Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

БАКАЛАВРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Спеціалізований комп'ютерний комплекс для полегшення роботи людям з вадами зору»

08-54.БКР.021.00.000 ПЗ

Виконав: студент 4 курсу, групи 2КІ-21б
спеціальності
123 Комп'ютерна інженерія,
(шифр і назва напряму підготовки, спеціальності)
освітня програма
«Комп'ютерна інженерія»
Антонов Е.П.
(прізвище та ініціали)

Керівник: к.т.н., доцент каф. ОТ
Томчук М.А.
(прізвище та ініціали)
Рецензент: к.т.н., доцент каф. ПЗ
Черноволик Г.О.
(прізвище та ініціали)

ДОПУЩЕНО
Завідувач кафедри ОТ
д.т.н., проф. Азаров О. Д

"18" 06 2025 р.

Вінниця ВНТУ – 2025 рік

Вінницький національний технічний університет

Факультет інформаційних технологій та комп'ютерної інженерії

Кафедра обчислювальної техніки

Освітньо-кваліфікаційний рівень бакалавр

Галузь знань – 12 «Інформаційні технології»

Спеціальність – 123 «Комп'ютерна інженерія»

Освітньо-професійна програма «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ

Завідувач кафедри ОТ

д.т.н., проф.

_____Азаров О. Д.

«21» березня 2025 року

З А В Д А Н Н Я

НА БАКАЛАВРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Антонову Едуарду Пантелейовичу

1 Тема роботи: «Спеціалізований комп'ютерний комплекс для полегшення роботи особам з вадами зору», керівник роботи Томчук М.А. к.т.н., доц., доцент кафедри ОТ затверджені наказом вищого навчального закладу від «25» березня 2025 року № 97

2 Термін подання студентом роботи _21.05.2025р.

3 Вихідні дані до роботи; технічна документація на модулі розпізнавання мови Vosk та синтезатора мовлення System.Speech, базові схеми підключення програмного забезпечення до голосових команд, середовище розробки –_JetBrains Rider.

4 Зміст розрахунково-пояснювальної записки: вступ, аналіз технологій і вибір засобів розробки, актуальність програми та її соціальне значення, розробка спеціалізованого комп'ютерного комплексу для полегшення роботи людям з

вадами зору, розробка графічного інтерфейсу та тестування, інструкція користувача, висновки.

5 Перелік ілюстративного матеріалу: стурктурна схема взаємодії компонентів голосового помічника, блок-схема обробки голосових команд.

6 Консультанти розділів роботи приведені в таблиці 1.

Таблиця 1.1 — Консультанти роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1-3	к.т.н., доц. каф. ОТ Томчук М.А.	21.03.25	13.05.25
Нормоконтроль	ас. каф. ОТ Швець С. І.	14.05.25	30.05.25

7 Дата видачі завдання 21 березня 2025 р.

8 Календарний план БКР приведений в таблиці 2

Таблиця 2 — Календарний план

№ з/п	Назва етапів бакалаврської кваліфікаційної роботи	Строк виконання	Підпис
1	Постановка задачі роботи	21.03.25	Викон.
2	Пошук матеріалів та інформаційних джерел	21.03.25— 30.03.25	Викон.
3	Аналіз існуючих прототипів та визначення вимог до створюваної програми.	21.03.25— 30.03.25	Викон.
4	Обґрунтування структури та принципів роботи системи	31.03.25— 13.04.25	Викон.
5	Розробка й тестування програмної системи	14.04.25— 27.04.25	Викон.
7	Оформлення пояснювальної записки та ілюстративного матеріалу	28.04.25— 11.05.25	Викон.
8	Перевірка якості виконання бакалаврської кваліфікаційної роботи та усунення недоліків	13.05.25	Викон.

Студент Антонов Е.П.

Керівник роботи к.т.н., доц. каф. ОТ Томчук М.А.

АНОТАЦІЯ

УДК 004.42

Антонов Е.П. Спеціалізований комп’ютерний комплекс для полегшення роботи людям з вадами зору. Бакалаврська кваліфікаційна робота зі спеціальності 123 «Комп’ютерна інженерія», освітня програма «Комп’ютерна інженерія».

Вінниця: ВНТУ, 2025. 79 с.

На укр. мові. Бібліогр.: 21 назв; рис.: 13.

У даній роботі проведено всебічний аналіз сучасних технологій голосового розпізнавання та синтезу мови для забезпечення доступності програмного забезпечення. Дослідження охоплює порівняльний огляд алгоритмів розпізнавання голосу, зокрема Vosk та NAudio, які дозволяють інтегрувати ефективне голосове керування в Windows Forms. Особливу увагу приділено аналізу синтезу мовлення через System.Speech, що дозволяє генерувати голосові відповіді у відповідь на команди користувача.

Проведено аналіз механізмів текстового вводу та управління програмними інтерфейсами через голос. Описано можливості впровадження клавіатурних скорочень та адаптивного управління користувацькими елементами через Windows Forms, що забезпечує високу інтерактивність та інтуїтивне керування. Особливу увагу приділено питанню доступності для людей із вадами зору, зокрема оптимізації взаємодії з програмним забезпеченням через голосові команди.

На основі проведеного аналізу розроблено функціональну голосову систему керування Windows Forms-програмою, яка включає налаштування алгоритмів розпізнавання та синтезу мови, розробку доступного графічного інтерфейсу та впровадження методів адаптивного вводу тексту. Запропоноване рішення має перспективи застосування у сферах розробки програмного забезпечення для людей із особливими потребами, автоматизації рутинних завдань та підвищення рівня доступності ІТ-продуктів.

Ключові слова: Голосове розпізнавання, Vosk, NAudio, System.Speech, Windows Forms, клавіатурні скорочення, доступність, адаптивне управління.

ABSTRACT

Antonov E.P. Specialized Computer System for Facilitating Work for Visually Impaired Individuals. Bachelor's Qualification Thesis in Specialty 123 "Computer Engineering", Educational Program "Computer Engineering".

Vinnytsia: VNTU, 2025. 79 pages.

In Ukrainian. Bibliography: 21 sources; illustrations: 13.

This paper provides a comprehensive analysis of modern speech recognition and synthesis technologies to improve software accessibility. The research includes a comparative overview of voice recognition algorithms, specifically Vosk and NAudio, which enable efficient voice control integration into Windows Forms applications. Special attention is given to speech synthesis analysis through System.Speech, which allows generating voice responses to user commands.

The study also examines mechanisms for text input and interface management via voice control. It describes the implementation of keyboard shortcuts and adaptive user interface control via Windows Forms, ensuring high interactivity and intuitive operation. Particular emphasis is placed on accessibility for visually impaired users, focusing on optimizing interaction with software through voice commands.

Based on the conducted analysis, a functional voice-controlled system for Windows Forms applications has been developed, incorporating speech recognition and synthesis algorithms, an accessible graphical interface, and adaptive text input methods. The proposed solution has prospects for application in software development for people with special needs, automation of routine tasks, and enhancing IT product accessibility.

Keywords: Speech recognition, Vosk, NAudio, System.Speech, Windows Forms, keyboard shortcuts, accessibility, adaptive control.

ЗМІСТ

ВСТУП

У сучасному світі, де технології доступності відіграють дедалі важливішу роль, зростає потреба у розробці ефективних рішень для покращення комфорту та автономності людей із вадами зору. Голосові асистенти дозволяють користувачам взаємодіяти з комп'ютером без необхідності візуального сприйняття, забезпечуючи зручність управління програмами та доступ до інформації. Завдяки розпізнаванню мовлення та синтезу голосу такі системи стають невід'ємною частиною повсякденного життя, сприяючи більшій інклюзивності у сфері технологій.

Актуальність теми бакалаврської кваліфікаційної роботи зумовлена необхідністю створення інтуїтивного та функціонального голосового помічника, здатного ефективно обробляти голосові команди та виконувати низку дій, включаючи запуск програм, управління файлами та забезпечення аудіовізуального зворотного зв'язку. Розвиток технологій розпізнавання мовлення, таких як Vosk, та синтезу голосу через System.Speech відкривають нові можливості для створення доступних інтерфейсів. Застосування Windows Forms дозволяє інтегрувати зручні засоби керування, а використання додаткових програмних компонентів, таких як NAudio, сприяє поліпшенню взаємодії з користувачем.

Об'єктом дослідження є технології голосового управління у програмному забезпеченні для людей із вадами зору.

Предметом дослідження є технічні та програмні рішення для реалізації голосового помічника, здатного ефективно обробляти мовні команди та забезпечувати інтерактивну взаємодію з користувачем.

Метою є створення голосового помічника для користувачів із вадами зору, який забезпечує зручну взаємодію з комп'ютером шляхом голосових команд та аудіовідповідей. Система повинна підтримувати розпізнавання мовлення для визначення команд, синтез мовлення для надання зворотного зв'язку та інтеграцію з Windows Forms для візуального доповнення взаємодії.

Для досягнення цієї мети в рамках роботи розв'язано такі **завдання:**

— аналіз сучасних технологій голосового управління, включаючи

розпізнавання мовлення (Vosk) та синтез голосу (System.Speech);

— вибір і обґрунтування архітектури голосового помічника, що включає модулі розпізнавання, синтезу мовлення та обробки команд;

— реалізація програмної частини помічника у середовищі JetBrains Rider на основі C#, забезпечення коректного розпізнавання команд та їх виконання;

— розробка графічного інтерфейсу у Windows Forms, що доповнює систему аудіовізуальним зворотним зв’язком;

— впровадження можливості запуску програм та виконання операцій на комп’ютері за допомогою голосових команд;

— проведення тестування системи, оцінка точності розпізнавання мовлення, швидкості реакції та зручності використання.

1 АКТУАЛЬНІСТЬ ПРОГРАМИ ТА ЇЇ СОЦІАЛЬНЕ ЗНАЧЕННЯ

1.1 Актуальність програми

Актуальність створення голосового помічника, що працює офлайн і підтримує українську мову, зростає з кожним роком. У світі стрімко розвивається тенденція до впровадження голосових інтерфейсів — цифрові асистенти, такі як Siri, Google Assistant, Amazon Alexa, стали звичними елементами повсякденного життя. Вони допомагають користувачам керувати пристроями, шукати інформацію, планувати події, або навіть керувати “розумним домом”. Проте майже всі вони мають спільну рису — повну залежність від підключення до інтернету та обмежену або відсутню підтримку української мови. Це ставить під сумнів як конфіденційність даних, так і зручність використання для україномовних користувачів, особливо у регіонах із нестабільним доступом до мережі.

Особливу цінність голосовий інтерфейс має для людей з порушеннями зору. Для них традиційні графічні інтерфейси часто є недоступними або складними в навігації. У таких випадках голосове керування не просто додає зручність — воно стає ключовим інструментом для забезпечення цифрової доступності. Програма, яка розпізнає голосові команди, озвучує відповіді і дозволяє запускати потрібні застосунки, може суттєво підвищити якість життя таких користувачів, дозволяючи їм повноцінно працювати з комп’ютером без сторонньої допомоги.

Важливим аспектом є конфіденційність. У більшості онлайн-сервісів голос передається на сервери великих компаній, де обробляється й аналізується. Це створює ризик витоку особистих даних, історії запитів, або навіть зберігання голосових записів без згоди користувача. На відміну від цього, офлайн-асистенти, збудовані на бібліотеках як Vosk, обробляють мовлення локально на пристрої. Усе — від захоплення аудіо до розпізнавання й виконання команд — відбувається без передачі інформації через інтернет. Такий підхід гарантує максимальну приватність і незалежність від зовнішніх серверів, що особливо важливо в умовах кібербезпеки та довіри користувачів.

Сьогодні голосові інтерфейси поступово витісняють традиційні методи взаємодії з комп’ютерами й мобільними пристроями. Вони забезпечують

швидший, інтуїтивніший та зручніший спосіб взаємодії, особливо в тих випадках, коли користування клавіатурою чи мишкою є ускладненим або взагалі неможливим. Світові технологічні гіганти інвестують мільярди в розвиток голосових помічників — Siri, Google Assistant, Alexa стали символами нової епохи цифрової комунікації. Проте навіть попри масштабність і технологічну досконаліть цих рішень, у них є кілька серйозних обмежень, які роблять їх недостатньо ефективними або навіть недоступними для значної частини українських користувачів.

Перше з таких обмежень — це мова. Більшість відомих голосових асистентів не мають повноцінної підтримки української. Наприклад, навіть якщо інтерфейс доступний українською, голосове розпізнавання або синтез мовлення працює лише англійською чи іншими більш комерційно поширеними мовами. Така ситуація створює цифрову нерівність, адже україномовні користувачі змушені або переходити на іншу мову, або взагалі відмовлятися від голосових сервісів. З огляду на сучасний контекст розвитку української мови в цифровому просторі, створення повноцінного україномовного голосового помічника — це не лише технологічне, а й культурне завдання, що підтримує мовну ідентичність.

Друге обмеження — це залежність від інтернету. Більшість комерційних голосових помічників працюють за принципом клієнт-сервер: аудіо передається на віддалені сервери, де обробляється штучним інтелектом, а потім відповідь повертається користувачеві. Така модель має переваги у великих обчислювальних потужностях, однак вона потребує стабільного з’єднання з мережею. У багатьох користувачів — особливо в сільських районах, у зоні бойових дій або при використанні автономних систем — інтернет може бути недоступним або ненадійним. Це робить онлайн-асистентів практично непридатними в критичних ситуаціях. Офлайн-голосовий асистент, натомість, може працювати незалежно від зовнішнього середовища: розпізнавання, синтез і виконання команд відбувається локально, прямо на пристрої користувача. Це забезпечує стабільність і автономність — важливі характеристики для систем, які повинні працювати завжди.

Третій важливий аспект — конфіденційність. Коли аудіо користувача передається на сервери третіх сторін, виникає ризик витоку або несанкціонованого використання персональних даних. Навіть великі компанії, які декларують захист конфіденційності, не завжди можуть гарантувати безпеку інформації — траплялися ситуації, коли записи розмов потрапляли до рук сторонніх осіб. Офлайн-асистент не потребує передавати дані назовні, що забезпечує повний контроль над особистою інформацією. У критичних сферах — як-от медицина, оборона, приватні організації — це має вирішальне значення.

Особливу роль така програма може відігравати у сфері інклюзії. Люди з порушеннями зору чи руху стикаються з серйозними труднощами при роботі зі стандартними комп'ютерними інтерфейсами. Для них голосове керування стає не просто альтернативою, а основним або навіть єдиним засобом доступу до інформації. Україномовний асистент, що не потребує інтернету й працює з мінімальними ресурсами, може відкрити нові можливості для тисяч користувачів — навчання, робота, спілкування, дозвілля стають доступними без бар'єрів.

Розробка голосового помічника з підтримкою української мови, що працює офлайн, є вкрай актуальною як з технологічної, так і з соціальної, етичної та культурної точок зору. Це відповідь на сучасні виклики, що поєднує незалежність, безпеку, інклюзивність і підтримку національної мови у цифровому просторі.

1.2 Проблеми людей з вадами зору

Люди з порушеннями зору щодня стикаються з численними бар'єрами при роботі з комп'ютерними системами. Найголовнішою проблемою для них є обмежений або повністю відсутній доступ до візуальної інформації, на якій базується абсолютна більшість графічних інтерфейсів. Елементи управління — кнопки, меню, іконки, повідомлення — стають недоступними без використання спеціальних програм або допоміжних технологій. Це ускладнює як повсякденну роботу з комп'ютером, так і отримання нових знань, перегляд новин, користування сервісами, спілкування чи навіть банальне відкриття потрібної програми.

Навігація по операційній системі, яка для звичайного користувача здається

простою і звичною, для людини з вадами зору перетворюється на складний процес, що потребує запам’ятовування комбінацій клавіш, структури меню та точної взаємодії з читачами з екрану. У багатьох випадках навіть такі допоміжні засоби, як NVDA або JAWS, не забезпечують повноцінної взаємодії, особливо в нестандартних або нестабільних інтерфейсах. А найменша зміна дизайну або оновлення системи може спричинити дезорієнтацію користувача, зламати звичні маршрути взаємодії.

Суттєвою проблемою є відсутність універсальних, інтуїтивно зрозумілих інструментів для голосового керування, які були б легкодоступними, безкоштовними та адаптованими до української мови. Більшість комерційних рішень або недоступні безкоштовно, або потребують глибоких технічних знань для встановлення та налаштування. Крім того, вони рідко враховують мовні потреби користувачів поза англомовним світом. Для людей, які не володіють іноземними мовами, це створює ще один рівень цифрової дискримінації — ніби технології існують, але не для них.

Далеко не всі користувачі мають постійний доступ до інтернету або можуть собі дозволити дорогі онлайн-сервіси. Для людей у сільській місцевості, у зоні бойових дій або просто в умовах обмежених ресурсів, необхідність стабільного з’єднання з мережею є великою проблемою. Вони потребують програм, які працюють локально, без передачі даних назовні, не залежать від серверів або хмарних сервісів. Офлайн-рішення в такому випадку не просто бажані — вони критично необхідні. Саме тому автономні голосові помічники, що підтримують українську мову та не вимагають інтернету, можуть значно покращити якість життя цієї категорії користувачів, надати їм нові можливості для самостійного життя, навчання й роботи.

Важливо розуміти, що створення таких рішень потребує не лише технічної майстерності, а й глибокого розуміння потреб користувачів з порушеннями зору. Розробники повинні співпрацювати з самими користувачами на всіх етапах створення продукту — від початкового проектування до тестування та впровадження. Лише тоді технології будуть дійсно корисними, зручними та

адаптованими до реального повсякденного досвіду людей, які з ними працюватимуть.

Не менш важливим є і питання стандартизації та підтримки доступності на рівні державної політики. Програмне забезпечення, що використовується у сфері освіти, охорони здоров'я, соціального захисту та інших критично важливих галузях, має бути доступним для всіх користувачів незалежно від їх фізичних можливостей. Це означає обов'язкову інтеграцію зчитувачів екрану, підтримку клавіатурної навігації, наявність описів зображень (альтернативного тексту), а також локалізацію мовних інтерфейсів.

Слід заохочувати відкриті ініціативи та розвиток спільнот, які працюють над створенням доступних рішень. Вільне програмне забезпечення з відкритим кодом може стати основою для створення недорогих і адаптованих до локального контексту інструментів. Така модель дозволяє залучати розробників, волонтерів і самих користувачів до покращення продукту, прискорює процес оновлення і забезпечує більшу гнучкість у врахуванні специфічних запитів.

Варто підкреслити, що доступність — це не лише про інструменти. Це про повагу до людської гідності, про право кожного на рівний доступ до інформації, навчання, праці та спілкування. У сучасному цифровому світі виключення з технологічного середовища означає виключення з активного суспільного життя. Тому питання цифрової доступності для людей з порушеннями зору — це не питання додаткової опції, а базова вимога до будь-якої сучасної системи, яка претендує на універсальність і справедливість.

