

Capstone Project Phase A

Using speech recognition to execute full website functionality

Project code: 23-1-D-6

[Github Link](#)

Supervisor:

Mr. Alexander Keselman

Alex@vir-tec.net

Students:

Michael Ravich – 320912546

Michael.Ravich@e.braude.ac.il

Tal Derai – 205810369

Tal.Derai@e.braude.ac.il

Abstract:

The SpeakEasy web service is a new and innovative solution for internet navigation that utilizes the power of speech recognition technology. The service is designed to improve accessibility and efficiency for individuals who may have difficulty navigating the internet through traditional methods. SpeakEasy uses the Google Speech-to-Text (STT) API for speech recognition and natural language processing, and a custom-built command execution module to perform actions on the targeted website. Also, SpeakEasy includes fill forms functionality.

The service is optimized for using on mobile devices, making it a convenient and user-friendly tool for internet navigation.

This project book will explore the advantages of the SpeakEasy web service, including its ability to improve accessibility and efficiency for users with disabilities, as well as its user-friendly interface and easy integration with existing technologies. Additionally, the service improves the accuracy level by adjusting to the database of specific websites, allowing for more accurate and functional speech commands. The document will delve into the research that was conducted on the speech recognition field and technologies, providing a comprehensive understanding of the current state of the industry and the potential for future advancements.

This project book will highlight the unique features of SpeakEasy that set it apart from other products on the market, including the ability to adjust speech commands to the specific functionalities of a website, and the potential to improve the accuracy percentage of the model.

This project book also will provide a detailed description of the SpeakEasy service through the use of user case diagrams, activity diagrams, and architecture diagram. This will give a clear understanding of how the service works and the different components that make it possible. Furthermore, proposed GUI will be presented to show how the service will look for the user.

Finally, to ensure that the SpeakEasy web service provides high-quality service for the customer, a comprehensive testing plan will be presented. This will include both functional and performance testing, to ensure that the service meets the needs and expectations of its users. Overall, SpeakEasy web service is a powerful and innovative solution that has the potential to revolutionize the way people interact with the internet on their mobile devices.

Table of Contents

1. Introduction.....	5
1. Stakeholders	6
2. Document review.....	6
2. Background.....	6
1. Machine learning	6
1. Definition	6
2. History	7
2. API.....	7
1. Google STT API.....	8
3. Speech recognition & STT	8
4. Mobile web development	9
1. Next.js.....	10
2.Json	10
3.NPM	10
5. SQLight.....	11
6.Web socket.....	11
7.Similarity index.....	12
8.Data index	13
9.People with disabilities	13
10.WER.....	14
3. Related work	14
4. Expected Achievements.....	15
1. Goals	16
2. Success Criteria	16
3. Unique Features.....	17

5. Research / Engineering process	17
1. Process	17
2. Product.....	18
1. Architecture Diagram	18
2. Use Case Diagram	19
3. Activity Diagram	19
4. GUI.....	20
6. Evaluation/Verification Plan	22
7. References.....	23

1. Introduction:

Speech is the future.

The world's technology giants are making a beeline for Speech-based applications.

A Speech Interface makes natter human interaction with computers possible, using speech recognition to catch spoken words and typically text to speech for more interactive session.

Speech Recognition software have been added to automobiles, home automation systems, computer operating systems, home appliances like washing machines, microwave ovens and television remote controls and more, which led to extraordinary comfort.

Our idea is to develop a mobile web service, which will be based on an existing model of artificial intelligence for Speech recognition (google cloud speech-to-text API).

The purpose is to create an external component named “SpeakEasy” suitable to any website, in order to control all the functionality that the website offers through Speech commands.

In detail, our service will improve the accuracy level of the API, through adjustment to the database of the specific website.

So basically, after the API returns us, command translated text, our component will match it against the website database with the help of similarity index.

In this way, we can both improve the accuracy of the translation from Speech to text API, and also perform the functional options that the site offers, with only a Speech command.

e.g: Customer enters ‘zap.co.il’ website and requests a price comparison between different products by Speech command, our web-service processes the command, and as a result the website shows him a detailed comparison between those products.

In addition, “SpeakEasy” will allow transcription of the Speech input into text for filling forms on websites.

What sets our development apart from the products on the market today, is that our component will be based on the database of the specific website, and in this way, we will be able to implement Speech commands that are adapted to the specific functionalities of the site.

In addition, as we mentioned earlier, we will be able to make another adjustment of the text against the websites database and thereby improve the accuracy percentage of the model.

The products that exist in the market today allow direct translation of the Speech to text but not the execution of commands adjusted per site. In other words, there are competitors such as Siri, but its technology is suitable for searching desired text in a search engine such as Google and not in a way that is adapted to specific website functionality.

1.1 Stakeholders:

The stakeholders in this project:

1. Website owners who wish to make their content more easily accessible.
2. Website owners who wish the use of Speech commands on their website to be more accurate and functional.
3. Anyone who prefers to use Speech commands for full functionality instead of clicks on the small cell phone screen.

1.2 Document review:

In this book we will present the idea of expanding an existing Speech-to-text recognition model, to a website service, in order to improve accuracy and implement the functionality that the website offers, and our related work and development.

Our product's design includes developing mobile web service, modeling the architecture, and finally testing.

