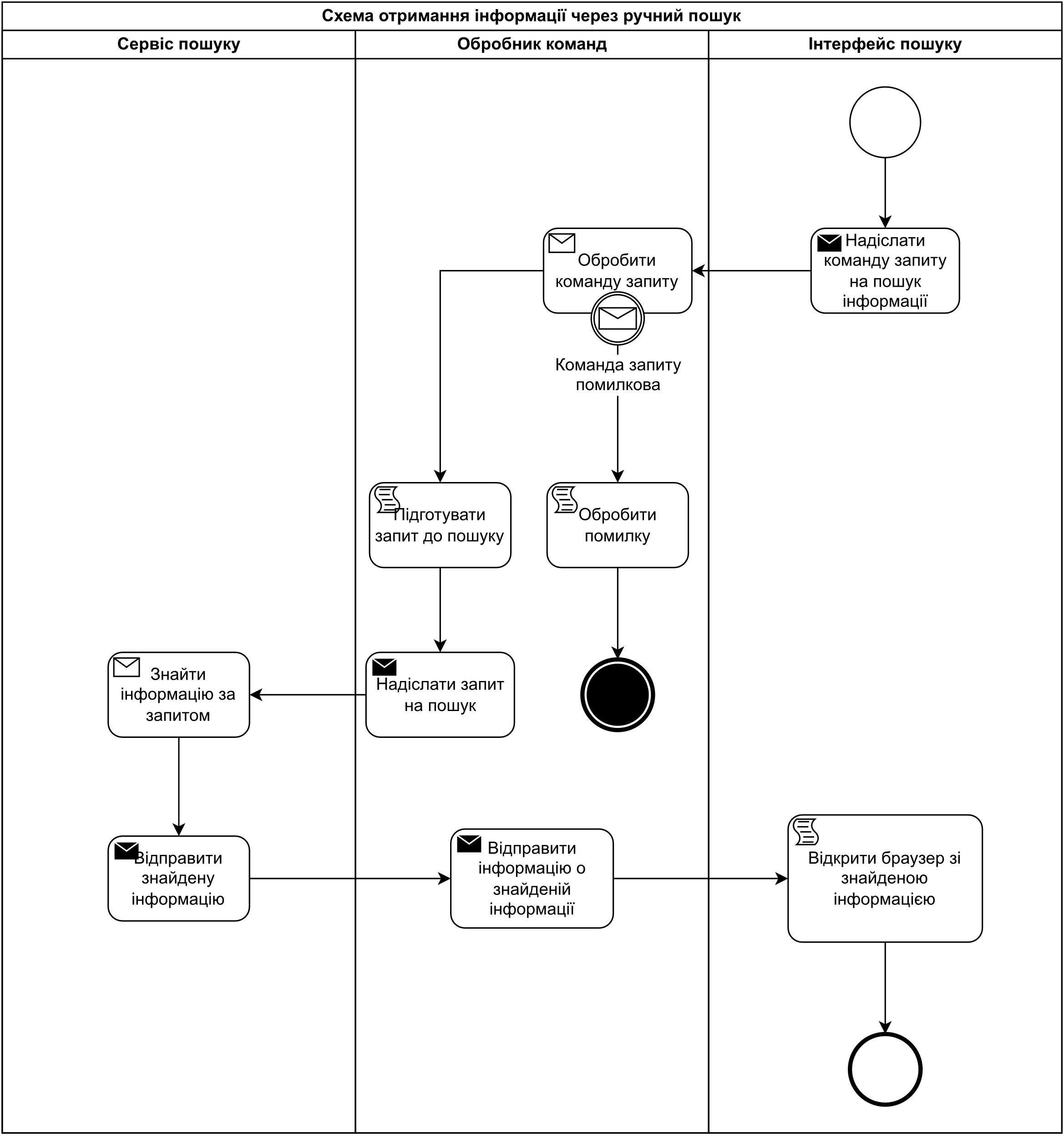
# МОДЕЛЮВАННЯ ТА КОНСТРУЮВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Моделювання та аналіз програмного забезпечення

Для опису бізнес процесу пошуку інформації в ручному режимі користувачем використовується BPMN модель (рисунок 2.1)[26].

Опис послідовності пошуку інформації в ручному режимі користувачем:

* користувач, знаходячись на головній сторінці, вводить команду для пошуку інформації;
* надісланий запит користувача обробляється на стороні обробника команд;
* у випадку наявності помилки в запиті, обробляється помилка, після чого процес зупиняється;
* підготовлений запит для пошуку відправляється до сервісу пошуку з боку обробника команд;
* сервіс пошуку знаходить інформацію за вказаним запитом і відправляє її до обробника команд;
* знайдена інформація передається користувачу через обробник команд та відображається в браузері.