1.3 Інтеграція в цифровий простір

У сучасному суспільстві цифрові технології пронизують усі сфери життя — від освіти й медицини до спілкування та працевлаштування. Люди з порушеннями зору, однак, часто залишаються осторонь цієї трансформації, адже більшість інтерфейсів орієнтована винятково на зорове сприйняття. У таких умовах голосове керування стає не просто зручністю, а необхідністю.

Програма, що дозволяє керувати комп'ютером голосом, надає користувачеві

відчуття контролю над цифровим середовищем. Вона зменшує бар’єри у взаємодії з технікою, відкриваючи доступ до базових функцій, які для більшості людей є звичними, але для незрячих або слабозорих — часто недоступними. Завдяки цьому користувач може самостійно виконувати дії, які раніше вимагали сторонньої допомоги, що значно підвищує рівень особистої автономії та гідності.

Крім базового зручнішого доступу до системи, програма відіграє вирішальну роль у сфері освіти. Люди з порушеннями зору отримують можливість не лише опановувати нові навички, а й інтегруватися в освітній процес на рівних умовах. Вони можуть слухати лекції, читати текстові матеріали за допомогою синтезу мовлення, писати власні роботи та брати участь у дистанційному навчанні, не відчуваючи себе виключеними з навчального середовища. Самостійність у навчанні, своєю чергою, сприяє зростанню мотивації, впевненості у власних силах та прагненню до подальшого особистісного розвитку.

Соціальна взаємодія — ще один надзвичайно важливий вимір. Коли людина може безперешкодно користуватися електронною поштою, спілкуватися у месенджерах чи брати участь у відеоконференціях, вона не лише підтримує зв’язок зі світом, а й утверджує свою присутність у спільноті. Голосовий помічник у цьому контексті стає інструментом інтеграції, що забезпечує участь у соціальному, культурному та громадському житті.

Професійна діяльність також виграє від запровадження подібної програми. Люди з порушеннями зору отримують інструмент, який дозволяє їм виконувати робочі завдання на рівних з іншими — вести документообіг, комунікувати з колегами, готувати звіти, вести бухгалтерію, створювати контент. Це розширює можливості працевлаштування, зменшує соціальну ізоляцію та дозволяє повноцінно реалізовувати себе у професійному середовищі.

Варто наголосити на важливості автономності рішення. У багатьох регіонах України, особливо в сільській місцевості або в умовах воєнного стану, стабільне інтернет-з’єднання або доступ до комерційних сервісів є обмеженим або відсутнім. У такому випадку можливість працювати без підключення до мережі стає критично важливою. Програма, яка не потребує онлайн-доступу, гарантує безперебійну

роботу, не залежить від зовнішніх серверів і забезпечує повну конфіденційність даних. Це не лише зручність, а й питання інформаційної безпеки.

У довгостроковій перспективі поява таких рішень сприятиме зменшенню цифрової нерівності. Вона змінює парадигму взаємодії з технологіями, роблячи її більш справедливою, людяною та універсальною. Технології мають служити всім — і саме завдяки таким ініціативам вони дійсно стають доступними для кожного, незалежно від фізичних можливостей. Голос — це не лише інструмент керування. Це символ рівноправ'я, що відкриває шлях до повноцінної участі в сучасному світі.

1.4 Роль голосового помічника у майбутньому

У сучасному світі цифрових технологій доступ до інформації є однією з базових умов для повноцінної участі людини в суспільному, професійному та особистому житті. Проте для людей з порушеннями зору ця участь часто ускладнюється або навіть унеможливлюється через недоступність інтерфейсів, що в більшості своїй візуальні за своєю природою. Стандартна комп'ютерна система, з її піктограмами, вікнами, кнопками, спливаючими підказками та візуальними повідомленнями, орієнтована на користувача, який має повноцінний зір. Для незрячих або слабозорих користувачів взаємодія з такою системою перетворюється на безперервну боротьбу за інформацію, яка для інших здається очевидною. Навігація у файловій системі, запуск програм, зміна налаштувань — усе це вимагає спеціальних знань, допоміжного програмного забезпечення та неабиякої наполегливості. Сучасні читачі з екрана, як-от NVDA чи JAWS, дозволяють частково вирішити цю проблему, але далеко не завжди гарантують стабільну роботу, особливо в середовищах з нестандартним дизайном, у нових версіях програм або в онлайн-сервісах із неадаптованими інтерфейсами. Кожна зміна розташування кнопки, зміна шрифту або кольору може стати бар'єром. Людина змушена переосмислювати спосіб навігації, шукати нові шляхи, часто через проби й помилки. Це потребує значного когнітивного навантаження, витрат часу і сил, які могли б бути спрямовані на продуктивну роботу або навчання.

Викликом є відсутність повноцінних, зрозумілих і доступних засобів

голосового керування українською мовою. Більшість сучасних голосових асистентів, що працюють на базі великих мовних моделей, вимагають постійного інтернет-з’єднання. Це зумовлено тим, що обробка запитів відбувається на віддалених серверах, де зберігаються обчислювальні потужності. Для користувача це означає залежність від стабільного зв’язку, доступу до хмарних сервісів, а також готовність ділитися особистими даними з сторонніми платформами. В умовах, коли людина проживає в сільській місцевості, де якість інтернету є низькою, або перебуває в зоні бойових дій, де зв’язок нестабільний або повністю відсутній, така модель втрачає практичну цінність. Крім того, значна частина рішень недоступна безкоштовно, вимагає ліцензій або платних підписок. Це ще більше обмежує доступ до технологій, які мали б бути інструментом для подолання бар’єрів. Люди, які не володіють іноземними мовами, стикаються з додатковими труднощами: інтерфейси англійською, голосові системи, що не розпізнають українську мову, або не можуть коректно відтворювати команди. Це створює парадокс: технологія існує, але вона недоступна тим, хто потребує її найбільше.

У цьому контексті ідея створення автономного голосового асистента, який працює локально, підтримує українську мову, не залежить від інтернету та не передає дані третім сторонам, є не просто актуальною — вона стає необхідною. Такий інструмент дозволяє людям з порушеннями зору самостійно керувати комп’ютером, відкривати потрібні програми, читати документи, шукати інформацію або просто взаємодіяти з системою без необхідності натискати клавіші або шукати потрібні пункти меню. Це принципово змінює модель використання техніки. Користувач отримує реальний контроль над процесами, стає незалежним від сторонньої допомоги, розширює свої можливості. Особливо важливим є те, що така технологія має потенціал інтеграції не лише в комп’ютери, а й у побутову техніку, розумні будинки, навчальні середовища. Коли світ навколо починає реагувати на голос, це відкриває двері до нового рівня доступності. Побутові речі — від телевізора до чайника — перестають бути пасивними об’єктами і стають помічниками.

Майбутнє таких рішень — не у винятковості, а в універсальності. Технології,

що створені для людей з порушеннями зору, можуть бути корисними всім. Голосовий інтерфейс зручний для людей похилого віку, для тих, хто має труднощі з моторикою, для користувачів у складних умовах або просто для зручного користування в щоденному житті. Така технологія має потенціал змінити саму парадигму взаємодії з комп'ютером: не користувач має адаптуватися до машини, а машина — до людини. І чим доступнішою буде ця технологія, чим гнучкіше вона враховуватиме індивідуальні потреби користувачів, тим більше людей зможуть повноцінно брати участь у цифровому житті.

Голос — це природна форма комунікації. І якщо ми зможемо забезпечити можливість керувати цифровими пристроями голосом, без обмежень і бар'єрів, ми відкриємо шлях до по-справжньому інклюзивного світу, де технології не відділяють людей, а об'єднують їх, не створюють нерівність, а руйнують її. Саме в цьому — справжня цінність автономного голосового асистента для тих, хто щодня стикається з невидимими, але реальними перешкодами. Це не просто програма. Це ключ до свободи, самостійності і гідності.

1.5 Використання голосового помічника у побуті

Автономний голосовий асистент у побуті відіграє надзвичайно важливу роль, адже він суттєво полегшує повсякденне життя людей з порушеннями зору, забезпечуючи їм можливість самостійно виконувати широкий спектр завдань, які раніше вимагали допомоги інших або складних маніпуляцій із комп'ютером. Одним із найпростіших, але водночас важливих аспектів є запуск та закриття програм. Завдяки можливості простою голосовою командою активувати потрібний додаток користувач може швидко переходити до виконання своїх справ, не витрачаючи час на пошук відповідних ярликів або запам'ятовування розташування кнопок у меню. Це стає особливо цінним, коли мова йде про важливі для роботи або навчання програми — текстові редактори, браузери, поштові клієнти. Закриття ж непотрібних програм також відбувається легко і без зайвих зусиль, що допомагає тримати робочий простір організованим і не відволікатися на технічні нюанси.

Вагомим напрямком є керування мультимедіа, яке зазвичай є невід'ємною

частиною сучасного життя. Людина з порушеннями зору, маючи у своєму розпорядженні голосового асистента, отримує змогу безперешкодно керувати програванням музики, відео, аудіокниг та подкастів. За допомогою простих голосових команд вона може запускати чи зупиняти відтворення, змінювати гучність, переходити до наступного треку або повертатися до попереднього. Така функція не лише розширює доступ до розваг і інформації, а й сприяє емоційному розвантаженню, дає змогу проводити час із комфортом та задоволенням. Водночас можливість керувати мультимедіа голосом робить цей процес максимально інтуїтивним і природним, що важливо для користувачів, які звикли орієнтуватися на слух.

Диктування простих текстів є однією з найпродуктивніших функцій голосового асистента. Для людей з порушеннями зору, які можуть стикатися з труднощами при роботі з клавіатурою, ця опція відкриває можливості для створення текстових документів, коротких заміток, листів чи повідомлень у месенджерах. Замість тривалого набору тексту вручну, який потребує від користувача точності і може викликати втому, можна просто промовити думки, і система перетворить їх у письмову форму. Це значно прискорює процес комунікації, дає змогу швидко фіксувати ідеї, організовувати робочі завдання або просто вести особистий щоденник. Водночас інтеграція функції диктування з можливістю озвучування повідомлень та документів створює замкнутий цикл взаємодії, який дозволяє повністю обходитися без традиційних пристроїв введення й виведення інформації.

Автономний голосовий асистент дозволяє швидко виконувати типові побутові завдання, які часто потребують використання різноманітних інструментів чи програм. Наприклад, користувач може легко встановити будильник, запланувати нагадування, дізнатися поточну дату і час, перевірити прогноз погоди або навіть виконати прості обчислення. Усі ці дії голосовий помічник виконує без необхідності переходити до складних меню або користуватися клавіатурою та мишею, що суттєво економить час і знижує когнітивне навантаження. Це особливо важливо для тих, хто живе самостійно, адже подібні можливості підтримують

повсякденну організацію життя і роблять її більш передбачуваною та комфортною.

Всі ці функції об’єднуються у потужний інструмент, який не лише підвищує рівень автономії людини з порушеннями зору, а й робить повсякденне життя більш простим і приємним. Завдяки відсутності залежності від інтернету та складних налаштувань, такий голосовий асистент стає доступним широкому колу користувачів, включаючи тих, хто проживає у віддалених регіонах або не має змоги користуватися дорогими онлайн-сервісами. Саме тому подібні рішення є важливим кроком на шляху до повноцінної інклюзії і рівних можливостей у цифровому суспільстві, дозволяючи долати бар’єри, що раніше здавалися нездоланними. Завдяки цьому голосовий помічник стає не просто технологічним нововведенням, а справжнім другом і підтримкою у повсякденному житті, що допомагає людині бути більш незалежною, впевненою та соціально активною.

1.6 Соціальне значення

Голосові помічники мають велике соціальне значення, адже вони суттєво спрощують доступ до цифрових технологій для різних категорій населення. Особливо це стосується людей з обмеженими фізичними можливостями, наприклад, з вадами зору чи руху, для яких традиційне керування комп’ютером або смартфоном може бути складним або навіть неможливим. Голосові асистенти надають цим людям можливість більш незалежно виконувати повсякденні завдання, спілкуватися, навчатися та працювати, що підвищує їхню якість життя та соціальну інтеграцію.

Голосові помічники допомагають знизити цифровий розрив між різними поколіннями. Старші люди, які часто мають менший досвід роботи з технологіями, отримують інтуїтивний і зручний спосіб взаємодії з гаджетами без необхідності вивчати складні інтерфейси. Це сприяє їхній активній участі у сучасному цифровому світі, збереженню зв’язку з родиною та суспільством, а також підтримці когнітивної активності.

Важлива роль голосових помічників у забезпеченні доступності інформації та послуг. Вони можуть автоматично надавати важливі повідомлення, нагадування,

допомагати з пошуком інформації, бронюванням або оплатою послуг, що особливо актуально для людей, які мають обмежений доступ до традиційних каналів комунікації. Це робить повсякденне життя більш комфортним і ефективним для широкого кола користувачів.

Голосові помічники також відіграють важливу роль у професійному середовищі, особливо для людей з обмеженими можливостями. Вони дозволяють виконувати робочі завдання без необхідності використовувати клавіатуру чи мишу, що розширює можливості працевлаштування та підвищує продуктивність. Такий інструмент сприяє створенню більш інклюзивного робочого простору, де кожен працівник має рівний доступ до ресурсів і можливостей.

Важливим аспектом є підтримка навчання за допомогою голосових технологій. Голосові помічники можуть допомогти студентам та учням з вадами зору або труднощами з моторикою брати активну участь у навчальному процесі, отримувати інформацію, диктувати відповіді та керувати навчальними програмами. Це сприяє доступності освіти і дозволяє здобувати знання без додаткових бар’єрів.

З соціальної точки зору голосові асистенти сприяють розширенню кола спілкування. Вони допомагають користувачам залишатися на зв’язку з близькими, навіть якщо вони фізично обмежені в пересуванні або користуванні звичними пристроями. За допомогою голосових команд можна надсилати повідомлення, робити дзвінки або користуватися соціальними мережами, що підтримує емоційну близькість і соціальну активність.

Голосові помічники також мають потенціал змінити підхід до медичного обслуговування. Вони можуть надавати інформацію про прийом ліків, нагадувати про візити до лікаря або допомагати у зв’язку з медичними пристроями, що особливо важливо для людей із хронічними захворюваннями чи літніх людей. Така підтримка покращує якість життя і здоров’я користувачів, роблячи медичні послуги більш доступними.

Соціальне значення голосових помічників відображається і у культурному аспекті — вони допомагають зберігати та популяризувати мови, зокрема рідні та

менш поширені, через інтеграцію локалізованих мовних моделей. Це підтримує мовне різноманіття і сприяє збереженню культурної спадщини в епоху глобалізації.

Голосові помічники виступають як міст між людьми і технологіями, допомагаючи подолати технічні бар'єри і забезпечуючи рівний доступ до цифрового світу. Їхній розвиток і впровадження сприяють створенню більш справедливого та відкритого суспільства, де кожен має можливість повноцінно брати участь у соціальному, освітньому і професійному житті.

Соціальна значущість голосових помічників зростає і в умовах швидкого розвитку «розумних» міст і інтелектуальних систем, де вони стають частиною більшої екосистеми, що підтримує рівноправність доступу до ресурсів і сервісів. Голосові інтерфейси роблять технології більш гуманними, наближаючи їх до природної комунікації людини, що сприяє більш гармонійному поєднанню людей і техніки у повсякденному житті.

2 АНАЛІЗ ТЕХНОЛОГІЙ І ВИБІР ЗАСОБІВ РОЗРОБКИ

2.1 Мова програмування C#

Мова програмування C# є однією з ключових технологій, яка використовується у сучасному програмуванні для створення різноманітних програмних продуктів, зокрема десктопних застосунків, вебсервісів, мобільних додатків та складних корпоративних систем. Вона була створена компанією Microsoft на початку 2000-х років у рамках розвитку платформи .NET. Основною метою створення C# було поєднання високого рівня абстракції з потужними інструментами керування ресурсами, надійною безпекою та широкими можливостями масштабування додатків.

C# має синтаксис, що нагадує мови C-подібного ряду, зокрема C++, Java та JavaScript, що забезпечує легкість засвоєння для розробників із досвідом роботи з іншими популярними мовами. Вона підтримує парадигму об’єктно-орієнтованого програмування, що дозволяє моделювати прикладні області через абстракції, інкапсуляцію, спадкування та поліморфізм. Об’єктно-орієнтований підхід у C# підвищує гнучкість програмного забезпечення, його повторне використання та простоту супроводу.

Особливістю мови C# є її глибока інтеграція з .NET-платформою. Завдяки цьому всі програми, написані на C#, виконуються у середовищі Common Language Runtime (CLR), що забезпечує автоматичне керування пам’яттю, безпечне виконання коду, перевірку типів під час виконання, обробку виключень та багато інших сервісів. CLR є ключовим елементом платформи .NET, який дозволяє компілювати код C# у проміжну мову (IL), що потім транслюється у машинний код відповідно до конкретної платформи виконання. Це дає змогу запускати програми на будь-якому пристрої, де встановлено відповідну реалізацію .NET.

C# також підтримує сучасні концепції функціонального програмування, що дозволяє розробникам ефективно використовувати лямбда-вирази, функції вищого порядку, LINQ (Language Integrated Query) для маніпуляцій з колекціями та базами даних. Завдяки LINQ C# забезпечує високу

читабельність коду, дозволяючи писати вирази, схожі на SQL-запити, без необхідності у створенні складної логіки вручну.

З розвитком технологій C# постійно вдосконалюється: з кожною новою версією мови додаються сучасні конструкції, які покращують продуктивність та знижують ймовірність помилок. Наприклад, такі можливості як pattern matching, nullable reference types, async/await для асинхронного програмування, та record types для створення незмінних типів суттєво полегшили розробку складних програмних рішень.

Завдяки кросплатформеній підтримці .NET Core (а тепер .NET 5+), C# вийшла за межі Windows, що дозволяє розробникам створювати програми під Linux, macOS, Android, iOS, веб та хмарні середовища. Це робить мову особливо привабливою для підприємств, які прагнуть універсальності, масштабованості та підтримки сучасних архітектур, таких як мікросервіси та контейнеризація.

У C# закладено потужну підтримку засобів розробки, таких як JetBrains Rider, Visual Studio та інші IDE, які забезпечують розширене автодоповнення, дебагінг, аналіз коду в реальному часі, автоматичне тестування та інструменти для розгортання програм. Це дозволяє підвищити ефективність команди розробки, мінімізуючи технічні помилки та спрощуючи процес супроводу програмного забезпечення.

C# активно використовується в розробці ігор завдяки популярному ігровому рушію Unity, який має вбудовану підтримку C#. Це відкриває великі можливості для студентів, інді-розробників і навіть великих студій у створенні інтерактивних додатків та ігор.

Безперечними перевагами C# є висока безпека, строгість типізації, багатий набір бібліотек, велика спільнота розробників, доступна документація та активна підтримка з боку Microsoft. Мова постійно еволюціонує у відповідь на зміни в галузі IT, що робить її актуальною як для навчання, так і для створення серйозних промислових рішень.

