

Designing an Application for Monitoring the Ukrainian Spoken Language

Tetiana Shestakevych, Leslav Kobylukh

Lviv Polytechnic National University, Stepana Bandery Street, 12, Lviv, 79000, Ukraine

Abstract

The study focuses on developing a tool aimed at assisting users in analyzing their native language speech and enhancing their communication abilities. The integration of a language analysis and control system can have benefits, such as minimizing the usage of inappropriate words and expressions, enhancing awareness of speech conventions and communication etiquette, upholding ethical standards across different contexts, and fostering a wholesome and constructive speech environment. The system was designed utilizing the UML notation technique. A conceptual representation of the system encompasses detailing input and output data, system functions and structure, along with its requisite specifications. Software methods and tools were analyzed and chosen to fulfill the system function requirements.

Keywords

Speech-to-Text, Natural Language Processing, Ukrainian language

1. Introduction

The Ukrainian language is not only a national heritage, but also a symbol of our identity and cultural wealth. However, over time it is affected by external factors, technological development and cultural changes, which leads to the need for its analysis and control. Over the last decade, important changes have taken place in Ukraine in the political, economic and cultural life of the country, which also affected the state of the Ukrainian language.


The use of the Internet and social networks has greatly affected the state of the Ukrainian language. The Internet and social networks make it possible to use the Ukrainian language for communication and work. However, unfortunately, language problems remain relevant and need to be further addressed. The number of people who speak Ukrainian is about 42 million people. This is the population of Ukraine, which has approximately 37 million inhabitants, as well as the Ukrainian diaspora around the world. The Ukrainian language has several varieties, which differ in their distance from the literary norm and peculiarities of use.


The most common of them:

- Literary Ukrainian language, which is the basis of the state language of Ukraine and is used in literature, science, mass media and official documents;
- Ukrainian folk language, which has a more colloquial and everyday character and is used in different regions of Ukraine;
- Idioms, which are local variants of the Ukrainian language and have their own peculiarities of pronunciation, vocabulary and grammar;
- Surzhyk, which is a mixture of Ukrainian and Russian languages and is used in some regions of Ukraine.

In addition, the Ukrainian language also has dialectal and regional variations that may differ from the language of the national language. Correct use of vocabulary is also an important factor. The use of words that do not belong to the Ukrainian language, or the use of Ukrainian words in the wrong context can lead to a change in the content and a violation of the speaking style.

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

 Tetiana.v.shestakevych@lpnu.ua (T. Shestakevych); leslav.b.kobylukh@lpnu.ua (L. Kobylukh)

 0000-0002-4898-6927 (T. Shestakevych); 0009-0006-5653-4943 (L. Kobylukh)



© 2024 Copyright for this paper by its authors.

Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

Another important aspect is the professional level of using the Ukrainian language. In the fields of science, education, art and business, the use of correct vocabulary and grammatical rules is necessary to achieve professional success and gain the respect of colleagues and clients.

2. Statement and justification of the problem

Considering the above, an urgent task is the development of information technology for analysis and control of speech, which would make it possible to detect and signal the use of unwanted words in the speech of users - to improve the quality of communication and create a healthy speech environment.

Place of application of the system: The system can be applied in various environments such as educational institutions, workplaces, social media and internet platforms. It can also be integrated with various devices such as smartphones, computers, tablets and other communication tools [1].

Rationale for the development and implementation of the system: The development and implementation of the system is based on the need to improve the quality of communication and reduce the negative impact of unwanted words in life. The system will promote the development of a positive speech culture and increase awareness of language norms and etiquette.

Expected effects of the implementation of the system: The implementation of the language analysis and control system can have the following positive effects:

1. Reducing the use of unwanted words and expressions in the speech of users.
2. Raising awareness of speech norms and culture of communication.
3. Ensuring compliance with ethical standards in various environments.
4. Promoting a healthy and positive speech environment.

Let's consider the components of information technology, the purpose of which is to improve the quality of communication and create a healthy speech environment.

3. Conceptual model of a speech monitoring application

On development of a conceptual model of the system for monitoring the Ukrainian spoken language, such features should be taken into account.

Description of input data: Input data consists of user speech signals that need to be analyzed and controlled. These may include recordings of conversations, text messages, audio and video clips from communication platforms and social media.

Description of output data: Output data includes speech analysis results, such as detected unwanted words, usage statistics, and audio alerts that inform users when such words are detected.

Description of system functions and structure: The system consists of a number of components, such as a sound identifier, a comparison engine, a database of unwanted words and a sound generator. Thanks to these components, the system can analyze speech signals, compare them with unwanted words in the database and generate appropriate sound signals when such words are detected.

Description of system requirements: Requirements for a speech analysis and monitoring system include speed and accuracy of speech signal analysis, real-time capability, compatibility with various devices and platforms, compliance with speech regulations and legislation, and ensuring data privacy and security.

Additional formal or generalized system models: Additional models such as deployment diagrams, component diagrams, sequence diagrams, and other models can be developed to provide a detailed representation of the operation of the language analysis and control system to help visualize and analyze various aspects of the system.

System scalability and adaptability: A language analysis and control system should be scalable so that it can easily adapt to changes in the number of users, data, or platforms. This means that

the system must be able to work efficiently with different load levels, ensuring stability and performance. In addition, the system must be adaptable to take into account different speech cultures, dialects and jargons, ensuring the accuracy and relevance of the analysis.

Interaction with users: The language analysis and control system should have a convenient and intuitive interface for users, which will allow them to easily interact with the system and receive useful information. This may include visual elements such as graphs and charts, as well as convenient options for customizing the system according to user needs.

System support and updates: To ensure the stable operation and relevance of the language analysis and control system, regular technical support and updates are required. This may include updating the database, improving analysis and voice identification algorithms, fixing bugs, and ensuring compatibility with new devices and platforms.

Evaluation of results and implementation of changes: In order for the language analysis and control system to remain effective and relevant, it is necessary to regularly evaluate its results and impact on users. This may include analyzing feedback from users, conducting performance and accuracy tests of the system, and studying new trends in speech culture and legislation. Based on these assessments, updates to the system can be designed and implemented to better meet user needs and meet current standards.

Training and education of users: Since the speech analysis and control system works with speech and communication, it is also important to provide training and education of users about the relevant speech norms and rules. This may include developing training materials, organizing training sessions or lectures, and providing support to users in using the system and understanding its results.

Developing a conceptual model using UML methods usually begins with defining key concepts and relationships between them. This is done by analyzing the system and identifying the main elements included in it. Now you need to select the appropriate UML diagrams to visualize these elements. For example, use case and activity diagrams can be used for the analysis stage, class, sequence, cooperation, and state diagrams for the design stage, and component and deployment diagrams for the development stage. The final step is to document the conceptual model and use it for further system or process design [2], [3].