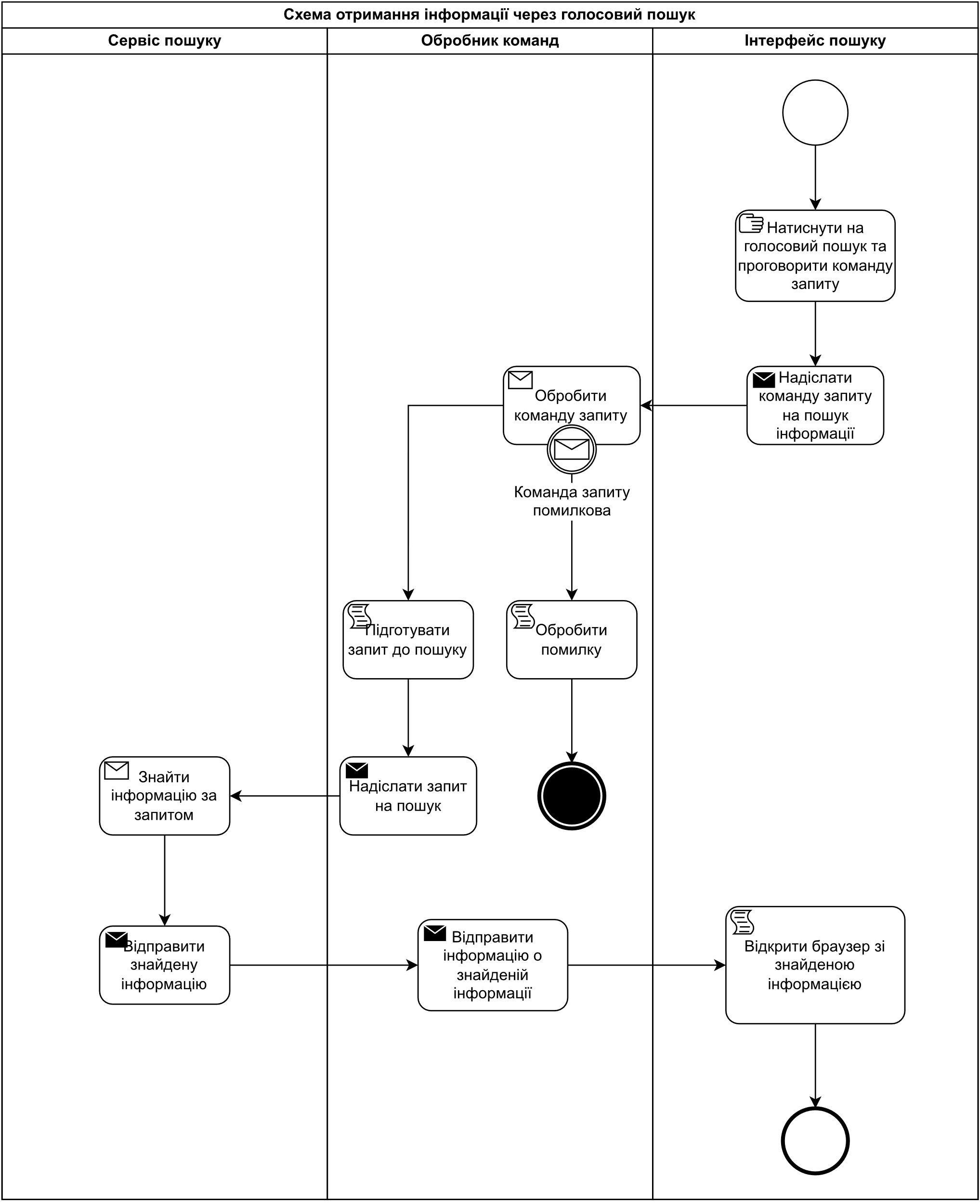


1. Схема бізнес-процесу отримання інформації через ручний пошук

Для опису бізнес процесу пошуку інформації в голосовому режимі користувачем використовується BPMN модель (рисунок 2.2).

Опис послідовності пошуку інформації в голосовому режимі користувачем:

* користувач, знаходячись на головній сторінці, натискає на голосовий пошук та говорить команду для пошуку інформації;
* надісланий запит користувача обробляється на стороні обробника команд;
* у випадку наявності помилки в запиті, обробляється помилка, після чого процес зупиняється;
* підготовлений запит для пошуку відправляється до сервісу пошуку з боку обробника команд;
* сервіс пошуку знаходить інформацію за вказаним запитом і відправляє її до обробника команд;
* знайдена інформація передається користувачу через обробник команд та відображається в браузері.



1. Схема бізнес-процесу отримання інформації через голосовий пошук

## Архітектура програмного забезпечення

При проєктуванні голосового помічника, використання патернів стає ключовим етапом для забезпечення ефективності, читабельності та легкості управління кодом. Одні із таких патернів об'єктноорієнтований підхід та синглтон патерн.

