Speech to Video – Ferramenta para construção de vídeos a partir do contexto de um áudio

Dálcio Garcia[1], Eufrane Neto[1], Grecio dos Santos[1], Manuel Muetunda[1]

[1]Instituto Superior Politécnico de Tecnologias e Ciências (ISPTEC) – Talatona – Luanda – Angola

{20170796, 20162116, 20170200, 20161278}@isptec.co.ao

Abstract. *This report portrays the techniques, tools and algorithms used to develop a tool that allows you to generate video through audio and image research. It also presents the tool itself and makes a comparative study with the competing tools already on the market. The same also speaks of the failures and difficulties that the group faced during the implementation of the tool as well as the improvements that the group intends to increase in future versions of the tool.*

Resumo. *O presente relatório retrata sobre as técnicas, ferramentas e algoritmos utilizados para desenvolver uma ferramenta que permite gerar vídeo por meio de áudio e pesquisa de imagens. O mesmo apresenta também a ferramenta em si e faz um estudo comparativo com as ferramentas concorrentes já existentes no mercado. O mesmo fala ainda das falhas e dificuldades que o grupo encarou durante a implementação da ferramenta bem como as melhorias que o grupo pretende incrementar nas futuras versões da ferramenta.*

1. Introdução

O reconhecimento de fala, também conhecido como reconhecimento automático de fala (ASR do inglês, Automatic Speech Recognition), reconhecimento de fala por computador ou fala para texto, é uma capacidade que permite que um programa processe a fala humana em um formato escrito. Embora seja comumente confundido com o reconhecimento de voz, o reconhecimento de fala se concentra na tradução da fala de um formato verbal para um texto, enquanto o reconhecimento de voz busca apenas identificar a voz de um usuário individual. E recentemente o reconhecimento de fala tem ganhado um grande espaço no mercado, tornando-se num dos principais métodos utilizados para a entrada de texto, facilitando o utilizador na inserção de texto, sobretudo de grandes volumes de texto.

Contudo, nos seus primórdios, não era vista como uma forma fiável de comunicação, isto desde o surgimento da "Shoebox" em 1962, visto que o sistema tinha capacidade para reconhecer somente 16 palavras diferentes. Isto porque a tecnologia que hoje conhecemos não tão desenvolvida, os dispositivos tinham uma baixa capacidade de armazenamento, hardwares fracos e muitos mais. Hoje, possuímos tecnologias mais poderosas que nos permitem processar grandes quantidades de informação, permitindo que os sistemas possam trabalhar de forma mais eficiente e eficaz.

Hoje, o avanço tecnológico no quesito multimídia, tem tornado possível o processamento de informação de diferentes formas, tal como transmissão de informação por intermédio de vídeo, tanto quanto a facilidade de acesso da informação, melhorias na forma de armazenamento de informação como cloud (processamento em nuvem).

Com o surgimento de tecnologias como HTML5 e Javascript, tem se tornado mais simples desenvolver soluções que fazem uso destes recursos, fornecendo soluções inovadoras e de grande praticidade.

Neste contexto a nossa ferramenta não passa de mais um fruto desta vasta área do saber.

A nossa ferramenta tem como o primeiro passo a transcrição de áudio captado de um microfone ou do dispositivo em texto e em seguida fazer uma busca de imagens referentes as palavras do texto na internet e por fim selecionar algumas destas imagens e montar um vídeo com as mesmas e o áudio captado.

2. Técnicas e procedimentos

Para a concepção do projeto usamos diversos processos para podermos chegar a uma solução palpável.

2.1. Processamento do áudio

O processamento do áudio envolveu duas etapas, nos permitindo assim fornecer aos usuários várias formas entrada de dados (áudio no caso).

2.1.1. Entrada a partir da escolha de um ficheiro

Esta forma de entrada permite ao usuário poder um ficheiro áudio do tipo flac existentes em seu dispositivo, para poder processar na aplicação.

Usamos um input do tipo file que nos permitiu carregar o conteúdo.

2.1.2. Gravação do áudio

Com isso, o usuário vai poder gravar seu áudio e assim poder processa-lo ou utiliza-lo na aplicação.

2.2. Processamento de Texto

O processamento do texto é feito usando o recurso da ferramenta IBM Watson Speech Recognition para transcrição do áudio para texto.

A IBM foi a pioneira no desenvolvimento de ferramentas e serviços de reconhecimento de voz que permitem às organizações automatizar seus processos de negócios complexos enquanto obtêm percepções de negócios essenciais. O Speech Recognition fornece soluções como, Speech to Text e Text to Speech.

Nós precisamos simplesmente do Speech to Text para a concessão da aplicação

2.2.1. Speech To Text

O Speech To Text, é uma solução nativa da nuvem que usa algoritmos de IA de aprendizado profundo para aplicar o conhecimento sobre gramática, estrutura de

linguagem e composição de sinal de áudio / voz para criar reconhecimento de voz personalizável para transcrição de texto ideal.

O Speech to Text faz uso de diversos algoritmos para fornecer resultados mais precisos, alguns dos algoritmos são: Processamento de linguagem natural (PNL), Modelos ocultos de markov (HMM), N-gramas, Redes neurais e Diarização de Locutor (SD)

2.2.1.1. Processamento de linguagem natural (PNL)

Embora a PNL não seja necessariamente um algoritmo específico usado no reconhecimento de fala, é a área da inteligência artificial que se concentra na interação entre humanos e máquinas por meio da linguagem por meio da fala e do texto. Muitos dispositivos móveis incorporam reconhecimento de voz em seus sistemas para conduzir pesquisa de voz - por exemplo, Siri - ou fornecer mais acessibilidade em torno de mensagens de texto.

2.2.1.2. Modelos ocultos de markov (HMM)

Modelos ocultos de Markov construídos no modelo de cadeia de Markov, que estipula que a probabilidade de um determinado estado depende do estado atual, não de seus estados anteriores. Enquanto um modelo de cadeia de Markov é útil para eventos observáveis, como entradas de texto, os modelos de markov ocultos nos permitem incorporar eventos ocultos, como tags de classes gramaticais, em um modelo probabilístico. Eles são utilizados como modelos de sequência no reconhecimento de fala, atribuindo rótulos a cada unidade - isto é, palavras, sílabas, sentenças, etc. - na sequência. Esses rótulos criam um mapeamento com a entrada fornecida, permitindo determinar a sequência de rótulos mais apropriada.