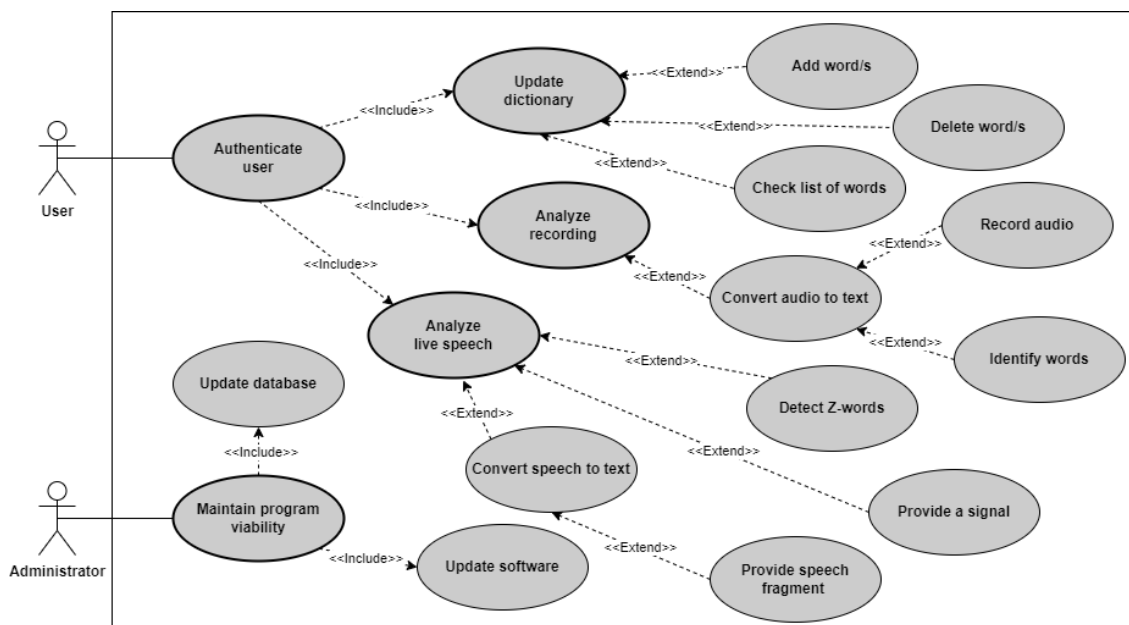


Figure 1: Use Case diagram for the modelled system.

The Use Case Diagram describes the interaction between the system and its users or external systems within specific interaction scenarios (Fig. 1).

Actors:

1. The user is a person who will use this system for speech control.
2. An administrator is a user who supports the viability of the program.

The main options for using the system are:

1. Authenticate the user - check and confirm the user's identity by entering a login and password (by voice).
2. Update the dictionary - make changes to the Z-word dictionary.
3. Analyze the record - download the record, carry out statistical processing and submit the results.
4. Analyze live speech - perform constant monitoring of speech and signal the appearance of Z-words.
5. Maintain the viability of the program - update the software.

The activity diagram of this system is presented in Fig. 2, sequence diagram - in Fig. 3, class diagram - in Fig. 4.

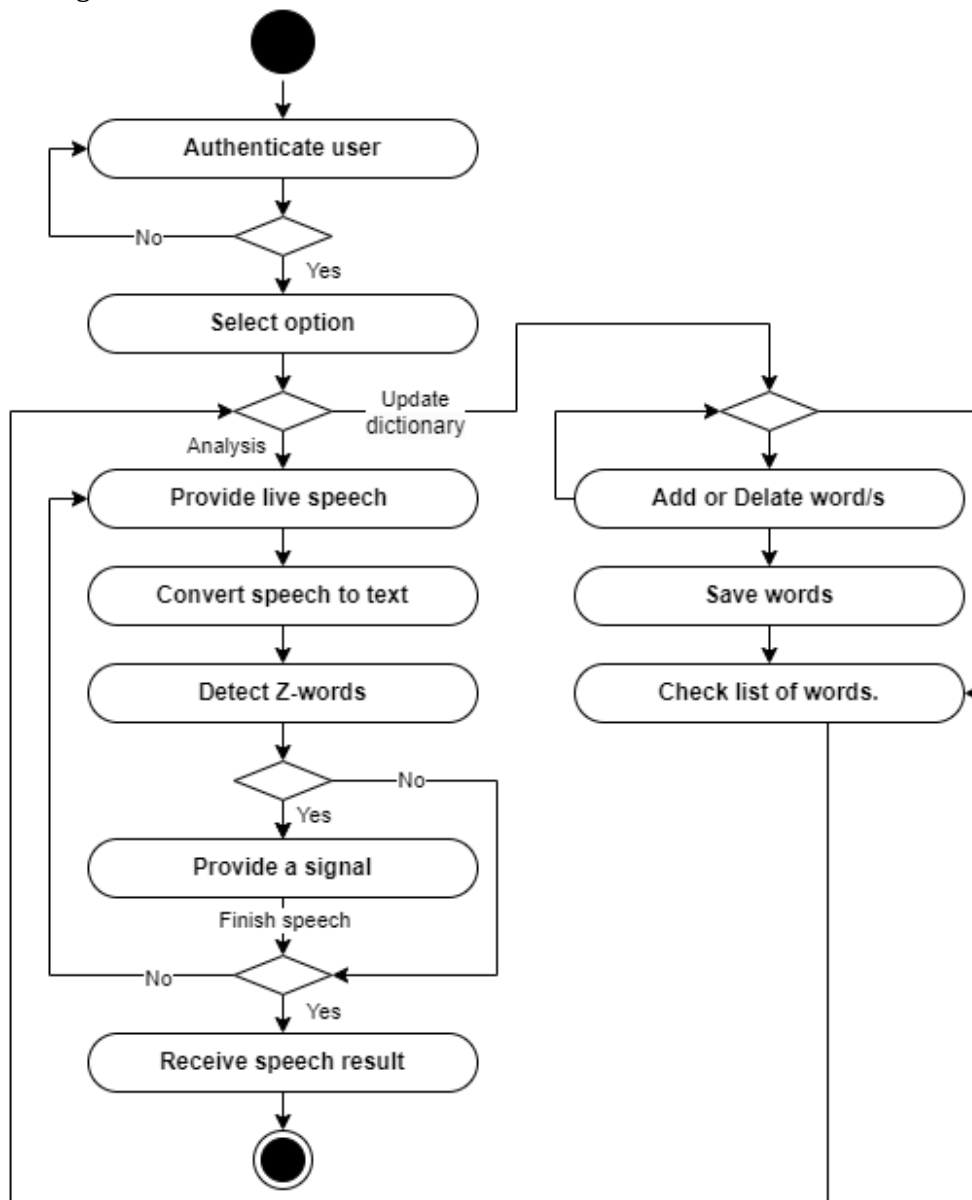


Figure 2: Activity diagram for the modelled system.