C# виступає універсальним інструментом програмування, який дозволяє

реалізовувати як прості навчальні проєкти, так і великі корпоративні рішення, забезпечуючи надійність, масштабованість і високу продуктивність програмного забезпечення. Її застосування охоплює практично всі сфери сучасної IT-індустрії — від десктопних додатків до хмарних сервісів і мобільних платформ.

2.2 C# як оптимальне середовище для розробки Windows-додатків з голосовим управлінням

У межах реалізації спеціалізованого комп'ютерного комплексу, призначеного для полегшення роботи особам із вадами зору, питання вибору мови програмування має принципове значення. Одним із найважливіших критеріїв при цьому є сумісність із операційною системою, підтримка голосових сервісів, продуктивність, а також наявність сучасного інструментарію для створення графічного інтерфейсу та взаємодії з апаратними та програмними ресурсами комп'ютера. Усі ці вимоги найповніше задовольняє мова програмування C#, яка стала ключовим інструментом реалізації даного проєкту.

Однією з головних переваг C# є її сильна типізація, яка знижує ймовірність логічних помилок та забезпечує високу надійність і передбачуваність поведінки програмного коду. Це особливо важливо в проєктах, де точність обробки голосових команд і системної взаємодії є критичною. Багатий стандартний класовий набір, що входить до складу .NET, дозволяє реалізовувати складні функціональні рішення без потреби у сторонніх залежностях, що скорочує час розробки й підвищує стабільність програми.

C# має повноцінну підтримку багатопотоковості, що дозволяє ефективно організовувати паралельну обробку команд, забезпечувати плавність взаємодії з користувачем і оперативно реагувати на голосові запити. Усе це значно підвищує продуктивність системи в режимі реального часу.

Для проєктів, орієнтованих на платформу Windows, C# є майже

безальтернативним вибором. Завдяки глибокій інтеграції з операційною системою Windows, мова забезпечує простий і безпосередній доступ до API та внутрішніх сервісів ОС. Це дозволяє реалізовувати пряме керування вікнами, системними діалогами, службами синтезу та розпізнавання мовлення, а також здійснювати взаємодію з пристроями введення/виведення на низькому рівні.

Важливою перевагою є нативна підтримка бібліотек для синтезу та розпізнавання мовлення. Наприклад, бібліотека System.Speech дозволяє швидко реалізувати як голосовий ввід, так і генерацію звукових повідомлень. Це особливо важливо для користувачів з вадами зору, адже дає змогу повноцінно взаємодіяти з додатком за допомогою голосу — без необхідності зорового контакту з інтерфейсом. У поєднанні з рушієм розпізнавання мовлення Vosk, який має вбудовану підтримку української мови, C# дозволяє створити зручне та локалізоване рішення, що повністю адаптоване під потреби українських користувачів.

Аспект, який не можна оминути, — це підтримка з боку спільноти. C# має потужну, активну та добре структуровану спільноту розробників, що дозволяє швидко знаходити відповіді на технічні запитання, користуватися великою кількістю прикладів, документації, відкритих бібліотек та розширень. Завдяки цьому розробка системи значно спрощується і пришвидшується, а потенційні труднощі розв’язуються без суттєвих затримок.

Якщо порівнювати C# з іншими мовами, які часто використовуються в аналогічних проєктах, переваги C# стають ще очевиднішими. Наприклад, мова Python широко застосовується у сфері машинного навчання і вважається зручною для створення прототипів, однак вона має низку обмежень при створенні повноцінних десктопних додатків, особливо для Windows. Інтеграція Python із системними ресурсами Windows та голосовими API, такими як System.Speech, є складнішою і вимагає сторонніх бібліотек чи обгорток, що підвищує складність реалізації та знижує стабільність. Java, в свою чергу, є кросплатформеною мовою, однак при створенні саме Windows-орієнтованих застосунків вона поступається C# у зручності, ефективності

доступу до системних функцій і сумісності з голосовими сервісами Windows.

C# виграє у порівнянні завдяки своїй нативній інтеграції з Windows, можливості просто і надійно використовувати системні компоненти, таким як бібліотеки для синтезу мови, доступ до API, а також завдяки повноцінній підтримці таких інструментів, як Visual Studio або Rider. Ці середовища розробки надають широкий спектр інструментів — від інтелектуального автодоповнення до інтегрованого тестування та дебагу, що значно покращує якість та швидкість написання коду.

Окрім технічних переваг, слід також звернути увагу на аспект довгострокової підтримки та масштабованості рішень, реалізованих на C#. Ця мова є одним із стратегічних напрямків розвитку компанії Microsoft, що гарантує її актуальність у найближчі десятиліття. Регулярні оновлення, поява нових функцій та вдосконалення середовища .NET Core, яке перетворилося на сучасну кросплатформену платформу .NET, створюють надійне підґрунтя для подальшої еволюції застосунків. Тобто проєкт, реалізований на базі C#, матиме можливість не тільки підтримуватися в актуальному стані, а й розвиватися — розширювати функціонал, адаптуватися під нові технології, інтегруватися з хмарними платформами та штучним інтелектом.

Розробка на C# дозволяє дотримуватися принципів модульності, що особливо важливо при створенні складних систем з багатьма компонентами. Завдяки об'єктно-орієнтованому підходу, C# сприяє розділенню логіки програми на незалежні частини, які легко підтримувати, оновлювати або навіть замінювати без ризику порушити роботу всього комплексу. Наприклад, окремий модуль синтезу мовлення можна буде вдосконалити або замінити на більш точну модель, не змінюючи основну логіку керування інтерфейсом чи роботою з файлами. Така гнучкість має вирішальне значення в довготривалих проєктах соціального призначення, де потреби користувачів можуть змінюватися з часом.

C# підтримує сучасні принципи безпеки, включаючи контроль доступу, обмеження на виконання шкідливого коду, а також можливість використання

криптографічних засобів для захисту персональних даних користувачів. Для програм, орієнтованих на осіб з вадами зору, це особливо важливо, адже нерідко такі додатки оперують конфіденційною інформацією — наприклад, контактами, документами, електронною поштою. Вбудовані засоби шифрування та автентифікації, доступні через API .NET, дають змогу організувати високий рівень захисту інформації без необхідності звертатися до сторонніх інструментів.

C# дозволяє створювати інтерфейси, які враховують вимоги доступності. За допомогою засобів Windows Forms або WPF (Windows Presentation Foundation) можна реалізувати елементи керування з підтримкою скрінрідерів, налаштування кольорів і шрифтів, а також інтеграцію з функціями Windows Accessibility. У поєднанні з голосовим управлінням, це формує цілісну екосистему, у якій користувач із порушеннями зору не відчуває себе обмеженим і може повноцінно користуватися комп'ютером без сторонньої допомоги.

У C#, є розвинена система логування та обробки винятків. У системах, орієнтованих на критичні дії користувача, важливо забезпечити надійний механізм виявлення та аналізу помилок. Завдяки широким можливостям журналювання подій у C# можна вести повну історію взаємодії з програмою, що спрощує як тестування, так і підтримку системи після її впровадження. У разі виникнення виняткових ситуацій розробник швидко зможе визначити причину збою і внести відповідні зміни, не порушуючи роботи решти системи.

Можливості інтеграції C# з іншими компонентами, які можуть бути додані до комплексу в майбутньому. Наприклад, можна легко під'єднати базу даних для збереження історії голосових команд або впровадити машинне навчання (через ML.NET) для адаптації системи до індивідуальних потреб користувача. Таким чином, C# дозволяє не лише реалізувати початкову функціональність, а й забезпечити платформу для масштабування та персоналізації у перспективі.

Варто відзначити якість доступної документації, прикладів та

навчальних матеріалів, які постійно оновлюються Microsoft та незалежними розробниками. Це суттєво скорочує час входження в нові технології навіть для малодосвідчених спеціалістів і створює передумови для формування команди підтримки або розвитку продукту після впровадження. Зокрема, існують безкоштовні курси, посібники, відеоуроки, які охоплюють як базові, так і просунуті аспекти розробки на C#, що сприяє стабільності довготривалого життєвого циклу програмного продукту.

C# — це не просто зручна і потужна мова програмування, а повноцінна екосистема для створення надійного, масштабованого та зручного програмного забезпечення. У випадку створення спеціалізованого комп'ютерного комплексу для осіб із вадами зору це рішення поєднує технічну ефективність із соціальною значущістю, відкриваючи нові можливості для інклюзивного використання комп'ютерних технологій.

2.3 Бібліотека Vosk

Vosk — це сучасний офлайн-движок для розпізнавання мовлення, який ідеально підходить для сценаріїв, де важлива автономність, безпека й конфіденційність. Його ключова особливість полягає у повній незалежності від інтернету, що дозволяє розробляти додатки, здатні працювати без доступу до мережі. Це особливо актуально для польових умов, локальних систем без зовнішніх з'єднань або застосунків із підвищеними вимогами до захисту персональних даних. Vosk підтримує розпізнавання мовлення на десятках мов, включно з українською, що робить його рідкісним і цінним інструментом із якісною локалізацією. Точність розпізнавання залишається високою навіть на слабких пристроях, завдяки оптимізованим мовним моделям, які не потребують значних обчислювальних ресурсів.

Перевага Vosk є простота інтеграції та використання. Розробникам не потрібно створювати складні алгоритми з нуля — достатньо завантажити готову мовну модель і підключити її до свого додатку. Модульний підхід дозволяє обробляти аудіо по частинах, отримуючи як проміжні, так і остаточні результати,

що забезпечує гнучкість у побудові логіки голосового керування. Для голосових помічників це означає можливість реагувати на команду ще до її повного проголошення, роблячи взаємодію швидкою і природною.

Vosk підтримує роботу з багатоканальним аудіо, що відкриває додаткові можливості для підвищення якості розпізнавання в умовах шуму, відокремлення джерел звуку та створення складних конфігурацій із кількома мікрофонами. Це особливо важливо для використання голосових систем у офісах, навчальних аудиторіях чи виробничих приміщеннях, де фонова активність може заважати точній роботі.

Слід відзначити активну підтримку спільнотою розробників і регулярні оновлення Vosk. Це гарантує не лише стабільність та сумісність із новими платформами, а й впровадження нових функцій, оптимізацій і покращень якості розпізнавання. Таким чином, вибір Vosk для розробки голосового помічника забезпечує довгострокову перспективу розвитку проєкту і його адаптацію до майбутніх викликів у сфері голосових технологій. Vosk не лише забезпечує високу точність розпізнавання, а й дозволяє ефективно працювати з різними типами аудіо сигналів. Наприклад, він здатний обробляти як чистий мовний сигнал, так і аудіо з фоновим шумом, що особливо важливо для реального використання голосових помічників у різних середовищах. Це означає, що користувачі можуть комфортно спілкуватися з системою навіть у шумному офісі, на вулиці або в інших складних акустичних умовах.

Особливістю Vosk є підтримка роботи в реальному часі. Завдяки оптимізованим алгоритмам і ефективній обробці аудіо, система здатна миттєво реагувати на голосові команди, що створює відчуття живого діалогу між користувачем і пристроєм. Це особливо актуально для голосових асистентів, де швидкість обробки запитів безпосередньо впливає на комфорт користування.

Vosk має гнучкі можливості налаштування мовних моделей. Розробники можуть адаптувати їх під специфічні галузі або завдання, наприклад, створювати словники спеціалізованих термінів, що підвищує точність розпізнавання у медичних, технічних або юридичних сферах. Така кастомізація розширює сферу

застосування системи і робить її універсальним інструментом для бізнесу та освіти.

Vosk інтегрується з різними платформами і мовами програмування, що дозволяє будувати кросплатформенні рішення — від десктопних додатків до мобільних і веб-сервісів. Це дає змогу розробникам швидко запускати продукти на різних пристроях, забезпечуючи однаковий користувацький досвід незалежно від операційної системи.

Використання Vosk у голосових асистентах сприяє підвищенню рівня інклюзії, оскільки офлайн-робота і підтримка української мови роблять технологію доступною для ширшого кола користувачів, включно з тими, хто має обмежений доступ до інтернету або потребує особливих засобів для взаємодії з комп’ютером. Це відкриває нові можливості для людей з особливими потребами та розширює горизонти цифрової взаємодії для всіх.

Vosk є одним із найкращих інструментів для тих, хто прагне поєднати ефективність, автономність і безпеку в системах розпізнавання мови. Його можливості дозволяють створювати інтуїтивні, швидкі та надійні голосові помічники, які працюватимуть у будь-яких умовах — від домашнього користування до професійних застосувань із підвищеними вимогами до якості й конфіденційності.

2.4 Бібліотека NAudio.Wave

NAudio.Wave — це одна з найпопулярніших бібліотек для роботи зі звуком у середовищі .NET, яка поєднує в собі простоту використання та потужний функціонал. Вона дозволяє розробникам отримувати прямий доступ до аудіопристроїв, зокрема мікрофонів, та зчитувати аудіопотік у реальному часі, що є критично важливим для створення інтерактивних застосунків, таких як голосові помічники. Завдяки підтримці різних форматів звуку та можливості гнучкого налаштування параметрів аудіо, NAudio дає змогу забезпечити високу якість запису й відтворення, а також швидко обробляти звук без значних затримок.

Перевага NAudio здатність працювати з аудіо у форматі WAV із частотою дискретизації 16 кГц та моно режимом — саме такі параметри рекомендуються для

передачі в системи розпізнавання мови, зокрема Vosk. Це забезпечує оптимальний баланс між якістю звуку та навантаженням на систему, дозволяючи досягти високої точності розпізнавання голосу. Використовуючи WaveInEvent, клас із бібліотеки NAudio, можна налаштувати зчитування аудіо у подієвому режимі, що спрощує інтеграцію в реальний час і дає можливість миттєво передавати аудіодані на розпізнавання або зберігання.

Крім базових функцій захоплення звуку, NAudio також надає інструменти для аудіоаналізу, таких як створення спектрограм, застосування фільтрів і ефектів, а також змішування різних аудіопотоків. Це відкриває широкі можливості для створення складних аудіододатків, що можуть адаптувати звук під конкретні потреби користувача або сценарії використання. Наприклад, у голосовому асистентові можна реалізувати автоматичне регулювання гучності або фільтрацію шумів, що значно підвищує якість розпізнавання і комфорт користування.

NAudio.Wave також відзначається своєю здатністю легко інтегруватися з іншими бібліотеками та сервісами для обробки аудіо, що робить її незамінною складовою у складі комплексних рішень. Наприклад, у системах голосового розпізнавання вона часто використовується у парі з такими інструментами, як Vosk або Microsoft Speech SDK, де NAudio відповідає за якісний захват звуку, а спеціалізовані модулі — за його трансформацію у текст. Така архітектура дозволяє максимально розділити функції і зберегти високу продуктивність при мінімальних затримках.

Важливим аспектом є гнучкість налаштувань, які пропонує NAudio.Wave. Розробник може визначати розмір буфера, кількість каналів, формат аудіо, що дозволяє підлаштувати захоплення звуку під різні апаратні можливості і специфіку проєкту. Це особливо важливо для застосунків, що працюють на різних пристроях — від потужних ПК до ноутбуків і навіть мобільних платформ, де ресурси можуть бути обмеженими. Завдяки цій універсальності NAudio використовується не лише в голосових помічниках, але й у музичних програмах, редакторах звуку, аудіоаналізаторах.

NAudio підтримує роботу з потоковим аудіо, що дає змогу створювати

застосунки, здатні приймати і обробляти звук у режимі онлайн. Це корисно для систем, які збирають голосові дані з мережі або підключених пристроїв, наприклад, у випадках віддаленого навчання, відеоконференцій чи голосових чатів. Завдяки цій функції бібліотека може стати основою для більш складних проектів, де аудіо потрібно не лише записати, але й транслювати чи аналізувати в режимі реального часу.

NAudio постійно розвивається, отримує оновлення і покращення, що відображає сучасні тенденції в роботі зі звуком і підтримує новітні формати та стандарти. Спільнота розробників активно доповнює бібліотеку новими можливостями, що дозволяє їй залишатися актуальною і сумісною з останніми версіями .NET і Windows. Такий рівень підтримки гарантує, що застосунки на базі NAudio будуть стабільними і безпечними протягом тривалого часу.

NAudio.Wave — це не просто інструмент для роботи зі звуком, а справжня платформа, яка дозволяє створювати різнопланові аудіо рішення. Вона ідеально підходить як для простих застосунків, які лише записують або відтворюють звук, так і для складних систем, що потребують тонкої настройки, аналізу і обробки аудіосигналів. Для розробників голосових помічників це означає, що з її допомогою можна побудувати надійний, швидкий та гнучкий модуль захоплення звуку, який стане основою для точного розпізнавання і комфортного користування.

NAudio має добре документований API і активне ком’юніті, яке постійно підтримує і розвиває бібліотеку. Це дозволяє швидко вирішувати виникаючі питання, знаходити готові рішення для типових задач і бути впевненим у стабільності роботи аудіокомпонентів навіть у складних проєктах. Таким чином, NAudio.Wave є надійним і гнучким інструментом для розробки програм, які потребують якісної роботи зі звуком у .NET середовищі, особливо коли мова йде про обробку голосу в режимі реального часу.

2.5 Бібліотека System.Speech.Synthesis

System.Speech.Synthesis є надзвичайно важливою і зручною складовою .NET, яка дозволяє програмам безпосередньо озвучувати текст голосом, використовуючи

вбудовані можливості операційної системи Windows. На відміну від сторонніх сервісів чи хмарних API, ця бібліотека не потребує додаткових інсталяцій або мережевого підключення, що робить її надзвичайно стабільною та автономною для роботи навіть у офлайн-режимі. Вона відкриває широкий спектр можливостей для розробників, які хочуть інтегрувати в свої додатки функції синтезу мовлення для створення голосових помічників, діалогових систем або інтерактивних голосових інтерфейсів.

Основний інструмент бібліотеки — клас SpeechSynthesizer — дозволяє легко керувати процесом озвучування тексту. Він надає інтуїтивно зрозумілі методи для початку мовлення, призупинки, відновлення та зупинки, що дає змогу реалізувати гнучкі сценарії взаємодії з користувачем. Програміст може налаштовувати параметри голосу: вибирати між чоловічим і жіночим голосом, змінювати швидкість мовлення, регулювати гучність і навіть задавати інтонацію. Якщо у системі встановлено відповідний мовний пакет, наприклад український голос, програма зможе якісно озвучувати текст саме цією мовою, що є надзвичайно важливим для користувачів, які віддають перевагу спілкуванню рідною мовою.

Ключовою перевагою System.Speech.Synthesis є її простота і мінімальні вимоги до реалізації. Для того, щоб озвучити текст, достатньо кількох рядків коду — це значно спрощує інтеграцію голосових функцій у вже існуючі проекти. Такий підхід дозволяє швидко створювати інтуїтивні і зрозумілі голосові інтерфейси без потреби залучати складні алгоритми або сторонні сервіси, що може бути особливо корисним у проєктах для людей з вадами зору, де надійність і простота мають критичне значення.

