UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO

FACULDADE DE CIÊNCIAS

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

GUILHERME GUMIERO DA NÓBREGA

LUCAS RAMOS DOMINGUES

MOODSTRINGS

Bauru

2025

UNIVERSIDADE ESTADUAL PAULISTA JÚLIO DE MESQUITA FILHO

FACULDADE DE CIÊNCIAS

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

GUILHERME GUMIERO DA NÓBREGA
LUCAS RAMOS DOMINGUES

MOODSTRINGS

Trabalho de conclusão do Curso de Bacharelado em Sistemas de Informação da Faculdade de Ciências – UNESP Campus Bauru.
Orientador: Dr. Leandro Aparecido Passos Junior

Bauru
2025

Nóbrega, Guilherme Gumiero da.
Moodstrings / Guilherme Gumiero da Nóbrega, Lucas Ramos Domingues. – Bauru, 2025
62 f. : il.

Trabalho de conclusão de curso (Bacharelado – Sistemas de Informação)-Universidade Estadual Paulista (UNESP), Faculdade de Ciências, Bauru
Orientador: Leandro Aparecido Passos Junior

1. Inteligência artificial. 2. Música. 3. Emoções na música. 4. Análise harmônica. 5. Análise de áudio. I. Domingues, Lucas Ramos. II. Universidade Estadual Paulista. Faculdade de Ciências. II. Título.

AGRADECIMENTOS

Agradecemos primeiramente às nossas famílias pelo apoio constante ao longo de toda nossa trajetória acadêmica, em especial aos nossos pais pelo incentivo, compreensão e presença em cada etapa deste trabalho.

Ao nosso orientador, Dr. Leandro Aparecido Passos Junior pela orientação cuidadosa, disponibilidade e contribuição nos campos tecnológicos, musicais e pessoais durante o desenvolvimento da pesquisa.

Agradecemos também à Universidade Estadual Paulista (UNESP) pela oportunidade de formação e pelos recursos oferecidos durante o curso que serviram de base para a pesquisa.

Por fim, agradecemos aos nossos amigos e colegas que, de maneira direta ou indireta, nos apoiaram e contribuíram para a nossa formação.

RESUMO

A teoria musical desempenha um papel significativo na experiência psicológica de quem aprecia música, sejam apenas ouvintes ou compositores, principalmente em suas emoções. Elas estão diretamente relacionadas à harmonia, à progressão de acordes e às escolhas melódicas desenvolvidas com o intuito de provocar alguma reação no público. Com o avanço tecnológico, a área musical pouco tem sido impactada pela inteligência artificial (IA), principalmente devido à sua complexidade: a criação de um trecho musical, muitas vezes é carregado de sentimentos e criatividade do compositor, algo que um sistema treinado não é capaz de transmitir, limitando-se apenas às regras presentes na teoria musical. Neste trabalho, exploramos a aplicação de um sistema capaz de analisar arquivos MIDI (Musical Instrument Digital Interface), um formato largamente utilizado na música digital que contém instruções dos instrumentos musicais. Essa análise é capaz de identificar informações como BPM (batidas por minuto), acordes tocados, tonalidade e progressão harmônica. Com base nesses dados, e com o uso de um modelo de IA treinado, é possível inferir as emoções associadas ao trecho musical trabalhado, enriquecendo o repertório do músico que busca provocar sentimentos em suas obras. O objetivo principal deste estudo é democratizar o acesso ao conhecimento musical, além de promover a integração da inteligência artificial como uma aliada no processo criativo, educativo e técnico de músicos em diferentes níveis de experiência.

Palavras-chave: inteligência artificial; música; emoções na música; análise harmônica; análise de áudio

ABSTRACT

Musical theory plays a significant role in the psychological experience of those who appreciate music, whether they are merely listeners or composers, particularly in relation to emotions. Emotions are directly associated with harmony, chord progressions, and melodic choices developed with the intent of provoking a reaction in the audience. Despite technological advancements, the field of music has been minimally impacted by artificial intelligence (AI), mainly due to its inherent complexity: the creation of a musical piece is often infused with the composer's emotions and creativity, elements that a trained system is not yet capable of conveying, as it is limited to following the rules of musical theory. In this study, we explore the application of a system capable of analyzing MIDI (Musical Instrument Digital Interface) files, a format widely used in digital music that contains musical instructions for instruments. This analysis is capable of identifying information such as BPM (beats per minute), chords played, key signature and harmonic progression. Based on these data and using a trained AI model, it becomes possible to infer the emotions associated with the analyzed musical excerpt, enriching the repertoire of musicians who aim to evoke feelings through their works. The main goal of this study is to democratize access to musical knowledge, as well as to promote the integration of artificial intelligence as a powerful ally of the creative, educational, and technical processes of musicians at different levels of experience.

Keywords: artificial intelligence; music; emotions in music; harmonic analysis; audio analysis

SUMÁRIO

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE FIGURAS

LISTA DE TERMOS E SIGLAS

API

Uma API (Interface de Programação de Aplicações) é um conjunto de regras que permite a comunicação entre diferentes sistemas ou *softwares*. Ela define como as funcionalidades de um programa podem ser acessadas por outro, facilitando integrações.

BACKEND

Camada lógica e estrutural de um sistema que processa regras de negócio, acessa banco de dados e fornece dados ao frontend.

DATASET

Nome inglês para "conjunto de dados". Faz referência ao grupo de dados que são usados para treinamento num modelo de IA.

DOWNLOAD

Processo de transferência de dados de um servidor ou rede para um dispositivo local do usuário.

FRONTEND

Camada visual e interativa de um sistema acessada pelo usuário final, responsável pela interface e experiência de uso no navegador ou aplicativo.

IA

Abreviação de Inteligência Artificial.

LIB

Abreviação em inglês de "*library*". Usado para fazer referência às bibliotecas que são importadas durante o desenvolvimento.

MIDI

Um arquivo MIDI (Musical Instrument Digital Interface) é um tipo de arquivo que contém instruções para reproduzir música digitalmente, como notas, tempo e instrumentos. Ele não

armazena áudio real, mas sim comandos que podem ser interpretados por sintetizadores ou *softwares* musicais.

SCRIPT

Script é um conjunto de instruções escritas em uma linguagem de programação ou de automação de tarefas, o qual executa comandos ou controla o comportamento de um sistema ou aplicação.

SOFTWARE

Palavra em inglês para referir-se ao sistema. Algo desenvolvido por meio da computação e programação

UX

UX (*User Experience*) refere-se à experiência geral do usuário ao interagir com um produto, sistema ou serviço. Envolve aspectos como usabilidade, acessibilidade e satisfação, buscando tornar a interação intuitiva e agradável.

WEB

Conjunto de tecnologias e serviços que permitem acesso e navegação a conteúdos e aplicações por meio da internet, usando protocolos como HTTP.

1 INTRODUÇÃO

A música é uma das formas de expressão mais antigas da humanidade, presente no cotidiano com a mesma naturalidade que atos essenciais, como respirar. Muitos usam-na como válvula de escape, outros como entretenimento e até como trabalho. O comum entre todos é que, de qualquer forma e qualquer seja o trecho ouvido, algum sentimento será provocado. A relação entre música e emoção pode ser interpretada por cada um, e, mais do que provocar emoções, a música funciona como um espelho para as emoções que projetamos nela (JUSLIN; SLOBODA, 2001).

O avanço tecnológico da inteligência artificial (IA) está se agregando em muitos campos de estudo nos dias atuais, inclusive na Música (como ciência). Contudo, a utilização de mecanismos de IA ainda não foram tão explorados para analisar certas características musicais, como a interpretação subjetiva da música e as emoções que ela proporciona. Um exemplo é a geração de música por algoritmos inteligentes que tem demonstrado potencial para a criação de composições com coerência e variabilidade, mesmo sem intervenção humana. De acordo com a pesquisa, abordagens como algoritmos genéticos conseguem capturar padrões musicais e gerar peças que seguem estruturas rítmicas e harmônicas reconhecíveis, embora ainda enfrentam desafios no que diz respeito à expressividade e à emoção presentes na música humana (LEE; AGRES; HERREMANS, 2013).