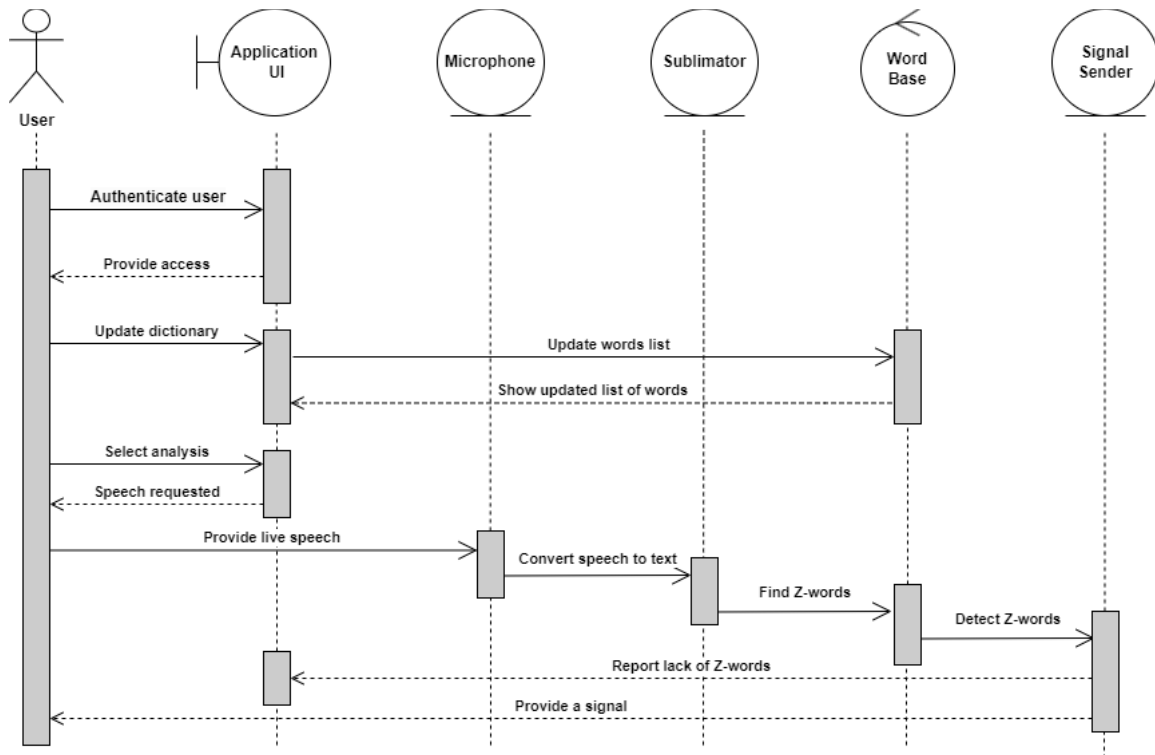


Figure 3: Sequence diagram for the modeled system.

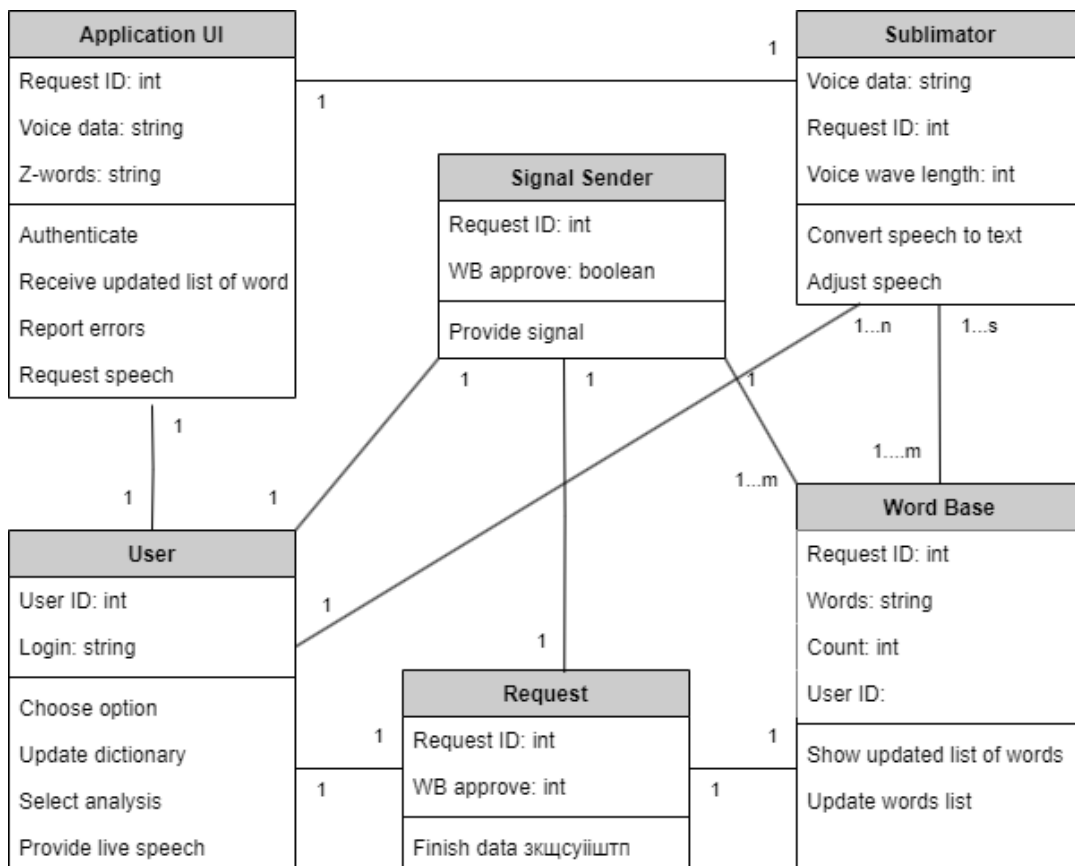


Figure 4: Diagram of the classes of the modeled system.

4. Information technologies for recognizing the Ukrainian language

Google Cloud Speech-to-Text API: The Google Speech-to-Text API supports more than 125 languages, including Ukrainian. This technology uses various neural network and deep learning algorithms for highly accurate transcription of spoken language into text. This API can handle various audio formats and work with live streaming or pre-recorded audio files [4].

Mozilla DeepSpeech: This is an open-source speech-to-text engine developed by Mozilla. It also uses a deep learning approach. Although it is focused more on the English language, the community has been very happy to support its expansion, including the addition of a model in Ukrainian [5].

Wit.ai: Acquired by Facebook in 2015, it is a platform that provides APIs for speech recognition and live speech understanding. It supports several languages, including Ukrainian. Developers can create a variety of programs that convert spoken language into written text and extract valuable and important information from it.

Kaldi: This is an open source speech recognition toolkit that provides tools and various scripts for building speech recognition systems. Although it does not have a pre-trained Ukrainian model, it allows researchers and developers to train their own models using the tools and resources provided. This flexibility makes it a good option for creating a non-standard Ukrainian speech-to-text system adapted to specific needs or applications.

CMUSphinx: CMUSphinx is another speech recognition toolkit with an “open code” methodology that can be used to build speech-to-text systems. It supports several languages, including Ukrainian, and offers various tools for acoustic model training, language model creation, and decoding. It is easily customizable, allowing users to tailor it for specific applications or work environments.

Microsoft Azure Speech Service: Microsoft Azure Speech Services is a cloud service that provides an API for speech recognition, including support for the Ukrainian language. It works on the basis of deep neural networks and provides high accuracy of transcription of speech into text [5].