2. Background:

2.1 Machine learning:

2.1.1. Definition:

Machine learning is a subfield of artificial intelligence (AI) that involves the development of algorithms that allow computers to learn from data and make decisions or predictions without explicit programming. These algorithms use statistical models and patterns in the data to make predictions or decisions, and they are able to improve their performance over time as they are exposed to more data by training the model.

In machine learning, a computer is fed a large amount of data, and an algorithm is used to identify patterns and relationships in the data. The algorithm is then tested on new data to see how well it can make predictions or decisions based on the patterns it has identified. As the algorithm is tested and refined, it becomes better at making accurate predictions or decisions.

Machine learning has many practical applications, including in areas such as natural language processing, image recognition, and recommendation systems. It has the potential to revolutionize many fields by automating tasks and making decisions based on data, rather than relying on human judgment.

2.1.2. History:

The history of machine learning can be traced back to the early 1950s, when researchers began exploring the use of computers to learn from data and make decisions or predictions. One of the earliest and most influential works in the field was published in 1959 by Arthur Samuel, who developed a program that could learn to play the board game checkers.

In the 1960s and 1970s, research in machine learning continued to advance, with the development of new algorithms and techniques for training computer systems to learn from data. This period saw the emergence of neural networks, which are inspired by the structure and function of the human brain and are used to recognize patterns and make predictions or decisions.

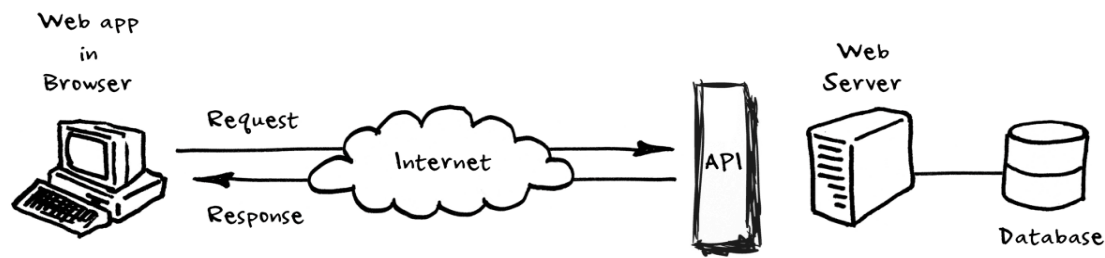
In the 1980s and 1990s, machine learning experienced a resurgence of interest, fueled in part by the availability of large amounts of data and the development of faster and more powerful computers. This period saw the emergence of new machine learning approaches, such as support vector machines and decision trees, as well as the development of tools and frameworks for building and deploying machine learning models.

Today, machine learning is a widely used and influential field. It continues to evolve and advance as researchers develop new algorithms and techniques, and as new sources of data become available.

2.2 API – Application Programming Interface:

API is an interface that computer programs use to interact with an application.

Application programming interfaces (APIs) expose services or data provided by a software application through a set of predefined resources, such as methods, objects, or URIs (Uniform Resource Identifier). By using these resources, other applications can access the data or services without having to implement the underlying objects and procedures. APIs are central to many modern software architectures, as they provide high-level abstractions that facilitate programming tasks, support the design of distributed and modular software applications and the reuse of code.[0]



2.2.1 Google Cloud Speech-to-text API:

Google Speech-to-Text is a cloud-based API that enables developers to convert spoken words to written text using machine learning. It is part of the Google Cloud Platform and can be accessed through a variety of methods, including through a web-based interface, as well as through APIs that can be integrated into third-party applications.

The Google Speech-to-Text API was first released in 2018, and it has since become a popular choice for developers looking to add speech-to-text capabilities to their applications. It is designed to be highly accurate and reliable, and it is able to recognize a wide range of accents and dialects.

The API is based on machine learning algorithms that are trained to recognize and transcribe speech patterns in various languages. It can be used for a variety of applications, including transcribing audio and video recordings, dictating text, and generating captions for multimedia content.

2.3 Speech Recognition & Speech to text:

Speech Recognition is the ability of machine/program to identify words and phrases in spoken language and convert them into machine-readable format.

Speech to text conversion is the process of converting spoken words into written texts. It is synonymous to speech recognition, but the latter is used to describe the wider process of speech understanding. STT follows the same principles and steps of speech recognition, with different combinations of various techniques for each step.

Here is a general overview of how a speech-to-text model might work:

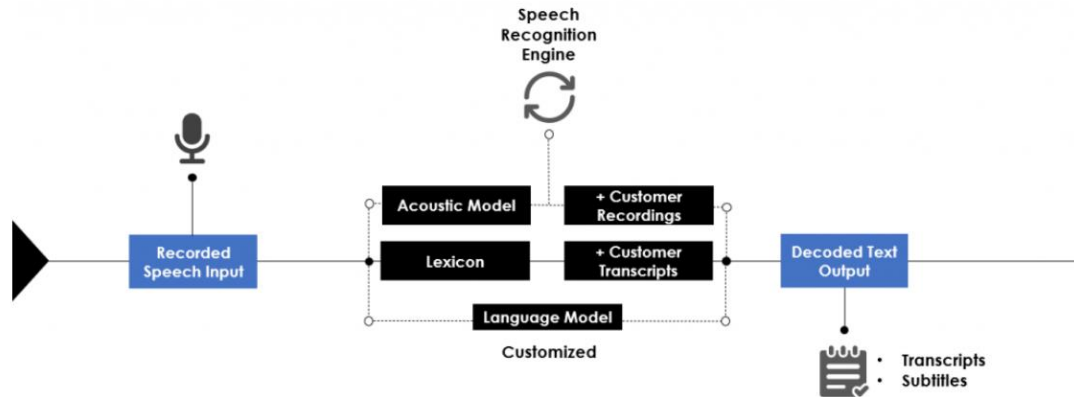
First, the model takes in an audio signal as input. This signal is typically a recording of someone speaking, but it could also be live speech captured in real-time.