A evocação de emoções na música está diretamente relacionada à teoria musical, cujas regras oferecem aos compositores recursos estruturais para modular a resposta emocional do ouvinte. Dentro das regras presentes na teoria musical, como cadência e tonalidade, são fatores determinantes para limitar o rol de sentimentos possíveis num trecho composto. Porém, a harmonia, principalmente a progressão de acordes, e o modo (maior e menor) são dois dos principais elementos que agregam nessa experiência. Segundo Dean *et al*. (2023), diferentes níveis de complexidade harmônica e variações no tempo musical são capazes de modular significativamente as emoções percebidas, indicando que a progressão de acordes e o andamento exercem papel fundamental na resposta emocional à música. Essa constatação reforça a importância de analisar a estrutura musical como possíveis gatilhos emocionais, especialmente em contextos computacionais, como sistemas inteligentes de reconhecimento de emoção.

Umas das muitas formas de análise estrutural de uma composição musical é por meio da análise de um áudio de formato MIDI (*Musical Instrument Digital Interface*). Esse tipo de arquivo é largamente utilizado em música digital, uma vez que representa uma estrutura simbólica da música, contendo informações detalhadas como notas, duração e intensidade.

Com essas informações detalhadas, é possível determinar o BPM (batidas por minuto), progressão de acordes, cadência e muitas outras informações relevantes. Modelos como o MusicBERT, desenvolvido pela Microsoft Research, demonstram a eficácia de utilizar grandes coleções de dados MIDI para treinar redes neurais profundas capazes de compreender a estrutura musical em alto nível, incluindo tarefas como classificação de gêneros, predição de notas e geração automática de conteúdo musical. Essa abordagem mostra como os dados simbólicos da música podem ser explorados por modelos linguísticos inspirados em NLP para capturar padrões musicais complexos (ZHANG *et al*., 2022).

Tais informações se mostram relevantes para os músicos, uma vez que com elas, são capazes de compôr novas melodias. Além disso, o processo criativo desses artistas pode ser facilitado com o uso da aplicação, uma vez que com ela ele será capaz de enviar seu próprio áudio para análise e obter a correspondência da emoção presente no trecho com a emoção que ele, como profissional, busca projetar em seus ouvintes.

Dado as possibilidades, a pesquisa propõe encontrar e classificar as emoções *angry*, *romantic*, *happy* e *sad* (irritado, romântico, feliz e triste) num presente trecho musical somando a progressão (sequência de acordes) com o modo (menor e maior). Essas informações serão extraídas com a ajuda de um sistema inteligente capaz de transcrever arquivos mp3 para o formato MIDI.

O desejo de implementar inteligência artificial com música era mútuo, assim, surgiu a oportunidade de provar que, baseado em informações harmônicas, principalmente progressão de acordes, era possível classificar e identificar uma emoção presente. O campo da Música é, apesar de suas características matemáticas e normativas complexas, muitas vezes subjugado à emoção e liberdade artística de um compositor. Então, promover um projeto que comprovasse a coligação entre emoção e música, é um enriquecimento à estes músicos, mesmo que a própria provocação de sentimento possa ser interpretada de forma única em cada indivíduo. Desse modo, a pesquisa busca explorar a correlação entre informações melódicas de um trecho musical com a emoção humana presente e contribuir para a democratização ao ensino musical com a pesquisa e *software* públicos.

1.1 Objetivos

1.1.1 Objetivo geral

Desenvolver uma aplicação que por meio da análise e transcrição de arquivos de áudio mp3 e análise de MIDI, extraia informações e identifique as emoções envolvidas no trecho reconhecido, visando explorar a correlação entre tecnologia e sentimentos.

1.1.2 Objetivos específicos

1. Empregar inteligência artificial como ferramenta para musicistas de diferentes níveis;
2. Explorar a conexão entre tecnologia e sentimentos humanos;
3. Descomplicar a teoria musical por meio da análise de arquivos MIDI e extração de informações relevantes para compositores;
4. Aplicar algoritmos simples capazes de reconhecer acordes numa sequência gravada;

1.2 Justificativa

Com o avanço tecnológico, especialmente da inteligência artificial, tem-se proporcionado novos horizontes para diversas áreas do conhecimento. Desse modo, a interconexão dessa ferramenta com o campo musical é ousada e potente, seja que essa junção ainda é emergente, seja a complexidade da aceitação do mesmo, a música com IA pode ser relevante.

Quanto se trata de música, a teoria musical é vista como complexa e muitas vezes inacessível para iniciantes. Porém, ela pode ser descomplicada por meio da visualização e análise automática de arquivos de áudio, em especial os arquivos MIDI. A transcrição desses conteúdos e a extração de informações relevantes da estrutura harmônica de maneira clara e objetiva promovem a democratização do acesso ao aprendizado musical, favorecendo estudantes e músicos de diferentes níveis técnicos e sociais. Em consequência disso, há a formação de novos talentos musicais capazes de inovar na área com o auxílio dessa ferramenta.

O emprego da inteligência artificial como ferramenta de apoio no processo criativo de um musicista representa uma mudança significativa na maneira como as músicas podem ser analisadas e também compostas. Ao explicitar quais as emoções presentes numa certa estrutura harmônica, a pesquisa expande o leque de possibilidades presentes num processo criativo de um compositor, abrindo novos caminhos para formas de experimentação musical.

Esse ponto pode favorecer desde compositores iniciantes até profissionais que buscam explorar ainda mais esse lado emotivo da composição.

Por fim, o sistema também contribui diretamente para campos acadêmicos que variam entre psicologia, computação e música. A análise da estrutura harmônica e sua correlação com as emoções envolvidas permite a compreensão dos mecanismos cognitivos e afetivos envolvidos na percepção musical. A pesquisa pode fornecer informações relevantes para próximos estudos, como a influência da música nos estados emocionais humanos, como a musicoterapia (BENENZON, 1989), bem como o refinamento da contribuição da IA na modelagem e interpretação de emoções, tornando-se mais que uma ferramenta de apoio, mas uma fonte de conhecimento e geração de novas pesquisas científicas interdisciplinares.

1.3 Hipótese de pesquisa

A hipótese central deste projeto é que algoritmos de análise de áudio, agregado à modelos treinados de inteligência artificial, são capazes de extrair informações relevantes presentes na estrutura harmônica (como tonalidade, progressões e BPM) via o tipo de arquivo MIDI, e relacioná-las com emoções percebidas, possibilitando uma nova abordagem para aprendizado musical.

2 SOLUÇÕES EXISTENTES

Durante a fase de pesquisa do projeto, algumas soluções que misturam música com inteligência artificial foram usadas e podemos tomá-las como referências e inspirações.

A primeira delas é um aplicativo chamado Chord AI (ChordAI, 2025). Ele tem como principal funcionalidade a representação de cifras dos acordes tocados em músicas em tempo real de vídeos do YouTube. Outro similar é o Songsterr (Songsterr, 2025) que consegue transcrever músicas do YouTube para tablatura.

O terceiro é o Chordwatch (Chordwatch, 2025) que identifica acordes tocados em tempo real e em faixas MIDI. Também é uma aplicação de grande inspiração para o desenvolvimento deste projeto.

Partindo de outras soluções existentes que seguem no tratamento de áudio, tem-se o Melodyne que é um *software* profissional para edição e conversão de áudio para MIDI e é largamente utilizado em estúdios de música. O Sonic Visualiser (Sonic Visualiser, 2025) é um grande *software* de análise detalhada de áudio, incluindo informações como notas e acordes daquela faixa. E, ao final, Samplab (SAMPLAB, 2025) que é uma ferramenta online com a capacidade de encontrar os acordes tocados num trecho de áudio (na versão gratuita, trechos de 10 segunods) pela transcrição via inteligência artificial.