2.2.1.3. N-gramas

Este é o tipo mais simples de modelo de linguagem (LM), que atribui probabilidades a sentenças ou frases. Um N-grama é uma sequência de N-palavras. Por exemplo, “peça a pizza” é uma trigrama ou 3 gramas e “peça a pizza” é uma trigrama de 4 gramas. A gramática e a probabilidade de certas sequências de palavras são usadas para melhorar o reconhecimento e a precisão.

2.2.1.4. Redes neurais

Utilizadas principalmente para algoritmos de aprendizado profundo, as redes neurais processam dados de treinamento imitando a interconectividade do cérebro humano por meio de camadas de nós. Cada nó é composto de entradas, pesos, uma tendência (ou limite) e uma saída. Se esse valor de saída exceder um determinado limite, ele “dispara” ou ativa o nó, passando os dados para a próxima camada da rede. As redes neurais aprendem essa função de mapeamento por meio do aprendizado supervisionado, ajustando-se com base na função de perda por meio do processo de descida do gradiente. Embora as redes neurais tendam a ser mais precisas e possam aceitar mais dados, isso tem um custo de eficiência de desempenho, já que tendem a ser mais lentas para treinar em comparação com os modelos de linguagem tradicionais.

2.2.1.1. Diarização de Locutor (SD)

Os algoritmos de diarização de locutor identificam e segmentam a fala pela identidade do locutor. Isso ajuda os programas a distinguir melhor os indivíduos em uma conversa e é frequentemente aplicado em call centers, distinguindo clientes e agentes de vendas.

Graças a esses algoritmos, podemos obter melhores resultados no processamento de áudios.

2.3. Web Scraping (coleta de dados)

Web scraping, ou coleta de dados, é uma forma de mineração que permite a extração de dados de sites da web convertendo-os em informação estruturada para posterior análise.

Usamos o Web Scraping para procurarmos pelas imagens pela internet (usamos o google imagens como alvo), essa busca nos forneceu um conjunto de imagens relacionadas a uma determinada palavra, e assim pormos dar a possibilidade ao utilizador de escolher as imagens que mais se adequem a palavra.

2.4. Uploading to cloud

Upload refere-se à transmissão de dados de um sistema de computador para outro por meio de uma rede.

Usamos o upload para carregarmos o áudio e o vídeo para a nuvem (utilizamos o cloudnary, para o armazenamento). Carregamos o áudio na nuvem porque precisávamos que o server tenha acesso ao áudio para que assim possa transcreve-lo e retornar o texto correspondente ao áudio e também porque o ficheiro é necessário para a concessão do resultado fina, a geração de vídeo. E carregamos o vídeo na nuvem para que o utilizador tenha acesso ao arquivo, podendo assim, baixa-lo ou partilhar o link para que outras pessoas tenham acesso.

2.5. Geração ou criação de vídeo

Para a concessão do vídeo fizemos o uso do ffmpeg, que é o framework multimídia líder, capaz de decodificar, codificar, transcodificar, mux, demux, transmitir, filtrar e reproduzir quase tudo que humanos e máquinas criaram. Suporta os formatos antigos mais obscuros até a vanguarda. Não importa se eles foram projetados por algum comitê de padrões, a comunidade ou uma empresa.

O FFmpeg contém libavcodec, libavutil, libavformat, libavfilter, libavdevice, libswscale e libswresample que podem ser usados por aplicativos. Bem como ffmpeg, ffplay e ffprobe, que podem ser usados por usuários finais para transcodificação e reprodução. O ffmpeg é uma ferramenta de linha de comando para converter arquivos multimídia entre formatos

Foi com o FFmpeg que foi possível criar o vídeo utilizando as imagens selecionadas pelo utilizador e o áudio que outrora tivera sido carregado na nuvem.

3. Descrição do projeto

Evitando rodeios, segue-se a ilustração de como funciona internamente o nosso projeto, desde um fluxograma geral a um desenho de arquitetura.

3.1. Arquitetura da aplicação

A seguir mostraremos a arquitetura do projeto, baseando-se no REST API, desenvolvemos as a nossa aplicação utilizando a seguinte arquitetura:

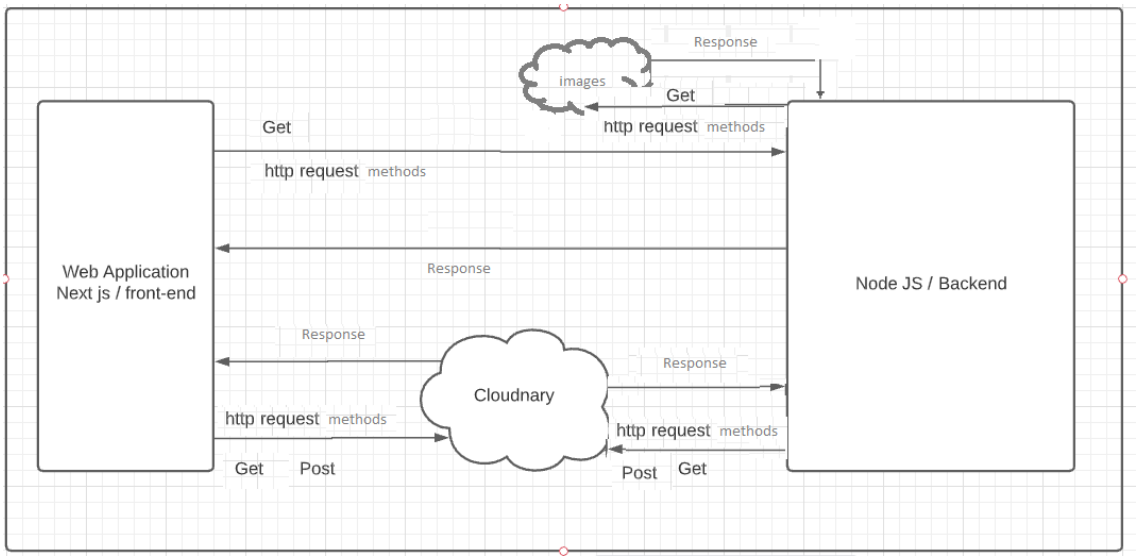


Figura 1. Arquitetura baseado no REST API architecture

Tudo começa no frontend (no lado do cliente), o usuário introduza o áudio na plataforma (gravando ou ficheiro) em seguida, é feito um upload, na nuvem **Cloudnary** através de uma requisição http post, em seguida é feita uma requisição http Get no server / backend, aí o server faz uma requisição http Get para o **Cloudnary** para adquirir o áudio carregado e depois gera uma response para o cliente retornando a transcrição do áudio, com a transcrição do áudio, o cliente segue o processo de obtenção de imagens(explicado na secção 3.2) ter sido concluído e iniciado o processo de geração do vídeo, é feito uma requisição na API (backend) do tipo Get e o server, antes de realizar um response ao cliente, ele faz uma requisição na nuvem (cloudnary) para pegar o áudio outrora uploaded e em seguida faz um Get nos repositório onde se encontram as imagens para baixa-las e posteriormente usar para a geração do vídeo, após a geração do vídeo o server executa um post no cloudnary fazendo o upload do vídeo, sendo retornado uma URL com o link do vídeo e depois de ser respondido ele responde (response) para o cliente com a URL do vídeo para que possa ser renderizada.

3.2. Armazenamento em nuvem

Como mencionamos antes, utilizamos o **Cloudnary** para o armazenamento de dados, esse processo nos permitiu conseguir persistência de dados na aplicação.

O armazenamento na nuvem é um modelo de computação em nuvem que armazena dados na Internet por meio de um provedor de computação na nuvem, que gerencia e opera o armazenamento físico de dados como serviço. O serviço é fornecido sob demanda, com capacidade e custos just-in-time, e elimina a compra e o gerenciamento de sua própria infraestrutura de armazenamento físico de dados. Com isso, os utilizadores poderão obter agilidade, escala global e resiliência, de dados podendo acessa-los a partir de qualquer lugar do mundo, a qualquer hora, não havendo necessidade de instalação de programas ou de armazenar dados. O acesso aos dados é

remoto, através da Internet (por isso a alusão à nuvem). E o uso desse modelo (ambiente ou forma de armazenamento de dados) é mais viável do que o uso de unidades físicas.

3.2.1. Como funciona o armazenamento em nuvem?

A nossa aplicação acessa as informações por meio por meio de protocolos de armazenamento **HTTP** isso na parte do cliente e diretamente usando a API do **Cloudnary**. Tanto por requisição http quanto por meio da API, nós conseguimos gerenciar os dados da plataforma.

3.2.2. Cloudnary

O Cloudinary oferece uma variedade de guias que descrevem os recursos disponíveis e apresentam casos de uso comuns, dessa variedade usamos o **Media Upload**, pois nos permite fazer o upload de imagens, vídeos, áudio ou qualquer outro tipo de arquivo de quase qualquer fonte usando uma API ou por meio de requisições.

O Cloudinary fornece uma API segura e abrangente para enviar facilmente arquivos de mídia de código do lado do servidor, diretamente do navegador ou de um aplicativo móvel. Nós pudemos fazer upload usando a API REST da Cloudinary e uma das bibliotecas de cliente (SDKs) isso, porque usamos o NodeJs para projetarmos a nossa solução, simplificando a integração com a aplicação e o nosso backend. As bibliotecas de cliente (SDKs) da Cloudinary envolvem a API de upload e simplificam bastante o uso dos métodos da API. Para fazer o upload de arquivos bastou uma chamada direta para a API no backend, e enviarmos uma solicitação HTTPS POST e GET.

3.3. Fluxo de funcionamento na parte do cliente ou usuário

Ilustraremos como funciona a interação do usuário com a aplicação.

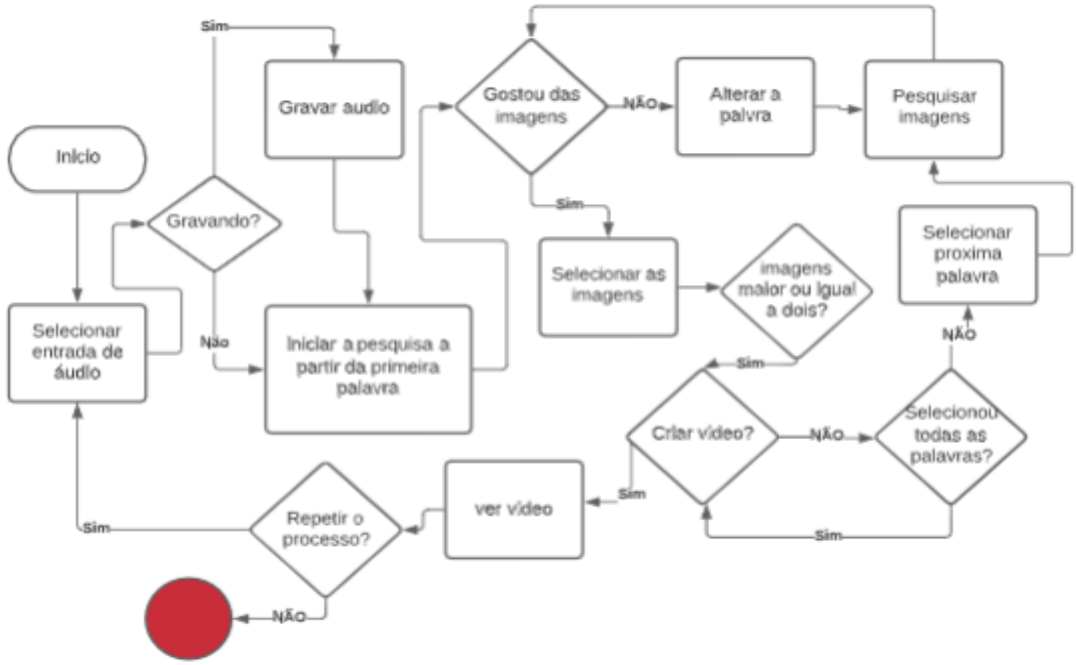


Figura 2. Fluxo de interação do usuário com a aplicação

3.3. Fluxo de funcionamento na parte lógica

Representaremos de uma forma esquemática como funciona internamente a aplicação de acordo com as operações do usuário.