У розробці голосового помічника використовується об'єктноорієнтований підхід до програмування (ООП), що сприяє організації коду у вигляді класів та об'єктів для спрощення розробки та утримання коду. Наприклад, існує клас Synthesizer, який втілює об'єкт синтезатора тексту в мову. Крім того, для забезпечення єдиності екземпляра класу Synthesizer використовується синглтон патерн[27][28].

Структура класу Synthesizer включає приватний конструктор при створенні екземпляра класу, де встановлюється системна властивість та ініціалізується об'єкт голосу. Синглтон реалізовано через статичний вкладений клас SynthesizerHolder, який забезпечує ініціалізацію та утримання єдиного екземпляра синтезатора.

Додатково, програма використовує механізм wait() та notifyAll() для реалізації Observer Pattern, який дозволяє спостерігати за змінами та сповіщати зареєстровані об'єкти про ці зміни. Компонент LockConnection слугує об'єктом спостереження, використовується для синхронізації між потоками та використовує механізм wait() та notifyAll() для ефективного керування виконанням операцій та уникнення зайвого очікування.

Інші частини програми, що взаємодіють із LockConnection, можуть реєструватися як спостерігачі та отримувати сповіщення про зміни стану, що додає до загальної ефективності та гнучкості системи.

Заданий контекст натякає на те, що побудова UML-діаграми може значно полегшити розуміння архітектурних аспектів системи. Це важливий інструмент для візуалізації структури, поведінки та взаємодії класів у системі. Діаграми UML дозволяють команді розробників та іншим учасникам проекту легше сприймати та спілкуватися з архітектурними концепціями[29].



1. UML-діаграма класів

Архітектура, побудована за патерном Singleton, дозволяє зберігати код чистим, оскільки вона гарантує, що у програмі існує лише один екземпляр класу і немає глобальних змінних чи складних систем управління життєвим циклом об'єктів.

Об'єктно-орієнтоване програмування дозволяє створювати код, який організований навколо об'єктів, які представляють конкретні екземпляри або класи об'єктів.

## Конструювання програмного забезпечення

### Оригінальні алгоритми та модифікації

Для досягнення паралельної анімації у Java Swing використовується підхід, де окремий потік відповідає за кожну окрему анімаційну частину. Наприклад, можна виділити окремий потік для обробки кожної кнопки меню на екрані, забезпечуючи гладкість та ефективне використання ресурсів[19].

Також розроблено можливість динамічного виділення потоків програмно, що дозволяє ефективно використовувати ресурси системи, адаптуючи кількість потоків під завдання[17].

У бібліотеці Sphinx4 використовується сінглтон-підхід для кожної ініціалізації об'єкта класу нової лінії мікрофону.

Це забезпечує ефективне управління ресурсами та уникнення зайвого створення екземплярів, зберігаючи стан мікрофону та структуру лише одного екземпляра.

Поміж іншими покращеннями до бібліотеки Sphinx4 введено оптимізації, які дозволяють ефективніше використовувати ресурси мікрофону та оптимізувати шлях обробки звукової інформації.

### Структури даних та програмні структури

* Об'єктно-орієнтований підхід та синглтон

Клас Synthesizer реалізовано за об'єктно-орієнтованим підходом, що дозволяє організувати код у вигляді класів та об'єктів. Застосовано синглтон патерн для забезпечення єдиності екземпляра синтезатора.

* Механізм wait() та notifyAll() для Observer Pattern

Клас LockConnection використовує механізми wait() та notifyAll() для реалізації Observer Pattern, що дозволяє ефективно спостерігати та сповіщати зареєстровані об'єкти про зміни.

* Діаграми UML для візуалізації структури

Застосовано побудову UML-діаграм для візуалізації архітектурних аспектів системи, сприяючи легшому розумінню та спілкуванню з архітектурними концепціями.

### Утиліти, бібліотеки та інше ПЗ

Опис утиліт, бібліотек та іншого стороннього програмного забезпечення, що використовується у розробці наведено в таблиці 2.22.

1. Опис утиліт

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 1 | IntelliJ IDEA | Головне середовище розробки програмного забезпечення курсової роботи. |
| 2 | Apache NetBeans | Програмне забезпечення, необхідне для швидкого та зручного створення інтерфейсної частини програмного забезпечення курсової роботи. |
| 3 | Sphinx4 | Бібліотека для розпізнавання мовлення, розробленою для використання в різних проектах. Вона базується на технологіях розпізнавання мовлення та надає інтерфейс для інтеграції з різними мовленнєвими застосунками. Sphinx4 може бути використаний для розпізнавання голосу в реальному часі, обробки команд голосового управління та інших завдань, пов'язаних із мовленням. |

Продовження Таблиці 2.1.