IBM Watson Speech-to-Text: IBM Watson provides a speech-to-text service that uses machine learning algorithms to transcribe spoken language into written text. Although Ukrainian may not be natively supported, the service allows users to potentially support Ukrainian by allowing users to create their own language models by training the system with domain-specific data [6].

To summarize, all the advantages and disadvantages of this or that technology are listed in the following Table (Table 1).

Table 1
Technology Comparison

Technology	Advantages	Disadvantages
Google Cloud Speech-to-Text API	High Accuracy Support more than 125 languages Real-time speech tracking and recorded files	Internet Connection Required API Usage Cost
Mozilla DeepSpeech	Open Access Can be used offline	Limited language support Technical expertise required for training and implementation
Wit.ai	Supports multiple languages Good for app development	Due to being tied to Facebook, may have privacy issues Internet connection required
Kaldi	Open Source Customizable Can be used offline	No previous Ukrainian model Technical expertise required for training

CMUSphinx	Open Source Supports Multiple Languages Can Be Used Offline	Lower accuracy compared to deep learning-based solutions Technical expertise required for training
Microsoft Azure Speech Service	High Accuracy Supports Multiple Languages Real-time Speech Tracking and Recorded Files	Internet Connection Required API Usage Cost
IBM Watson Speech-to-Text	Supports native language models Can be used for domain-specific applications	The Ukrainian language is not supported Internet connection required API Usage Cost

5. Hierarchy analysis methods for decision-making regarding tools for recognizing the Ukrainian language

Let's consider the method of analysis of hierarchies for making a decision on choosing one of the three speech recognition systems [14], which we will evaluate according to five criteria. The alternatives to analyze are: Google Cloud Speech-to-Text API, Mozilla DeepSpeech and Kaldi, using five criteria - the Analytic Hierarchy Process [15] is applied.

First, let's define the criteria as follows:

- Recognition accuracy (K1): Determination of how accurately the system converts speech to text. The higher the accuracy, the better the system.
- Processing speed (K2): An estimate of the time required for the system to process the incoming audio and return the recognized text. Speed can be a key factor for some applications.
- Language support (K3): Consideration of which languages are supported by the speech recognition system. A wider range of supported languages can make the system more versatile.
- Cost (K4): Estimate the cost of using the system, including the cost of using the API and other costs.
- Scalability (K5): Assessment of the ability of the system to work effectively when increasing the amount of data or the load without a significant decrease in performance.

Among the analogues listed in Table 1, we will choose three for analysis:

1. Google Cloud Speech-to-Text API (A1) is a speech recognition service developed by Google that provides the ability to convert speech to text with high accuracy and speed using cloud computing.

2. Mozilla DeepSpeech (A2) is an open source speech recognition software developed by the Mozilla Foundation. It uses neural networks for speech recognition and tries to provide accuracy comparable to commercial solutions.

3. Kaldi (A3) is open source software for automatic speech recognition, which provides a library of tools and algorithms for creating self-made speech recognition systems.

After successfully defining the systems, we build a hierarchy. It will include the main goal (goal) of this project, five selected criteria and alternatives that will be depicted in the form of information systems. The hierarchy itself is shown in Fig. 5.

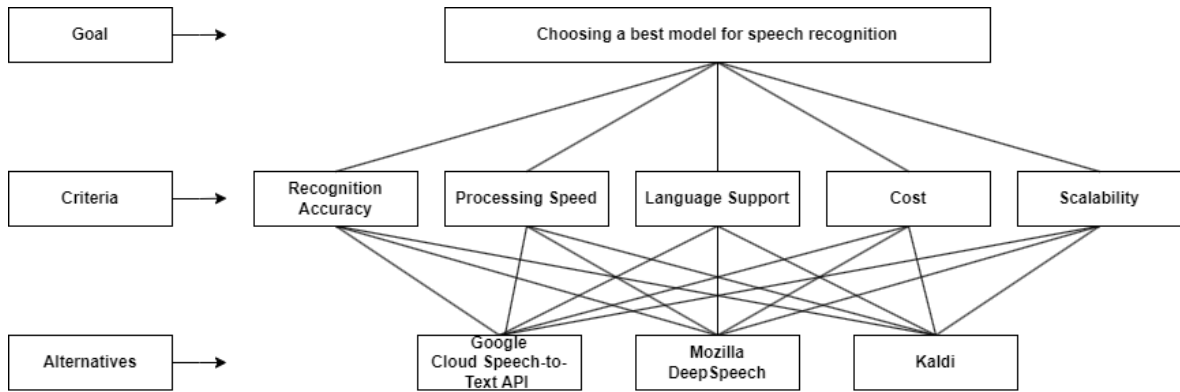


Figure 5: Hierarchy.

To create a matrix of pairwise comparisons of criteria, a list of criteria to be compared was defined. Now a matrix is being built that will show expert evaluations (Fig. 6). The names of the columns and columns will be the criteria mentioned above. Columns will also be added that will define eigenvalues and vectors.

	K1	K2	K3	K4	K5
K1	1	3	2	5	2
K2	0,33	1	2	4	3
K3	0,50	0,50	1	3	3
K4	0,20	0,25	0,33	1	0
K5	0,50	0,33	0,33	3,00	1

Figure 6. Matrix of expert evaluations.

Let's calculate the value of eigenvalues according to this formula (2.1):

$$B\check{C} = \sqrt[n]{\prod_{j=1}^n a_{ij}}, \quad (2.1)$$

Calculation:

$$\begin{aligned}
 B\check{C}(K1) &= \sqrt[5]{1 * 3 * 2 * 5 * 2} = 2,27 \\
 B\check{C}(K2) &= \sqrt[5]{0.33 * 1 * 2 * 4 * 3} = 1,52 \\
 B\check{C}(K3) &= \sqrt[5]{0.5 * 0.5 * 1 * 3 * 3} = 1,18 \\
 B\check{C}(K4) &= \sqrt[5]{0.20 * 0.25 * 0.33 * 1 * 0.33} = 0,35 \\
 B\check{C}(K5) &= \sqrt[5]{0.55 * 0.33 * 0.33 * 3 * 1} = 0,70
 \end{aligned}$$

Let's calculate the value of the eigenvectors according to the formula

$$B\check{B} = \frac{w_i}{\sum_{i=1}^n w_i}, \quad (2.2)$$

We calculate the denominator as follows:

$$2,27 + 1,52 + 1,18 + 0,35 + 0,70 = 6,01$$

In the next step, we will carry out a similar manipulation for alternatives. Accordingly, the size of the matrix will be 3*3 and two additional columns of vectors. In fig. 7 provides a matrix of comparisons for alternatives according to the *recognition accuracy* criterion.

	A1	A2	A3	BЧ	BB
A1	1	5	7	3,27	0,74
A2	0,20	1	2	0,74	0,17
A3	0,14	0,50	1	0,41	0,09

Figure 7: Matrix of pairwise comparisons of alternatives for K1 (*Recognition accuracy*)

The final step is to create comparisons of alternatives based on the previously calculated values of each alternative with respect to each criterion (Figure 8).