Використання вбудованих голосових синтезаторів дозволяє уникнути багатьох проблем, пов’язаних із захистом приватності, адже весь процес обробки тексту відбувається локально на комп’ютері користувача без передачі даних через інтернет. Це особливо важливо у випадках, коли користувачі працюють з конфіденційною інформацією або перебувають у середовищах з обмеженим доступом до мережі. Таким чином, бібліотека забезпечує не лише зручність, а й безпеку.

Важливою перевагою System.Speech.Synthesis є її гнучкість і розширюваність. Вона підтримує події, які можна відстежувати у програмі, наприклад, момент початку і завершення мовлення, що дозволяє синхронізувати візуальні або інші аудіо-відповіді з озвучуванням тексту. Це робить можливим створення комплексних мультимедійних додатків, у яких голосова взаємодія є лише однією зі складових загального користувацького досвіду.

System.Speech.Synthesis — це не просто інструмент для перетворення тексту на мовлення, а ціла платформа, яка надає можливості для створення гнучких та адаптивних голосових інтерфейсів. Вона підтримує роботу з різними мовними пакетами, що дозволяє розробникам легко інтегрувати синтез мови для численних мов і діалектів, забезпечуючи широку аудиторію користувачів. Це особливо важливо у багатомовних країнах або для застосунків, які орієнтовані на міжнародний ринок. Наявність української мови, зокрема, робить цю бібліотеку привабливою для розробників, що працюють у нашому регіоні, оскільки дозволяє створювати продукти, адаптовані до потреб місцевих користувачів.

SpeechSynthesizer має можливість не лише відтворювати текст, але й керувати процесом озвучування у динамічному режимі. Наприклад, можна програмно контролювати паузи між словами або реченнями, змінювати темп мовлення залежно від контексту або емоційного забарвлення тексту. Це дає змогу створювати більш природні та живі голосові відповіді, що підвищує зручність і комфорт взаємодії з користувачем. Для голосових помічників та інших інтерактивних систем це особливо цінно, адже таке налаштування допомагає зробити спілкування максимально наближеним до живого діалогу.

Особливість — можливість збереження синтезованої мови у звукові файли різних форматів. Це відкриває широкі можливості для застосування бібліотеки у створенні аудіокниг, голосових повідомлень, навчальних матеріалів та мультимедійних презентацій. Розробник може попередньо підготувати озвучені фрагменти і зберегти їх для подальшого використання, що зменшує навантаження на процесор у реальному часі і підвищує швидкодію програми. Ця функція особливо корисна для мобільних чи вбудованих систем, де обмежені ресурси.

System.Speech.Synthesis підтримує різноманітні події і зворотні виклики, які дозволяють відслідковувати стан синтезу в режимі реального часу. Це відкриває можливості для глибшої інтеграції з іншими компонентами програмного забезпечення — наприклад, для синхронізації озвучування з анімацією, візуальними підказками чи логікою діалогів. Завдяки такій гнучкості розробники можуть створювати комплексні, багатофункціональні рішення, які відповідають сучасним вимогам до доступності та зручності використання. У підсумку, System.Speech.Synthesis є надійним і потужним інструментом для реалізації голосового синтезу в будь-яких типах додатків — від простих голосових помічників до складних інтерактивних систем.

System.Speech.Synthesis — це ідеальний вибір для розробників, які хочуть додати до своїх програм можливість природного озвучування тексту без зайвих складнощів. Завдяки її простоті, ефективності та інтеграції з операційною системою, вона є базовим інструментом для створення голосових помічників, навчальних програм, аудіокниг, а також програмної підтримки користувачів із особливими потребами, зокрема з вадами зору.

2.6 Бібліотека System.Diagnostics

System.Diagnostics — це один із найважливіших і найпотужніших інструментів у .NET, який забезпечує широкий доступ до функціональності операційної системи. Його ключова роль полягає в тому, щоб дати розробникам можливість запускати, контролювати і взаємодіяти з процесами, а також збирати різноманітні діагностичні дані, необхідні для моніторингу та налагодження програм. У рамках розробки голосового помічника для людей з вадами зору саме ця бібліотека дає основу для реалізації критично важливих функцій, які роблять програму корисною, інтерактивною та автономною.

Клас Process, що є центральним елементом System.Diagnostics, дозволяє не просто запускати сторонні додатки, а й повністю керувати їхнім життєвим циклом. Можна не лише відкривати програму за командою користувача, але й перевіряти, чи вона вже запущена, припиняти її роботу, отримувати зворотній зв’язок про

статус процесу, а також налаштовувати спосіб запуску — наприклад, запуск у прихованому режимі, щоб не відволікати користувача, або з перенаправленням виводу, що може бути використано для складніших інтерактивних сценаріїв. Такі можливості особливо важливі для голосового помічника, який має реагувати на різноманітні запити користувача максимально швидко і безпомилково.

System.Diagnostics дозволяє програмі збирати дані про продуктивність комп'ютера, відслідковувати використання системних ресурсів і отримувати інформацію про всі активні процеси. Це відкриває широкі перспективи для створення адаптивного помічника, який міг би аналізувати стан системи і оптимізувати роботу залежно від навантаження. Наприклад, у випадку, якщо система перевантажена, помічник може порадити користувачу закрити деякі додатки або самостійно завершити неважливі процеси, щоб покращити продуктивність. Це значно підвищує зручність та безпеку використання, особливо для людей, які не можуть швидко реагувати на технічні складнощі через обмежені можливості зору.

System.Diagnostics є стандартною частиною .NET Framework і .NET Core, що робить цю бібліотеку кросплатформенною і надійною. Для розробника це означає відсутність необхідності підключати сторонні пакети чи платити за додаткові ліцензії — усе, що потрібно, вже доступне у стандартному наборі інструментів. Завдяки цьому голосовий помічник, створений на базі цієї технології, може працювати стабільно і ефективно на різних комп'ютерах з Windows без додаткових налаштувань.

Важливим аспектом є безпека та контроль. Використовуючи System.Diagnostics, розробник може встановити строгі обмеження на запуск певних програм, захистити користувача від потенційно небезпечних дій та забезпечити логування всіх виконаних команд. Це створює додатковий рівень захисту для користувачів із вадами зору, які можуть випадково запустити невідповідні або шкідливі програми. Система звітності допомагає у випадку потреби діагностики помилок або підтримки користувачів.

Використання System.Diagnostics вбудовує в голосовий помічник

фундаментальні механізми інтеграції з операційною системою, які дозволяють максимально автоматизувати керування комп’ютером голосом, одночасно забезпечуючи гнучкість, безпеку та стабільність роботи. Це робить програму не просто інструментом для виконання базових команд, а потужним помічником, який підвищує якість життя і сприяє самостійності користувачів із порушеннями зору, дозволяючи їм повноцінно взаємодіяти з комп’ютерними системами у сучасному цифровому світі.

2.7 Порівняння Vosk та аналогів

Vosk вважається одним із найкращих варіантів для голосового розпізнавання, особливо коли йдеться про проєкти, що мають працювати офлайн, підтримувати українську мову та бути максимально гнучкими. Його основна перевага полягає саме в автономності: для роботи йому не потрібне підключення до інтернету, на відміну від багатьох інших систем, які покладаються на хмарні сервіси (як-от Google Speech API чи Microsoft Azure Speech). Це критично важливо для програм, які запускаються в умовах обмеженого доступу до мережі, наприклад у військових, медичних або мобільних застосунках.

На відміну від PocketSphinx, який теж працює офлайн, Vosk підтримує набагато ширший перелік мов, включаючи українську, причому з досить високою якістю розпізнавання як показано на рисунку 2.1. PocketSphinx — це дуже легкий і старий інструмент, який швидко працює навіть на слабких пристроях, але його точність значно нижча, а якість моделей часто вимагає ручного налаштування граматик, лексиконів, що ускладнює його використання. Vosk же йде з готовими мовними моделями, які не потрібно навчати з нуля — їх достатньо просто завантажити та підключити, після чого система вже готова до роботи. Це суттєво спрощує інтеграцію.

У порівнянні з бібліотекою SpeechRecognition у Python, яка також підтримує Vosk як бекенд, але за замовчуванням використовує онлайн-сервіси, основна перевага Vosk у версії для C# полягає в тому, що він працює автономно й безпосередньо через API, не потребуючи сторонніх обгорток. Крім того,

SpeechRecognition не дуже добре інтегрується з .NET-середовищем, оскільки спочатку створювався для Python. Це робить Vosk набагато привабливішим варіантом, якщо ти працюєш у C# або хочеш створити голосового асистента для Windows.

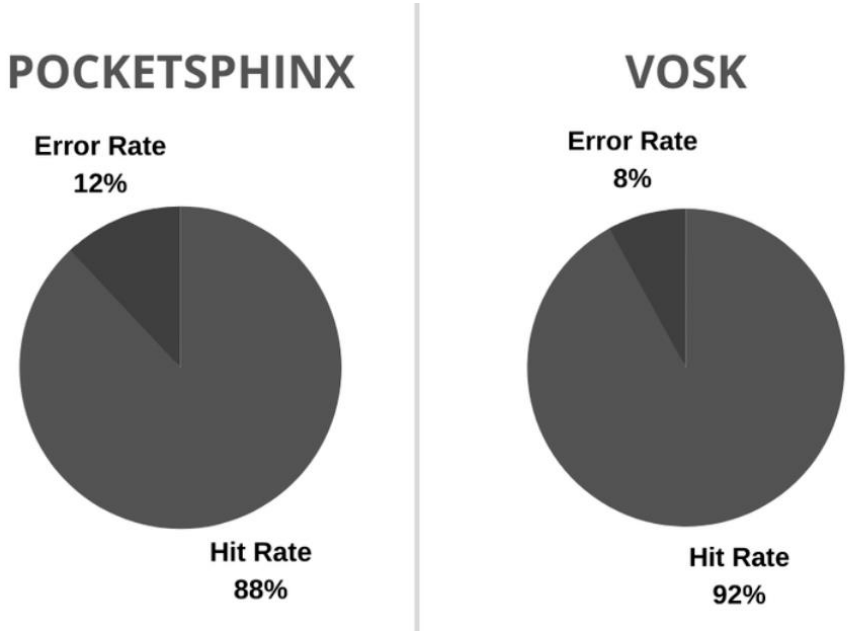

Рисунок 2.1 — Порівння Vosk і Pocketsphinx за точністю розпізнавання

Vosk також має високу швидкість обробки мовлення. Завдяки використанню ефективних моделей на основі Kaldi, він здатний працювати в реальному часі навіть на середньому за характеристиками комп'ютері. Це дає змогу створити інтерфейс, де користувач не відчуває затримки між тим, як він говорить, і тим, як програма реагує. Однак варто зазначити, що моделі Vosk досить великі — україномовна модель, наприклад, може важити кілька сотень мегабайт. Це означає, що програмі все ж потрібно більше оперативної пам'яті, ніж PocketSphinx, але ця вартість компенсується якістю та точністю.

Vosk має відкритий код, активне співтовариство і постійно розвивається, що робить його сучасним вибором для офлайн-розпізнавання. Він добре підходить не лише для простих додатків, а й для складних рішень, таких як діалогові системи, голосові інтерфейси, командні панелі, застосунки для стенографування чи автоматичного субтитрування.

2.8 Клас SpeechSynthesizer

Клас SpeechSynthesizer із простору імен System.Speech.Synthesis є невід'ємною частиною .NET Framework, що забезпечує можливість перетворення тексту в мовлення, що має надзвичайне значення для створення голосових інтерфейсів. Ця технологія є базою для численних застосунків, які полегшують життя людей, особливо тих, хто має проблеми зі зором або труднощі у сприйнятті візуальної інформації. Використання SpeechSynthesizer дозволяє зробити програмне забезпечення більш інклюзивним, відкриваючи доступ до інформації через голос, що є однією з найприродніших форм спілкування людини.

Основна функціональність класу полягає у підтримці широкого спектра голосів, які встановлені у Windows. Це дає змогу обирати між різними варіантами озвучення, враховуючи мовні та гендерні переваги користувача. Наприклад, наявність українського голосу, такого як Microsoft Irina, є важливим фактором для локалізації голосових помічників для українськомовних користувачів. Крім вибору голосу, SpeechSynthesizer дозволяє регулювати параметри швидкості та гучності, що допомагає підлаштувати відтворення тексту під індивідуальні потреби людини, зробити його більш комфортним для слухання.

Технологія синтезу мовлення, реалізована через SpeechSynthesizer, є особливо корисною у сценаріях, де необхідна оперативна взаємодія з користувачем. Наприклад, голосовий асистент може миттєво озвучувати відповіді на запити користувача, підтверджувати виконання команд або повідомляти про помилки. Такий зворотній зв'язок є ключовим для людей з вадами зору, адже він замінює візуальні сигнали і дозволяє повноцінно працювати з комп'ютером без допомоги інших осіб. Реалізація функції озвучення тексту без затримок і з високою якістю є можливою саме завдяки ефективності SpeechSynthesizer.

Клас підтримує не тільки відтворення мовлення в реальному часі, але й збереження аудіо у форматі WAV або інших підтримуваних форматах. Це дозволяє створювати аудіофайли з інструкціями, довідковими матеріалами чи будь-якою іншою інформацією, які можна відтворювати незалежно від роботи самої програми. Така можливість розширює сфери застосування голосових технологій,

дозволяючи використовувати їх у навчанні, освіті, професійній діяльності та повсякденному житті користувачів.

Технічна простота використання SpeechSynthesizer також є однією з його найбільших переваг. Для інтеграції синтезу мовлення у додаток достатньо кількох рядків коду, що значно прискорює розробку і знижує бар'єри для впровадження голосових функцій у програмне забезпечення. Це дає змогу навіть невеликим командам чи окремим розробникам створювати інноваційні рішення для людей з особливими потребами, які раніше були доступні лише у дорогих або складних у налаштуванні системах.

Завдяки роботі без підключення до інтернету SpeechSynthesizer забезпечує повну автономність роботи голосового асистента, що є критично важливим для користувачів у регіонах з обмеженим доступом до мережі або для тих, хто цінує конфіденційність своїх даних. Відсутність залежності від онлайн-сервісів виключає ризики витоку особистої інформації і робить систему більш стабільною та передбачуваною у використанні.

Впровадження такого синтезатора мовлення в голосових помічниках суттєво підвищує інклюзивність цифрових технологій, відкриває нові горизонти для навчання, роботи, спілкування та розваг людей з вадами зору. Голосові повідомлення, які звучать природно і зрозуміло, створюють комфортне середовище для користувача, дозволяють швидко орієнтуватися в системі, виконувати необхідні дії та отримувати інформацію без зайвих складнощів. Таким чином, SpeechSynthesizer стає не просто технічним компонентом, а справжнім містком між користувачем і комп'ютером.

Цей клас є незамінним інструментом для розробки сучасних голосових інтерфейсів, які допомагають людям з особливими потребами відчувати себе більш незалежними та впевненими у цифровому світі. Він поєднує у собі простоту, гнучкість, надійність і високу якість відтворення, що робить його оптимальним вибором для створення дружніх і доступних голосових помічників.

2.9 Клас Process

Клас Process із простору імен System.Diagnostics є надзвичайно важливим і потужним інструментом для запуску та керування зовнішніми процесами в операційній системі. Його головне призначення полягає в тому, щоб дозволити програмі створювати, запускати, взаємодіяти та контролювати виконання інших програм або системних команд. Для розробників, які створюють голосових помічників, ця можливість є ключовою, оскільки більшість завдань, що виконує асистент, напряму пов’язані з відкриттям та керуванням різноманітними додатками — текстовими редакторами, браузерами, файловими менеджерами, іграми тощо.

При використанні класу Process, розробник може дуже просто запускати будь-яку програму, просто вказавши ім’я виконуваного файлу або його повний шлях. Наприклад, команда «відкрити Блокнот» реалізується викликом Process.Start("notepad.exe"), після чого система сама завантажить і відкриє відповідний додаток. Це значно спрощує інтеграцію голосових команд із реальними діями на комп’ютері, роблячи взаємодію інтуїтивною і природною для користувача.

Крім базового запуску, клас Process підтримує численні параметри і опції, що роблять його надзвичайно гнучким. Можна налаштовувати середовище запуску, передавати аргументи у програми, визначати робочий каталог, а також контролювати вікно, в якому запущено додаток — приховувати його або навпаки робити активним. Це особливо корисно для створення більш складних сценаріїв роботи голосового асистента, де потрібно запускати кілька програм одночасно або керувати їхнім інтерфейсом без прямої участі користувача.

Process дозволяє розробнику контролювати хід виконання запущених програм. Можна очікувати їх завершення, отримувати коди виходу, що допомагає відслідковувати успішність виконання команд, або у разі потреби примусово завершувати процес. Ці можливості особливо важливі для забезпечення стабільної роботи голосового помічника, який повинен адекватно реагувати на помилки або некоректні дії користувача, забезпечуючи безперебійну роботу і зворотний зв’язок.

Клас Process є частиною базової бібліотеки .NET Framework/.NET Core, що гарантує його стабільність, сумісність і безпеку. Розробник не потребує додаткових

сторонніх бібліотек або компонентів для реалізації запуску і контролю програм, що знижує складність проєкту і підвищує надійність роботи голосового помічника.

Використання Process дає змогу інтегрувати голосового помічника з найрізноманітнішим програмним забезпеченням, починаючи від системних утиліт і закінчуючи спеціалізованими бізнес-додатками. Це відкриває широкі можливості для автоматизації повсякденних завдань, що особливо актуально для людей з вадами зору, яким голосовий помічник допомагає подолати бар'єри у використанні комп'ютера.

Клас Process підтримує перенаправлення стандартних потоків вводу-виводу, що дозволяє створювати більш інтерактивні сценарії. Наприклад, голосовий помічник може запускати програму, передавати їй команди або отримувати від неї результати для подальшої обробки. Це розширює функціональність системи і дозволяє реалізовувати складні діалоги між користувачем і комп'ютером.

Клас Process є фундаментальною складовою будь-якого голосового помічника, що прагне надати користувачу повний контроль над комп'ютером голосом. Завдяки своїй простоті, гнучкості і потужності він забезпечує безперебійне і швидке виконання команд, що є ключовим для комфортного і ефективного використання технології людьми з обмеженими можливостями зору. Його можливості дозволяють створювати справді інклюзивні комп'ютерні системи, які роблять цифровий світ доступним для всіх.

2.10 Клас VoskRecognizer

VoskRecognizer є центральним компонентом бібліотеки Vosk, яка сьогодні вважається одним із найсучасніших і найефективніших рішень для офлайн-розпізнавання мовлення. Його головна функція полягає у прийомі аудіосигналу, аналізі цього потоку звуку і перетворенні його у текстовий формат у режимі реального часу. Це робить VoskRecognizer ідеальним для застосувань, де потрібна швидка та надійна взаємодія голосом, особливо у випадках, коли неможливо або небажано використовувати хмарні сервіси через обмеження в доступі до інтернету або з міркувань приватності.

