Robert Bosch Ltda

Hackathon ETS 8° Edição

Gabriel Resende da Silva

Geovanna Souza Santos

Joyce Kelly de Souza Santos

Pedro Alves Estevam Magaieski

Raphaela Tavares Fonseca

Relatório Técnico do Projeto BFlash

Campinas

2025

Sumário

Introdução

O projeto “BFlash”, visa ser uma ferramenta IA que gera resumos e criação de atas para gravações de reuniões e vídeos. Este projeto busca não apenas ser uma ferramenta para resumo e criação de atas, mas também otimizar o tempo do usuário, tendo o resumo de um vídeo de minutos/horas em questão de segundos, com fidelidade ao assunto tratado.

Visão Geral

A plataforma “BFlash” é uma ferramenta baseada em inteligência artificial desenvolvida para facilitar a criação de resumos e atas de reuniões e vídeos gravados. Seu principal objetivo é reduzir o tempo necessário para revisar conteúdos longos, oferecendo resumos concisos e precisos em questão de segundos. A solução é especialmente útil para profissionais que necessitam otimizar suas agendas e garantir que informações chave sejam rapidamente acessadas. Com um foco na fidelidade ao conteúdo original, o “BFlash” busca proporcionar uma experiência eficiente, ágil e de alta qualidade na gestão de informações, tornando-se um aliado valioso no cotidiano dos colaboradores da ETS.

Problemática

Os colaboradores possuem muitas das vezes agendas cheias com reuniões estratégicas e de importância o tempo todo. Se uma reunião de urgência for marcada em um horário que o colaborar já está com outra reunião marcada, ele teria que se ausentar desta reunião e dar preferência a de urgência, o que faria com que o colaborador independentemente do nível de importância da reunião, perdesse assuntos tratados e pontos importantes. A reunião é gravada do início ao fim, fazendo com que muitas das vezes o colaborador não tenha o tempo devido para assisti-la completa.

Objetivos

1. Transcrever vídeos

- O vídeo deve transcrito e resumido, e conforme baseado em feedbacks o BFlash se torne cada vez melhor e preciso.

2. Identificar integrantes

- Os integrantes da reunião devem ser identificados para melhor compreensão do que foi falado em reunião/vídeo.

3. Funções do usuário

- Pode fazer login ou caso não tenha uma conta, pode se cadastrar;
- Adicionar título e data para ficar salvo no histórico
- Pode adicionar um vídeo ou reunião no campo especificado e ter seu resumo;

Plataformas Utilizadas

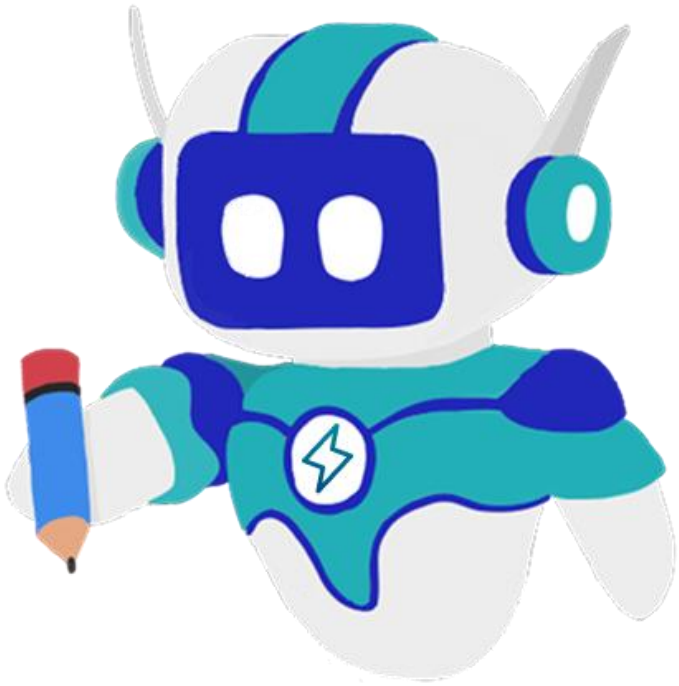
- Figma
- Canva
- Krita
- VS Code (Visual Studio Code)
- Word

Identidade Visual BFlash

1. Design logo

- O nome “BFlash” surgiu incialmente de B (Bosch) e Flash (Rápido), mas atualmente o B se refere ao verbo Be (ser) em inglês e o Flash (Rápido) ou seja, Ser Rápido. Do qual faz parte das intenções do projeto em entregar a solução rápido para o usuário e economizar tempo.

2. Mascote



- Nosso mascote se chama Ray (Raio), ele foi criado e desenvolvido na plataforma Krita.