	K1	K2	K3	K4	K5	Generalized priorities
	0,38	0,25	0,20	0,06	0,12	
A1	0,74	0,21	0,66	0,27	0,10	0,488797464
A2	0,17	0,24	0,21	0,63	0,47	0,255450937
A3	0,09	0,55	0,13	0,10	0,43	0,255751599

Figure 8: Results of applying the method of analysis of hierarchies for alternatives (A) and criteria (K)

After analyzing the results and using the method of hierarchies to choose the best type of system that we will involve in the development, very similar results were obtained for all four alternatives. However, the first alternative turned out to be better. Accordingly, the Google Cloud Speech-to-Text API was chosen for development.

6. Development environment

Visual Studio Code is a universal code editor with an Open-Code structure developed by Microsoft. It supports a wide range of programming languages, including Python, which was used in the development of an intelligent language analysis and control system depending on the criteria set by the user. The choice to use VSCode as the primary development environment for this project was based on several factors:

1. Cross-platform compatibility: VSCode is available on Windows, macOS, and Linux, which ensures a consistent cross-platform development environment, making it easier to collaborate with other programs.
2. Extensibility: VSCode has a rich ecosystem of extensions (called plugins) that can be easily added to improve and extend its functionality. For an intelligent system of language analysis and control, depending on the criteria specified by the user, special Python extensions, such as Python Language Server, can be used to provide syntax highlighting, code completion, and debugging support.
3. Integrated terminal: VSCode includes an integrated terminal that allows developers to run scripts, test code, and manage project dependencies without leaving the editor.
4. Version Control Integration: VSCode has built-in Git support, making it easy to track changes, commit code, and collaborate with other developers.
5. Customizability: VSCode allows developers to customize the appearance and behavior of the editor to suit their preferences and workflow.

An alternative to VSCode is PyCharm. It is an integrated development environment (IDE) specially designed for Python programming. The environment offers advanced features such as code refactoring and integrated testing support. However, it can be heavier and slower

compared to VSCode, and its regular version lacks some features that are only available in the paid pro version.

Advantages of this environment:

- Advanced features designed for Python development
- Integrated testing support

Cons of PyCharm:

- Heavier and potentially slower compared to VSCode
- Some features are limited to the paid professional version

The next analogue environment is Sublime Text. Also a fairly simple, fast text editor with the possibility of internal customization. It supports several programming languages, including Python. However, it lacks some of the built-in features and extension ecosystem available in VSCode.

Advantages of this environment:

- Easy and fast
- Highly customizable

Cons of Sublime Text:

- Less extensive ecosystem of extensions compared to VSCode
- Missing some built-in features available in VSCode

The third environment is Jupyter Notebooks. An interactive web environment that allows you to create and share documents containing live-code, equations, visualizations, and descriptive text. It is great for prototyping and data analysis, but may not be the best fit for developing full-fledged systems such as intelligent language analysis and control.

Advantages of this environment:

- Interactive and great for prototyping
- Ideal for data analysis and visualization

Cons of Jupyter Notebooks:

- Not intended for full application development
- Limited collaboration and management features compared to VSCode

Summarizing the above, Visual Studio Code was chosen as the main development environment for an intelligent system of language analysis and control depending on the criteria set by the user due to its cross-platform compatibility, compatibility, extensibility, integrated terminal, version control integration and the possibility of customization to the user's taste. While alternatives such as PyCharm, Sublime Text, and Jupyter Notebooks have their merits, the combination of features and ease of use provided by VSCode make it the best option for this idea.

7. Programming languages

To develop the system, a universal, multi-level programming language Python was chosen, which gained and is gaining huge popularity among developers due to its simplicity, readability and a large set of libraries. To implement an intelligent language analysis and control system [7] - [11] depending on the criteria set by the user, Python was used for several reasons:

1. Clarity: Python's syntax is designed to be as clean, simple, and easy to understand as possible for other users, which helps to write user-friendly and efficient code.
2. Choice of libraries: Python has a huge ecosystem of libraries and packages, which makes it easy to implement various functions in the system, such as audio processing, machine learning, and analysis of existing data.
3. Compatibility: Python code can be executed on different platforms such as Windows, macOS, and Linux with minimal modifications, allowing easy deployment of this system on different operating systems.
4. Support: Python has a large and active community that contributes to its continuous development and provides great resources such as documentation, tutorials, and forums where you can find useful information and answers to difficult questions.

5. Integration: Python is widely used in the field of machine learning and artificial intelligence, which allows you to easily integrate the most advanced machine learning technologies, such as speech recognition, in an intelligent language analysis and control system depending on the criteria set by the user.

It should be noted that this system in the first version is being developed for the Windows platform. In the future, it can be easily transformed for other systems using programming languages for a particular environment.

An analogue of Python for developing a suitable system is JavaScript, which allows you to execute scripts on the server side and can be used to develop web applications. Using WebRTC and other web APIs, implementing this system as a web application can ensure compatibility between devices, including Android-based smartphones, tablets, and desktops. Advantages include cross-device compatibility and integration with web APIs, while disadvantages include limited access to native hardware features compared to native applications and the need for additional efforts to optimize performance and user experience.

The next example and analogue for other systems is the Java language (for Android): It is the main programming language for developing native applications for Android devices. Using Java to create an Android application for this system will provide access to native hardware features and an optimized experience with Android devices. If this language is used, access to native hardware functions will be obtained, optimizing the experience of working on Android devices. But the disadvantages will be the limitation of the Android platform (iOS is not supported) and development for other platforms will be required. Therefore, if we talk about the development of applications for iOS, Swift is mentioned. It is the main programming language for iOS applications. The implementation of an intelligent language analysis and control system depending on the criteria set by the user as an iOS application using Swift will provide access to native hardware functions and an optimized experience of working with iOS devices. In turn, this will give advantages in access to own hardware functions and experience of working with iOS devices. But the iOS operating system itself supports applications only verified by the App Store platform, which in turn affects the implementation of the system. That is why, our choice stopped on the Python language for the implementation of the first version of this program on Windows.

8. Vosk Model

The Vosk model will be applied in an intelligent language analysis and control system depending on the criteria set by the user for speech recognition. In particular, it will be used to convert spoken speech into text, allowing the system to analyze and process the user's speech. Details about these models can be found on the [alphacephei](http://alphacephei.com) website.

In this system, the Vosk model performs several functions:

1. Audio-to-Text: Vosk will be used to transcribe recorded audio files, converting the user's speech into text that will be analyzed and processed by the program.

2. Real-time speech analysis: Vosk will also be used for real-time speech recognition, which will allow the system to monitor and analyze a user's speech as they speak. It will also allow the system to provide instant feedback on the user's speech and use of specific words.

3. Word recognition and analysis: By converting speech to text, the Vosk model will allow the system to search for specific words or phrases in the transcribed text, helping users identify areas for improvement and track their progress over time.