Todos esses citados são *softwares* presentes no mercado que serviram de inspiração para o desenvolvimento deste trabalho. Funcionalidades comuns como transcrição de áudio para MIDI, análise de informações de áudio como notas musicais, acordes e tempo e formatação para partitura são pontos chave encontrados. Acompanhe o Quadro 1 que compara algumas funcionalidades em comum entre eles.

Quadro 1: Comparação de softwares existentes

Software	**Áudio para acordes**	**Nível de uso**	**Valores**
ChordAI	Sim	Casual	Gratuito, mas possui compras via aplicativo
Songsterr	Sim (via YouTube)	Casual	Poucos recursos; assinatura por R$19,90 mensal
Chordwatch	Sim (tempo real e MIDI)	Casual	Compra por R$80,00 aproximadamente
Melodyne	Sim	Profissional	Teste grátis limitado; Planos de R$134,00 até R$2150,00 aproximadamente
Sonic Visualiser	Sim (mas com plugins)	Profissional	Grátis e de código aberto
Samplab	Sim	Ambos	Gratuito e limitado; assinaturas mensais de R$43,00 até R$53,00

Fonte: elaborado pelos autores (2025)

3 METODOLOGIA

3.1 Contexto

A escolha da transcrição de áudio justifica-se pela necessidade de uma funcionalidade que agregasse usuários comuns, aqueles que podem desconhecer o que um arquivo MIDI representa. O processo de importar MP3 (extensão mais usada entre indivíduos) para serem convertidos para WAV dá-se a este tipo de áudio ser feito para armazenagem sem compressão, garantindo qualidade e em formato de ondas que, via código, podem ser interpretadas e transcritas para um arquivo MIDI.

Ademais, o MIDI, conhecido da sigla em inglês *Musical Instrument Digital Interface* não contém propriamente áudio, mas um conjunto de dados que descrevem a música. Dessa forma, notas, duração e velocidade podem funcionar no padrão de instruções que um sintetizador ou *software* para gerar o som. Assim, é possível diversos equipamentos musicais se comunicarem entre si, interpretando a música de forma digital, bem como sua manipulação via algoritmos.

Portanto, por meio de uma pesquisa e desenvolvimento de uma aplicação que fosse capaz de interpretar acordes num trecho musical, extrair informações melódicas e ainda ser capaz de classificar as emoções, dentre das possíveis (*angry*, *romantic*, *sad* e *happy*) foi a solução encontrada para solucionar os problemas previamente abordados na introdução do documento.

3.2 Tipo de pesquisa

Para assegurar rigor científico na análise e validação do sistema proposto, optou-se pelo desenvolvimento de uma pesquisa aplicada com abordagem quantitativa e caráter experimental. Consistiu-se em solucionar o problema real da diminuição da complexidade da teoria musical por meio de um sistema computacional que reconheça emoções num trecho musical por meio da análise de arquivos de áudio e extração de informações harmônicas. A transcrição de arquivos MP3 para MIDI foi uma funcionalidade criada visando facilitar a aceitação pelos usuários mais leigos.

Adicionalmente, a abordagem quantitativa se sobressai uma vez que buscou-se mensurar variáveis, tratamento e análise de dados e comparação estatística de resultados, assegurando maior precisão de modelo de inteligência artificial e capacidade de reprodução dos fatos.

Ao fim, justifica-se o caráter experimental dado a necessidade de testar hipóteses por meio do controle e da manipulação de variáveis em um ambiente estruturado, permitindo avaliar respostas como a correspondência das emoções classificadas e congruências das informações melódicas extraídas e comparação com ferramentas e pesquisas similares para validação dos resultados.

3.3 Procedimentos técnicos

Os procedimentos técnicos adotados nesta pesquisa visam garantir a coleta, processamento, análise e validação sistemática dos dados musicais, assegurando rigor metodológico e reprodutibilidade dos resultados. O tópico está estruturado em etapas, não necessariamente sequenciais, dito que algumas etapas possam ter sido executadas paralelamente, visando compreender desde a coleta de arquivos musicais, pré-processamento, transcrição de áudio e extração de informações melódicas dos arquivos MIDI.

3.3.1 Seleção do conjunto de dados musicais

Durante o planejamento do projeto, foi necessário encontrar um conjunto de dados que compreendesse, em sua totalidade, todos os requisitos que pudessem agregar valor para a conclusão do *software*. São eles:

1. Categorização de emoções baseado na progressão de acordes e modo (menor e maior);
2. Padrões de sequência de acordes que pudessem inferir a emoção dentre as presentes no conjunto;
3. Possuir grande quantidade de dados para enriquecimento do projeto.

Assim foi encontrado um conjunto de dados intitulado de XMIDI (XMUSIC-PROJECT, 2025) que pode ser acessado por meio do link: <https://github.com/xmusic-project/XMIDI_Dataset>. Neste, uma grande coleção de arquivos MIDI estão nomeados com o padrão *XMIDI_<Emoção><Gênero><ID_Arquivo>.midi.* Ele compreende cerca de 108.023 arquivos, totalizando 5.278 horas de som tocado. Dentre os rótulos, estavam presentes 11 emoções iniciais em inglês: *angry, exciting, fear, funny, happy, lazy, magnificent, quiet, romantic, sad, warm*, ou seja, irritado, empolgante, amedrontador, engraçado, feliz, preguiçoso, magnífico, quieto, romântica, triste, calma, respectivamente.

Logo, todos os requisitos elencados foram suficientemente supridos em um único conjunto de dados rico e preciso.

3.3.2 Pré-processamento dos dados

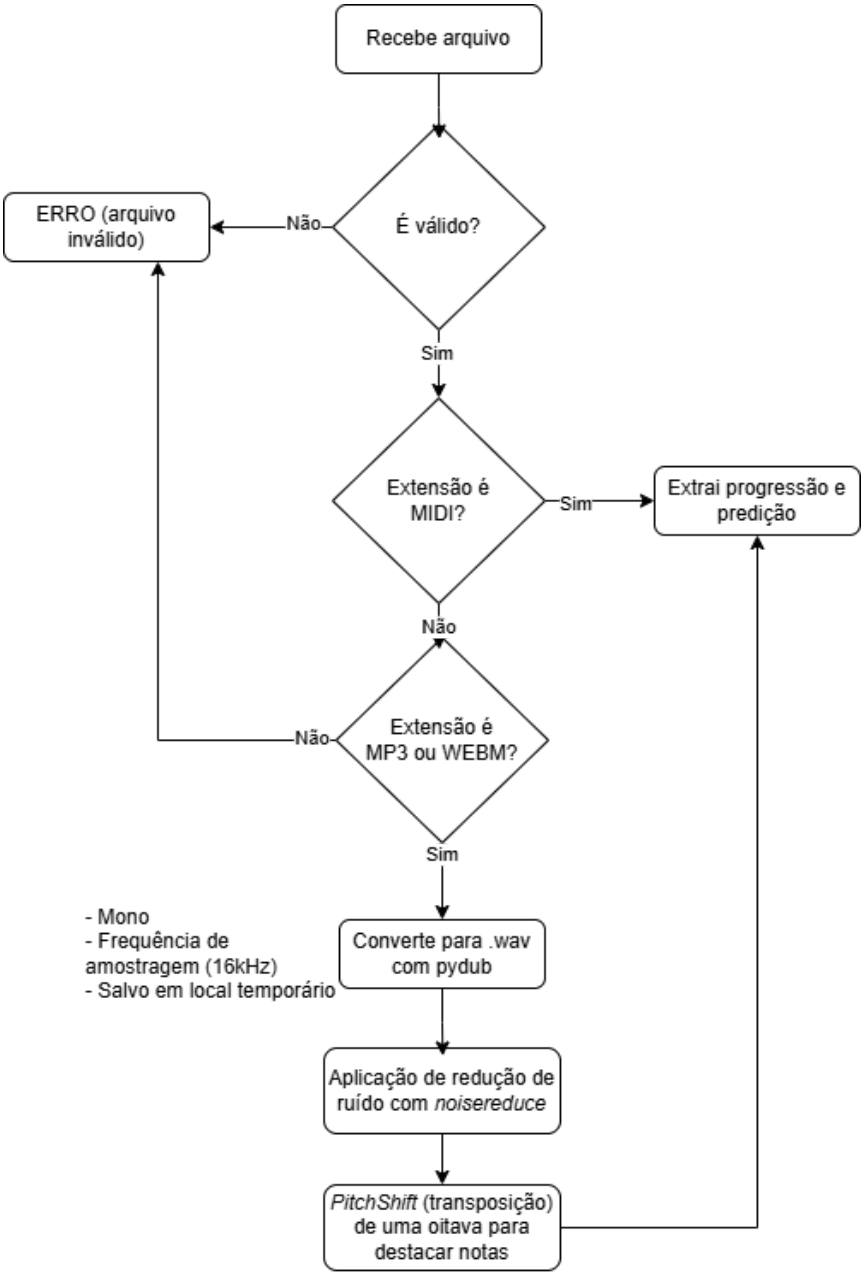
Este procedimento foi a adequação dos dados do conjunto de dados descrito no tópico anterior, para sua manipulação via código. Por meio de algoritmos em Python (Python Software Foundation, 2025) que realizaram a extração de informações harmônicas do MIDI (as progressões de acordes dos mais de 100 mil arquivos) e de seu rótulo, a emoção relativa foi retirada. Esse *script* realizou a transposição das informações para um arquivo do tipo *.csv*, o qual é um tipo de arquivo de texto simples que armazena os dados de forma tabular, como uma planilha. A escolha foi meramente convencional, uma vez que métodos de inteligência artificial usam esse tipo de arquivo dado sua velocidade de leitura e pouco peso computacional para treinamento.