3. Paleta de cores

- Código cores:

046AB3

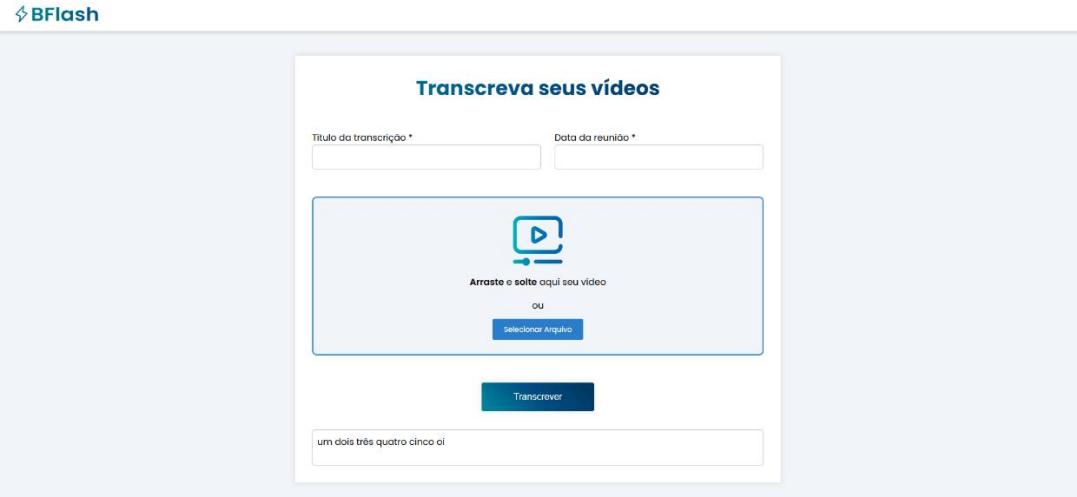
008B9E

01365B

4. Telas

- Tela BFlash

- Tela Home



Front-End

O front-end do site é composto por duas páginas principais:

1. Página Informativa (Home)

- Função: Esta página serve como uma introdução ao site, fornecendo informações gerais sobre o que o site oferece.
- Conteúdo: Texto explicativo sobre o site, seus objetivos e funcionalidades. Também há, uma imagem do mascote.

- Elementos:
    - Título e descrição do site.
    - Botão que redireciona para a página principal (home.html).

2. **Página Principal (home.html)**

- Função: Esta página é a principal, onde os usuários interagem com o conteúdo ou funcionalidades do site.
- Conteúdo:
    - Barra de menu com logo.
- Elementos:
    - Menu de Navegação (Logo BFlash):
        - Logo que redireciona para a página "BFlash".
    - Formulário:
        - Campos para inserção de dados (título da reunião, data da reunião, local para depositar os arquivos);
        - Campo de texto invisível, onde aparecerá somente quando a transcrição for gerada;
        - Botão de envio.

3. Estrutura do HTML

BFlash.html

C: > Users > Administrador > Desktop > BFlash.html > ...

```html
<!DOCTYPE html>
<html lang="pt-br">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>BFlash - Transcrições de vídeos</title>
    <link rel="stylesheet" href="styles/style.css">
</head>
<body id="body-BFlash">

    <!-- Barra de menu -->
    <div class="barra-menu">
        <a href="home.html">
            <img src="imagens/logo.png" alt="">
        </a>
    </div>


    <!-- Conteúdo principal -->
    <main class="conteudo-principal">
            <form action="" class="formulario-container">

                <h1 class="titulo-transcricao test">Transcreva seus vídeos</h1>

                <div class="titulo-container">
                    <label for="titulo">Título da transcrição *</label>
                    <input type="text" id="titulo">
                </div>

                <div class="data-container">
                    <label for="data">Data da reunião *</label>
                    <input type="text" id="dataInput" maxlength="10">
                </div>

                <div class="upload-area">
                    <div class="test">
                        <img src="imagens/marketing-de-video.png" alt="">
                        <p><span>Arraste</span> e <span>solte</span> aqui seu vídeo</p>
                    </div>

                    <p>ou</p>

                    <div>
                        <label for="fileInput">Selecionar Arquivo</label>
                        <input type="file" id="fileInput" style="display: none;" accept="video/mp4, audio/mpeg" />
                    </div>

                </div>

                <textarea class="textarea-container" placeholder="Sua transcrição será gerada aqui..." readonly></textarea>

                <div class="botao-enviar" >
                    <button type="submit">Gerar transcrição</button>
                </div>
            </form>
    </main>

    <script src="scripts/main.js"></script>
</body>
</html>
```

Back-End

O backend consiste em um modelo que recebe um arquivo de áudio e retorna um texto, este modelo é inserido em um formulário que posteriormente interage com o formulário HTML do frontend.