The model then pre-processes the audio signal to extract relevant features that can be used for transcription. This might involve filtering out background noise, identifying the different phonemes or sounds in the speech, or extracting other relevant characteristics of the signal.

The extracted features are then fed into a machine learning model, which is trained to recognize and transcribe speech patterns in various languages. This model might use techniques such as deep learning or neural networks to analyze the features and make predictions about what words or phrases are being spoken.

The output of the machine learning model is then processed to generate a written transcript of the spoken words.

This might involve applying rules or constraints to ensure that the transcript is grammatically correct and follows the conventions of the target language.



[9]

2.4 Mobile web development:

Mobile web development refers to the development of websites and web applications specifically for use on mobile devices, such as smartphones and tablets. Mobile web development has become increasingly important in recent years with the proliferation of mobile devices and the increasing use of the internet on these devices.

The background of mobile web development can be traced back to the early days of the World Wide Web, when the first websites were designed to be accessed from desktop computers. As mobile devices became more widespread and internet access from these devices became more common, it became necessary to design websites that could be accessed and used effectively on these devices.

To accommodate the smaller screens and touch-based input of mobile devices, mobile web development involves designing websites and web applications that are optimized for these devices. This often involves using responsive design techniques, which allow a website to adapt to the size and capabilities of the device being used to access it.

In addition to responsive design, mobile web development may also involve the use of specific technologies and frameworks, such as Next.JS, HTML5, CSS3, to build web applications that are optimized for mobile devices. Mobile web development also often involves the use of mobile-specific features, such as GPS and camera access, to create rich and engaging user experiences.

2.4.1 Next.js:

Next.js is a popular open-source JavaScript framework for building server-rendered and statically generated web applications. It is built on top of React, a popular JavaScript library for building user interfaces, and it allows developers to easily create web applications that are optimized for performance and SEO (Search Engine Optimization).

One of the key features of Next.js is its ability to automatically generate static versions of pages, which can be pre-rendered on the server and delivered to the client. This can improve the performance of web applications and make them more accessible to search engines.

Next.js also provides a range of features and tools for building and deploying web applications, including support for server-side rendering, automatic code splitting, and automatic optimization of assets. It also integrates with a number of popular tools and libraries, such as GraphQL and TypeScript, to make it easier for developers to build and deploy web applications.

Overall, Next.js is a powerful and flexible framework that makes it easy for developers to build high-performance and scalable web applications.

2.4.2 JSON:

JSON (JavaScript Object Notation) is an open standard file format and data interchange format that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and arrays (or other serializable values). It is a very common data format, with a diverse range of applications, one example being web applications that communicate with a server.

2.4.3 NPM:

NPM (short for Node Package Manager) is a package manager for the JavaScript programming language. It is the default package manager for the JavaScript runtime environment Node.js. npm helps manage and share the packages (or modules) of code that are needed for a project, making it easier to develop and maintain applications. With npm, you can easily install, update, and manage packages, as well as share your own packages with the npm community. npm is widely used for building and deploying web applications, and it is an essential tool for any modern JavaScript developer.

2.5 SQLite:

SQLite is a popular open-source SQL database engine that is designed to be lightweight and easy to use. It is implemented as a library that can be embedded in applications and does not require a separate server process to operate.

SQLite is widely used in many applications, including web browsers, mobile devices, and desktop applications. It is known for its reliability, simplicity, and small size, and it is often used as an embedded database for applications where a full-featured database management system is not required.

SQLite supports a wide range of data types and SQL commands, and it includes a number of built-in functions and utilities for working with data. It is also highly extensible, with support for user-defined functions and extensions.

Overall, SQLite is a popular choice for developers who need a reliable and easy-to-use database engine for their applications.

2.6 Web Socket:

WebSocket is a computer communication protocol that provides a full-duplex communication channel over a single TCP connection. It is designed to be implemented in web browsers and web servers, but it can be used by any client or server application.

WebSocket was developed as a way to enable real-time communication between web applications and servers. It allows a client (such as a web browser) to initiate a connection to a server and then send and receive messages over that connection in real time, without the need to send HTTP requests and wait for responses.

WebSocket uses a handshake process to establish a connection, after which it can send and receive data as a stream of messages. It supports both text and binary data, and it can be used for a wide range of applications, including chat, multiplayer games, real-time data visualization, and more.

WebSocket is supported by most modern web browsers and is widely used in web-based applications that require real-time communication. It is also supported by many server-side technologies, such as Next.js, making it easy to implement on both the client and server sides.

2.7 Similarity index:

Similarity index is a measure of the similarity between two texts or documents. There are several different ways to calculate similarity index, but they generally involve comparing the content of the texts and determining the percentage of overlap or commonality.

One common method for calculating similarity index is to use the cosine similarity measure, which compares the vectors representing the texts and calculates the cosine of the angle between them. The resulting value ranges from 0, indicating no similarity, to 1, indicating a perfect match.