|  |  |  |
| --- | --- | --- |
| №  п/п | Назва утиліти | Опис застосування |
| 4 | Swing | Бібліотека для розробки графічного інтерфейсу користувача (GUI) в мові програмування Java. Ця бібліотека надає набір компонентів і контейнерів, які дозволяють створювати візуально привабливі та інтерактивні програми. Swing використовується для розробки віконних додатків, де користувач може взаємодіяти з програмою за допомогою миші, клавіатури та інших пристроїв. |
| 5 | FreeTTS | Бібліотека для синтезу мовлення в мові програмування Java. Вона дозволяє розробникам створювати додатки, які можуть генерувати звукові файли або безпосередньо виводити мовлення на аудіо-пристрої. |
| 6 | IPinfo | Бібліотека (та сервіс) для отримання інформації про IP-адреси та геолокацію. Використовуючи цей сервіс у розробці, ви можете отримувати дані про місцезнаходження користувачів, що може бути корисним для точкового аналізу. |

## Аналіз безпеки даних

Ця курсова робота виконує функції голосового помічника та, важливо відзначити, не здійснює збір персональних даних користувача. Таким чином, аспекти безпеки даних не становлять частину цього проекту.

## Висновки до розділу

В даному розділі висвітлено результати аналізу бізнес-процесів з пошуку інформації в ручному та голосовому режимах. Представлено деталі архітектури програмного забезпечення голосового помічника, охоплюючи об'єктно-орієнтований підхід, синглтон патерн, використання механізмів wait() та notifyAll() для Observer Pattern та використання UML-діаграми класів для візуалізації структури.

Розглянуто оригінальні алгоритми та модифікації, спрямовані на паралельну анімацію та ефективне управління ресурсами. Подано опис структур даних та програмних конструкцій, що використовуються у процесі розробки, і визначено їхню раціональність та ефективність.

Висвітлено використання утиліт, бібліотек та іншого зовнішнього програмного забезпечення у розробці. Проведено аналіз безпеки даних, в результаті якого підтверджено, що дана курсова робота не здійснює збір персональних даних користувача.

Зазначено, що об'єктно-орієнтований підхід та синглтон патерн сприяють ефективній організації коду, спрощуючи розробку та утримання кодової бази. Механізм wait() та notifyAll() для Observer Pattern дозволяє ефективно спостерігати за змінами та інформувати зареєстровані об'єкти про ці зміни. Діаграми UML виявляються значущим інструментом для візуалізації структури, поведінки та взаємодії класів у системі. Паралельна анімація та ефективне управління ресурсами сприяють підвищенню продуктивності та ефективності системи.

# АНАЛІЗ ЯКОСТІ ТА ТЕСТУВАННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Аналіз якості ПЗ

Аналіз якості ПЗ було проведено за допомогою двох плагінів: MetricsReloaded та Qodana[30][31].

Основні метрики, отримані за допомогою MetricsReloaded, та їх опис:

* JLOC (Javadoc lines of code) – 205:

Кількість рядків коду, які містять коментарі Javadoc (документаційні коментарі для Java коду). Вказує на розмір документації в коді.

* LOC (Lines of code) – 1854:

Загальна кількість рядків вихідного коду. Ця метрика включає як код, так і коментарі. Вказує на обсяг реального виконуваного коду.

* v(G)avg (Average cyclomatic complexity) – 1.88:

Середнє значення цикломатичної складності коду. Цикломатична складність вимірює кількість ліній виходу (alternative paths) в коді. Низьке значення (менше 5) вказує на меншу складність коду.

Щодо результатів аналізу плагіном Qodana:

A diagram of a problem

Description automatically generated

1. Результат аналізу коду за допомогою плагіну Qodana

Знайдено тільки 4 проблеми, які є залежностями в Maven та ніяк не впливають на даний проект.

За результатами аналізу вказано, що код проекту в цілому має низьку цикломатичну складність, хорошу документацію (включаючи Javadoc), та містить дуже мало проблем, які, крім того, не впливають на функціональність проекту.

## Опис процесів тестування

Було виконане мануальне тестування програмного забезпечення, опис відповідних тестів наведено у таблицях 3.1 – 3.8.

1. Тест VO\_1

|  |  |
| --- | --- |
| Test Case ID | VO\_1 |
| Summary | Підтвердити, що переклад за запитом в додатку "Vocalia" ручним вводом працює без помилок |
| Priority | Високий |
| Precondition | Користувач має комп'ютер з JDK >= 8 із доступом в інтернет та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.1.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку два рази ЛКМ |  | Поле пошуку активувалося для вводу |  |
| 3 | Введіть команду для перекладу тексту та сам текст після неї в поле пошуку клавіатурою | translate Hello I'm a Nickita |  |  |
| 4 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Асистент вивів на екран переклад фрази та програв звук що команда виконана успішно |  |
| 5 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_2

|  |  |
| --- | --- |
| Test Case ID | VO\_2 |
| Summary | Підтвердити, що прогноз погоди за запитом в додатку "Vocalia" голосовим вводом працює без помилок |
| Priority | Високий |
| Precondition | Користувач має комп'ютер з JDK >= 8 із доступом в інтернет та встановлений додаток "Vocalia" |
|  |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на кнопку голосового вводу |  | Іконка голосового вводу смінилася на активну |  |