Для роботи з VoskRecognizer спочатку необхідно завантажити та ініціалізувати мовну модель, яка відповідає мові користувача — у нашому випадку це українська. Мовна модель містить інформацію про фонетику, лексику, граматику та інші характеристики мови, що дає змогу розпізнавати слова та фрази з високою точністю. Після ініціалізації в систему надходять аудіодані у вигляді невеликих фрагментів, які VoskRecognizer аналізує по мірі їх надходження, швидко визначаючи текстовий вміст. Такий підхід дозволяє створювати системи, здатні реагувати на голосові команди майже миттєво, без відчутних затримок.

Ключовою перевагою VoskRecognizer є підтримка як проміжних, так і остаточних результатів розпізнавання. Проміжні (partial) результати надають частковий текст, який система вже «бачить» на певному етапі прослуховування, але може змінитися до фінального варіанту. Це дозволяє будувати динамічні сценарії взаємодії, наприклад, реагувати на окремі ключові слова чи фрази ще до того, як користувач закінчить говорити, що значно підвищує швидкодію і зручність роботи голосового помічника. Остаточні (final) результати — це повністю оброблений і затверджений текст, який можна використовувати для виконання команд або інших дій.

VoskRecognizer повністю працює офлайн і не вимагає підключення до інтернету. Це не лише забезпечує приватність даних користувача, але й робить технологію доступною для людей, які живуть у регіонах із поганим або нестабільним зв’язком, а також для користувачів, які свідомо обирають автономні рішення. Завдяки цьому голосові помічники, побудовані на VoskRecognizer, можуть успішно застосовуватися у різних сферах — від побуту до професійної діяльності.

Крім підтримки української мови, VoskRecognizer має відкриту архітектуру, що дозволяє легко інтегрувати його у різноманітні додатки та системи. Він сумісний із різними платформами, включно з Windows, Linux, Android, iOS, що робить його універсальним інструментом для розробників. Його застосування варіюється від простих голосових помічників і систем диктування тексту до складних інтелектуальних систем управління, де точність і швидкість

розпізнавання є критичними.

VoskRecognizer підтримує багатоканальне розпізнавання, що дає змогу працювати з кількома джерелами звуку одночасно. Це особливо корисно в складних аудіоумовах, де необхідно відокремити голос користувача від фонового шуму або розпізнавати мову декількох співрозмовників.

VoskRecognizer — це потужний і гнучкий інструмент, який поєднує в собі високу точність, швидкість, автономність і зручність інтеграції. Саме завдяки цим якостям він став невід'ємною складовою сучасних систем голосового управління, особливо тих, які розробляються для людей з обмеженими можливостями, надаючи їм можливість комфортно і ефективно взаємодіяти з комп'ютером без необхідності користуватися клавіатурою чи мишкою.

2.11 Клас WaveInEvent

Клас WaveInEvent із бібліотеки NAudio є однією з найважливіших складових для організації аудіозахоплення в режимі реального часу в додатках на платформі .NET. Його конструкція базується на подієвій моделі, яка суттєво спрощує обробку аудіопотоку, що надходить із мікрофона, забезпечуючи миттєву реакцію програми на голос користувача. Завдяки цьому WaveInEvent став популярним вибором серед розробників, які створюють голосові помічники, системи розпізнавання мови, програми для обробки звуку та інші аудіододатки.

Принцип роботи WaveInEvent полягає в тому, що він періодично зчитує аудіодані з підключеного пристрою введення (наприклад, мікрофона) і розбиває цей потік на невеликі буфери фіксованого розміру. Коли один з таких буферів наповнюється, генерується подія DataAvailable, до якої можна підписатися. Це дозволяє програмі оперативно отримувати нові аудіодані без необхідності постійно опитувати пристрій вручну. Таким чином, WaveInEvent знижує складність коду, уникає надмірного використання ресурсів і забезпечує ефективне асинхронне захоплення звуку.

Важливою перевагою WaveInEvent є можливість тонкого налаштування параметрів аудіозахоплення. Розробник може вибрати частоту дискретизації,

кількість каналів (моно або стерео), а також розрядність звуку (кількість біт на зразок). Для голосових систем розпізнавання, таких як Vosk, стандартною є частота 16 кГц і моно-режим, що забезпечує оптимальний баланс між якістю звуку та обсягом даних, які необхідно обробити. Крім того, правильний вибір цих параметрів суттєво впливає на точність та швидкість роботи системи розпізнавання мови.

WaveInEvent підтримує роботу з широким спектром аудіоінтерфейсів, що робить його гнучким інструментом для застосування на різних пристроях — від вбудованих мікрофонів ноутбуків і планшетів до зовнішніх професійних аудіокарт. Це забезпечує сумісність розроблених програм із різноманітними конфігураціями користувачів і дозволяє досягти максимальної якості захоплення аудіо.

Важливим аспектом є підтримка асинхронної роботи, яка дає змогу одночасно обробляти аудіопотік і виконувати інші обчислювальні задачі, не блокуючи при цьому інтерфейс користувача. Для голосових помічників це критично, адже затримки у захопленні чи обробці звуку можуть значно погіршити користувацький досвід, знизити швидкість і точність взаємодії. WaveInEvent допомагає уникнути цих проблем, забезпечуючи плавність і стабільність роботи.

Клас WaveInEvent підтримує буферизацію, що дозволяє адаптувати розмір аудіобуферів залежно від конкретних потреб застосунку. Менші буфери дозволяють швидше отримувати аудіодані і швидше реагувати на голосові команди, але можуть збільшувати навантаження на систему, тоді як більші буфери зменшують частоту обробки подій, що економить ресурси, але може додавати затримку. Така гнучкість дає змогу оптимізувати роботу системи під конкретні умови.

WaveInEvent також забезпечує високу стабільність роботи і надійність навіть у складних умовах — наприклад, при наявності кількох аудіопристроїв або під час зміни конфігурації звукової системи в реальному часі. Він автоматично обробляє внутрішні помилки пристрою та відновлює захоплення аудіо, що дозволяє уникнути збоїв і втрати звуку під час роботи.

WaveInEvent є незамінним інструментом для розробників, які створюють

голосові помічники та інші аудіододатки, особливо ті, що орієнтовані на користувачів з обмеженими можливостями. Його функціональність, простота використання та можливість точного налаштування роблять його ідеальним вибором для реалізації систем, які потребують швидкої, надійної та високоякісної обробки аудіосигналу в реальному часі. Завдяки цьому класу голосові асистенти можуть миттєво реагувати на команди, забезпечуючи комфортну і ефективну взаємодію з комп’ютером для людей з вадами зору.

3 РОЗРОБКА АРХІТЕКТУРИ КОМП'ЮТЕРНОГО КОМПЛЕКСУ ДЛЯ ПОЛЕГШЕННЯ РОБОТИ ОСОБАМ З ВАДАМИ ЗОРУ

3.1 Середовище розробки JetBrains Rider та розробка програми

JetBrains Rider — це інтегроване середовище розробки (IDE) для .NET, яке підтримує програмування мовою C# та роботу з платформами .NET, ASP.NET, Unity, Unreal Engine та іншими технологіями. Воно є потужним та продуктивним, забезпечує глибоку інтеграцію з ReSharper, розширені можливості аналізу коду, швидке автодоповнення та ефективне налагодження.

Основною мовою програмування в Rider є C#, що дозволяє створювати високопродуктивні додатки з використанням об'єктно-орієнтованого підходу.

Для того щоб створити програму спочатку заходимо у JetBrains Rider та після відкриття натискаємо "New Solution" як на рисунку 3.1.

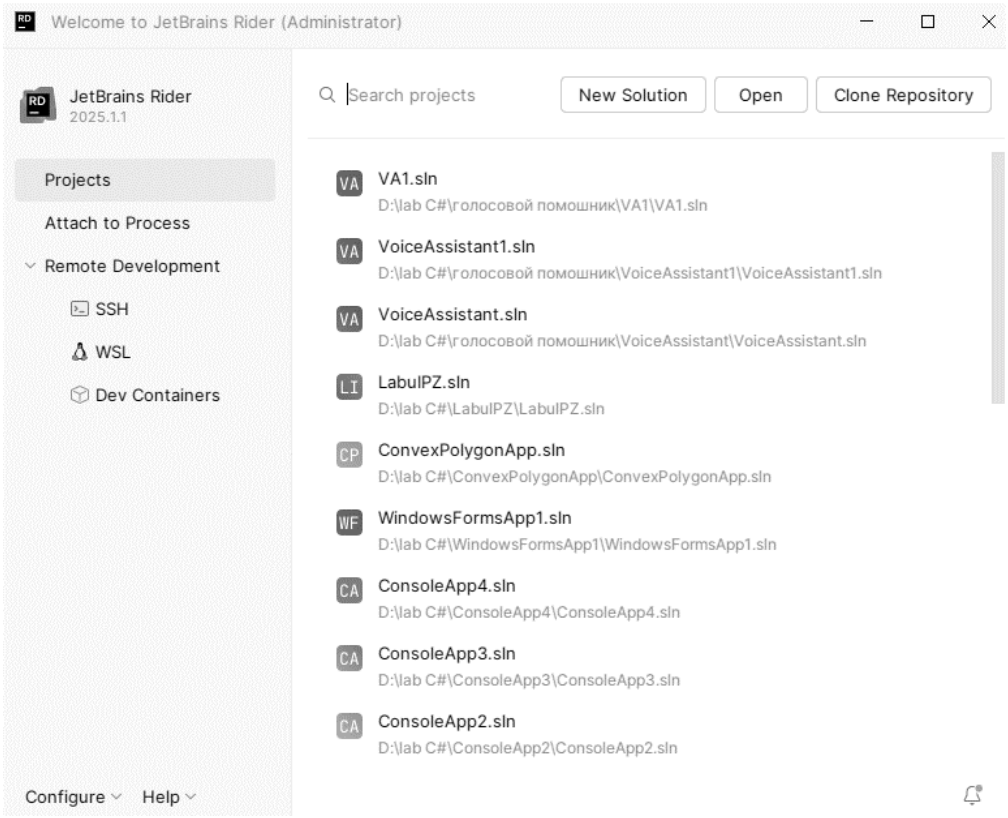

Рисунок 3.1 — Вікно запуску JetBrainsRider

Після того як натиснули обираємо "Desktop" та задаємо назву проекту після чого у "Template" обираємо "Windows Forms App" та натискаємо "Create" як на

рисунку 3.2.

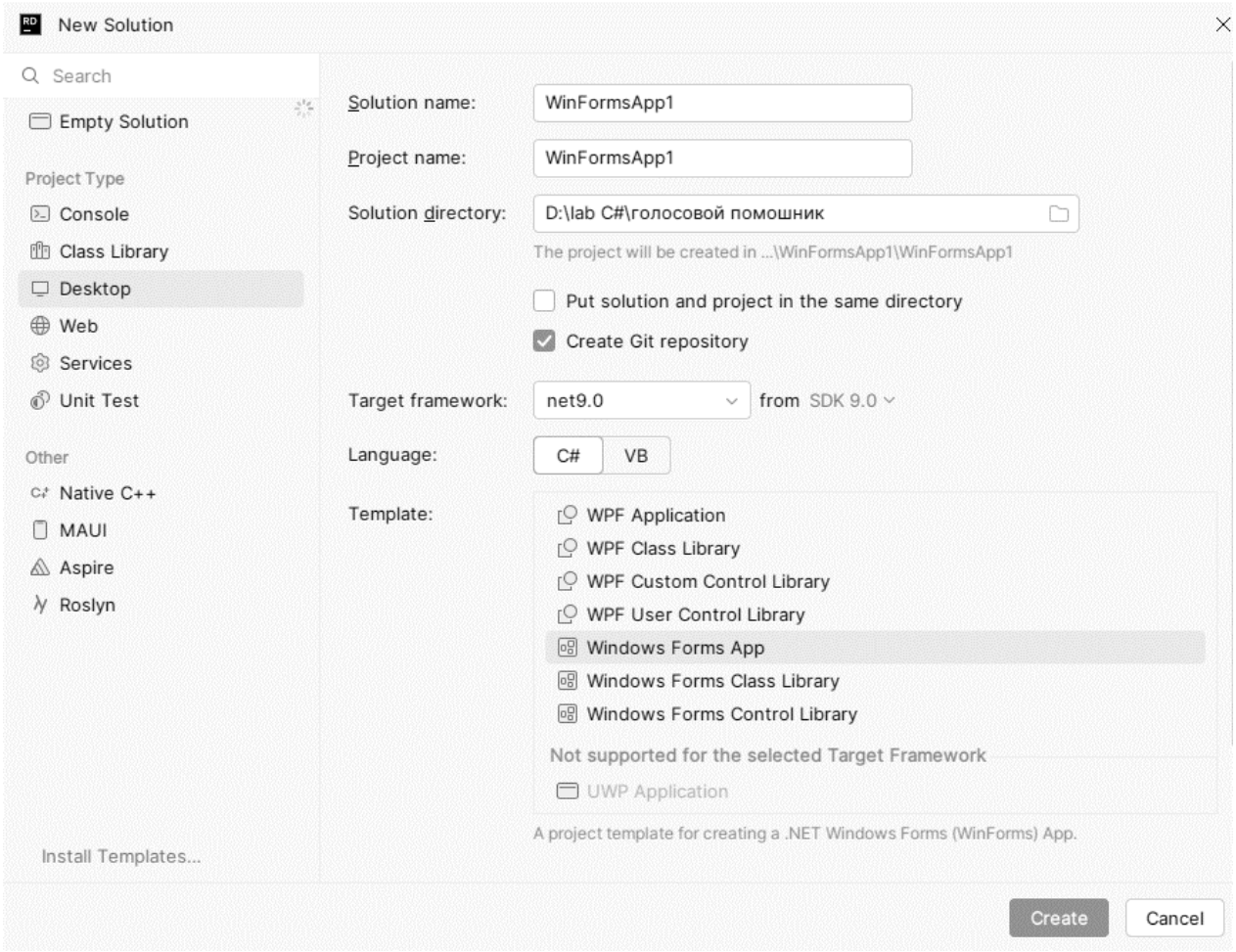

Рисунок 3.2 — Створення програми

Щоб було можливо працювати з кодом у програмі, нам потрібно додати бібліотеки: Vosk.Speech, System.Diagnostics, System.Speech та натиснути Install як на рисунку 3.3.

Рисунок 3.3 — Додавання бібліотек у програму

Після цього нам потрібно завантажити офлайн бібліотеку Vosk та прописати у коді шлях до неї, для цього потрібно зайти на офіційний сайт Vosk прогортати до української мови та обрати там файл vosk-model-uk-v3 як на рисунку 3.4.

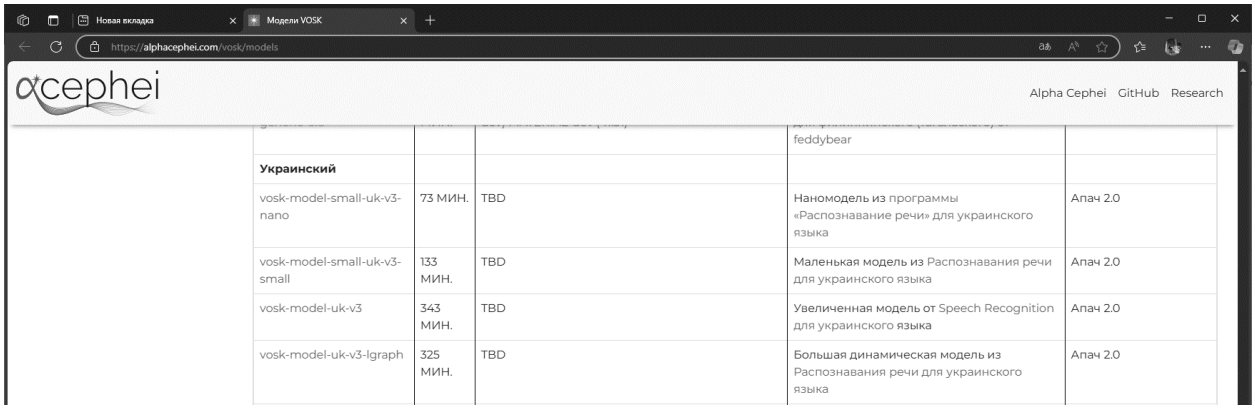

Рисунок 3.4 — Завантаження бібліотеки Vosk

Завантажувати потрібно у папку з англійськими літерами інакше програма буде вибивати помилку що не може знайти шлях до файлу, після того як все вірно завантажили прописуємо у коді шлях до файлу як на рисунку 3.5.

```
private string RecognizeSpeech()
{
    string modelPath = @"D:\Games\vosk-model-uk-v3";

    if (!Directory.Exists(modelPath))
    {
        Speak( text: $"❌ Помилка: Модель не знайдена у {modelPath}");
        return null;
    }
```

Рисунок 3.5 — Прописуємо шлях до бібліотеки

Після цього необхідно створити UML-діаграму як на рисунку 3.6, яка відображає взаємозв’язки між основними компонентами голосового помічника та сценарії його роботи. UML-діаграма є важливим етапом розробки, оскільки допомагає структуровано представити архітектуру програмного комплексу, визначити взаємодію між класами та зрозуміти алгоритм виконання команд.