Esse modelo então passa pela view(controla alterações no modelo) onde existem funções para receber o arquivo de áudio que veio no upload e devolver o texto gerado a partir de uma outra função da api Vosk que transcreve o arquivo passado como parâmetro para essa função.

1. Upload de aúdio

```python
def upload_audio(request):
    transcription = None
    if request.method == 'POST':
        form = TranscriptionForm(request.POST, request.FILES)

        if form.is_valid():
            transcription = form.save(commit=False)
            transcription.save()
            audio_path = os.path.join(settings.MEDIA_ROOT, transcription.audio_file.name)

            # Convert audio file to WAV format if necessary
            audio_format = transcription.audio_file.name.split('.')[-1]
            if audio_format not in ['wav']:
                sound = AudioSegment.from_file(audio_path)
                audio_path = audio_path.replace(audio_format, 'wav')
                sound.export(audio_path, format='wav')

            # Perform transcription
            try:
                transcription.text = transcribe_audio(audio_path)
                transcription.save()
            except Exception as e:
                return render(request, 'transcriber/upload_audio.html', {
                    'form': form,
                    'error': f"Could not transcribe audio; {e}"
                })
    else:
        form = TranscriptionForm()
    return render(request, 'transcriber/BFlash.html', {'form': form, 'transcription': transcription})
```

2. Transcrição de vídeos/reunião

```python
    # Transcrever audio com argumento de caminho de arquivo
def transcribe_audio(file_path):

    #Inicializar reconhecimento de fala com frame rate la de cima

    #Ler arquivo de audio que foi passado como argumento r-read b-binary (le o arquivo em forma binaria)
    wave_file = wave.open(file_path,"rb")

    #Tratativa de canais de audio, largura da amostra e frequencia da amostra
    if wave_file.getnchannels() != 1:
        raise ValueError("O arquivo de audio deve ser de formato  mono")
    elif wave_file.getsampwidth() != 2:
        raise ValueError("O arquivo de audio deve ser de formato WAV")
    elif wave_file.getcomptype() != "NONE":
        raise ValueError(f'O arquivo de audio deve ser de formato PCM(pulse code modulation) ')
    #inicializa modelo de treino de transcrição em portugues
    model = Model(model_name="vosk-model-small-pt-0.3")

    #reconhecimento de fala comparando modelo de treino com arquivo de entrada com sua respectiva taxa de quadros
    recognizer = KaldiRecognizer(model,wave_file.getframerate())
    recognizer.SetWords(True)
    recognizer.SetPartialWords(True)
    #inicializa variavel que recebe texto gerado pela transcricao
    transcription = ""
    # Realizar reconhecimento de fala
    while True:
        # Le 4000 frames(cerca de 133,3 segundos ou 2 min) de audio e adiciona a variavel dados
        dados = wave_file.readframes(4000)
        # Se não ouver dados em dados, parar de ler
        if len(dados) == 0:
            break
        #passa dados pelo reconhecimento de fala
        if recognizer.AcceptWaveform(dados):
            #cria um resultado da transcrição em json
            result = recognizer.Result()
            #transforma o json em um dicionario python
            result_dict = json.loads(result)
            #adiciona o texto do dicionario ao final da variavel transcricao
            transcription += result_dict.get("text","") + " "
            print(transcription)
        # devolve a transcrição final formatada
    return transcription.strip()
```

IA utilizada

A IA utilizada no desenvolvimento do projeto foi a IA Vosk que é uma biblioteca de reconhecimento de fala (speech-to-text) de código aberto, que permite transcrever áudio em texto. Ele é muito eficiente para ser usado em dispositivos com recursos limitados, como dispositivos móveis e embarcados, pois não requer uma conexão com a internet para realizar as transcrições.

A IA Vosk suporta múltiplos idiomas e pode ser usado para transcrever áudios de diferentes fontes, como gravações de voz, que era o que precisávamos para pegar o aúdio do vídeo e transcrevê-lo.

Público-Alvo

Nosso público-alvo são os colabores da ETS que muitas das vezes por conta de sua rotina acabam não tendo tempo para participar ou assistir as gravações de reuniões e vídeos a parte.

Então a utilização do BFlash irá suprir a dor desta área e dos colaboradores sendo mais prático, rápido, eficiente, ágil e confiável.

Conclusão

O BFlash se apresenta como uma solução inovadora que utiliza a inteligência artificial para resumir reuniões e vídeos facilitando e otimizando o tempo dos colaboradores. Ao oferecer a geração de resumos e atas de forma rápida e precisa, o projeto não apenas otimiza o tempo dos usuários, mas também assegura que os conteúdos essenciais sejam capturados com fidelidade. Com isso, o BFlash torna-se uma ferramenta indispensável para aqueles que buscam eficiência e agilidade na organização de dados oriundos de gravações de reuniões e vídeos.