Similarity index can be used to compare the content of documents for a variety of purposes, such as detecting plagiarism or analyzing the similarity of different documents. It can also be used to compare the content of texts within a single document, such as in the context of text summarization or automatic translation.

Using similarity index in this way can help to improve the relevance and accuracy of search results, particularly when working with large databases or when the search query is not an exact match for the documents in the database.

There are several ways to implement a similarity index, but some common methods include:

Cosine Similarity: This method compares the angle between the vectors representing the two pieces of text. It is based on the dot product of the vectors and is commonly used in text classification and information retrieval.

Jaccard Similarity: This method is based on the Jaccard index, which is the size of the intersection of two sets divided by the size of their union. In text similarity, the sets are typically the sets of words or tokens in each text.

Levenshtein Distance: This method calculates the minimum number of single-character edits (insertions, deletions, or substitutions) required to change one string into another. It is used as a measure of similarity between two strings and is also known as the "edit distance" between the strings.

Longest Common Subsequence (LCS): It's a measure of similarity between two strings, it's a length of longest common subsequence between the two strings.

Word embeddings: Word embeddings are a way to represent words in a numerical format, this allows us to use mathematical operations like cosine similarity to find similarity between two sentences, by comparing the similarity between their embeddings.

It's worth noting that the choice of similarity method will depend on the specific use case and the type of data you're working with. Also, it's possible to use combination of different methods to improve the accuracy and robustness of the similarity index.

2.8 Data index:

A Data index is a data structure that allows efficient search, insertion, and deletion of data items in a database. An index is a separate data structure that is stored in a specific way to allow for faster access to data. There are several different types of indexes, each with its own specific characteristics and uses. For example, a primary index is a type of index that is used to locate records in a database table based on the primary key of the table. A secondary index is a type of index that is used to locate records in a database table based on a non-primary key attribute. Other types of indexes include clustered indexes, non-clustered indexes, and full-text indexes. Indexes are commonly used in database management systems to speed up the process of searching for specific data. By creating an index on a table, you can significantly improve the performance of queries that search for specific data in the table. This is because an index allows the database to quickly locate the data it needs, rather than having to search through the entire table to find the data. Indexes can also be used to improve the performance of insert and delete operations, as they allow the database to efficiently update the index when new data is added or removed from the table. However, it is important to note that indexes can also have a negative impact on the performance of some types of queries and operations, so they should be used carefully and only when they are needed.

2.9 People with disabilities:

People with disabilities can benefit from speech recognition programs. For individuals that are Deaf or Hard of Hearing, speech recognition software is used to automatically generate a closed captioning of conversations such as discussions in conference rooms, classroom lectures, and/or religious services.

Speech recognition is also very useful for people who have difficulty using their hands, ranging from mild repetitive stress injuries to involve disabilities that preclude using conventional computer input devices. In fact, people who used the keyboard a lot and developed RSI became an urgent early market for speech recognition.

Speech recognition is used in deaf telephony, such as Speechmail to text, relay services, and captioned telephone. Individuals with learning disabilities who have problems with thought-to-paper communication (essentially, they think of an idea, but it is processed incorrectly causing it to end up differently on paper) can possibly benefit from the software, but the technology is not bug proof.

2.10 WER:

WER stands for Word Error Rate, and it is a common metric used to evaluate the performance of speech recognition systems. WER is a measure of the accuracy of a speech recognition system, and it is calculated as the number of words in the transcript that are incorrect, divided by the total number of words in the reference transcript.

For example,
if a speech recognition system produces the following transcript for a given audio:

The cat sat on the mat

And the reference transcript for the same audio clip is:

The cat sat on the hat

Then the WER would be calculated as:

$$(2 \text{ incorrect words}) / (4 \text{ total words}) = 0.5$$

In this example, the WER is 50%, which means that the speech recognition system made errors on half of the words in the transcript. A lower WER indicates better performance of the speech recognition system, and a higher WER indicates poorer performance.

WER is often used in speech recognition research and development to compare the performance of different speech recognition systems or to track the performance of a single system over time.

3. Related Work:

Speech recognition has been an active area of research for decades, and there has been significant progress in the development of models and systems for converting spoken language into written text. In this section, we review some of the most relevant and influential works in the field of speech recognition.

One of the earliest and most influential works in speech recognition is the Hidden Markov Model (HMM), which was introduced by Rabiner and Juang in 1986 [1]. The HMM is a probabilistic model that is widely used for modeling sequential data, and it has been applied to many tasks, including speech recognition, language modeling, and machine translation.

Another influential approach to speech recognition is the use of deep neural networks (DNNs). DNNs have achieved state-of-the-art performance on many speech recognition

tasks, and they have been used to improve the accuracy and robustness of speech recognition systems [2, 3].

More recent approaches to speech recognition have focused on the use of attention mechanisms, which allow the model to selectively focus on certain parts of the input when making predictions. Attention mechanisms have been shown to improve the performance of speech recognition models, particularly when dealing with long sequences or noisy inputs [4, 5].

In addition, it is known that the google speech to text API has a WER error percentage of about 15 percent! That means we have many percentages that can be improved [6].

In summary, the field of speech recognition has seen many advances over the years, and there are many different approaches and models that have been developed for converting spoken language into written text. Our project builds on these previous efforts and aims to improve the accuracy and robustness of speech recognition models through the use of additional text analysis using the client's (website's) database.