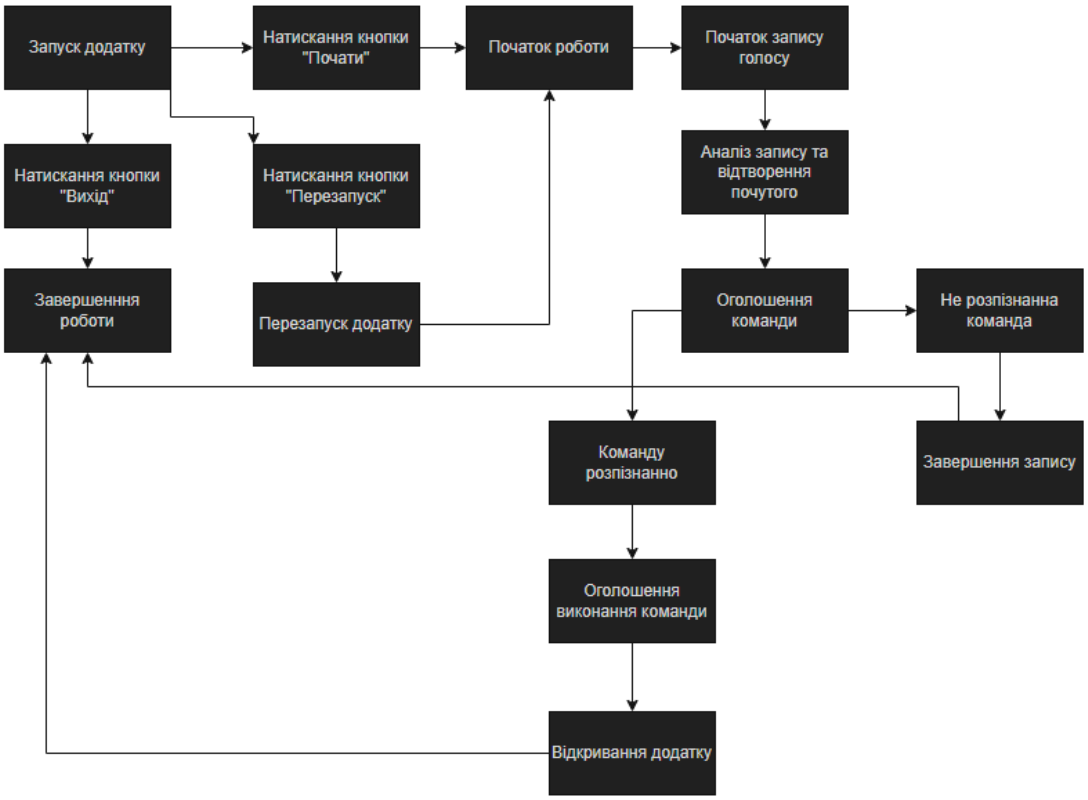

Рисунок 3.6 — UML-діаграма

3.2 Розпізнавання голосу

Розпізнавання голосу є ключовим елементом роботи автономного голосового асистента, що забезпечує можливість точного та швидкого перетворення мовлення користувача у текстову форму без необхідності підключення до інтернету. Для реалізації цієї функції використовується модель Vosk, яка підтримує локальне розпізнавання української мови, що робить її особливо цінною для користувачів, які не мають постійного доступу до мережі або хочуть зберегти конфіденційність своїх даних. Підключення моделі передбачає завантаження спеціалізованої мовної моделі, адаптованої під особливості української мови, що дозволяє підвищити точність розпізнавання та зменшити кількість помилок.

Процес розпізнавання голосу починається із запуску режиму реального часу, коли програма прослуховує звуковий потік, захоплений з мікрофона користувача. VoskRecognizer приймає ці аудіодані, які надходять у форматі з частотою дискретизації 16 кГц — стандарті, що забезпечує достатню якість сигналу для коректної обробки. Модель здатна працювати з багатоканальним аудіо, що

особливо важливо в умовах домашнього середовища, де можуть бути присутні різні фонові шуми, і дозволяє більш ефективно відокремити голос користувача від інших звуків.

Після отримання аудіосигналу VoskRecognizer обробляє його, розпізнає мовленнєві патерни та перетворює їх у текстовий рядок, який далі використовується для виконання команд або введення тексту. Особливістю цієї моделі є можливість адаптації під голос конкретного користувача, що покращує якість розпізнавання з часом і дозволяє враховувати індивідуальні особливості вимови, інтонації та швидкості мовлення. Завдяки цьому процес взаємодії стає більш природним, а точність обробки команд — максимально високою навіть у складних акустичних умовах.

Інтеграція Vosk у систему автономного голосового асистента забезпечує надійне і ефективне розпізнавання української мови, що є фундаментом для подальшого керування комп’ютером голосом без потреби у сторонніх сервісах чи інтернет-з’єднанні. Це робить технологію більш доступною, безпечною і комфортною для широкого кола користувачів, зокрема для людей з порушеннями зору, яким важливо зберегти автономність і контроль над власними пристроями.

3.3 Запис голосу

Запис голосу є важливим етапом у системі автономного голосового асистента, оскільки якість захопленого аудіосигналу безпосередньо впливає на точність розпізнавання мовлення. Для реалізації цього процесу широко використовується бібліотека NAudio, яка забезпечує стабільне та гнучке захоплення звуку з мікрофона користувача. Завдяки своїй ефективності NAudio дозволяє працювати з аудіопотоком у реальному часі, що є критично важливим для оперативного реагування голосового асистента на команди.

Під час запису встановлюється формат WAV із параметрами 16 кГц частоти дискретизації, 16-бітної глибини та монофоничного звуку. Такий формат є оптимальним для подальшої обробки за допомогою моделі Vosk, оскільки він забезпечує достатню якість звуку при порівняно невеликому обсязі даних, що

дозволяє зберігати високу швидкодію системи навіть на малопотужних пристроях. Монофоничний режим спрощує обробку, адже модель розпізнавання працює з одним аудіоканалом, що зменшує навантаження на обчислювальні ресурси.

Вхідні аудіодані обробляються у режимі реального часу з використанням буферизації, що дозволяє зберігати короткі фрагменти звуку для подальшого аналізу або збереження. Буферизація забезпечує безперервний потік даних, уникаючи втрат інформації навіть у разі тимчасових затримок у роботі системи. Це особливо важливо, адже мова користувача часто містить паузи або змінюється швидкість, а завдання системи — точно зафіксувати всі звукові сигнали для коректної інтерпретації.

Запис коротких аудіофрагментів дає можливість не лише негайно обробляти команду, а й зберігати її у вигляді файлу, що може бути корисним для ведення журналу взаємодії або подальшого аналізу. Такий підхід сприяє підвищенню надійності роботи системи і дає змогу при необхідності покращувати модель розпізнавання або відновлювати інформацію у разі помилок.

Використання бібліотеки NAudio для запису голосу в поєднанні з оптимальним аудіоформатом і механізмом буферизації створює надійну основу для якісного й ефективного захоплення аудіоданих, що є ключовою складовою автономного голосового асистента, який працює без підключення до хмарних сервісів. Це гарантує швидкий і точний зв’язок між користувачем та системою, забезпечуючи комфорт і доступність технології для широкого кола користувачів.

3.4 Виконання команд

Виконання команд у системі автономного голосового асистента базується на аналізі розпізнаного тексту, що отримується після перетворення голосових сигналів у текстову форму. Першим кроком є пошук у цьому тексті ключових слів, які відповідають конкретним діям або програмам, що користувач хоче запустити. Наприклад, якщо у команді містяться слова "гра", "провідник" чи "блокнот", система розпізнає їх як інструкції відкрити відповідні програми — ігри, файловий менеджер або текстовий редактор. Такий підхід дозволяє швидко і просто

зв’язувати мовленнєві команди з реальними діями на комп’ютері, забезпечуючи інтуїтивність та зручність користування.

Після визначення наміру користувача відбувається виклик відповідної програми за допомогою функції Process.Start, яка запускає потрібний додаток у операційній системі. Цей метод є стандартним і надійним, що дозволяє інтегрувати широкий спектр програм та сервісів без складних налаштувань. Голосовий асистент при цьому виступає посередником між користувачем і комп’ютером, перетворюючи усні команди на конкретні дії, які виконуються миттєво.

Важливою складовою системи є обробка помилок, що можуть виникати під час виконання команд. Наприклад, якщо користувач видає команду, яка не містить розпізнаних ключових слів, або ж зазначена програма відсутня на пристрої, система повинна коректно проінформувати про це користувача, не припиняючи роботу і не викликаючи плутанину. Це може бути реалізовано через голосові повідомлення або текстові сповіщення, які підказують, що команда не була зрозуміла або програма не знайдена, і пропонують повторити запит або спробувати іншу команду. Такий підхід забезпечує більш плавну взаємодію та дозволяє користувачеві швидко орієнтуватися у можливостях асистента, навіть якщо виникають непередбачені ситуації.

Система виконання команд у голосовому асистентові поєднує простоту розпізнавання ключових слів із надійністю запуску програм і грамотним управлінням помилками, що разом забезпечує зручність, ефективність і стабільність роботи, особливо важливі для користувачів з порушеннями зору, які потребують безперебійної та інтуїтивної взаємодії з комп’ютером.

3.5 Синтез мовлення

Синтез мовлення в автономному голосовому асистенті виконує важливу роль зворотного зв’язку, озвучуючи результати виконання команд та інформуючи користувача про стан системи. Для цієї задачі застосовується компонент SpeechSynthesizer, який перетворює текстову інформацію у звуковий сигнал природного звучання. Це дозволяє користувачам з порушеннями зору отримувати

оперативні відомості про виконані дії, помилки або підтвердження команд без необхідності дивитися на екран, що значно підвищує комфорт і автономність користування комп’ютером.

Вибір голосу є ключовим фактором для забезпечення природності та зрозумілості мовлення. У цьому випадку для синтезу звуку використовується голос Microsoft Irina Desktop, який підтримує українську мову та відзначається приємним тембром і чіткістю вимови. Завдяки цьому користувачі можуть легко сприймати озвучену інформацію навіть у складних акустичних умовах, що важливо для безперебійної та ефективної роботи системи.

Крім озвучення результатів виконання команд, синтезатор мовлення також інформує користувача про успішність або наявність помилок. Наприклад, після запуску програми або виконання певної дії, асистент може підтвердити успіх голосовим повідомленням, а у разі проблем – повідомити про неможливість виконати запит або про необхідність повторення команди. Такий діалог робить взаємодію більш зрозумілою та дружньою, допомагаючи уникнути непорозумінь і зменшуючи рівень стресу користувача.

Для адаптації озвучення до індивідуальних потреб і вподобань користувача передбачено налаштування параметрів швидкості та гучності синтезованого мовлення. Це дозволяє регулювати темп озвучення, роблячи його більш спокійним або динамічним, а також підлаштовувати рівень звуку під навколишні умови чи особисті вподобання. Така гнучкість підвищує загальну зручність і ефективність використання голосового асистента, особливо для людей з різним ступенем порушень зору, які можуть мати різні вимоги до сприйняття аудіоінформації.

3.6 Аналіз завдання та порівняння аналогів

Порівнюючи розроблену автономну систему голосового асистента з популярними голосовими помічниками, такими як Cortana від Microsoft, Siri від Apple чи Google Assistant, варто детальніше розглянути ключові відмінності, що роблять цю розробку особливо актуальною для користувачів з порушеннями зору та обмеженим доступом до інтернету. В першу чергу, більшість відомих асистентів

працюють у режимі онлайн і значною мірою покладаються на хмарні сервіси для обробки голосових команд, збереження даних та навчання моделей. Це створює ряд обмежень і ризиків: залежність від стабільного інтернет-з’єднання, ризики втрати конфіденційності, а також можливі проблеми з доступністю у віддалених або малозабезпечених регіонах.

Натомість автономний голосовий асистент не вимагає підключення до мережі, що є його вагомою перевагою. Така автономність дозволяє користувачам бути незалежними від зовнішніх факторів, що особливо важливо для людей, які проживають у сільській місцевості, зонах з нестабільним інтернетом або у випадках, коли доступ до мережі є обмеженим через економічні чи інші обставини. Крім того, автономна робота підвищує рівень приватності — всі голосові команди та особисті дані обробляються виключно локально на пристрої користувача, що усуває ризики збору і зберігання інформації третіми сторонами.

Ключовий аспект — підтримка української мови. Хоча сучасні голосові асистенти поступово впроваджують нові локалізації, їхні можливості для української мови часто залишаються обмеженими або недостатньо адаптованими до особливостей мовлення та культури. Розроблена система передбачає повноцінну підтримку української, що включає коректне розпізнавання команд і якісний синтез мовлення, що звучить природно і зрозуміло. Це має велике значення для комфортної взаємодії користувачів з вадами зору, яким важливо не тільки почути, а й легко сприйняти інформацію.

Простота і легкість налаштувань — ще одна вагома перевага розробки. Популярні голосові помічники часто вимагають реєстрації у відповідних сервісах, налаштування облікових записів, синхронізації з іншими пристроями та додатками. Для користувачів з обмеженими можливостями це може стати серйозним бар’єром, що ускладнює процес адаптації і знижує мотивацію до використання технологій. У автономній системі всі необхідні функції доступні одразу після встановлення без додаткових складних процедур, що робить технологію більш дружньою і зрозумілою для людей з вадами зору.

Відсутність потреби у високошвидкісному інтернеті значно розширює коло

потенційних користувачів. Люди, які мешкають у віддалених регіонах, військових зонах або у місцях з обмеженою інфраструктурою, часто позбавлені можливості користуватися сучасними онлайн-сервісами. Автономний голосовий асистент надає їм можливість отримати доступ до цифрових технологій, підвищити рівень самостійності, долучитися до навчання, роботи та соціального життя, що є важливим кроком у напрямку інклюзії.

Поглиблюючи порівняння, варто звернути увагу на питання адаптивності і гнучкості голосових асистентів. Комерційні рішення, як правило, мають широкий спектр функцій, але не завжди можуть бути адаптовані під особливі потреби конкретних користувачів, особливо тих, хто має серйозні порушення зору. Вони часто орієнтовані на масовий ринок і стандартні сценарії використання, що може не враховувати індивідуальні особливості взаємодії. Автономний асистент, навпаки, розробляється з урахуванням цих специфічних вимог, забезпечуючи більш персоналізований і точний досвід, що критично важливо для користувачів з обмеженими можливостями.

Важливий аспект — безпека і контроль над даними. У традиційних голосових асистентів, що працюють через хмарні платформи, інформація користувачів передається на віддалені сервери, що створює потенційні ризики для приватності і безпеки. Для людей з особливими потребами, які часто вразливіші до соціальних та технічних загроз, можливість автономної роботи без передачі особистих даних є значним плюсом. Це не лише захищає конфіденційність, а й підвищує довіру до технології, стимулюючи її активне використання.

Не менш важливим є питання підтримки та оновлення системи. У популярних онлайн-асистентів оновлення відбуваються автоматично і часто вимагають постійного доступу до інтернету. В автономній системі оновлення можна здійснювати локально, що дозволяє краще контролювати процес і уникати небажаних змін у роботі асистента, які можуть дезорієнтувати користувача або порушити звичний режим взаємодії. Такий підхід робить систему більш стабільною і передбачуваною для користувачів з порушеннями зору, для яких навіть незначні зміни інтерфейсу можуть стати серйозною перешкодою.

Варто зазначити економічний аспект. Вартість доступу до комерційних онлайн-сервісів часто є недоступною для багатьох категорій користувачів, особливо в регіонах з низьким рівнем доходу або у людей, які змушені обмежувати свої витрати. Автономний голосовий асистент, який працює без підключення до інтернету і не вимагає підписки на платні сервіси, стає набагато більш доступним рішенням. Це відкриває нові можливості для соціальної підтримки та інтеграції, знижуючи бар’єри, які часто виникають через економічні обмеження.

Автономність і локальна підтримка української мови створюють базу для подальшого розвитку та адаптації технології. Це дозволяє швидше реагувати на потреби користувачів, впроваджувати нові функції, адаптувати систему під різні категорії користувачів і сценарії використання, включаючи освітні, професійні та побутові задачі. Завдяки цьому голосовий асистент стає не просто технічним інструментом, а повноцінним партнером у житті людей з порушеннями зору, підтримуючи їх незалежність і якість життя.

3.7 Розуміння, розпізнавання та відтворювання команд

Для того, щоб програма могла розпізнавати голосові команди, перш за все вона повинна отримати звуковий сигнал з мікрофона. Це забезпечує спеціальний компонент, який захоплює аудіо в режимі реального часу. Він приймає звукові хвилі, перетворює їх у цифровий аудіопотік і передає далі для обробки. Такий запис у реальному часі дозволяє миттєво реагувати на голос користувача без затримок. Програма використовує буфери для прийому невеликих порцій звуку, що дозволяє розбивати голосові дані на частини і передавати їх для розпізнавання поступово, з можливістю коригувати результати під час проміжного прослуховування.

Наступним кроком є розпізнавання мови — процес, у якому аудіопотік перетворюється у текст. Для цього використовуються спеціальні алгоритми і моделі, які аналізують звукові хвилі, ідентифікують слова і фрази, а потім формують текстову команду. Важливо, що сучасні системи можуть працювати офлайн, не потребуючи доступу до інтернету, що підвищує приватність і швидкість обробки. Під час розпізнавання система може навіть передбачати, що користувач

збирається сказати, обробляючи проміжні результати, щоб швидко реагувати на команду.

Після того як команда розпізнана, програма аналізує її зміст і визначає, яку дію необхідно виконати. Якщо команда коректна і зрозуміла, вона передається модулю керування, який запускає відповідні програми, відкриває документи чи виконує інші дії. Якщо команда не впізнана, система може повідомити користувача про помилку або запропонувати повторити голосовий запит. У будь-якому випадку користувачу надається зворотній зв’язок у вигляді звуку.

Для озвучення відповідей або повідомлень використовується модуль синтезу мовлення, який перетворює текст у голосовий сигнал. Цей компонент дозволяє програмі "говорити" з користувачем природною мовою, роблячи взаємодію більш комфортною і інтуїтивною. Голос може бути настроєний за швидкістю, гучністю і тембром, що дає змогу підлаштуватися під індивідуальні уподобання. Завдяки цьому користувач отримує не просто текстовий результат, а повноцінне голосове спілкування.

Програма починає свою роботу з налаштування аудіоінтерфейсу, який забезпечує якісний запис голосу. Використання спеціалізованих бібліотек дозволяє правильно налаштувати частоту дискретизації, формат звуку і буферизацію, щоб отримувати чіткий сигнал без перешкод і затримок. Це надзвичайно важливо, адже будь-який шум або затримка можуть знизити точність розпізнавання голосу. Оптимізація процесу запису забезпечує комфортну і стабільну роботу голосового помічника навіть у складних умовах.

Далі аудіодані передаються до модуля розпізнавання мови, який працює за допомогою сучасних нейронних мереж і статистичних моделей. Ці моделі навчалися на величезних обсягах мовних даних, щоб розпізнавати не тільки слова, а й інтонації, наголоси, діалекти. Завдяки цьому програма розуміє не лише формальні команди, а й природну мову, що робить спілкування більш людяним і зручним. Розпізнавання відбувається майже миттєво, що дає змогу реалізувати інтерактивний діалог у реальному часі.

Коли текст команди отриманий, він проходить етап синтаксичного і

семантичного аналізу, щоб зрозуміти контекст і наміри користувача. Програма шукає ключові слова або фрази, які співпадають із заздалегідь визначеними командами або сценаріями дій. Якщо знайдено відповідність, запускається потрібна функція або застосунок. Якщо ж команда не розпізнана, система може запропонувати альтернативні варіанти або попросити користувача повторити фразу, тим самим підвищуючи зручність використання і знижуючи рівень помилок.

Важливу роль відіграє система зворотного зв’язку, яка інформує користувача про стан виконання команди. За допомогою голосового синтезатора програма може підтвердити успішний запуск програми, повідомити про помилку або надати додаткову інформацію. Такий діалог робить роботу з голосовим асистентом інтуїтивною і природною, адже користувач завжди знає, що відбувається і як система сприймає його команди.

Наприкінці процесу відтворення голосової відповіді, програма може вести журнал виконаних дій, що допомагає в майбутньому аналізувати використання і покращувати функціональність. Зберігаються як розпізнані команди, так і відповіді асистента, що дозволяє навчати систему, покращувати розпізнавання і автоматизувати більш складні сценарії взаємодії. Такий підхід робить голосового помічника не тільки зручним інструментом, але й системою, що постійно вдосконалюється.