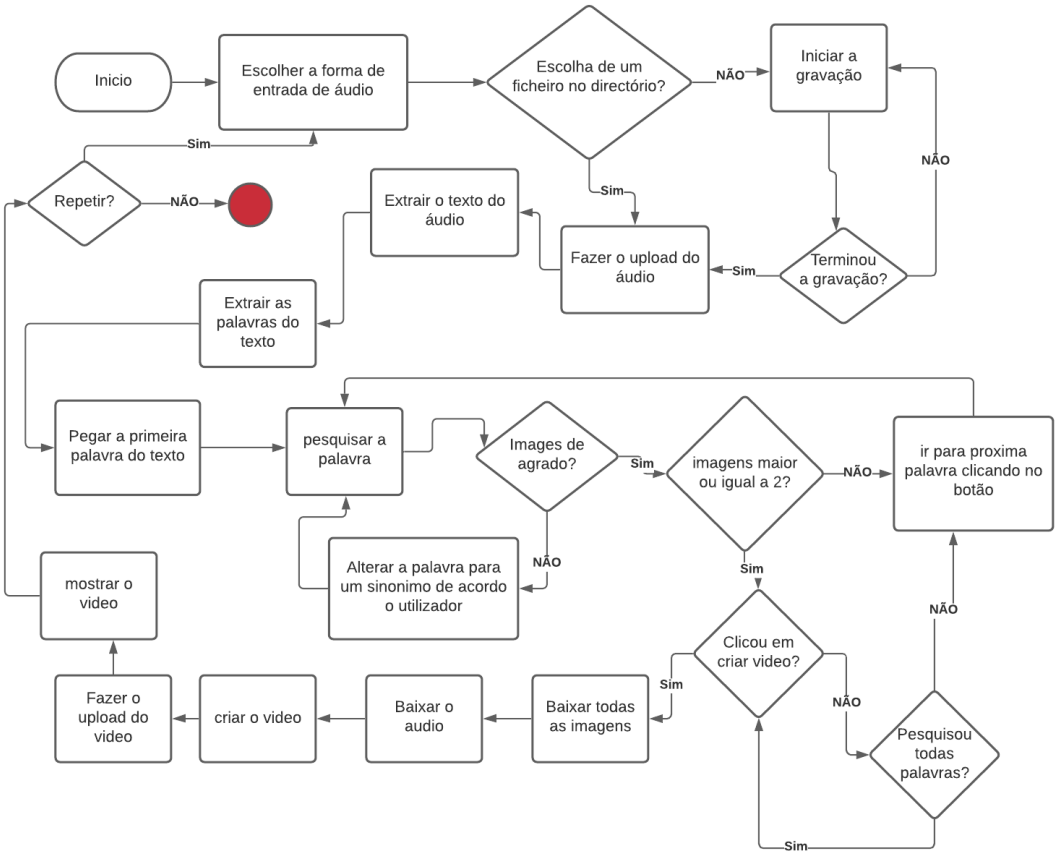


Figura 3. Fluxo da aplicação para a geração do vídeo

4. Aplicações concorrentes

Quanto a concorrência, acabamos por não encontrar aplicações que fazem o mesmo que a nossa, mas encontramos algumas que possuem algumas funcionalidades semelhantes.

4.1. Canva Slides

O Canva é uma ferramenta de design que permite fazer apresentações de slides impressionantes em segundos. Com o Canva, você pode criar apresentações de slides de fotos e vídeos com música para impressionar seus seguidores, amigos ou familiares.

4.2. Google Slides

O aplicativo permite que os usuários criem e editem arquivos online enquanto colaboram com outros usuários em tempo real. As edições são rastreadas pelo usuário com um histórico de revisão que apresenta as alterações.

4.3. Adobe Spark

O Adobe Spark é um conjunto integrado de aplicativos de criação de mídia para celular e web desenvolvido pela Adobe Systems. É composto por três aplicativos de design separados: Spark Page, Spark Post e Spark Video.

O serviço faz parte da Creative Cloud, cujo conteúdo criado é salvo automaticamente na nuvem. O aplicativo da web gratuito Adobe Spark sincroniza com os aplicativos móveis Spark Page, Spark Post e Spark Video iOS, permitindo aos usuários criar, editar e compartilhar sua história visual a partir de qualquer dispositivo. Os três aplicativos de design permitem aos usuários criar e projetar conteúdo visual que pode ser usado para negócios, educação, mídia social, etc. Os usuários podem importar/pesquisar imagens usando qualquer um dos três aplicativos, apenas com imagens marcadas com a licença Creative Commons estando disponível com a ferramenta de pesquisa.

Como pudemos ver, nenhuma das soluções se baseia no áudio como fonte de entrada principal para a geração de seus resultados finais. Já a nossa solução, é a partir de um áudio que é processado o resultado.

5. Viabilidade

A nossa ferramenta tem como requisito os seguintes

- Um smartphone ou um computador
- Acesso à internet
- Um navegador em seu dispositivo
- O dispositivo precisa permitir a captação de áudio (Recomendado, mas não obrigatório)

6. Público alvo

Hoje em dia, a mídia é setor em evolução continua, e novas formas de fornecer produtos desta natureza vêm surgindo constantemente.

A nossa aplicação foi feita pensando nas pessoas que pretendem criar conteúdos visuais dinâmicos, de baixo custo e fácil acesso, sendo assim todas as pessoas que desejam tal feito estão aptas para o uso da nossa aplicação. E isso, faz com que tenhamos os músicos, palestrantes, professores, estudantes, pessoal de marketing e outros façam parte do nosso público alvo.

7. Resultados

Feito a implementação da segunda fase da aplicação, pudemos obter os seguintes resultados:

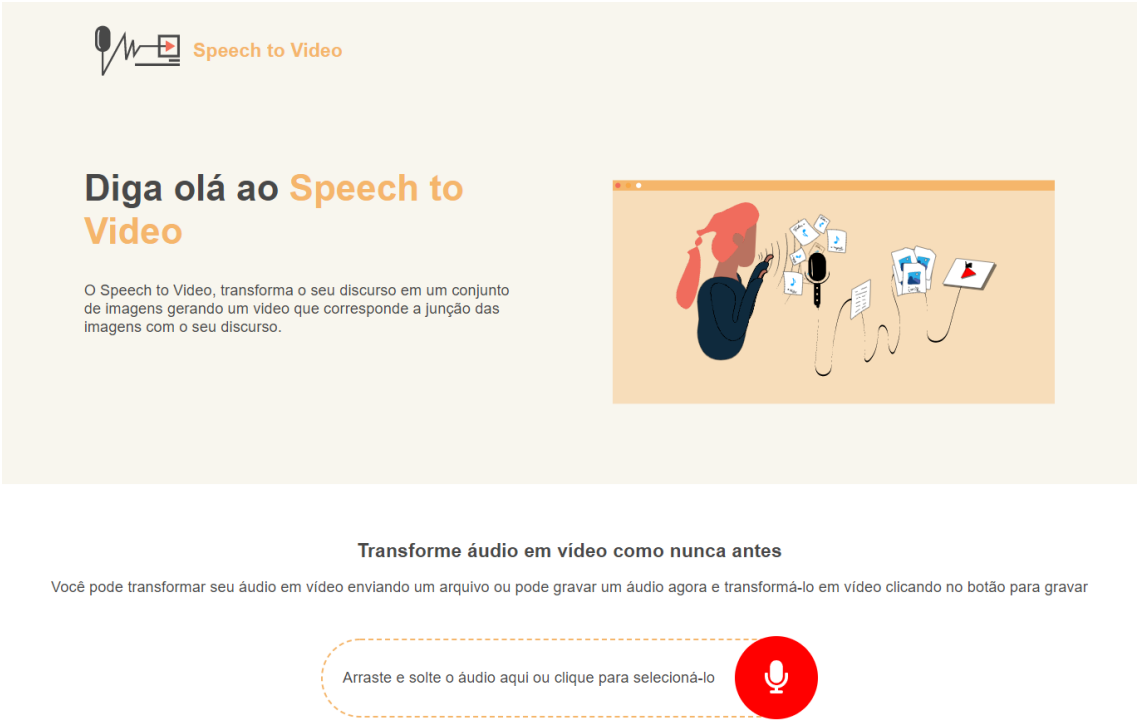


Figura 4. Página inicial da aplicação

A **figura 4,** mostra a tela inicial, contendo informações sobre a plataforma e um campo que nos permite selecionar o ficheiro clicando ou arrastando e largando o áudio em nosso dispositivo e um botão com Mic que nos permite gravar o áudio em vez de selecionarmos.

Figura 5. Gravando o áudio para a aplicação

A **figura 5,** mostra o estado do botão quando for clicado para gravar.

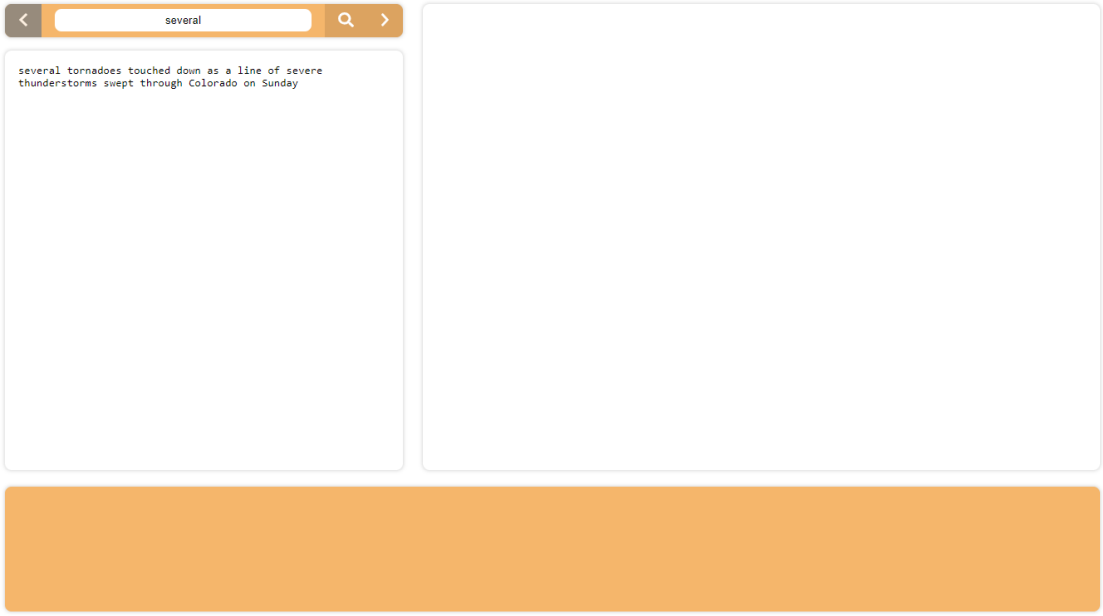


Figura 6. Texto processado e primeira palavra selecionada

A **figura 6,** mostra um campo que contém o texto extraído do áudio, um campo que contém a palavra selecionada (primeira palavra no caso) que será pesquisada futuramente, a palavra selecionada está entre três botões, um para avançar para a próxima palavra, outro para voltar para palavra anterior e o outro para iniciar a busca pelas imagens. Contém também um campo onde será mostrado as imagens que serão usadas após clicarmos no botão pesquisar e um campo onde será mostrado as imagens que foram selecionadas.

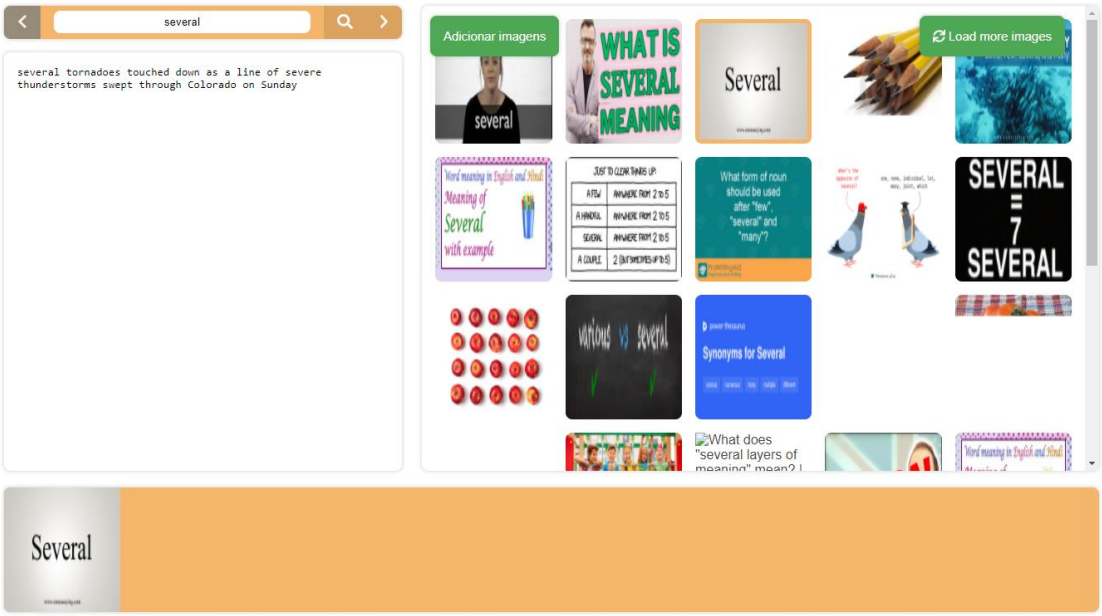


Figura 7. Texto processado e primeira palavra selecionada

A **figura 7,** mostra os resultados da busca da primeira palavra selecionada (este resultado é mostrado a direita, do texto do áudio) além de mostrar a imagem que

pretendemos selecionar e em baixo (no rodapé) é mostrado a imagem que fora selecionado.