Portanto, esse primeiro passo foi imprescindível para a inicialização do projeto. Na seção DESENVOLVIMENTO, tópico 4.2 desse documento, alguns outros procedimentos para refinamento e melhoria de performance do projeto serão descritos passo a passo. Essa primeira etapa norteou boa parte dos testes e escolhas referidas à pesquisa.

3.3.3 Transcrição de áudio

Dentre os procedimentos técnicos, a transcrição de áudio foi desenvolvida com base em pesquisa e testes manuais, uma vez que sua sensibilidade e complexidade técnica de execução poderiam criar barreiras de difícil transposição. A necessidade de receber arquivos MP3 para identificação foi acordada para que contribuísse com a integração de usuários que não conheçam arquivos MIDI propriamente ditos. Então, visando a facilitação dessa etapa, um fluxograma ilustrado pela Figura 1 foi desenvolvido para esclarecer os processos que o áudio deve seguir até que tenha suas informações extraídas.

Figura 1: Fluxo de transcrição MP3 para MIDI



Fonte: elaborado pelos autores (2025)

Era reconhecido que essa funcionalidade seria uma das mais problemáticas dado sua alta sensibilidade à ruídos. Dito isso, uma certa sequência de acordes tocadas num ambiente com pessoas e repetido, com os mesmos equipamentos, porém em um estúdio profissional, a resposta seria diferente. Com base nisso, foi levantado a necessidade de equalização e pré-processamento de áudio para percorrer o caminho MP3 - WAV - MIDI a fim de minimizar interferências externas que pudessem prejudicar a classificação da emoção posteriormente.

A equalização, processo de melhoria no áudio, foi possível com bibliotecas da própria linguagem Python (Python Software Foundation, 2025), especificamente a biblioteca *noisereduce* (SAINBURG, 2024) teve resultados satisfatórios quanto à testes feitos. Essa ferramenta usa um método conhecido por *spectral gating*. Em termos simples, é uma forma mais avançada de "noise gate", uma técnica muito utilizada para redução de ruídos. Ela funciona analisando frequências do áudio e diminuindo apenas aquelas que parecem ser "sujeira" (ruídos), limpando a gravação como um todo.

O segundo ponto é o pré-processamento do tipo *.wav* para *.midi*. Nessa etapa, foi identificado a necessidade de aumentar em uma oitava o que foi ouvido, uma vez que feito isso, os acordes são deslocados de grave para agudo. Em outras palavras, para facilitar a extração e evitando que notas naturalmente mais baixas sejam removidas no processo, o ajuste não afeta diretamente no acorde tocado, mas deixa toda a sequência mais aguda, numa frequência que não há perda de informação.

3.3.4 Modelos de classificação

Para a classificação das emoções musicais, foram selecionados quatro modelos de aprendizado de máquina amplamente utilizados e reconhecidos pela eficiência em tarefas de classificação: *Support Vector Machine* (SVM), *K-Nearest Neighbors* (KNN), *Random Forest* e *Naive Bayes*. A escolha desses algoritmos comparou diferentes abordagens de modo que *Random Forest* apresentou um desempenho melhor na análise das características harmônicas extraídas das transcrições MIDI.

O SVM (*Support Vector Machine*) é um método de classificação supervisionado que encontra o hiperplano ótimo que melhor separa as classes no espaço de características. Ele maximiza a margem entre os vetores de suporte, que são os pontos mais próximos da fronteira de decisão. Por esse alto custo computacional e a necessidade de ajuste fino dos parâmetros do kernel, o método SVM apresentou um resultado inicial insatisfatório para os fins da pesquisa.

Já o KNN (*K-Nearest Neighbors*) é baseado em proximidade. Ele classifica uma nova amostra considerando as k amostras mais próximas no conjunto de treinamento, definidas por uma métrica de distância e a mais frequente entre os vizinhos determina a predição. Esse método, apesar de simples, mostrou-se ineficiente quanto ao tempo de predição, já que precisava calcular a distância entre o novo ponto e todo o conjunto de treinamento, tornando-o problemático.

O terceiro método escolhido, *Naive Bayes*, é um classificador probabilístico baseado no Teorema de Bayes, assumindo independência entre as variáveis preditoras. Ele calcula a probabilidade de uma amostra pertencer a cada classe e seleciona aquela com maior valor. Tal método é simples e rápido, funciona relativamente bem, porém a hipótese de independência entre as variáveis preditoras resultou numa precisão baixa na classificação emocional.

Por fim, o *Random Forest* é um método que combina múltiplas árvores de decisão independentes, cada uma treinada com subconjuntos aleatórios de dados e atributos. A decisão final é obtida por votação majoritária entre as árvores. Essa abordagem reduz o risco de *overfitting* (é quando o modelo treinado aprende muito bem os detalhes de ensino, mas apresenta resultados ruins com novos dados) e aumenta a robustez e a precisão do modelo. Esse tipo de classificação foi determinante, uma precisão aceitável e numa velocidade consideravelmente boa com um grande conjunto de dados.

3.4 Etapas de execução

Nesse tópico do capítulo 3, estão descritos todas as etapas realizadas para a criação do projeto. Os mesmos processos estão, com maior detalhamento, presentes no capítulo seguinte, com um rigor técnico mais acentuado.

3.4.1 Etapa 1

A primeira etapa foi definida para que todo o ambiente local estivesse funcionando com as tecnologias escolhidas, especialmente pelo uso da tecnologia Docker (Docker, 2025) como facilitadora e que as arquiteturas de *software* estivessem claras para regulamentação do padrão. Tudo isso deveria ser colocado em prática e adicionado à um repositório em nuvem com a ferramenta GitHub (GitHub, 2025) para que tivesse controle total de versionamento e acesso geral à todos os desenvolvedores.

Dessa forma, foi estabelecido um ambiente usando containers, assim como a determinação da arquitetura *backend* separado do *frontend* em ambientes diferentes interconectados. Assim, tanto explicitou-se a responsabilidade de cada parte quanto facilitou a integração. É possível acessar o repositório do projeto no GitHub (RAMOS DOMINGUES, 2025) e um maior detalhamento do processo de criação dos containers poderá ser encontrado na seção DESENVOLVIMENTO, tópico 4.1.2.

Após aplicação dos passos descritos, foi possível obter um código concreto onde os containers descritos isolavam *backend* e *frontend*, de forma interconectada. Os padrões de

código foram devidamente implementados, assim como a primeira versão do código já estivesse salva em um repositório online em nuvem.