Весь процес — від захоплення звуку до його розпізнавання, аналізу команди і відтворення голосової відповіді — працює як єдина система, що забезпечує швидку і ефективну взаємодію користувача з комп’ютером через голос. Це дозволяє створювати зручні голосові помічники, які полегшують роботу, особливо для людей з обмеженими можливостями.

4 РОЗРОБКА КОРИСТУВАЦЬКОГО ІНТЕРФЕЙСУ, ІНСТРУКЦІЇ КОРИСТУВАЧА ТА ТЕСТУВАННЯ ЗАСТОСУНКУ

4.1 Розробка графічного інтерфейсу

Графічний інтерфейс розроблено з використанням технології Windows Forms, що забезпечує зручність і доступність для користувачів із вадами зору. Інтерфейс має мінімалістичний та інтуїтивно зрозумілий дизайн, який сприяє легкій взаємодії з програмою.

На головному екрані представлені три функціональні кнопки:

— "Почати" – ініціює процес голосового розпізнавання та виконання команд;

— "Перезапуск" – дозволяє повторно активувати розпізнавання мовлення у випадку неточного визначення команди;

— "Вихід" – завершує роботу застосунку.

Для додаткового спрощення управління програмою передбачено вбудовану підтримку клавіатурних комбінацій. Відповідні текстові підказки розміщені під кнопками:

— "Натисніть X" – активує кнопку "Почати";

— "Натисніть C" – викликає команду перезапуску;

— "Натисніть V" – завершує роботу голосового помічника.

Графічний інтерфейс оптимізовано для використання як із мишею, так і з клавіатурою, що розширює можливості взаємодії як на рисунку 4.1. Крім того, підтримується звуковий зворотний зв’язок, який забезпечує аудіоінформування про виконані дії.

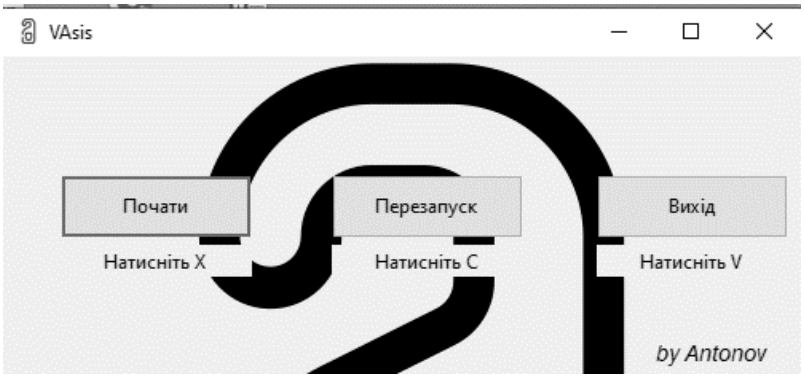

Рисунок 4.1 — Графічний інтерфейс програми

4.2 Тестування програми

Для запуску програми потрібно натиснути на додаток або протиснути комбінацію клавіш “Ctrl+Alt+G”. Після чого відкривається вікно програми як на рисунку 4.1. Для того щоб почати роботу голосового помічника потрібно натиснути клавішу “X” або кнопку “Почати” після чого програма почне працювати як на рисунку 4.2.

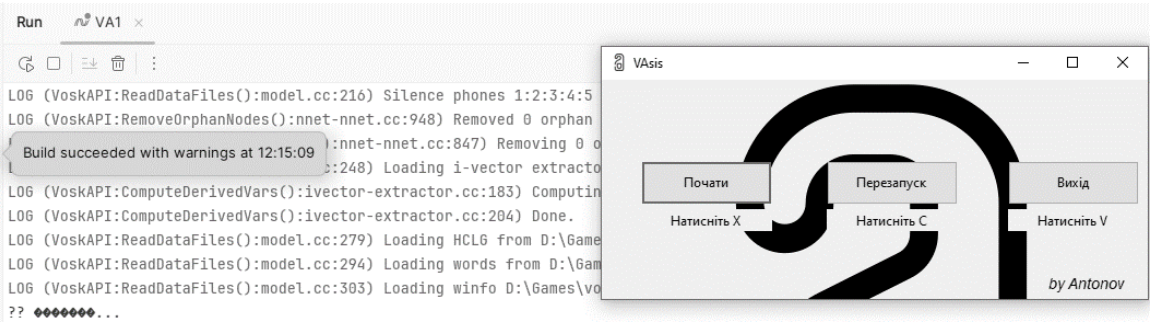

Рисунок 4.2 — Запуск програми після натискання клавіші “X” або кнопки “Почати”

Після запуску асистент говорить команду “Говоріть” і починає запис голосу, після того як час запису закінчився він дублює інформацію яку почув та починає пошук ключових слів, якщо програма знайшла знайоме слово то оголошує про відкриття додатку та завершує роботу але графічний інтерфейс не закривається. Якщо потрібно знову запустити чи сказати якусь команду то потрібно натиснути клавішу “C” або кнопку “Перезапуск” після чого програма перезапуститься і почне знову алгоритм як на рисунку 4.3.

Після виконання команди програми оголошує що відкриває додаток та на вашему екрані відкривається програма яку ви оголошували. Щоб вийти з програми достатньо натиснути клавішу “V” або кнопку “Вихід” як на рисунку 4.4.

Рисунок 4.3 — Блок-схема роботи програми

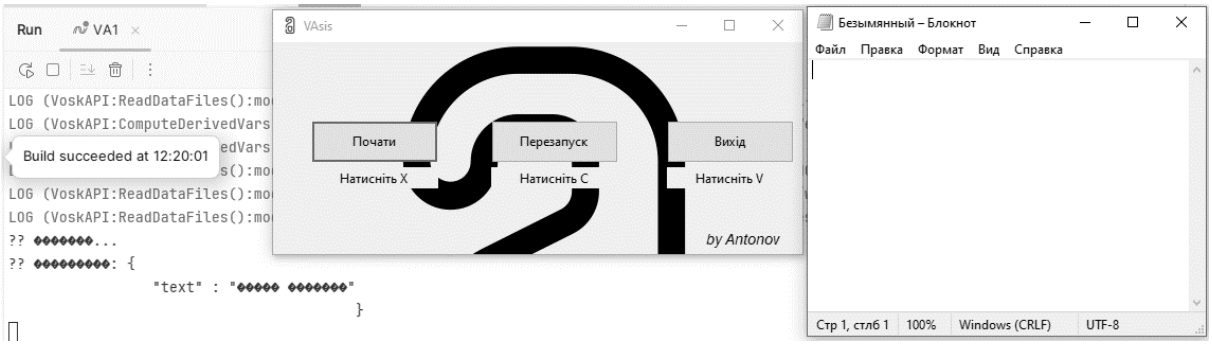

Рисунок 4.4 — Успішне виконання програми та завершення роботи

4.3 Системні вимоги

Системні вимоги для Windows:

— операційна система 64-розрядна версія Windows 10 або новіша;

— процесор 2 або більше ядер;

— оперативна пам’ять 4Гб або більше;

— місце на диску 5Гб або більше.

Системні вимоги для macOS:

— операційна система macOS Mojave (версія 10.14) або новіша;

— процесор Intel або Apple Sillicon (M1/M2);

— оперативна пам’ять 4Гб або більше;

— місце на диску 5Гб або більше.

Системні вимоги для Linux:

— операційна система 64-розрядна версія Linux з glibc 2.29 або новіша;

— процесор 2 або більше ядер;

— оперативна пам’ять 4Гб або більше;

— місце на диску 5Гб або більше.

Робота програми розпочинається з відкриття файлу «VA1.exe».

Після запуску програми асистент вам подасть команду “Говоріть” після чого програма почне записувати ваш голос протягом 10 секунд після цього починається аналіз і пошук ключових слів, якщо програма не знайшла жодного ключового слова тоді ви почуєте “Не вдалось і виконати команду” і голосовий асистент повторить те що він встиг розпізнати. Якщо все пройшло добре то програма повторить слова які вона розпізнала і повідомить що виконує команду, після чого відкриється додаток і програма буде очікувати наступних команд.

ВИСНОВКИ

У межах бакалаврської кваліфікаційної роботи було розроблено спеціалізований комп’ютерний комплекс для полегшення роботи осіб з вадами зору, що базується на сучасних технологіях голосового управління. В ході дослідження було проведено аналіз методів розпізнавання мовлення та синтезу голосу, що дозволило створити ефективну систему інтерактивної взаємодії користувача з комп’ютером без необхідності використання традиційних засобів вводу.

Розроблена система має архітектуру, яка дозволяє здійснювати запуск та керування програмами за допомогою голосових команд, використовуючи локальні мовні моделі та автономні технології, що не потребують доступу до мережі інтернет. Це забезпечує конфіденційність даних користувача та стабільність роботи асистента в будь-якому середовищі.

Основним елементом розпізнавання мовлення стала бібліотека Vosk, що дозволяє обробляти мовні команди в режимі реального часу. Для захоплення аудіопотоку використано технологію NAudio, а механізм синтезу мовлення реалізовано через System.Speech, що надає користувачеві зворотний зв’язок і підвищує ефективність взаємодії з програмою.

Під час розробки було враховано потреби користувачів із порушеннями зору, що дозволило створити простий та інтуїтивний інтерфейс, доповнений голосовими повідомленнями та підтримкою клавіатурних команд. Проведені тестування підтвердили працездатність системи та її здатність коректно виконувати голосові команди, запускати програми, керувати файлами та здійснювати інші операції.

Створений комп’ютерний комплекс є ефективним рішенням, яке підвищує рівень доступності цифрових технологій для людей із вадами зору. Застосування сучасних методів розпізнавання мовлення та синтезу голосу формує основу для подальшого розвитку таких систем, розширення їхнього функціоналу та інтеграції з мобільними пристроями.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 123 «Комп'ютерна інженерія» (освітня програма «Комп'ютерна інженерія») [Електронний ресурс] / уклад. О. Д. Азаров, С. В. Богомолов, С. І. Швець. – Вінниця : ВНТУ, 2023. – 105 с.

2. Документаці C#. — Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/csharp/tour-of-csharp/

3. Про середовище програмування C#. — Режим доступу: https://foxminded.ua/seredovyshche-prohramuvannia-si-sharp/

4. Бібліотека розпізнавання Vosk. — Режим доступу: https://wirenboard.com/wiki/index.php?title=VOSK&mobileaction=toggle_view_desktop

5. Vosk Models. — Режим доступу: Модели VOSK

6. NAudio.Wave. — Режим доступу: https://github.com/naudio/NAudio

7. System.Speech.Synthesis. — Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/api/system.speech.synthesis?view=net-9.0-pp

8. System.Diagnostics. — Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/api/system.diagnostics?view=net-9.0

9. Process клас. — Режим доступу: https://learn.microsoft.com/ru-ru/dotnet/api/system.diagnostics.process?view=net-8.0

10. Голосові помічники: що це і навіщо вони. — Режим доступу: https://hurma.work/blog/voice-assistants-shho-cze-i-navishho-voni-hr-ok-google-alexa-siri-cortana/

11. Досин Д.Г. Інтелектуальні системи, базовані на онтологіях // Д.Г. Досин, В.В. Литвин, Ю.В. Нікольський, В.В. Пасічник. – Львів: Цивілізація, 2009. – 414 с

12. Огляд базових принципів роботи незрячих за комп'ютером. — Режим доступу: https://inc.kiev.ua/index.php/16-statti/47-oglyad-bazovikh-printsipiv-roboti-nezryachikh-za-komp-yuterom

13. Розумний дім та голосові помічники. — Режим доступу:

https://www.iqdim.ua/post/rozumnyy-dim-ta-holosovi-pomichnyky

14. 7 ключових прогнозів щодо майбутнього голосових помічників. — Режим доступу: https://www.unite.ai/uk/7-key-predictions-for-the-future-of-voice-assistants-and-ai/

15. Використання голосових помічників для керування системою «розумного дому»: переваги та обмеження. — Режим доступу: https://derevko.com.ua/vykorystannya-golosovyh-pomichnykiv-dlya-keruvannya-systemoyu-rozumnogo-domu-perevagy-ta-obmezhennya/

16. Розвиток голосових помічників на основі ШІ. — Режим доступу: https://uk.shaip.com/blog/role-of-voice-assistant-in-enhancing-quality-of-healthcare/

17. Що таке .NET?. — Режим доступу: https://campus.epam.ua/ua/blog/301

18. Як вивчити мову програмування C# та стати .NET розробником. — Режим доступу: https://edu.cbsystematics.com/ua/blog/learn-csharp-become-dnet

19. Найпопулярніший у світі голосовий асистент. — Режим доступу: http://ti.ua/ua/news/samyy_populyarnyy_v_mire_golosovoy_assistent/?srsltid=AfmBOoo0yWfb3iK3mLNRsuViy6xo4piW4us9n4mh7w4IjY2SJoDoE0lv

20. JetBrains Rider. — Режим доступу: https://www.jetbrains.com/ru-ru/lp/rider-web/

21. Спеціалізований комп’ютерний комплекс для полегшення роботи особам з вадами зору / Е.П.Антонов, М.А. Томчук // Матеріали конференції Молодь в науці: дослідження, проблеми, перспективи. — Режим доступу: https://conferences.vntu.edu.ua/index.php/mn/mn2025/author/submission/25496

ДОДАТОК А
Технічне завдання

Міністерство освіти і науки України
Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра обчислювальної техніки

ЗАТВЕРДЖУЮ
Завідувач кафедри ОТ
проф., д.т.н.. Азаров О.Д.
21 березня 2025 р.

ТЕХНІЧНЕ ЗАВДАННЯ
на виконання бакалаврської кваліфікаційної роботи
«Спеціалізований комп'ютерний комплекс для полегшення
роботи людям з вадами зору»
08-54.БКР.021.00.000 ТЗ

Науковий керівник: к.т.н., проф. каф. ОТ
_____Томчук М.А.

Студент групи 2КІ-21б
_____Антонов Е.П.

м. Вінниця — 2025

1 Підстава для використання бакалаврської кваліфікаційної роботи (БКР)

1.1 У сучасному світі, де технології доступності відіграють ключову роль у створенні інклюзивного цифрового середовища, особливого значення набувають рішення, що полегшують взаємодію користувачів із вадами зору з комп'ютерними системами. Одним із таких рішень є розробка голосового помічника, який дозволяє керувати програмами та виконувати команди голосом, забезпечуючи зручний і безпечний спосіб навігації без необхідності використання клавіатури та миші.

1.2 Наказ про затвердження теми бакалаврської кваліфікаційної роботи від 21 березня 2025 р. №97.

2 Мета і призначення БКР

2.1 Метою роботи є створення голосового помічника для користувачів із вадами зору, який забезпечує розпізнавання голосових команд, синтез мовлення та керування операційною системою. Система реалізована на основі локальних мовних моделей, що не потребують доступу до інтернету, що гарантує автономність та конфіденційність використання.

2.2 Призначення розробки полягає у створенні програмного забезпечення, яке забезпечує розпізнавання мовлення через бібліотеку "Vosk", синтез голосу через "System.Speech", а також інтеграцію з Windows Forms для графічного відображення роботи асистента. Система дозволяє користувачам запускати програми, керувати файлами та отримувати аудіозворотний зв'язок без необхідності візуального контролю.

3 Вихідні дані для виконання БКР

3.1 Проведення аналізу сучасних технологій голосового управління, методів розпізнавання мовлення та синтезу голосу.

3.2 Розробка структури, логіки взаємодії компонентів та макетів інтерфейсу Windows Forms.

3.3 Реалізація програмної частини голосового помічника у середовищі "Visual Studio", інтеграція розпізнавання мовлення та синтезу голосу.

3.4 Оптимізація алгоритмів обробки аудіосигналів з використанням “NAudio” для покращення роботи з аудіопотоком.

3.5 Тестування точності розпізнавання мовлення, швидкості виконання команд та загальної продуктивності системи.

3.6 Впровадження голосового управління для запуску програм, доступу до системних функцій та створення зворотного аудіозв’язку з користувачем.

4 Вимоги до виконання БКР

Головна вимога – створити адаптивний, функціональний та стабільний голосовий помічник, який забезпечує ефективне розпізнавання мовлення, синтез голосу та управління комп’ютерними програмами без необхідності використання миші чи клавіатури.

Для досягнення цієї мети система має відповідати наступним критеріям:

— використання нейронної моделі Vosk для якісної обробки мовлення;

— коректне виконання команд, мінімізація затримок під час обробки мовлення та озвучення відповідей;

— використання локальних мовних моделей без потреби підключення до інтернету;

— підтримка голосових підказок та інтеграція з Windows Forms для додаткового візуального зворотного зв’язку;

— можливість додавання нових функцій, адаптації до інших мов та розширення списку підтримуваних команд.

5 Етапи БКР та очікуванні результати

Етапи розробки БКР та очікуванні результати наведено у Таблиці А.1.

6 Матеріали що подаються до захисту БКР

До захисту подаються: пояснювальна записка БКР, ілюстративні матеріали (блок-схема алгоритму системи, ілюстрації функціонування програми), протокол попереднього захисту БКР на кафедрі, відгук наукового керівника, анотації до БКР

українською та іноземною мовами, довідка про відповідність оформлення БКР чинним вимогам.

Таблиця А.1 — Етапи БКР

№ етапу	Назва етапу	Термін виконання		Очікувані результати
		Початок	Кінець	
1	Постановка задачі	21.03.25	21.03.25	Розділ 1
2	Аналіз існуючих прототипів та визначення вимог до створюваної системи	21.03.25	30.03.25	Розділ 1
3	Аналіз та обґрунтування вибору технологій	21.03.25	30.04.25	Розділ 1
4	Аналіз та обґрунтування вибору технічної частини	21.03.25	30.04.25	Розділ 2
5	Розробка програмного коду	31.03.25	13.04.25	Розділ 3
6	Тестування програми на першому етапі розробки	14.04.25	27.04.25	Розділ 4
7	Підготовка тезових матеріалів для публікації	21.04.25	28.04.25	Тези
8	Оформлення пояснювальної записки та демонстраційної презентації для доповіді	29.04.25	12.05.25	ПЗ, презентація
9	Підготовка супровідної документації, перевірка відповідності нормам та проходження тесту на плагіат	13.05.25	13.05.25	ПЗ, презентація

7 Порядок контролю виконання та захисту БКР

Контроль виконання етапів здійснюється науковим керівником згідно з встановленими термінами. Захист БКР проходить на засіданні екзаменаційної комісії, затвердженої наказом ректора.

8 Вимоги до оформлення та порядок виконання БКР

8.1 При оформленні БКР використовуються:

— ДСТУ 3008 : 2015 «Звіти в сфері науки і техніки. Структура та правила оформлювання»;

— ДСТУ 8302 : 2015 «Бібліографічні посилання. Загальні положення та правила складання»;

— методичні вказівки до виконання бакалаврських кваліфікаційних робіт для студентів спеціальності 123 «Комп'ютерна інженерія» (освітня програма «Комп'ютерна інженерія») — кафедра обчислювальної техніки, ВНТУ 2023.