The Vosk model is an automatic speech recognition (ASR) library that can work with various languages, including Ukrainian. It uses deep learning techniques to convert spoken language into text. Below is a general description of how the Vosk model works with the Ukrainian language:

- input audio is subjected to pre-processing, which may include noise reduction, normalization and segmentation into smaller fragments to improve recognition accuracy;
- the pre-processed audio is converted into a set of fragments, which are numerical representations of the audio signal. These fragments capture the basic characteristics of

speech, such as pitch, tone, and phonemes.

- Vosk uses a language model to determine the most likely sequence of words given a set of features extracted from the audio. The language model is trained on a large corpus of Ukrainian text, which allows you to understand the structure, grammar and vocabulary of the language.
- the acoustic model is used to map selected features into phonemes or other verbal units. The acoustic model is trained on a large set of Ukrainian speech data, which allows recognizing specific sounds and speech patterns.
- the Vosk model combines the output of speech and acoustic models to generate sequences of words that best match the input audio. This involves a search algorithm, such as the Viterbi Algorithm, to find the most likely sequence of words based on estimates of the acoustic and speech model.
- the decoded text may undergo post-processing such as capitalization, punctuation and error correction to improve the readability and accuracy of the final transcription.

9. Description of the implementation of the task

The software is developed using the Python programming language and includes the following main modules:

- User authentication and registration module
- Audio recording and analysis module and results
- Live speech analysis module
- Word list correction module
- Module for playback of audio files
- Data analysis creation and display module

Among the main libraries and methods used will be the following:

1. The `os` library is used to interact with the operating system. This allows the function to clear the console screen (`os.system('cls')`) and join file paths using the `os.path.join()` function.
2. The `datetime` library is used to generate unique file names based on the current date and time using the `datetime.datetime.now()` command. `strftime("%Y%m%d-%H%M%S")`.
3. Speech Recognition: The speech recognition library is used in the `audio_to_text` function (called from parameter 1) to transcribe recorded speech into text. The library provides an interface for working with various speech recognition services, including the Google Web Speech API.
4. The `pyaudio` library is used in the `record_audio` function (called from parameter 1) to record the input audio from the user's microphone. This library provides a Pythonic interface to the `PortAudio` library, which is a cross-platform audio input/output library.
5. The `openpyxl` library is used to read and write Excel files. In context (option1), it is used in the `check_word_list`, `analyze_text` and `save_to_excel` functions to interact with the user's Excel file, where the word list for analysis and the speech-to-text results are stored.
6. The `analyze_text` function uses basic string manipulation and Python's built-in string functions to search for specified words in the transcribed text. It counts occurrences of each word and returns a list of found words and their count.

These libraries and methods provide the necessary functionality to record and analyze an audio file, transcribe it to text, search for specific words in the text, and save the results.

9.1. Live speech analysis module

This function (option2) implements the approach of using the user's speech and analyzing it in real time. Accordingly, if the user's speech contains Z-words, then the program will play a match signal, thereby notifying the user about the use of an unwanted word. The function responsible for the implementation in the code is called `process_speech`, designed to perform real-time speech recognition and detect unwanted words in a given list [12]. In order to achieve

the goal and obtain the final result, the following libraries and variables will be imported: pyaudio, queue, logging, json, vosk and keyboard.

We will also set variables for sound processing: chunk, format, channels and rate. The audio stream that will be received by the system is configured with a fragment size of 1024 samples, 16-bit integer format, one audio channel (mono) and a frequency of 48,000 Hz. These settings are typical for real-time audio processing, providing a balance between audio quality and processing speed.

As mentioned in the third chapter, Vosk is a model for speech recognition, and KaldiRecognizer is a class that uses Vosk. The given model used for speech recognition is passed to the function as a file path. It is loaded using the Model class from the Vosk library. The choice of model can significantly affect the accuracy and performance of speech recognition.

The pyaudio library is also used to record the input audio from the user's microphone in real time in this function [13].

The callback module is called by the audio stream whenever new audio data is received. It adds the audio data to the queue and returns the data and a variable indicating that the stream should continue. The queue module provides a thread-safe implementation of a queue that is used to store incoming audio data.

The function continuously processes the audio data from the queue using the Vosk KaldiRecognizer. When the parser receives a piece of signal data, it returns a JSON string containing the parsed text. The JSON is then parsed to extract the text.

The recognized text is split into words and each word is checked against the provided words_list. If a word from the list is found in the recognized text, the sound is played using the play_sound function, and the unwanted word is added to the list of unwanted_words.

Real-time speech recognition and unwanted word detection will continue until the spacebar is pressed. This is checked using the keyboard.is_pressed(" ") function. The function then returns a Counter object containing the occurrences of each junk word found in the recognized speech.

9.2. Module for playing audio files

This module is defined by the play_audio_file function, which allows the user to play audio files from a specific folder based on the recording date. The function uses the pygame library to handle audio playback and the keyboard library to detect keyboard keystrokes.

The pygame.mixer.init() function initializes the pygame mixer module for audio playback. The data directory is then determined using os.path.dirname(os.path.abspath(__file__)) to get the path to the script directory and os.path.join() to create the path to the user record folder. Creates a list of audio files (.wav format) in the user's recording folder using the list method.

If there are no audio files in the folder, the function outputs a message and returns to the beginning, skipping the rest of the code.

9.3. Data analysis results display module

The create_charts function is responsible for the implementation of this module. She creates two interactive charts using Plotly based on data stored in an Excel file. Charts are saved as HTML files and automatically open in your default web browser. So, it first sets the paths to an Excel file named {username}_text.xlsx and reads three sheets into three separate Pandas DataFrames: data_word_time, data_overall_text, and data_unwanted_words. We group the data_word_time DataFrame by words and calculate the number of words for each word. We store the result in a new DataFrame called word_counts.

To implement the first graph, we will select the 10 most frequently used words from the word_counts DataFrame and store them in a new DataFrame named top_words. We create a diagram (histogram) using Plotly, displaying the first 10 words and their number. Save the chart as an HTML file named chart_top_words.html and open it in your default web browser.

10. The result of the program execution

For example, let's take one of the points of this work and read it as a speech fragment. A visual result of speech analysis will be given below (Fig. 9).

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Запис та аналіз аудіо.
Запис починається...
Запис закінчено.