Useful innovations that use Speech recognition for example:

- Siri is a digital personal assistant that performs searches and completes actions in response to an end user's natural Speech commands and learns from a user's behavior and routines to provide predictive recommendations and information.
- Alexa is Amazon's digital assistant, first launched in 2014 along with the Amazon Echo smart speaker. Alexa can provide results for web searches, order products from Amazon, and act as a hub for compatible IoT devices all via Speech commands.

4. Expected Achievements:

What differentiates our project from all these assistants that use Speech recognition, is that our service will be adapted according to the website's DB and according to the functionality it offers. The technologies we mentioned above do indeed process Speech into text and know how to perform certain actions but did not know how to realize all the existing functionality of a certain website using only the human Speech.

In addition, as we mentioned, we will perform additional text analysis after we receive it from the google API and thanks to this, we will be able to increase our level of accuracy.

4.1 Goals:

Our main goal is to extend existing software component to web-service that will be applicable for mobile web services.

Our additional goals are:

1. Search in database of mobile application will be accurate as possible. E.g: few errors as possible in Speech recognition.
2. Make the use of websites via mobile phones more convenient for all users.
3. The web service should be able to integrate with other systems and platforms as needed.
4. The interface should be easy for users to understand and use.
5. Help to peoples with disabilities. Especially, to adjust our service to mobile applications that are used most for disabled/elderly population. For example: Bituah Leumi, banks, shopping sites and more.
6. Do not infringe on user privacy rights (not saving recording Speech or distribution to third-party application).

4.2 Success Criteria:

To evaluate the performance of our proposed web service model, we will use a number of different metrics and benchmarks. These will include:

1. Increase number of clients (websites) per year.
2. Positive feedback from our clients – websites.
3. Fix bugs infrequently.
4. The web service should be secure, protecting sensitive data and preventing unauthorized access.
5. Word error rate (WER): We will use WER to measure the overall accuracy of our model.
6. Speed: We will measure the speed of our model in terms of the number of words it can process per second, and we will compare it to other models to see how it performs in terms of efficiency.

4.3 Unique Features:

Our proposed web service model has several unique features that set it apart from other models in the field. These include:

1. Possibility to adapt the service to **each** site.
2. Additional text analysis after we receive it from the google API, by this we aim to improve the accuracy of the model. We expect it to provide significant benefits in terms of performance.
3. Full implementation of the client's functionality through Speech commands, by interfacing with the site's database.

Overall, we believe that our proposed model has the potential to make significant contributions to the field of speech recognition and to improve the accuracy and robustness of speech recognition systems.

5. Research / Engineering Process:

5.1 Process:

First, we learned about the architecture of the intended system, with an emphasis on the API which we will use - Google Speech to Text.

We divided the learning equally by topics until the end of phase A.

We worked in a top-down methodology, meaning that we first performed a high-level review of all the components used in the project, and then dived into learning each of them in depth. Our learning and work process is divided into two parts:

1. Learning the tools need for our project:
IDE's (Integrated Development Environments), Code languages for development, System infrastructure, Web development, Data Base, Google speech to text API.
2. Theoretical learning for research:
Investigating the topic of speech recognition systems, existing models, products that use this technology, integrating the API into our system.
Reading articles in the speech recognition field, and learning about relevant concepts such as WER, similarity index, data index, etc.

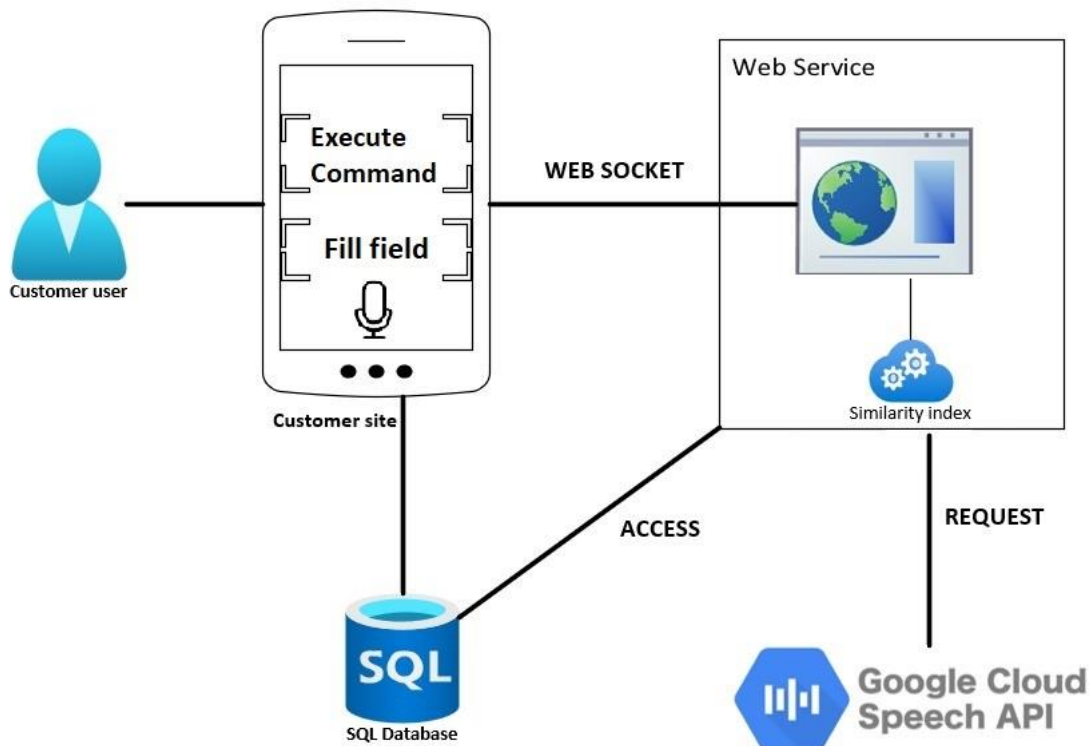
Challenges we face:

The main question we asked ourselves - how to improve and expand the google speech to text API to an improved model with better accuracy, and full functionality. We learned about methods to improve accuracy, and we decided to use the similarity index, through which the output from the model can be compared to keywords from the database, and then determine above what threshold to use the commands we extracted. This way we can actually give higher percentages of accuracy in our web-service. Regarding the realization of full functionality, we have decided that we will realize the option of making Speech commands, meaning that the user can use every option that the site offers via Speech command.

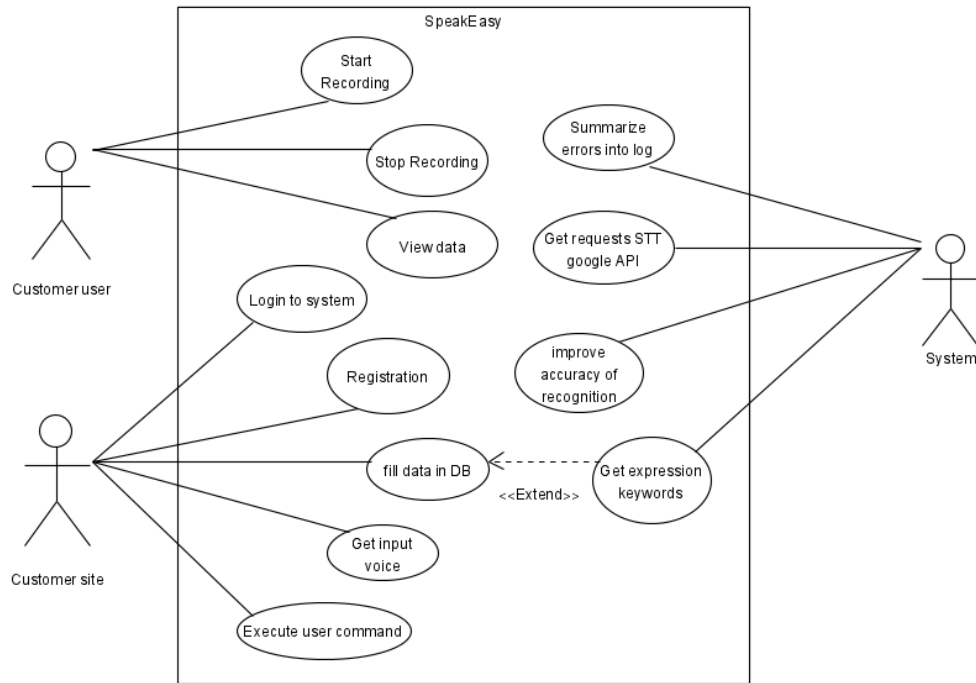
In order to actually implement all the tasks mentioned above, in phase B, we need to learn new programming languages, web development, infrastructure, etc. Since we have no prior knowledge of these subjects from our degree, we study them independently.

5.2 Product:

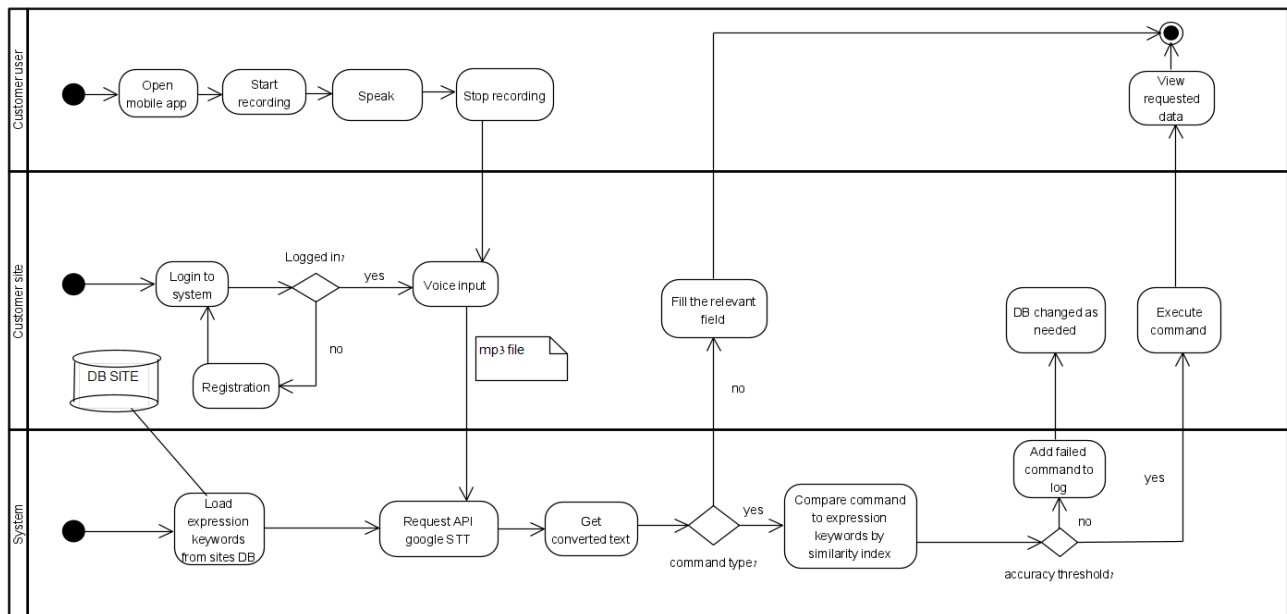
5.2.1. Architecture Diagram:



5.2.2. Use Case Diagram:

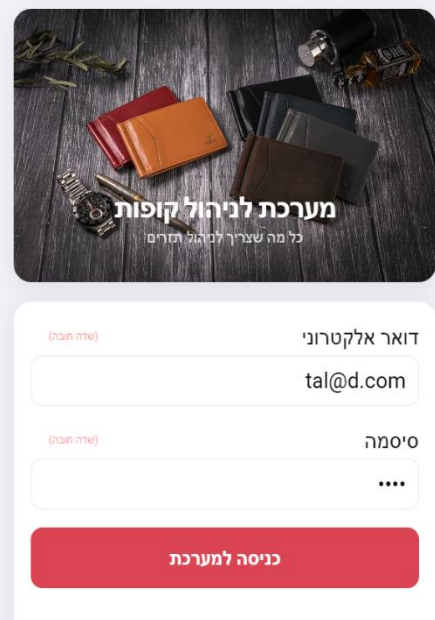


5.2.3. Activity Diagram:



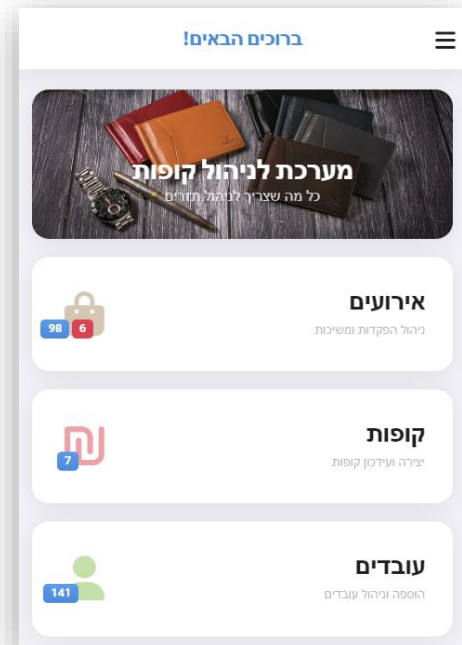
5.2.4. GUI:

User Login Page



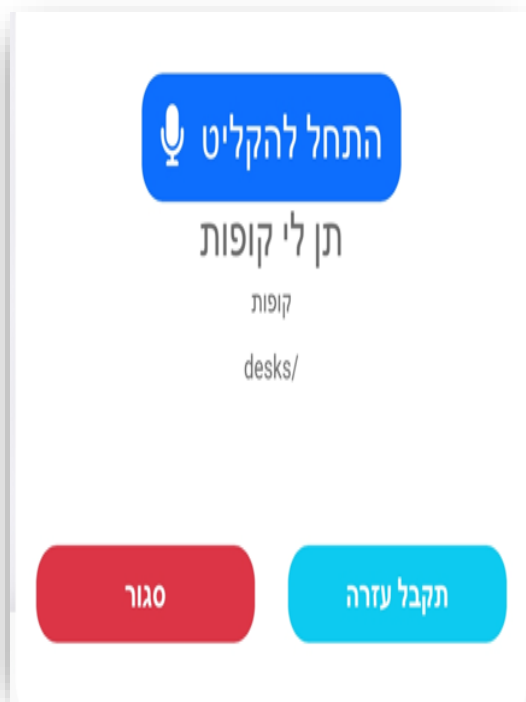
The User Login Page features a header image showing wallets and keys with the text "מערכת לניהול קופות" (Wallet Management System) and "כל מה שצריך לניהול תזרים" (Everything you need for cash flow management). Below the header is a login form with two input fields: "דואר אלקטרוני" (Email) with the placeholder "tal@d.com" and "סיסמה" (Password) with masked characters. A red button at the bottom is labeled "כניסה למערכת" (Login to system).

Main Page



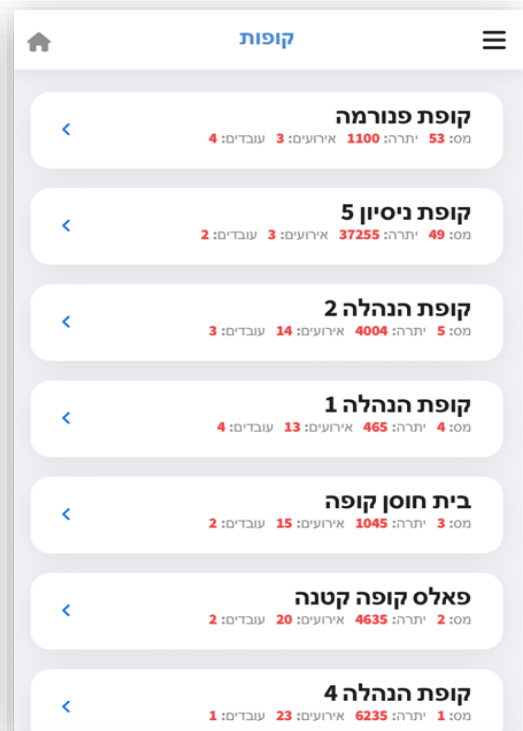
The Main Page has a header with the text "ברוכים הבאים!" (Welcome!) and a hamburger menu icon. Below the header is a large image of wallets and keys with the text "מערכת לניהול קופות" (Wallet Management System) and "כל מה שצריך לניהול תזרים" (Everything you need for cash flow management). The page contains three main sections: "אירועים" (Events) with a lock icon and "ניהול הפקדות ומשיכות" (Manage deposits and withdrawals), "קופות" (Wallets) with a wallet icon and "יצירה ועידכון קופות" (Create and update wallets), and "עובדים" (Employees) with a person icon and "הוספה וניהול עובדים" (Add and manage employees).