3.4.2 Etapa 2

A segunda etapa consistiu na limpeza inicial dos dados fornecidos pelo conjunto robusto de arquivos MIDI, bem como a criação das classes, seguindo padrões estabelecidos anteriormente que fossem responsáveis por cada um dos métodos escolhidos de classificação. Esperava-se ter um *dataset.csv* criado com extração de acordes e emoção no formato convencional de arquivos de análise de dados – CSV.

A segunda parte constituiu a criação da primeira rota da API em Python (Python Software Foundation, 2025). Ela era capaz de receber um arquivo do tipo MIDI via upload e já extrair as primeiras informações, como tom e sequência de acordes. Foi feita pensando nos testes iniciais de integração do projeto todo, mas seria alterada pelo processo geral de transcrição.

Ao final dessa etapa, tinha-se um novo conjunto de dados formatados e recolhidos à partir do conjunto de arquivos MIDI inicialmente catalogado, bem como quatro classes criadas na API que foram responsabilizadas cada uma por seu método de classificação escolhido anteriormente e uma rota *backend* capaz de receber um arquivo MIDI para extração de informações melódicas.

3.4.3 Etapa 3

A terceira etapa tratou da integração dos modelos de IA com a API diretamente, por meio do incremento da rota de *upload* para receber arquivos MP3, iniciando o processo de transcrição de áudio. Ainda nesse período, foram gravados alguns arquivos de áudio simples, em ambiente controlado, para servirem de comparação para os testes de extração da progressão tocada, garantindo seu rigor profissional e com baixíssima ou inexistente interferência. Ainda na terceira etapa os treinamentos dos modelos escolhidos com o *dataset* criado foram realizados.

Ademais, as primeiras telas foram criadas, de maneira simplista, visando validar a integração entre todas as partes do projeto.

Portanto, ao final dessa etapa, tinha-se o projeto com as funcionalidades implementadas (transcrição de áudio e predição de emoção), áudios de testes gravados e a primeira tela desenhada. Porém, constatou-se a necessidade de refinamento tanto o método *Random Forest*, já que apresentou melhor desempenho, quanto a transcrição de áudio. Essa

imprecisão já havia sido mapeada: o desafio era melhorar o que estava sendo feito para atingir um patamar satisfatório e coerente para o projeto.

3.4.4 Etapa 4

Por conseguinte, dado que os objetivos estavam sendo alcançados, houve a necessidade de realizar um adequação ao modelo de classificação para atingir uma acurácia satisfatória com mais de 50% (parâmetro definido internamente) para que os resultados fossem, de fato, relevantes. Da mesma forma, a transcrição de áudio estava funcional, mas sofria interferência de qualquer ruído. Então nessa etapa, houve um trabalho focando a limpeza dos arquivos de áudio enviados pelo usuário, evitando ruídos e problemas na detecção da progressão tocada.

Dessa forma, ambos foram feitos paralelamente, mas iniciou-se com a transcrição. O problema dessa rotina foi a sua alta sensibilidade para ruídos, ou seja, qualquer que seja o som capturado durante um acorde tocado poderia alterar o resultado final. Outro ponto identificado com testes é que, ao transcrever de MP3 para WAV e depois para MIDI, havia uma redução, tornando o som mais grave, e aplicados filtros de compressão, algumas informações eram perdidas. Assim sendo, houve a necessidade de aumentar a frequência antes da conversão (uma oitava, para não perder informações relevantes do que foi tocado) e aplicação de filtro de *noise reduction* (redução de ruído, do inglês) para a transcrição.

Em seguida, para as emoções detectadas dos modelos de inteligência artificial, foi necessário um estudo intenso e aplicações de métodos de limpeza do *dataset* usado. A princípio, o largo conjunto de dados sofria intensamente com emoções desbalanceadas em quantidade de linhas e de acordes, o que resultava em resultados enviesados da predição, e portanto afetando a precisão. O tratamento dos dados iniciais foi essencial para alcançarmos um equilíbrio entre as *features* para reduzir o caráter esparso dos dados, característica essa que contamina o aprendizado do modelo. A seleção de classes para serem inferidas pelo modelo de classificação também foi um ponto de destaque por proporcionar uma quantidade menor de possíveis resultados, auxiliando o processo de refinamento e aumento da precisão do modelo para aproximadamente 61%.

Portanto, após desenvolvimento para adequação das etapas prometidas, a transcrição de áudio foi melhorada regulando parâmetros de equalização bem como adição de uma etapa intermediária de correção manual pelo usuário. Da mesma forma, pela aplicação dos métodos de regulação de modelos de inteligência artificial explicitados, a acurácia da classificação foi atingida à um patamar aceitável para realização da pesquisa e conclusão dos objetivos.

3.4.5 Etapa 5

Como última etapa, era necessário realizar testes de integração, validação manual e adição de novas informações relevantes para o usuário final. Também havia necessidade de melhorar o *frontend* com um visual mais recente e usável.

Desse modo, com a utilização das bibliotecas de processamento de áudio e interpretação de MIDI, novas informações foram adicionadas ao visual do projeto, como o tom estimado, andamento e seu nome em italiano, função de cada acorde dentro de uma escala, a escala que pertence e também escalas relativas. Informações acerca das notas que compõem o acorde, seu nome por extenso e cifra, além da possibilidade de reenvio dos arquivos e gravação direta do navegador.

Certas melhorias de impacto foram feitas quanto ao *frontend*. A definição dos padrões de cores, componentização de certas partes reutilizáveis, responsividade e fluxo de uso do usuário também foram afetadas diretamente, contribuindo para um projeto mais concreto e usável.

Logo, dado os refinamentos das duas partes importantes do código (transcrição de áudio e classificação da emoção) e o alcance de um patamar satisfatório, o projeto foi finalizado. Ao final de tudo, tem-se uma rotina de *upload* de arquivos de áudio, transcrição e extração de acordes, confirmação e customização por parte do usuário e classificação da emoção.

4 DESENVOLVIMENTO

4.1 Visão geral

Esta seção busca explicar os processos durante a etapa efetiva de desenvolvimento do projeto, contendo informações técnicas relevantes e detalhadas. Ademais, também é possível encontrar informações sobre a interface desenvolvida, limitações técnicas e validações realizadas.

4.1.1 Tecnologias, bibliotecas e modelos utilizados

Para a realização do projeto foi necessário o conhecimento de algumas tecnologias e ferramentas úteis de desenvolvimento. A primeira foi a linguagem escolhida para trabalhar tanto na criação do modelo de inteligência artificial quanto a API: Python (Python Software Foundation, 2025), para a API, especificamente o framework FastAPI (FastAPI, 2025) foi determinante na criação de rotas e integração com *front end*. Escolheu-se essas tecnologias pela sua praticidade com assuntos de inteligência artificial e bibliotecas úteis para tal finalidade, além do framework simples, escalável e de fácil entendimento, garantindo assim uma API construída na linguagem sem variações.

No segundo momento, o framework VueJS (Vue.js, 2025) foi responsável pela criação da interface do usuário e comunicação com a API desenvolvida. Esse framework é ótimo por sua reatividade, o que torna a construção de SPA (*single page application*) mais fácil e otimizada. A linguagem torna a construção de uma interface mais reativa e usável para o usuário, já que é uma tecnologia de ponta para essa finalidade. A nível de código, a sua componentização permitiu o reuso de partes sem código duplicado. Agregada à ela, o design foi construído seguindo os padrões de TailwindCSS (TAILWIND LABS, 2025).

Quando se diz respeito ao próprio modelo de inteligência artificial treinado, a biblioteca Python (Python Software Foundation, 2025) usada foi *scikit-learn* (Scikit-learn, 2025) que abrange uma gama enorme de métodos de classificação e funções necessárias para a rotina de classificação das emoções.

Exclusivamente usou-se um *dataset* chamado de XMIDI (XMUSIC-PROJECT, 2025), o qual contém um grande acervo de arquivos MIDI rotulados. Essa grande quantidade de informação agregada facilitou o treinamento do modelo conciso para o fim desejado.