8.2 Порядок виконання БКР викладено в «Положення про кваліфікаційні роботи на першому (бакалаврському) рівні вищої освіти СУЯ ВНТУ-03.02.02-П.001.01:21»

ДОДАТОК Б
ПРОТОКОЛ ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Назва роботи: Спеціалізований комп'ютерний комплекс для полегшення роботи людям з вадами зору

Тип роботи: _____ бакалаврська кваліфікаційна робота _____

(бакалаврська кваліфікаційна робота / магістерська кваліфікаційна робота)

Підрозділ _____кафедра обчислювальної техніки_____

(кафедра, факультет, навчальна група)

Коефіцієнт подібності текстових запозичень, виявлених у роботі системою StrikePlagiarism ___0,5____%

Висновок щодо перевірки кваліфікаційної роботи (відмітити потрібне)

✓ Запозичення, виявлені у роботі, є законними і не містять ознак плагіату, фабрикації, фальсифікації. Роботу прийняти до захисту

☐ У роботі не виявлено ознак плагіату, фабрикації, фальсифікації, але надмірна кількість текстових запозичень та/або наявність типових розрахунків не дозволяють прийняти рішення про оригінальність та самостійність її виконання. Роботу направити на доопрацювання.

☐ У роботі виявлено ознаки плагіату та/або текстових маніпуляцій як спроб укриття плагіату, фабрикації, фальсифікації, що суперечить вимогам законодавства та нормам академічної доброчесності. Робота до захисту не приймається.

Експертна комісія:

_____Азаров О.Д., зав. каф. ОТ _____ _____

(прізвище, ініціали, посада) (підпис)

____Крупельницький Л.В., доц. каф ОТ_____ _____

(прізвище, ініціали, посада) (підпис)

Особа, відповідальна за перевірку _____ ___Захарченко С.М.____

(підпис) (прізвище, ініціали)

З висновком експертної комісії ознайомлений(-на)

Керівник _____ _____Томчук М.А._____

(підпис) (прізвище, ініціали, посада)

Здобувач _____ _____Антонов Е.П._____

(підпис) (прізвище, ініціали)

ДОДАТОК В
Ілюстративна частина

СПЕЦІАЛІЗОВАНИЙ КОМП’ЮТЕРНИЙ КОМПЛЕКС ДЛЯ ПОЛЕГШЕННЯ РОБОТИ ЛЮДЯМ З ВАДАМИ ЗОРУ

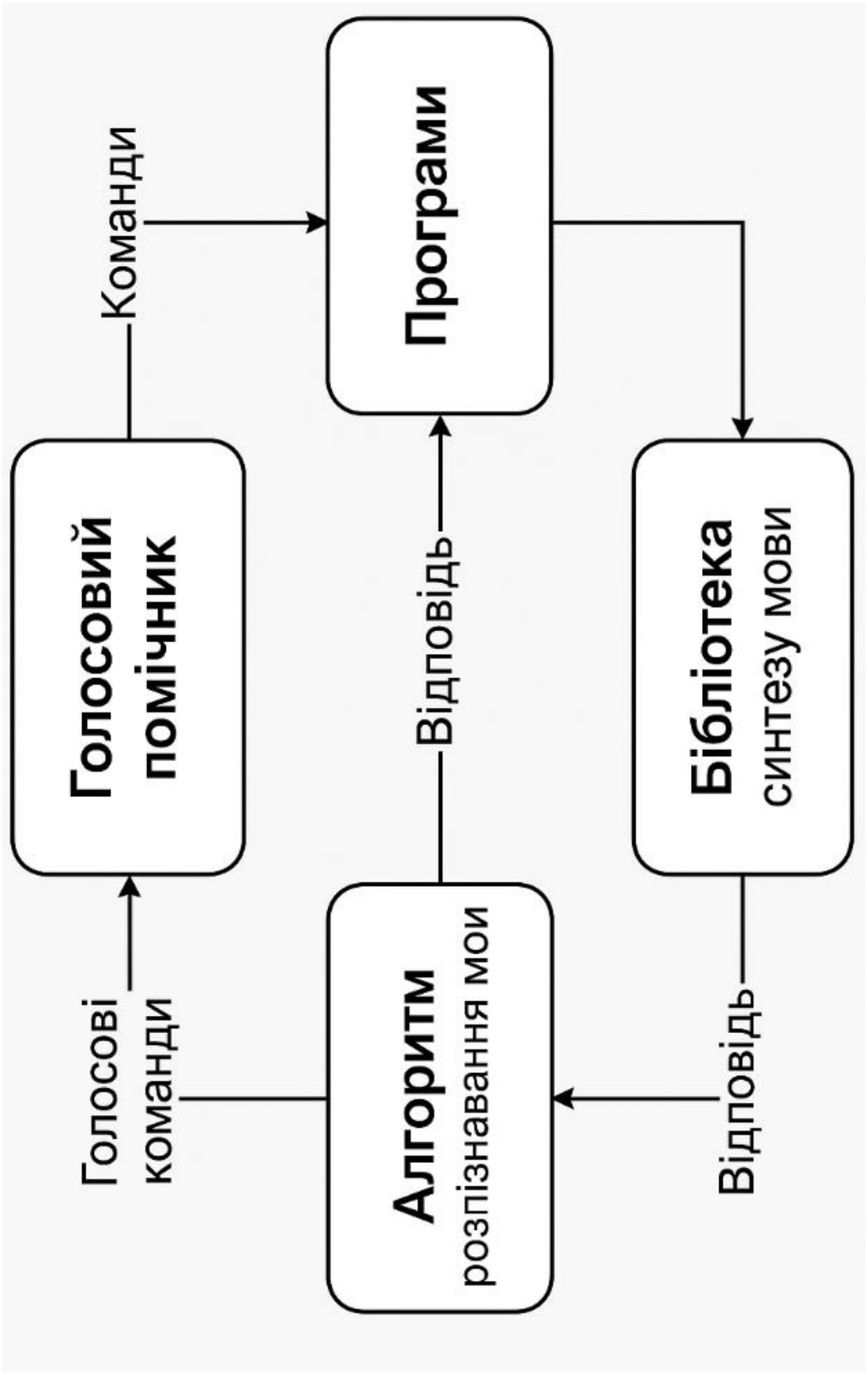

Рисунок В.1 — Стуктурна схема взаємодії компонентів голосового помічника

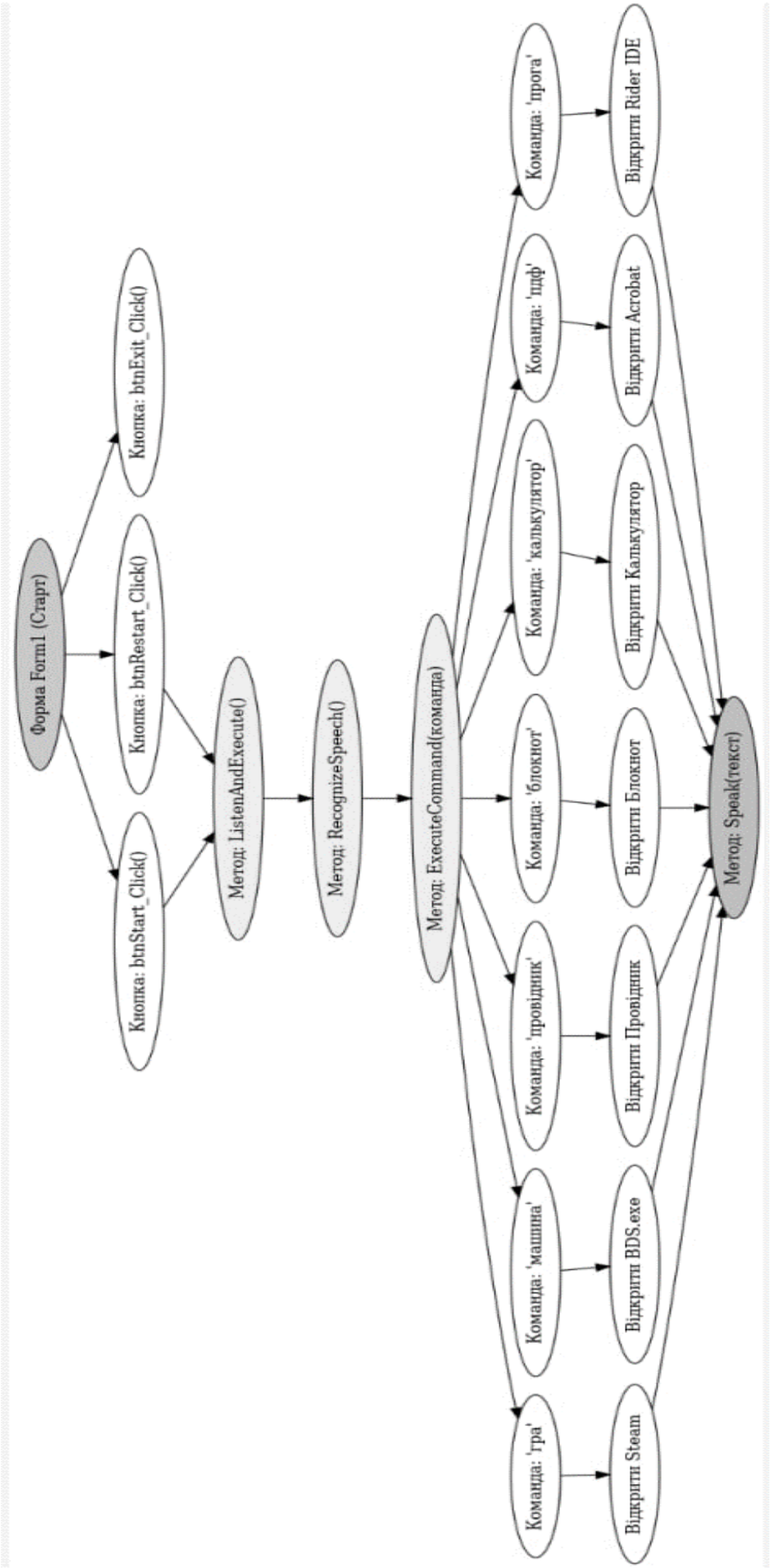

Рисунок В.2 — Блок-схема обробки голосових команд

ДОДАТОК Г

Лістинг програмного коду

Лістинг Г.1 – основний файл вихідного коду Form1.cs

```
using System;
using System.Diagnostics;
using System.IO;
using System.Windows.Forms;
using Vosk;
using NAudio.Wave;
using System.Speech.Synthesis;

namespace VA1
{
   public partial class Form1 : Form
   {
      SpeechSynthesizer synth = new SpeechSynthesizer();

      public Form1()
      {
         InitializeComponent();
         this.KeyPreview = true;
         this.KeyDown += new KeyEventHandler(Form1_KeyDown);
      }

      private void btnStart_Click(object sender, EventArgs e)
      {
         ListenAndExecute();
      }

      private void btnRestart_Click(object sender, EventArgs e)
      {
         ListenAndExecute();
      }

      private void btnExit_Click(object sender, EventArgs e)
      {
         Application.Exit();
      }

      private void Form1_KeyDown(object sender, KeyEventArgs e)
      {
         if (e.KeyCode == Keys.X)
```

```
        {
            btnStart.PerformClick();
        }
        else if (e.KeyCode == Keys.C)
        {
            btnRestart.PerformClick();
        }
        else if (e.KeyCode == Keys.V)
        {
            btnExit.PerformClick();
        }
    }

    private void ListenAndExecute()
    {
        Console.WriteLine("🎤 Очікуємо команду...");

        string userCommand = RecognizeSpeech()?.ToLower();
        if (!string.IsNullOrEmpty(userCommand))
        {
            Speak($"Розпізнано команду: {userCommand}");
            ExecuteCommand(userCommand);
        }
    }

    private string RecognizeSpeech()
    {
        string modelPath = @"D:\Games\vosk-model-uk-v3";

        if (!Directory.Exists(modelPath))
        {
            Speak($"❌ Помилка: Модель не знайдена у {modelPath}");
            return null;
        }

        Model model = new Model(modelPath);
        using (var waveIn = new WaveInEvent())
        {
            waveIn.WaveFormat = new WaveFormat(16000, 1);
            var rec = new VoskRecognizer(model, 16000f);

            waveIn.DataAvailable += (_, a) => { rec.AcceptWaveform(a.Buffer,
a.BytesRecorded); };
            waveIn.StartRecording();
```

```
            Console.WriteLine("🎤 Говоріть...");
            Speak("Говоріть...");
            System.Threading.Thread.Sleep(5000);
            waveIn.StopRecording();

            string result = rec.Result();
            Console.WriteLine($"🔹 Розпізнано: {result}");
            Speak($"Я розпізнав: {result}");
            return result;
        }
    }

    private void ExecuteCommand(string command)
    {
        if (command.Contains("гра"))
        {
            Speak("Запускаю Steam");
            Process.Start("D:\\ste\\steam.exe");
        }
        else if (command.Contains("машина"))
        {
            Speak("Запускаю BDS");
            Process.Start("D:\\ste\\steamapps\\common\\Backseat Drivers
Demo\\BDS.exe");
        }
        else if (command.Contains("провідник"))
        {
            Speak("Запускаю Провідник");
            Process.Start("explorer.exe");
        }
        else if (command.Contains("блокнот"))
        {
            Speak("Запускаю Блокнот");
            Process.Start("notepad.exe");
        }
        else if (command.Contains("калькулятор"))
        {
            Speak("Запускаю Калькулятор");
            Process.Start("win32calc.exe");
        }
        else if (command.Contains("пдф"))
        {
            Speak("Запускаю PDF");
```

```
            Process.Start("C:\\Program Files\\Adobe\\Acrobat
DC\\Acrobat\\Acrobat.exe");
        }
        else if (command.Contains("прога"))
        {
            Speak("Запускаю JetBrains Rider");
            Process.Start("D:\\JetBrains Rider 2025.1.1\\bin\\rider64.exe");
        }
        else
        {
            Console.WriteLine("⚠ Команду не розпізнано.");
            Speak("Не вийшло виконати команду");
        }
    }

    private void Speak(string text)
    {
        synth.SelectVoice("Microsoft Irina Desktop");
        synth.Speak(text);
    }
  }
}
```

Лістинг Г.2 – файл дизайнерського коду Form1.Designer.cs

```
namespace VA1
{
  partial class Form1
  {
    private System.Windows.Forms.Button btnStart;
    private System.Windows.Forms.Button btnRestart;
    private System.Windows.Forms.Button btnExit;
    private System.Windows.Forms.Label lblSignature;
    private System.Windows.Forms.Label lblStartKey;
    private System.Windows.Forms.Label lblRestartKey;
    private System.Windows.Forms.Label lblExitKey;

    /// <summary>
    /// Required method for Designer support - do not modify
    /// the contents of this method with the code editor.
    /// </summary>
    private void InitializeComponent()
    {
      System.ComponentModel.ComponentResourceManager resources = new
System.ComponentModel.ComponentResourceManager(typeof(Form1));
```

```
        btnStart = new System.Windows.Forms.Button();
        btnRestart = new System.Windows.Forms.Button();
        btnExit = new System.Windows.Forms.Button();
        lblSignature = new System.Windows.Forms.Label();
        lblStartKey = new System.Windows.Forms.Label();
        lblRestartKey = new System.Windows.Forms.Label();
        lblExitKey = new System.Windows.Forms.Label();
        SuspendLayout();
        //
        // btnStart
        //
        btnStart.FlatAppearance.BorderColor = System.Drawing.Color.White;
        btnStart.Location = new System.Drawing.Point(36, 74);
        btnStart.Name = "btnStart";
        btnStart.Size = new System.Drawing.Size(120, 40);
        btnStart.TabIndex = 0;
        btnStart.Text = "Почати";
        btnStart.UseVisualStyleBackColor = true;
        btnStart.Click += btnStart_Click;
        //
        // btnRestart
        //
        btnRestart.Location = new System.Drawing.Point(206, 74);
        btnRestart.Name = "btnRestart";
        btnRestart.Size = new System.Drawing.Size(120, 40);
        btnRestart.TabIndex = 1;
        btnRestart.Text = "Перезапуск";
        btnRestart.UseVisualStyleBackColor = true;
        btnRestart.Click += btnRestart_Click;
        //
        // btnExit
        //
        btnExit.Location = new System.Drawing.Point(372, 74);
        btnExit.Name = "btnExit";
        btnExit.Size = new System.Drawing.Size(120, 40);
        btnExit.TabIndex = 2;
        btnExit.Text = "Вихід";
        btnExit.UseVisualStyleBackColor = true;
        btnExit.Click += btnExit_Click;
        //
        // lblSignature
        //
        lblSignature.Font = new System.Drawing.Font("Arial", 10F,
System.Drawing.FontStyle.Italic);
        lblSignature.ForeColor = System.Drawing.Color.Black;
```

```
            lblSignature.Location = new System.Drawing.Point(406, 178);
            lblSignature.Name = "lblSignature";
            lblSignature.Size = new System.Drawing.Size(86, 20);
            lblSignature.TabIndex = 4;
            lblSignature.Text = "by Antonov";
            //
            // lblStartKey
            //
            lblStartKey.Location = new System.Drawing.Point(36, 118);
            lblStartKey.Name = "lblStartKey";
            lblStartKey.Size = new System.Drawing.Size(120, 20);
            lblStartKey.TabIndex = 1;
            lblStartKey.Text = "Натисніть X";
            lblStartKey.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // lblRestartKey
            //
            lblRestartKey.Location = new System.Drawing.Point(206, 118);
            lblRestartKey.Name = "lblRestartKey";
            lblRestartKey.Size = new System.Drawing.Size(120, 20);
            lblRestartKey.TabIndex = 2;
            lblRestartKey.Text = "Натисніть C";
            lblRestartKey.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // lblExitKey
            //
            lblExitKey.Location = new System.Drawing.Point(372, 118);
            lblExitKey.Name = "lblExitKey";
            lblExitKey.Size = new System.Drawing.Size(120, 20);
            lblExitKey.TabIndex = 3;
            lblExitKey.Text = "Натисніть V";
            lblExitKey.TextAlign = System.Drawing.ContentAlignment.MiddleCenter;
            //
            // Form1
            //
            BackgroundImage = ((System.Drawing.Image)
resources.GetObject("$this.BackgroundImage"));
            ClientSize = new System.Drawing.Size(500, 200);
            Controls.Add(btnStart);
            Controls.Add(lblStartKey);
            Controls.Add(btnRestart);
            Controls.Add(lblRestartKey);
            Controls.Add(btnExit);
            Controls.Add(lblExitKey);
            Controls.Add(lblSignature);
```

```
            Icon = ((System.Drawing.Icon) resources.GetObject("$this.Icon"));
            Text = "VAsis";
            ResumeLayout(false);
        }
    }
}
```

Лістинг Г.3 – файл точки входу в застосунок Program.cs

```
using System;
using System.Windows.Forms;

namespace VA1
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Form1());
        }
    }
}
```