Продовження Таблиці 3.2.

|  |  |  |  |
| --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |
| 3 | Скажіть команду для прогнозу погоди англійською | weather forecast | Іконка голосового вводу смінилася на пасивну та асистент вивів на екран прогноз погоди для локації зовнішнього IP |  |
| 4 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_3

|  |  |
| --- | --- |
| Test Case ID | VO\_3 |
| Summary | Підтвердити, що програма правильно обрабляє ситуації, коли користувач без доступа в інтернет |
| Priority | Середній |
| Precondition | Користувач має комп'ютер з JDK >= 8 без доступу в інтернет та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку |  | Додаток виведе повідомлення про відсутність інтернет з'єднання та спробує під'єднатиса до інтернету через 1.5 секунди |  |
| 3 | Натисніть на кнопку пошуку |  | Додаток виведе повідомлення про відсутність інтернет з'єднання та спробує під'єднатиса до інтернету через 1.5 секунди |  |

Продовження Таблиці 3.3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | Натисніть на кнопку голосового пошуку |  | Додаток виведе повідомлення про відсутність інтернет з'єднання та спробує під'єднатиса до інтернету через 1.5 секунди |  |
| 5 | Ввімкніть доступ до інтернету |  |  |  |
| 6 | Натисніть на поле пошуку |  | Повідомлення про відсутність інтернет з'єднання пропадає та поле пошуку стає активним для вводу |  |
| 7 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_4

|  |  |
| --- | --- |
| Test Case ID | VO\_4 |
| Summary | Підтвердити, що в додатку "Vocalia" працює захист від спаму мишкою |
| Priority | Середній |
| Precondition | Користувач має комп'ютер з JDK >= 8 із доступом в інтернет та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.4.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на іншу вкладку ЛКМ та швидко натисніть на інші |  | Додаток буде ігнорувати кліки ЛКМ декілька секунд після чого можна буде натискати інші вкладки |  |
| 3 | Натисніть на іншу вкладку один раз |  | Додаток відкриє натиснуту вкладку |  |

Продовження Таблиці 3.4.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | Натисніть на відкриту вкладку |  | Додаток ігнорує натискання на відкриту вкладку |  |
| 5 | Натисніть на іншу вкладку один раз |  | Додаток відкриє натиснуту вкладку |  |
| 6 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_5

|  |  |
| --- | --- |
| Test Case ID | VO\_5 |
| Summary | Підтвердити, що пошук за запитом в додатку "Vocalia" ручним вводом працює без помилок |
| Priority | Високий |
| Precondition | Користувач має комп'ютер з JDK >= 8 з ОС Windows із доступом в інтернет >=7 та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.5.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку два рази ЛКМ |  | Поле пошуку активувалося для вводу |  |
| 3 | Введіть команду для пошуку запиту в браузері та сам запит після неї в поле пошуку клавіатурою | search for youtube |  |  |
| 4 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Додаток відкрив браузер Google Chrome (якщо є, інакше Microsoft Edge) із відповідним запитом в ньому |  |
| 5 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_6

|  |  |
| --- | --- |
| Test Case ID | VO\_6 |
| Summary | Підтвердити, що генератор жартів в додатку "Vocalia" ручним вводом працює без помилок |
| Priority | Низький |
| Precondition | Користувач має комп'ютер з JDK >= 8 із доступом в інтернет та встановлений додаток "Vocalia" |
|  |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку два рази ЛКМ |  | Поле пошуку активувалося для вводу |  |
| 3 | Введіть команду для генерації жарту в поле пошуку клавіатурою | tell me a joke |  |  |

Продовження Таблиці 3.6.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Асистент вивів на екран сгенерований жарт |  |
| 5 | Введіть команду для генерації жарту в поле пошуку клавіатурою знову | tell me a joke |  |  |
| 6 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Асистент вивів на екран новий сгенерований жарт |  |
| 7 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_7

|  |  |
| --- | --- |
| Test Case ID | VO\_7 |
| Summary | Підтвердити, що сервіс музики в додатку "Vocalia" ручним вводом працює без помилок |
| Priority | Середній |
| Precondition | Користувач має комп'ютер з JDK >= 8 з ОС Windows із доступом в інтернет >=7 та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.7.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку два рази ЛКМ |  | Поле пошуку активувалося для вводу |  |
| 3 | Введіть команду для музики в поле пошуку клавіатурою | play music |  |  |
| 4 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Асистент відкрив YouTube Music випадковий плейліст у браузері Google Chrome (якщо є, інакше Microsoft Edge) та ввімкнув його на прослуховування |  |

Продовження Таблиці 3.7.

|  |  |  |  |
| --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |
| 5 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

1. Тест VO\_8

|  |  |
| --- | --- |
| Test Case ID | VO\_8 |
| Summary | Підтвердити, що, поки асистент говорить, користувач не може користуватися пошуком |
| Priority | Високий |
| Precondition | Користувач має комп'ютер з JDK >= 8 з ОС Windows із доступом в інтернет >=7 та встановлений додаток "Vocalia" |
|  |