Recording Page



The Recording Page features a large blue button with a microphone icon and the text "התחל להקליט" (Start recording). Below this is the text "תן לי קופות" (Give me wallets), "קופות" (Wallets), and "desks/". At the bottom are two buttons: a red one labeled "סגור" (Close) and a blue one labeled "תקבל עזרה" (Get help).

Results



The Results Page has a header with a home icon, the text "קופות" (Wallets), and a hamburger menu icon. Below the header is a list of seven items, each with a back arrow, a title, and statistics. The items are: "קופת פנורמה" (Panorama Wallet) with 53 items, 1100 events, 3 employees, and 4 desks; "קופת ניסיון 5" (5 Experience Wallet) with 49 items, 37255 events, 3 employees, and 2 desks; "קופת הנהלה 2" (2 Management Wallet) with 5 items, 4004 events, 14 employees, and 3 desks; "קופת הנהלה 1" (1 Management Wallet) with 4 items, 465 events, 13 employees, and 4 desks; "בית חוסן קופה" (Resilience Wallet) with 3 items, 1045 events, 15 employees, and 2 desks; "פאלס קופה קטנה" (Small False Wallet) with 2 items, 4635 events, 20 employees, and 2 desks; and "קופת הנהלה 4" (4 Management Wallet) with 1 item, 6235 events, 23 employees, and 1 desk.

Brief explanation about the GUI screens:

(1) Login screen for the client site.

On the right side of the fields there is a microphone button in order to fill them with a speech transcription.

(2) After connecting to a cinema website (for example), there are several options and user can make a voice command by clicking on start recording.

(3) After clicking start recording, the button will turn red and stop recording will appear and the listening will start until the user clicks the button to stop.

(4) Output option:

The command went through successfully and the requested screen appeared.

(5) Output option:

The command failed even though it has logic, in that case there will be a log and the client can add the command to the database.

(6) Output option:

The command failed because the voice recognition was not successful.

6. Evaluation/Verification Plan:

Test	Tested Function	Tested Module	Testing Plan
1	Pressing record button	Customer Site	Customer user presses the record button. <u>Expected Result:</u> Starts listening to speech and displays a "stop recording" button.
2	Pressing stop recording	Customer Site	Customer user presses stop recording. <u>Expected Result:</u> The system transfers the recording file to the STT Google API and website displays identification results. Also, the “stop recording” button switches to “start recording”.
3	After speech command, the identification results don't pass threshold.	Customer Site, System	System checks the results by comparing to key expressions in DB via similarity index, detects that there is insufficient accuracy. <u>Expected Result:</u> Error message appears: “There isn't matching result”.
4	Filling forms	Customer Site	Customer user tries to fill form by speech input. <u>Expected Result:</u> Customer Site filling the form with the extracted text from the STT Google API.
5	Command didn't pass the threshold.	System	Customer user tries to execute some website functionality via speech command, but it doesn't pass threshold. <u>Expected Result:</u> Filling a log with the content of the extracted command for future improvement.
6	Executing Speech command successfully	Customer Site	Customer user tries to execute some website functionality via speech command, and it passes threshold. <u>Expected Result:</u> Customer site executes the requested functionality (shows required data).
7	Checking all Speech commands	System	System checks all possible commands in DB. <u>Expected Result:</u> Returns the matching action for each command.

7. REFERENCES:

[0] Michael Meng , Stephanie Steinhardt and Andreas Schubert -Journal of Technical Writing and Communication :Application Programming Interface Documentation: What Do Software Developers Want?

[1] Rabiner, L.R. and Juang, B.H. (1986). An introduction to hidden Markov models. IEEE ASSP Magazine, 3(1), 4-16.

[2] Hinton, G., et al. (2012). Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. IEEE Signal Processing Magazine, 29(6), 82-97.

[3] Amodei, D., et al. (2016). Deep speech 2: End-to-end speech recognition in English and Mandarin. arXiv preprint arXiv:1512.02595.

[4] Bahdanau, D., et al. (2014). Neural machine translation by jointly learning to align and translate. arXiv preprint arXiv:1409.0473.

[5] Chorowski, J., et al. (2015). Attention-based models for speech recognition. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, 1412-1421.

[6] <https://www.assemblyai.com/blog/speech-to-text-wer-google-aws-may-2020/>

[7] Speech to text - <https://cloud.google.com/speech-to-text>

[8] Review on Speech Recognition System for Disabled People Using Speech Recognition - https://www.researchgate.net/publication/348165820_Review_on_Speech_Recognition_System_for_Disabled_People_Using_Automatic_Speech_Recognition_ASR

[9] <https://www.scriptix.io/speech-to-text/>