O áudio carregado, contém o seguinte conteúdo. *"several tornadoes touched down as a line of severe thunderstorms swept through Colorado on Sunday lef"*.

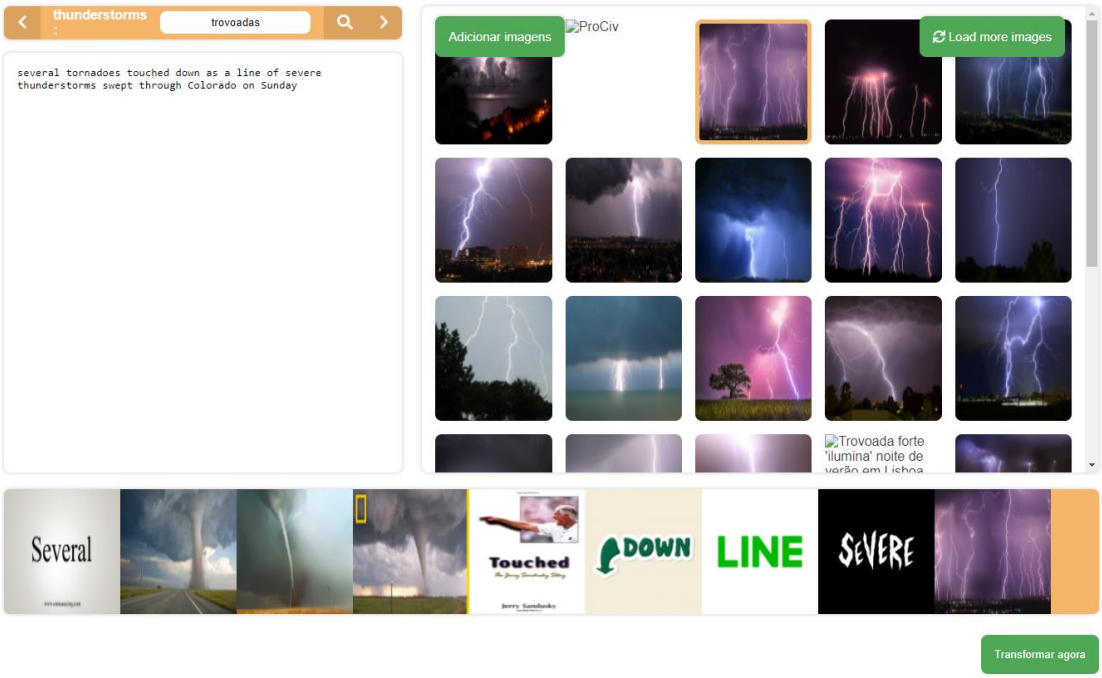


Figura 8. Texto processado e primeira palavra selecionada

A **figura 8,** mostra no campo da palavra selecionada a possibilidade de alternarmos a busca da palavra atual por um sinônimo ou por uma palavra que acharmos adequada, no caso pesquisamos por **trovoadas** ao invés de **thunderstorms**, e também mostra no canto inferior direito o botão para a geração de vídeo com o texto **transformar agora**.

Figura 9. Texto processado e primeira palavra selecionada

A **figura 9,** mostra a possibilidade de visualizarmos uma imagem caso queiramos, clicando na mesma imagem.

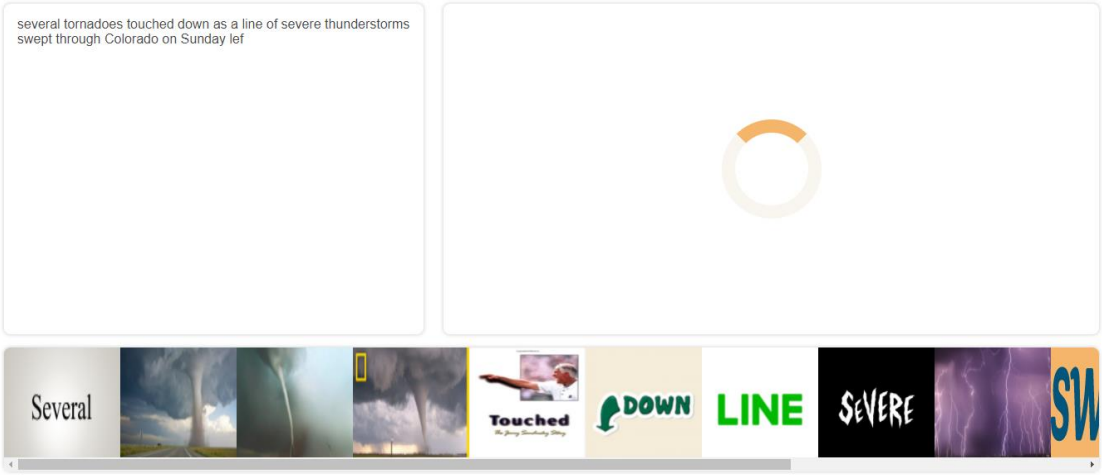
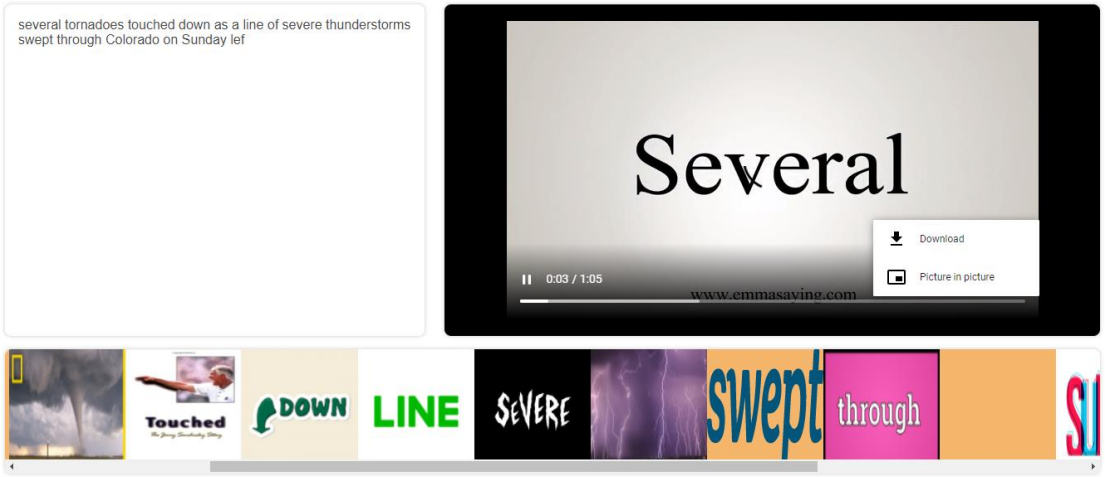