Para o campo musical, a biblioteca chamada music21 (Cuthbert *et al*., 2025) é largamente utilizada para análise harmônica e detecção dos acordes. Além dela, a biblioteca intitulada de basic-pitch (SPOTIFY AB, 2025) foi utilizada na rotina de reconhecimento de

acordes e extração de informações musicais, bem como *librosa* (LIBROSA DEVELOPERS, 2025) e pydub (ROBERT, 2025) atenderam à finalidades de manuseio de arquivos de áudio, equalização harmônica, tratamento de áudio e conversão para WAV. Ao final, noisereduce (SAINBURG, 2024) foi usada exclusivamente para redução de ruídos no momento de transcrição de áudio.

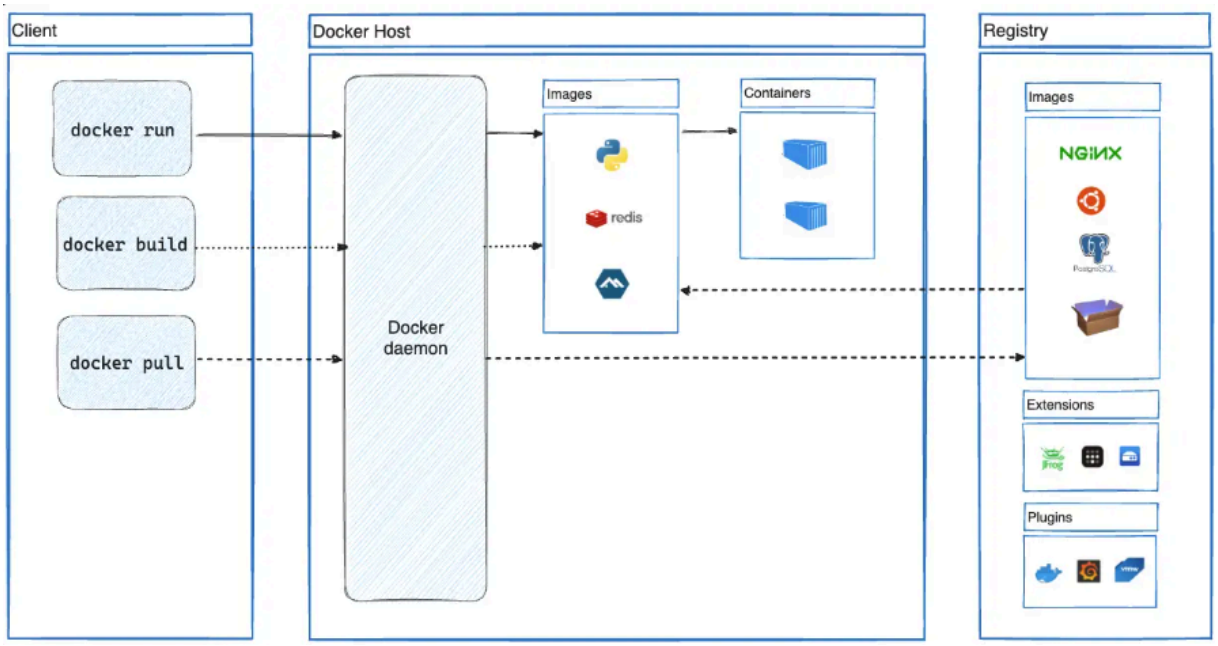
Em seguida, para o ambiente de desenvolvimento, containers Docker (Docker, 2025) foram preparados para rodar a interface (*front end*) separadamente da API em si (*backend*). Essa ferramenta é útil para construir espaços isolados, além da sua generalização, permitindo o ambiente idêntico em qualquer máquina e atomização.

A plataforma GitHub (GitHub, 2025) foi usada para armazenamento e versionamento dos códigos, sendo a mais conhecida no mercado atualmente. Essa ferramenta abrange atividades de divisão em *branches*, tornando o desenvolvimento mais seguro e organizado. Ele também é responsável por manter a qualidade do código com seus *pipelines* e avisos de quebra de segurança, principalmente caso haja vazamentos de chaves primárias e senhas em arquivos do repositório.

4.1.2 Arquitetura da aplicação

É importante o destaque do tipo de arquitetura de *software* adotado para o código criado. Antecipadamente, containers Docker (Docker, 2025) foi montado com a finalidade de equiparar o ambiente de desenvolvimento num padrão em que seja idêntico em qualquer que seja a máquina hospedeira. A Figura 2, da própria documentação da ferramenta, ilustra seu funcionamento. A coluna *client* são comandos para o usuário rodar no terminal onde os arquivos de configuração estão escritos. Então, o *daemon* é como um gerenciador de ciclo de vida dos containers. Assim, as imagens (normalmente onde as linguagens de programação são instaladas) são geradas e seladas em um *container*.

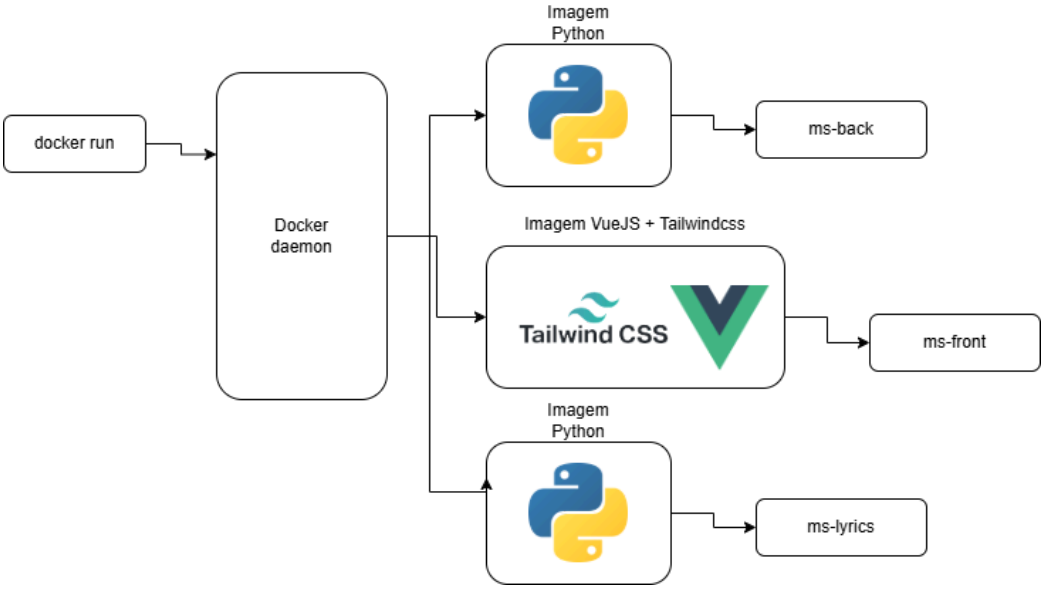
Figura 2: Diagrama do funcionamento Docker



Fonte: Docker (2025).

Em seguida, a Figura 3 demonstra a forma que foi separado os containers e imagens Docker (Docker, 2025) do projeto seguindo um padrão parecido com o da figura anterior. Ao final, ms-lyrics, ms-back e ms-front são as iniciais de *MOODSTRINGS-LYRICS, MOODSTRINGS-BACKEND* e *MOODSTRINGS-FRONT*. Esses são os nomes dados aos containers gerados para facilitar o referenciamento durante uma linha de código.