Продовження Таблиці 3.8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Steps | Test Step | Test Data | Expected result |  |
| 1 | Відкрийте додаток "Vocalia" двічі клікнувши на нього ЛКМ |  | Додаток відкрився і ніяких помилок не сталося, при відкритті програми програвся звук про те, що додаток готовий для роботи |  |
| 2 | Натисніть на поле пошуку два рази ЛКМ |  | Поле пошуку активувалося для вводу |  |
| 3 | Введіть будь яку команду в поле пошуку клавіатурою | weather forecast |  |  |
| 4 | Натисніть на клавіатурі <ENTER> або на кнопку пошуку |  | Асистент виконав відповідний запит |  |

Продовження Таблиці 3.8.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5 | Поки асистент каже відповідь, натисніть на поле пошуку або на кнопки пошуку |  | Додаток повідомить про те, що асистент ще говорить відповідь на команду та не дасть користуватися пошуком |  |
| 6 | Закрийте додаток "Vocalia" |  | Додаток закрився без помилок |  |

## Висновки до розділу

В цьому розділі проведено аналіз якості та тестування програмного забезпечення додатку "Vocalia". Для оцінки якості ПЗ використано два плагіни: MetricsReloaded і Qodana. Результати показали, що код додатку має низьку цикломатичну складність, добру документацію (зокрема, Javadoc) і містить мало проблем, які не впливають на функціональність проекту.

Для тестування ПЗ виконано 8 мануальних тестів, що охоплюють основні функціональні можливості додатку. Усі тести пройдені успішно. Основний висновок аналізу полягає в тому, що код "Vocalia" характеризується низькою цикломатичною складністю, якісною документацією та мінімальною кількістю проблем, які не впливають на функціональність.

У результаті тестування виявлено, що додаток має високу якість коду, стабільність функціоналу та коректну роботу основних функцій. Знайдені проблеми із залежностями в Maven не мають впливу на роботу проекту, а усі мануальні тести пройшли успішно, що підтверджує надійність додатку.

# ВПРОВАДЖЕННЯ ТА СУПРОВІД ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

## Розгортання програмного забезпечення

Для розгортання голосового додатку потрібно виконати наступні кроки:

* Встановити JDK 21 або більше з офіційного сайту:

<https://www.oracle.com/cis/java/technologies/downloads/>.

* Створити новий проект в Intellij IDEA, чи в іншому зручному середовищі, для Java та склонувати в нього репозиторій Vocalia:

git clone https://github.com/TheMegistone4Ever/Vocalia.git

* Збудувати проєкт за допомогою Maven, для цього запустіть в локальному терміналі:

mvn install

* Створити JAR-артефакт:

Натиснути Cntrl+Alt+Shift+S (або пройти за шляхом: File -> Project Structure).

Зайти на вкладку артефактів.

Натиснути “+”, далі обрати JAR та From modules with dependencies.

A screenshot of a computer

Description automatically generated

1. Створення JAR-артефакту

Обрати модуль Vocalia та основний клас com.nickmegistone.vocaliamaven.Vocalia.

Натиснути “ОК”.

A screenshot of a computer

Description automatically generated

1. Налаштування створеного JAR-артефакту

Натиснути “Apply”.

* Побудувати проект: пройти за шляхом: Build -> Build Artifacts -> Vocalia:jar -> Build)
* Після побудови скопіювати повний шлях JAR-файлу в out теки.

A screenshot of a computer

Description automatically generated

1. Копіювання повного шляху створеного JAR-файлу

* Скомпілювати JAR-файл з кореневої папки проекту:

“C:\Program Files\Java\jdk-<version>\bin\java.exe” -jar “ABSOLUTE PATH”

* Можна користуватися голосовим помічником.

## Підтримка програмного забезпечення

Для ефективного управління ПЗ, проект буде зберігатися на GitHub, і кожен коміт буде представляти декілька пушів. Кожен коміт має тег у форматі "v\*..", де "\*" вказує на конкретну версію. Користувачі можуть завантажити архів і встановити додаток, дотримуючись інструкцій у файлі README.md. Цей підхід дозволяє легко відслідковувати версії та забезпечує зручний спосіб для користувачів встановлювати та оновлювати ПЗ.

Використання GitHub для зберігання ПЗ допомагає в створенні ефективного та спільного середовища розробки. Платформа надає можливість простоти відслідковування змін, зручного спілкування між розробниками та документування проекту. Система комітів і тегів "v\*.." дозволяє легко розрізняти різні версії ПЗ, а також фіксує важливі зміни та оновлення.

Крім того, GitHub забезпечує можливість роботи з гілками, що дозволяє працювати паралельно над різними функціональностями чи виправленнями. Вбудовані інструменти для ведення багтрекінгу, такі як Issues, дозволяють ефективно вирішувати проблеми та взаємодіяти з користувачами.