Перетворення аудіо в текст.
Цей модуль визначається функцією play_audio_file, яка дозволяє користувачеві відтворювати аудіо-файли з певної папки на основі дати запису.
Функція використовує бібліотеку pygame для обробки відтворення аудіо та бібліотеку keyboard для виявлення натискань на клавіші клавіатури.
Функція pygame.mixer.init() ініціалізує модуль мікшера pygame для відтворення аудіо. Потім визначається каталог даних за допомогою os.path.
dirname (os.path.abspath(_файл_)) для отримання шляху до каталогу сценарію та os.path.join() для створення шляху до користувача папки запису.
Сторостається список аудіофайлів (формат .wav) у папці запису користувача за допомогою методу списку. Якщо в папці немає аудіофайлів,
функція виводить повідомлення та повертається до початку, пропускаючи решту фрагменту коду. Функція реалізує структуру вкладеного циклу для
створення інтерфейсу на основі меню для користувача. Зовнішній цикл обробляє вибір дати, тоді як внутрішній цикл обробляє вибір аудіофайлу
у цю структуру дозволяє користувачеві переходити між двома меню, поки він не вибере вихід. Важливим пунктом у даній системі є те, що функція
іє розпізнає аудіофайли, витягує та аналізує їх назви (пропускаючи, що дата міститься в перших 8 символах назви файлу). Саме тому старі
ені файли (з option) створювались з назвою яка еквівалентна даті проведення запису мовлення. Потім дані сортується в порядку зростання, функція
виводить список унікальних дат запису аудіо та пропонує користувачеві вибрати дату, ввівши число. Якщо користувач вводить "0", функція
виводить із циклу та очікує екран. Для вибраної дати функція фільтрує аудіофайли, назва яких починається з вибраної дати, і відображає список
аудіофайлів для цієї дати. Наступним етапом користувачеві вибрати аудіофайл для відтворення, ввівши номер. Якщо користувач знову
вводить "0", функція виводить із циклу та очікує екран. Та якщо користувач вибрав дійсний номер файлу, функція відтворює вибраний
аудіофайл за допомогою модуля pygame.mixer.music. Він завантажує файл за допомогою pygame.mixer.music.load() і починає відтворення за
д допомогою pygame.mixer.music.play(). Для припинення відтворення функція визначає, чи натиснуто клавішу "esc" за допомогою методу keyboard.is
_pressed(). Якщо клавішу натиснули, функція зупиняє відтворення аудіо за допомогою pygame.mixer.music.stop(). У випадку помилки, функція за
безпечує зворотній зв'язок щодо неправильних дій під час процесу вибору дати та файлу. Користувачеві пропонується повторити спробу, якщо він
ввів невірне значення. У майбутньому дану функцію можна модернізувати декількома способами, щоб покращити її функціональність, зручність
та у використанні та продуктивність. Дякуємо за покращення коду вклучати в себе: 1. Інтерфейс користувача можна розробити більш зручний
і візуально привабливий інтерфейс за допомогою бібліотеки GUI, такої як Tkinter, Pyqt або Kivy. Це дозволить користувачам взаємодіяти з пр
ограною за допомогою мишки, повзуноків та інших графічних елементів замість текстового інтерфейсу. 2. Фільтрування файлів покращить пара
метри навігації та фільтрації файлів, дозволить користувачеві фільтрувати аудіофайли на основі додаткових критеріїв, таких як тривалість файлу, розмір файлу або певні ключові слова, присутні в текстовому форматі файлу. 3. Елементи к
ерування Реалізувавши розширені елементи керування для відтворення аудіо, як-от пауза, продовження відтворення, зупинка і регулювання гучності. Це забезпечить користувачеві більше контролю над відтворенням аудіо та покращить їхній загальн
ий досвід користування. 4. Асинхронне відтворення можна зробити відтворення аудіо асинхронним, щоб програма могла виконувати інші завдання на фоні під час відтворення аудіо, як-от відображення візуалізації аналізу даних або оновленн
я інших елементів інтерфейсу користувача. 5. Аналіз аудіо Інтегрування можливості аналізу аудіо, наприклад визначення настрою, темпу або гучності, та надання користувачеві аналізу даних параметрів розширить функціонал даної системи. 6
. Редагування аудіо Дозволить користувачеві виконувати основні завдання з редагування аудіо, наприклад обрізати, розділити на менші та об'єднувати аудіофайли. Це дозволить користувачеві налаштувати аудіо-файли та керувати ними безпос
ередньо в програмі.

Пошук слів в тексті.
Знайдені слова: ['користувач', 'функція', 'користувачеві', 'функція']
Кількість кожного знайденого слова: {'користувач': 3, 'функція': 13, 'користувачеві': 6, 'функція': 1}