Figura 10. Texto processado e primeira palavra selecionada

Após clicarmos em transformar o vídeo é processado, a figura 10, **mostra o campo onde o vídeo será mostrado processando. Figura 11. Vídeo processando**



A **figura 11,** mostra o vídeo processado.

8. Análise e Discussão dos resultados

Após testes realizados durante a implementação do projeto, pudemos notar alguns pontos que iremos comentar abaixo.

8.1. Falhas e Possíveis Soluções

8.1.1. Falhas

- Algumas URL's de imagens quebram e provocam falhas no processamento de vídeo fazendo com que o vídeo não seja gerado.

- Baixa conexão de internet pode provocar falhas no processo de aquisição dos dados por parte do servidor.
- Diferentes formatos de imagens podem provocar falhas

8.1.2. Possíveis soluções para algumas falhas por implementar

Esta subsecção trata de possíveis formas que podem ser usadas para tentar ultrapassar ou resolver as falhas encontradas, não passam de um conjunto de teorias não provadas. As possíveis soluções são:

- Uma possível solução para a falha das url's é garantir no front que as imagens estejam em condições fazendo o upload na nuvem da pasta com as imagens todas.
- Outra possível solução para a falhas das url's das imagens é fazer o upload das imagens na nuvem antes de passar para o servidor ao invés de trabalhar com as url's de cada imagem.

8.2. Melhorias

Um sistema informático dificilmente está terminado e o nosso não é diferente da maioria, possuindo algumas melhorias que serão feitas ao longo do tempo:

- Correção das falhas
- Suportes outros idiomas diferentes do inglês
- Possibilidade de reordenar as imagens já selecionadas
- Implementação de algoritmo para melhorar o conjunto de palavras mais eficientes (Mining text algorithms) e obtermos melhores resultados na busca por imagens
- Implementação de algoritmo para facilitar o usuário na busca por sinônimos, mostrando os sinônimos que melhor se adequem com a palavra
- Melhorias na interface do usuário

8.3. Tempo

O tempo consumido pela aplicação depende de quatro fatores, a velocidade de internet, o tamanho do áudio (quanto maior o áudio, maior o tempo que a aplicação vai levar para processa-lo), da agilidade do utilizador na tomada de decisões (embora este seja um fator irrelevante) e do vídeo a ser processado (quanto maior o volume de informações maior será o tempo consumido.

8.4. Consumo de recursos

Desde o tamanho do áudio, até a quantidade de imagens que serão processadas podemos ter uma grande quantidade de dados como uma ínfima quantidade se o tamanho de informação for menor.

8.5. Comparação com a concorrência

Conforme mostra a tabela a baixo, podemos ver que todas as aplicações conseguem processar áudio tanto do microfone quanto do armazenamento do dispositivo, e que

dentre elas, só o **Canva Slide** não é capaz de processar áudio para texto, porém somente o **Speech to Video** é capaz, permite a busca de imagens por intermédio do texto processado. Contudo todas elas permitem termos um vídeo como resultado final do nosso processamento de informações.

Tabela 1. Variables to be considered on the evaluation of interaction techniques

	Entrada de áudio do armazenamento	Gravação de áudio	Áudio para texto	Pesquisa de imagens por palavra	Criação de vídeo
Speech to Video	Sim	Sim	Sim	Sim	Sim
Google Slides	Sim	Sim	Sim	Não	Sim
Adobe Spark	Sim	Sim	Sim	Não	Sim
Canva Slides	Sim	Sim	Não	Não	Sim

9. Portabilidade da aplicação

A plataforma pode ser executada em qual ambiente seja Linux, Windows, Mac OS ou em dispositivos moveis. Sendo uma aplicação web, ele é simplesmente limitado pela capacidade dos navegadores.

Porém como a nossa plataforma suporta diferentes formas de entrada de dados, tanto pelo microfone do computador ou smartphone tanto quanto a partir da seleção de um ficheiro de áudio da memória de armazenamento do dispositivo, faz dela capaz de rodar ou funcionar caso o navegador não de suporte a entrada de dados por microfone.

Para que o utilizador tenha uma melhor experiencia com a plataforma, recomendamos a utilização de um dos navegadores que se segue:

9.1. Plataformas Windows

- Google Chrome última versão
- Microsoft Edge última versão
- Mozilla Firefox última versão

9.2. Plataforma Linux

- Mozilla Firefox

9.3. Plataforma Mac OS

- Safari última versão
- Google Chrome última versão

9.4. Dispositivos Android

- Google Chrome

- Edge
- Samsung Browser
- Mozilla Firefox

9.5. Iphones

- Safari
- Google Chrome
- Mozilla Firefox

10. Melhoria das funcionalidades

Durante a terceira fase pudemos melhoras ou implementar as seguintes funcionalidades:

- Integração do áudio com vídeo
- Suporte a ficheiros .mp3

11. Persistência de dados

Quanto a persistência de dados, agora a nossa aplicação é capaz de armazenar dados (áudio e o vídeo) na nuvem.

Como mencionamos antes, utilizamos o C**loudnary** para o armazenamento e persistência de dados. A 3.2. Armazenamento em nuvem, explica o processo de funcionamento do **Cloudnary**, em casos de dúvida, revise a secção.