## Висновки до розділу

У даному розділі було детально висвітлено процес впровадження та супроводу програмного забезпечення голосового додатку. Починаючи з встановлення JDK версії 21 або вище та створення проекту в Intellij IDEA, розгортання програми передбачало кілька кроків, таких як збірка проекту за допомогою Maven і створення JAR-артефакту.

Процедура створення JAR-артефакту включала в себе важливі кроки, такі як вибір основного класу та модуля, налаштування артефакту та його побудова. Результатом був готовий JAR-файл, який можна було використовувати для запуску голосового додатку.

Використання системи контролю версій GitHub для зберігання програмного забезпечення дозволило забезпечити прозорість та легкість завантаження ПЗ. Вона створює зручне середовище для відслідковування змін, вирішення проблем, спілкування та документування проекту. Використання гілок також сприяло паралельній роботі над різними аспектами програми. Успішне впровадження та супровід програмного забезпечення включає етапи встановлення середовища розробки, розгортання програми та ефективне управління версіями через систему контролю версій, зокрема GitHub.

# ВИСНОВКИ

У ході виконання курсової роботи було розроблено голосовий помічник "Vocalia" як Desktop додаток. Додаток дозволяє користувачам здійснювати голосовий та ручний пошук інформації в Інтернеті, а також виконувати інші завдання, такі як відтворення музики, генерація жартів, прогноз погоди тощо.

У першому розділі курсової роботи було проведено аналіз вимог до додатка, визначено його функціональні та нефункціональні вимоги. На основі цього аналізу було обрано оптимальні мову програмування Java, середовище розробки Intellij IDEA та інші технології для реалізації додатка.

У другому розділі було розроблено архітектуру додатка, яка включає об'єктноорієнтований підхід, синглтон патерн, механізми wait() та notifyAll() для Observer Pattern, UML-діаграми класів, паралельну анімацію та ефективне управління ресурсами. Оптимізовано бібліотеку Sphinx-4 за допомогою багатопоточності та виправлено деякі проблеми з цією бібліотекою, наприклад, неефективне використання мікрофона пристрою.

У третьому розділі було проведено аналіз якості та тестування додатка. Для оцінки якості ПЗ були використані два плагіни: MetricsReloaded і Qodana. Результати показали, що код додатка має низьку цикломатичну складність, добру документацію та містить мало проблем, які не впливають на функціональність проєкту. У результаті тестування виявлено, що додаток має високу якість коду, стабільність функціонала та коректну роботу основних функцій.

У четвертому розділі було детально висвітлено процес впровадження та супроводу програмного забезпечення голосового додатка. Розгортання програми передбачало кілька кроків, таких як встановлення JDK версії 21 або вище та створення проєкту в Intellij IDEA, збірка проєкту за допомогою Maven і створення JAR-артефакту. Використання системи контролю версій GitHub для зберігання програмного забезпечення дозволило забезпечити прозорість та легкість завантаження ПЗ.

За результатами виконання курсової роботи можна зробити наступні висновки:

* Додаток "Vocalia" відповідає усім поставленим вимогам і може бути використаний для практичних цілей.
* Додаток має високу якість коду, стабільність функціонала та коректну роботу основних функцій.
* Розгортання та супровід додатка є ефективним та прозорим.

Для подальшого розвитку додатка можна розглянути такі напрямки:

* Розширення функціональності додатка, наприклад, додаванням підтримки інших мов, додаткових функцій пошуку, а також можливістю взаємодії з іншими пристроями.
* Покращення продуктивності та ефективності додатка, наприклад, шляхом використання нових алгоритмів та структур даних.
* Підвищення безпеки додатка, наприклад, шляхом використання сучасних методів шифрування та захисту від вразливостей.

Розроблений голосовий помічник має широкий спектр потенційних застосувань. Його можна використовувати для покращення доступності інформації та послуг для людей з обмеженими можливостями, а також для підвищення продуктивності праці та навчання.

# СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Everything you need to know about voice assistants [Електронний ресурс] // santander.com. - Режим доступу до ресурсу: https://www.santander.com/en/stories/everything-you-need-to-know-about-voice-assistants
2. Voice assistants: What they are and what they mean for marketing and commerce [Електронний ресурс] // insiderintelligence.com. - Режим доступу до ресурсу: https://www.insiderintelligence.com/insights/voice-assistants
3. What is artificial intelligence (AI)? [Електронний ресурс] // ibm.com. - Режим доступу до ресурсу: https://www.ibm.com/topics/artificial-intelligence
4. How do voice assistants work? [Електронний ресурс] // thisisdmg.com. - Режим доступу до ресурсу: https://www.thisisdmg.com/en/how-do-voice-assistants-work
5. GOOGLE ASSISTANT, SIRI, BIXBY, ALEXA, ЯКИЙ НАЙКРАЩИЙ? [Електронний ресурс] // gizchina.com.ua. - Режим доступу до ресурсу: https://gizchina.com.ua/uk/2022/12/26/google-assistant-siri-bixby-alexa-yakyy-naykraschyy
6. The Paradox of Intelligent Assistants: Poor Usability, High Adoption [Електронний ресурс] // nngroup.com. - Режим доступу до ресурсу: https://www.nngroup.com/articles/intelligent-assistants-poor-usability-high-adoption
7. Google's Privacy Policy [Електронний ресурс] // policies.google.com. - Режим доступу до ресурсу: https://policies.google.com/privacy?hl=en-US
8. Google's Speech-to-Text documentation [Електронний ресурс] // cloud.google.com. - Режим доступу до ресурсу: https://cloud.google.com/speech-to-text/docs
9. Google's Speech-to-Text [Електронний ресурс] // cloud.google.com. - Режим доступу до ресурсу: https://cloud.google.com/speech-to-text?hl=en
10. CMUSphinx Documentation [Електронний ресурс] // cmusphinx.github.io. - Режим доступу до ресурсу: https://cmusphinx.github.io/wiki
11. Welcome to DeepSpeech’s documentation! [Електронний ресурс] // deepspeech.readthedocs.io. - Режим доступу до ресурсу: https://deepspeech.readthedocs.io/en/r0.9
12. wit.ai - Build Natural Language Experiences [Електронний ресурс] // wit.ai. - Режим доступу до ресурсу: https://wit.ai
13. Java Documentation [Електронний ресурс] // docs.oracle.com. - Режим доступу до ресурсу: https://docs.oracle.com/en/java
14. Python 3.12.1 documentation [Електронний ресурс] // docs.python.org. - Режим доступу до ресурсу: https://docs.python.org/3
15. IntelliJ IDEA – the Leading Java and Kotlin IDE [Електронний ресурс] // jetbrains.com. - Режим доступу до ресурсу: https://www.jetbrains.com/idea
16. ECLIPSE IDE - The Leading Open Platform for Professional Developers [Електронний ресурс] // eclipseide.org. - Режим доступу до ресурсу: https://eclipseide.org
17. Sphinx-4 - Pure Java speech recognition library [Електронний ресурс] // github.com. - Режим доступу до ресурсу: https://github.com/cmusphinx/sphinx4
18. Transcribe speech to text by using client libraries [Електронний ресурс] // cloud.google.com. - Режим доступу до ресурсу: https://cloud.google.com/speech-to-text/docs/transcribe-client-libraries#client-libraries-install-java
19. Package javax.swing [Електронний ресурс] // docs.oracle.com. - Режим доступу до ресурсу: https://docs.oracle.com/javase/7/docs/api/javax/swing/package-summary.html
20. JavaFX [Електронний ресурс] // openjfx.io. - Режим доступу до ресурсу: https://openjfx.io
21. FreeTTS 1.2.3 - A speech synthesizer written entirely in the JavaTM programming language [Електронний ресурс] // freetts.sourceforge.io. - Режим доступу до ресурсу: https://freetts.sourceforge.io
22. Amazon Alexa [Електронний ресурс] // alexa.com. - Режим доступу до ресурсу: https://www.alexa.com
23. Apple Siri [Електронний ресурс] // support.apple.com. - Режим доступу до ресурсу: https://support.apple.com/uk-ua/guide/iphone/iph83aad8922/ios
24. Google Assistant [Електронний ресурс] // assistant.google.com. - Режим доступу до ресурсу: https://assistant.google.com/intl/en
25. Samsung Bixby [Електронний ресурс] // samsung.com. - Режим доступу до ресурсу: https://www.samsung.com/ca/apps/bixby
26. BPMN [Електронний ресурс] // uk.wikipedia.org. - Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/BPMN
27. WHAT IS OBJECT-ORIENTED PROGRAMMING (OOP)? EXPLAINING FOUR MAJOR PRINCIPLES [Електронний ресурс] // softserveinc.com. - Режим доступу до ресурсу: https://career.softserveinc.com/en-us/stories/what-is-object-oriented-programming-oop-explaining-four-major-principles
28. Java Singleton Design Pattern Best Practices with Examples [Електронний ресурс] // digitalocean.com. - Режим доступу до ресурсу: https://www.digitalocean.com/community/tutorials/java-singleton-design-pattern-best-practices-examples
29. What is Unified Modeling Language (UML)? [Електронний ресурс] // visual-paradigm.com. - Режим доступу до ресурсу: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/
30. MetricsReloaded - IntelliJ IDEs Plugin | Marketplace [Електронний ресурс] // plugins.jetbrains.com. - Режим доступу до ресурсу: https://plugins.jetbrains.com/plugin/93-metricsreloaded
31. Qodana Documentation [Електронний ресурс] // jetbrains.com. - Режим доступу до ресурсу: https://www.jetbrains.com/help/qodana/qodana-ide-plugin.html#UI+overview