1 -- Повернутися в головне меню.
2 -- Розпочати аналіз це раз.
: █
    
```

Figure 9: Performing User Speech Analysis.

As you can see in Figure 9, the program starts recording and analyzing the audio, which the user is notified by appropriate messages on the terminal. After recording an audio file, it is saved in the recording folder that was created during registration. In the next step, the intelligent language analysis and control system, depending on the criteria specified by the user, converts the audio file into text and analyzes it for the presence of Z-words specified at the previous stage. As a result, information was obtained about the found words and their number encountered during the speech of this text.

If the user wants to view and analyze his data, he uses the "Graphics" option. After execution, the program will open two graphs in the browser (Fig. 10 and Fig. 11) where you can see the number of Z-words in speech and the top 10 words that were used throughout the time.

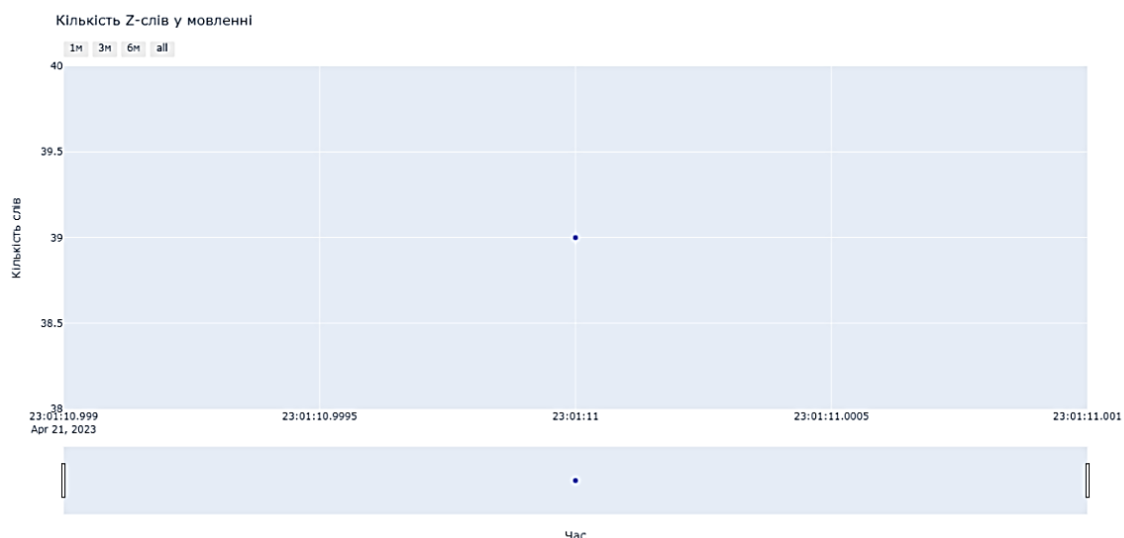


Figure 10: Graph of the number of Z-words in speech.

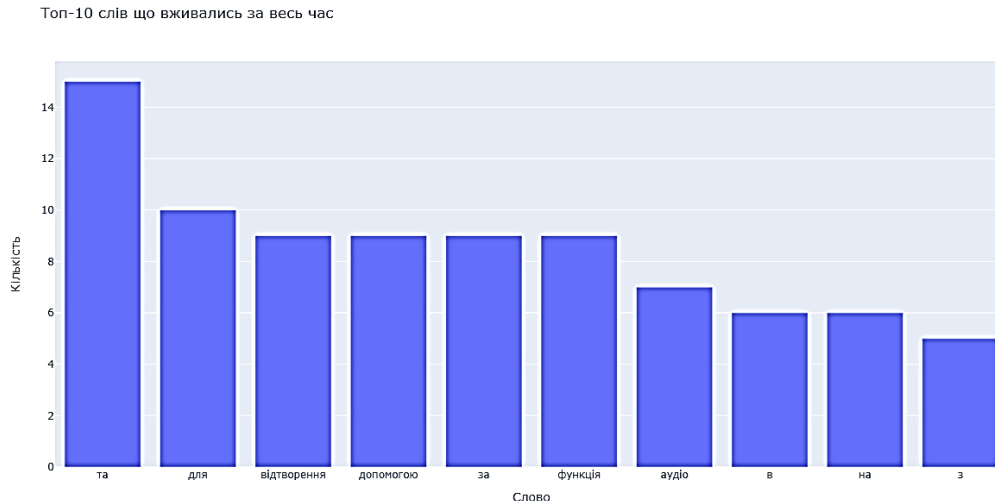


Figure 11: Chart of the top 10 words used in speech.

11. Conclusions

The work is devoted to modeling a tool for users who want to analyze their speech and improving their communication skills. Implementation of language analysis and control system can have positive consequences. These effects include reducing the use of unwanted words and phrases, increasing awareness of speech norms and communication culture, ensuring compliance with ethical standards in various environments, and promoting a healthy and positive speech environment.

The system was modeled using the UML notation method. A conceptual model of a system contains a description of the input and output data, functions and structure of the system, as well as its requirements.

The choice of Python as a programming language was due to its simplicity, versatility, and extensive library support, making it an ideal choice for implementing the project's functions. Alternative programming languages and platforms such as JavaScript or Swift can be considered to extend the project's reach to other devices such as mobile phones and web applications. The Vosk model was chosen due to its compatibility with the Ukrainian language, open source code and offline functionality.

The application uses Python libraries such as PyAudio, Vosk, Pandas, and Plotly to record and process audio, recognize speech, manage data, and create visualizations. The details of various functions that handle various methods of the program such as speech recording, speech processing, managing unwanted words in Excel, playing audio files and creating charts have also been covered. In addition, alternative methods for implementing some of these features, possible future enhancements, and additional visualization and analysis techniques that can be applied to provide users with more insight into their data are explored.

Some possible improvements to the developed application may include the following.

User Interface: A more user-friendly and visually appealing interface can be developed using a GUI library such as Tkinter, PyQt, or Kivy. This will allow users to interact with the program using buttons, sliders, and other graphical elements instead of a text interface.

File Filtering: Improved file navigation and filtering options will allow users to filter audio files based on additional criteria such as file length, file size, or specific keywords present in the text file format.

Controls: By implementing advanced controls for audio playback, such as pause, resume, stop, and adjust volume. This will give users more control over audio playback and improve their overall user experience.

Asynchronous playback: You can make audio playback asynchronous so that the application can perform other tasks in the background while the audio is playing, such as displaying data analysis visualizations or updating other user interface elements.

Audio Analysis: Integrating audio analysis capabilities such as mood, tempo, or loudness and allowing the user to analyze these parameters will expand the functionality of the system.

Audio Editing: Allow users to perform basic audio editing tasks such as trimming, splitting and merging audio files. This will allow users to customize and manage audio files directly in the program.

References

- [1] O. Matviykyv, N. Bokla, T. Klymkovych, U. Marikutsa, I. Farmaha, M. Lobur, M. Banas, Lab-chip diagnostic device for the rainwater monitoring system using wireless sensors network (2019) Proceedings of the 26th International Conference "Mixed Design of Integrated Circuits and Systems", MIXDES 2019, art. no. 8787185, pp. 241 - 245. DOI: 10.23919/MIXDES.2019.8787185
- [2] UML Diagram Types | Learn About All 14 Types of UML Diagrams. Creately Blog. URL: <https://creately.com/blog/diagrams/uml-diagram-types-examples/>.
- [3] What is Unified Modeling Language (UML)?. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. URL: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-uml/>
- [4] Google Cloud. (n.d.). Cloud Speech-to-Text. URL: <https://cloud.google.com/speech-to-text>
- [5] Microsoft Azure. (n.d.). Speech Services. URL: <https://azure.microsoft.com/en-us/services/cognitive-services/speech-services/>
- [6] Amazon Web Services. (n.d.). Amazon Transcribe. URL: <https://aws.amazon.com/transcribe/>
- [7] R. Peleshchak, V. Lytvyn, I. Peleshchak, A. Khudyy, Z. Rybchak, S. Mushasta, Text Tonality Classification Using a Hybrid Convolutional Neural Network with Parallel and Sequential Connections Between Layers (2022) CEUR Workshop Proceedings, 3171, pp. 904 - 915.
- [8] N. Melnykova, N. Shakhovska, V. Melnykov, M. Logoyda, Y. Peleshchak, The problem of analysing the relationships between individual characteristics of individuals with COVID`19 (2020) CEUR Workshop Proceedings, 2753, pp. 473 - 482.
- [9] Y. Tverdokhlib, V. Andrunyk, L. Chyrun, L.Chyrun, N. Antonyuk, I. Dyyak, O. Naum, D. Uhryn, V. Basto-Fernandes, Analysis and estimation of popular places in online tourism based on machine learning technology (2020) CEUR Workshop Proceedings, 2631, pp. 457 - 470.
- [10] I. Zheliznyak, Z. Rybchak, I. Zaviruschak, Analysis of clustering algorithms (2017) Advances in Intelligent Systems and Computing, 512, pp. 305 - 314. DOI: 10.1007/978-3-319-45991-2_21
- [11] V. Zheliznyak, V. Khavalko, V. Sherega, A.Boichuk, A.Barna, Biosignal and image processing system for emotion recognition applications (2021) CEUR Workshop Proceedings, 2824, pp. 181 - 191.
- [12] A. Hannun, Speech Recognition Is Not Solved. URL: <https://awni.github.io/speech-recognition/>
- [13] M. Swerts, J. Jansen, and J. Colpaert, "Speech Recognition for Language Learning: A Study of Usefulness, Learner Involvement and Effectiveness," Computer Assisted Language Learning, vol. 27, no. 4, 2014, pp. 349-369. doi: 10.1080/09588221.2014.913056.
- [14] L. Kobylukh, Z. Rybchak, O. Basystiuk, Analyzing the Accuracy of Speech-to-Text APIs in Transcribing the Ukrainian Language (2023) CEUR Workshop Proceedings, 3396, pp. 217 - 227.
- [15] T. Shestakevych, M. Logoyda, U. Marikutsa Expert group decision-making in corporate identity restyling (2022) International Scientific and Technical Conference on Computer Sciences and Information Technologies, 2022, November, pp. 230 - 234. DOI: 10.1109/CSIT56902.2022.10000501