12. Hospedagem

Infelizmente, para nós não foi possível a hospedagem da aplicação, sendo que ela é composta por duas partes, cliente e servidor, os sites que encontramos requisitavam a entrada de dados de um cartão de divisas internacional (como visa), para que posteriormente possa ser feita a hospedagem da API.

Devido isso, não conseguimos hospedar a aplicação, pois só uma parte que estaria na hospedada (parte do cliente) e não poderia funcionar devido à ausência do servidor.

13. Fator hibrido ou nativo

De acordo a essas definições, um app **nativo** costuma a ser desenvolvido para uma plataforma especifica, explorando assim todas as potencialidades daquele sistema operacional. O **hibrido**, no entanto, consegue unir o nativo com a linguagem da web, podendo ser utilizado tanto em um como no outro, podemos dizer que a nossa solução não é hibrida nem nativa pois ela não foi desenvolvida para uma plataforma especifica e nem requer de recurso específicos de uma plataforma para que funcione. Os recursos que por ela são requisitados, fica de responsabilidade dos navegadores.

14. Trabalhos pendentes

Quanto aos trabalhos pendentes, a que dizer que antes de tudo, que conseguimos construir o básico da aplicação (processar áudio, pesquisar imagens e formar vídeo com

o áudio integrado), mas infelizmente não podemos terminar o projeto (no que tange a melhoria da aplicação), deixando assim funcionalidades e recursos pendentes que tornariam melhor a experiência do usuário com a plataforma, também não conseguimos corrigir todos os erros da plataforma, deixando assim para a próxima versão.

Em suma, a 8.1. Falhas **e Possíveis Soluções** e 8.2. Melhorias explicam de uma forma mais detalha o que não foi feito e corrigido.

15. Guia de utilização

O facto da plataforma, ser constituída por dois lados, **cliente e servidor**, infelizmente fez com que não conseguíssemos colar o projeto no ar, tornando possível simplesmente acessa-lo a partir do modo de desenvolvimento (as secções **15.1. Requisitos a nível de servidor** e **15.2. Requisitos a nível do cliente** explicam como acessa-las), isso porque a hospedagem do server exigia-nos custos adicionais, ou seja, para que pudéssemos hospedar o server o grupo ou um dos elementos do grupo teria que ter um cartão internacional (visa, PayPal) isso porque os sites que forneciam sistemas de hospedagem grátis como **Amazon Aws, Microsoft Azure**, precisam que o utilizador tenha um cartão para podermos fazer o que chamamos de Billing que consiste em conectar a conta com um cartão de credito internacional para que possamos usufruir dos recursos grátis e pagos fornecidos pelos memos.

Já, quanto ao **cliente** o cenário foi diferente isso porque existe muitas empresas que fornecem serviços de hospedagem grátis sem a burocracia do cartão de crédito (impossibilitando simplesmente o usuário de lucrar com a plataforma caso esteja utilizando o pacote grátis) empresas como a Netlify e a Vercel, porém, nós optamos por não hospedar o cliente, já que o mesmo não funciona sem que esteja conectado ao servidor, ou seja, se hospedássemos o cliente, seria como fornecer aos consumidores um carro sem motor ou um computador sem processador, tornando a aplicação inútil para o cliente. Com isso vimos que seria melhor não hospedarmos o mesmo até que futuramente consigamos colocar o pacote completo no ar.

15.1. Acessando a plataforma

Para ser possível o acesso a plataforma, a partir do modo de desenvolvimento, é preciso ter a certeza que possui os seguintes componentes em sua máquina:

- **NodeJs 14 ou superior**
- **Git para controle de versão** opcional

Após ter a certeza que possui os requisitos acima mencionados, para adquirir o código fonte do projeto siga um dos passos a seguir:

- Caso não tenhas o Git instalado, **baixe o projeto** a partir do repositório.
- Caso tenhas o Git instalado, abra um terminal e execute o seguinte comando: **git clone https://github.com/Dalcio/speech-to-video.git**

15.1. Requisitos a nível de servidor

Realizando o que foi dito acima, para que consigas rodar o servidor de modo funcional, é preciso garantir que possuas em sua máquina os seguintes requisitos:

- FFmpeg **Guia de instalação para Windows,** ou procurar o equivalente para sua plataforma.
- Em seguida abra o terminal no diretório root do projeto e rode os seguintes comandos na ordem:
- **cd server**
- **npm install**
- **npm start.**

Com isso, o servidor estará rodando em sua máquina, na porta 8080 ou em outra a ser definida caso esta esteja ocupada.

15.2. Requisitos a nível do cliente

Para o acesso a interface (cliente) da plataforma, é preciso seguir os seguintes passos:

- Abra um outro terminal no diretório root do projeto
- Rode **yarn install**
- E em seguida **yarn dev**

O cliente rodara na porta **http://localhost:3000/pt**, portando abra um navegador e cole o link

16. Conclusão

A crescente evolução das tecnologias, recursos como a inteligência artificial, processamento de mídia, tem facilitado o processo de automação de sistemas multimidia, aumentando o leque de desenvolvimento de novas soluções e de forma mais criativas e inovadoras.

Dessa maneira, foi possível concebermos devido a utilização de alguns dos frutos de evolução, como Speech Recognition e Ffmpeg, produzir uma tecnologia capaz de processar áudio em texto e produzir como resultado um vídeo devido a varredura por imagens que correspondem ao contexto do áudio.

Referências

“IBM Watson Documentation”, https://cloud.ibm.com/docs

“IBM Speech to text”, https://cloud.ibm.com/apidocs/speech-to-text?code=node

“FFmpeg Documentation”, https://www.ffmpeg.org/

“Cloudnary Documentation”, https://cloudinary.com/documentation

“Media Record Repository”, https://github.com/0x006F/react-media-recorder#readme

“What is cloud storage?”, https://aws.amazon.com/pt/what-is-cloud-storage/

“Canva Slides”, https://www.canva.com/

“Google Slides”, https://www.google.com/slides/about/

"Adobe Spark Home Page", https://spark.adobe.com/sp/

"Adobe Voice Show your story on the App Store", https://apps.apple.com/us/app/adobe-voice-show-your-story/id852555131

“Speech to Text – Repositório do Projeto”, https://github.com/Dalcio/speech-to-video