Figura 3: Ambiente de desenvolvimento Moodstrings



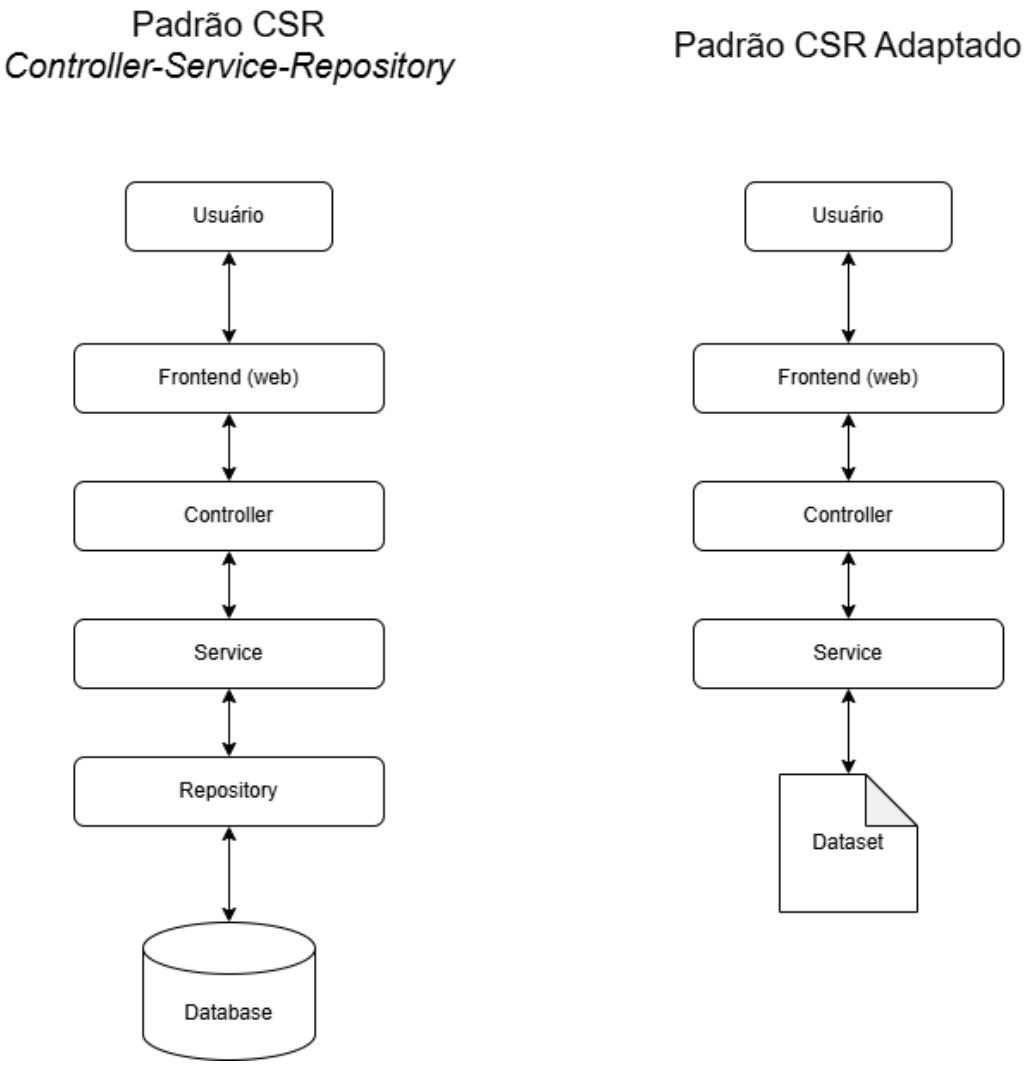
Fonte: elaborado pelos autores (2025).

Além da “conteinerização” do código com esta ferramenta, padrões internos de código foram definidos. Um dos padrões de código existentes é o CSR (*Controller-Service-Repository*) para APIs.

De arquitetura interna, a opção mais viável foi o padrão conhecido de aplicações: o CSR *pattern* (*Controller-Service-Repository pattern*), onde um arquivo é considerado o controlador (*controller*), recebendo informações das requisições do navegador e enviando para um serviço (*service*). Esta classe tem a responsabilidade de aplicar as regras de negócio e validações adicionais, bem como se comunicar com o banco de dados por meio do repositório (*repository*). No caso desse projeto, foi necessário uma adaptação, uma vez que não foi usado banco de dados, mas sim um conjunto de dados *.pkl* (dados treinados) ou *.csv* (informações retiradas para uso estatístico), então removemos a camada *repository*. Veja a Figura 4 para entender o processo.

Em resumo, o usuário acessa a aplicação *web* via o próprio navegador, encontrando o *frontend*. Assim, qualquer informação (como envio do arquivo de áudio) é passado para uma classe de controle (*controller*) responsável para analisar qual a requisição e direcionar para uma classe realizar um serviço (*service*). Eventualmente essa classe pode requisitar o *dataset* (modelo treinado) para a classificação da emoção do trecho enviado pelo usuário. A informação que será passada ao usuário segue o caminho reverso deste descrito.

Figura 4: Adaptação do modelo CSR para arquitetura interna



Fonte: elaborado pelos autores (2025)

4.2 Preparação do modelo de inteligência artificial

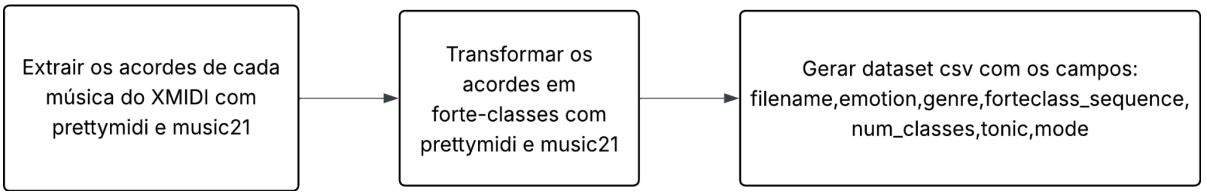
Recapitulando o que foi descrito no tópico 3.3.1, foi utilizado o conjunto de dados XMIDI (XMUSIC-PROJECT, 2025) que pode ser acessado por meio do link: <https://github.com/xmusic-project/XMIDI_Dataset>. Esse compreende uma grande variedade de arquivos MIDI com melodias tocadas e rotulados como *XMIDI_<Emoção><Gênero><ID_Arquivo>.midi.* O mesmo possui 108.023 arquivos, totalizando cerca de 5.278 horas de música. Porém, foi necessário a realização de alguns passos de refinamento, limpeza e melhoria das informações presentes nesse conjunto de dados para que se adequasse às necessidades do projeto.

4.2.1 Tratamento dos dados

A primeira etapa consistiu na limpeza inicial dos dados fornecidos pelo conjunto robusto de arquivos MIDI, bem como a criação das classes, seguindo padrões estabelecidos anteriormente, que fossem responsáveis por cada um dos métodos escolhidos de classificação.

Inicialmente foi necessário extrair informações para criação de um novo conjunto de dados. O pacote utilizado é chamado de XMIDI (XMUSIC-PROJECT, 2025) e cada arquivo seguia o padrão *XMIDI_<Emotion><Genre><ID_len_8>.midi,* onde *Emotion* é o rótulo da emoção referente ao áudio, importantíssimo para análise. Em seguida, um *script* simples em Python (Python Software Foundation, 2025) que lê o arquivo midi e toma informações da sequência reproduzida, gerando uma progressão de acordes. Por fim, após um determinado tempo, um *dataset.csv* pôde ser gerado. Na Figura 5 é possível analisar o fluxograma inicial de criação desse conjunto de dados:

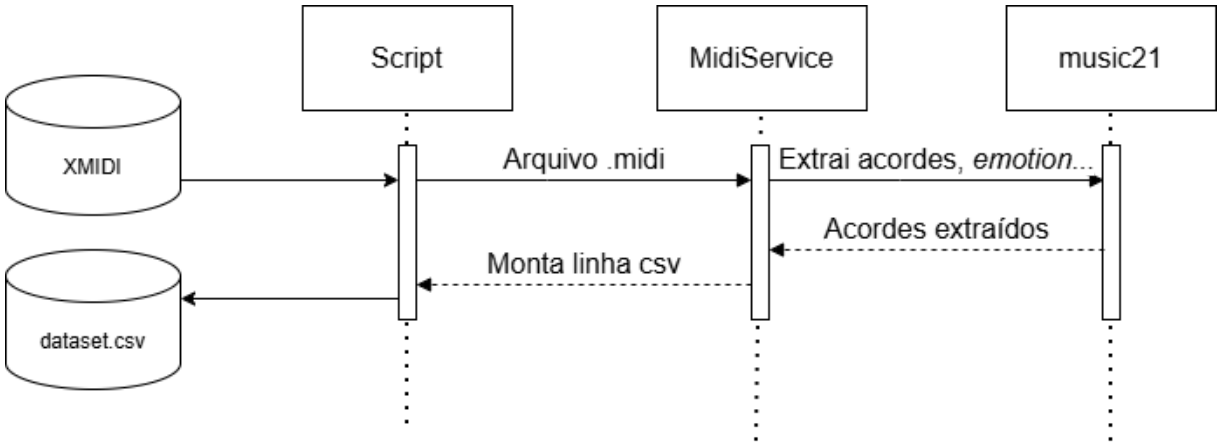
Figura 5: Fluxograma do script de criação do novo dataset



Fonte: elaborado pelos autores (2025)

Partindo do mesmo fluxo, a Figura 6 explicita um diagrama de sequência que mostra como foi realizado a construção do *dataset*. XMIDI trata-se do conjunto de arquivos MIDI, o qual é enviado para o *script* que extrai um arquivo por vez, passa para um serviço (*MidiService*) que trata da extração em si com a biblioteca dita. Assim a informação toma o caminho inverso até ser inserida no *dataset.csv*.

Figura 6: Diagrama de Sequência - Script de criação do novo dataset



Fonte: elaborado pelos autores (2025)

Num próximo passo, foram criadas quatro classes que são responsáveis pelos métodos de classificação escolhidos para o teste inicial. Os métodos em questão são: KNN, SVM, *Random Forest* e *Naive Bayes*. Foram estes escolhidos pela convenção de serem métodos largamente utilizados em classificação com inteligência artificial. A justificativa da escolha dos métodos foi descrita na seção, tópico 3.3.4.

4.2.2 Testes iniciais de treinamento

Com o *dataset* gerado, foram treinados três métodos, uma vez que o KNN é um método treinado toda vez que é executado. Dos outros, após certo tempo de execução, foram criados os arquivos *.pkl* de treinamento finalizado, mas continham certos pontos de destaque, eram eles:

A. Qualquer um dos quatro métodos de classificação possuía um tempo considerável de resposta, causado pela quantidade altíssima de dados de treinamento usados;
B. Todas possuíam acurácia menor que 50%, sendo *Random Forest* o melhor com aproximadamente 40% de precisão, enquanto os outros modelos não passavam de 20%, devido a grande quantidade de emoções e de amostras.

A seguir estão as tabelas com os cálculos dos métodos de avaliação de um modelo. Foi preciso manter os nomes dos indicadores em inglês por convenção, mas são eles:

- *Precision*: A precisão em determinar tal alvo, ou seja, o quanto o modelo erra quando diz que algo é de uma classe. Em outras palavras, seria o grau de confiabilidade

daquela classificação. Exemplo: se o modelo diz que 10 músicas são *happy* e 7 realmente são, então a *precision* é de 0,7.

- *Recall*: é um indicativo de quão o modelo consegue encontrar todos os exemplos reais daquela classe. Útil para determinar a capacidade do modelo não deixar escapar dados positivos. Por exemplo, de 20 músicas rotuladas como *happy* o modelo encontrou 15, então o *recall* é 0.75.
- *F1-Score*: é a média harmônica entre as duas métricas anteriores. Ele mede o desempenho geral da classe. É dado por uma fórmula, é ela:

Figura 7: Fórmula F1-Score

$$F1 = 2 * \frac{\text{precisão} * \textit{recall}}{\text{precisão} + \textit{recall}}$$

Fonte: elaborado pelos autores (2025)

- *Support*: quantos exemplos daquela classe existem dentro do *dataset* de teste.
- Acurácia: porcentagem de previsões corretas no geral.
- Média: é a média aritmética simples das métricas de cada classe
- Média proporcional a classe: média ponderada, como cada classe contribui para a média proporcionalmente ao seu *support*.

A Tabela 1 possui os dados gerados no primeiro treinamento, no caso, do modelo *Naive Bayes.*

Tabela 1: Análise de resultado - Naive Bayes

Análise de resultado - Naive Bayes				
Emoção	***Precision***	***Recall***	***F1-Score***	***Support***
Irritado (*angry*)	0,26	0,53	0,35	878
Empolgante (*exciting*)	0,30	0,19	0,23	2074
Amedrontrador (*fear*)	0,07	0,14	0,09	320
Engraçado (*funny*)	0,29	0,11	0,16	1299
Feliz (*happy*)	0,23	0,31	0,26	1334
Preguiçoso (*lazy*)	0,34	0,63	0,44	494
Magnífico (*magnificent*)	0,06	0,17	0,09	277
Quieto (*quiet*)	0,06	0,06	0,06	412
Romântica (*romantic*)	0,28	0,11	0,15	1200
Triste (*sad*)	0,14	0,14	0,14	902
Calma (*warm*)	0,21	0,16	0,18	1469
Acurácia			**21,79%**	10659
Média	0,20	0,23	0,20	10659
Média proporcional a classe	0,24	0,22	0,21	10659

Fonte: elaborado pelos autores (2025)

Ao analisar os dados transcritos do algoritmo para a tabela acima, pontua-se a acurácia de 21,79%, o que é considerado baixíssimo. Além disso, nenhuma precisão das classes sequer chegou próximo de 0,5, tirando a confiabilidade no método escolhido. Portanto, segue-se a interpretação da Tabela 2 com os dados do modelo *Support Vector Machine*.

Tabela 2: Análise de resultado - Support Vector Machine (SVM)

Análise de resultado - Support Vector Machine (SVM)				
Emoção	***Precision***	***Recall***	***F1-Score***	***Support***
Irritado (*angry*)	0,28	0,53	0,37	878
Empolgante (*exciting*)	0,27	0,58	0,37	2074
Amedrontrador (*fear*)	0,20	0,01	0,01	320
Engraçado (*funny*)	0,34	0,16	0,22	1299
Feliz (*happy*)	0,29	0,17	0,21	1334
Preguiçoso (*lazy*)	0,53	0,45	0,49	494
Magnífico (*magnificent*)	0,00	0,00	0,00	277
Quieto (*quiet*)	0,00	0,00	0,00	412
Romântica (*romantic*)	0,27	0,12	0,17	1200
Triste (*sad*)	0,07	0,00	0,00	902
Calma (*warm*)	0,20	0,30	0,24	1469
Acurácia			**27,46%**	10659
Média	0,22	0,21	0,19	10659
Média proporcional a classe	0,25	0,27	0,23	10659

Fonte: elaborado pelos autores (2025)

Ao comparar com os dados do modelo anterior, o SVM se saiu um pouco melhor quanto à acurácia de 27,46% e a classe “Preguiçoso” com 0,5 de confiabilidade. Mesmo que seu desempenho foi levemente melhor, ainda assim não se mostrou útil para a pesquisa. Por fim, vê-se a Tabela 3 com as informações do *Random Forest*.

Tabela 3: Análise de resultado - Random Forest

Análise de resultado - Random Forest				
Emoção	***Precision***	***Recall***	***F1-Score***	***Support***
Irritado (*angry*)	0,48	0,48	0,48	878
Empolgante (*exciting*)	0,39	0,60	0,47	2074
Amedrontrador (*fear*)	0,37	0,25	0,30	320
Engraçado (*funny*)	0,44	0,34	0,38	1299
Feliz (*happy*)	0,37	0,29	0,33	1334
Preguiçoso (*lazy*)	0,67	0,62	0,64	494
Magnífico (*magnificent*)	0,55	0,22	0,31	277
Quieto (*quiet*)	0,62	0,21	0,31	412
Romântica (*romantic*)	0,37	0,45	0,40	1200
Triste (*sad*)	0,43	0,28	0,34	902
Calma (*warm*)	0,30	0,33	0,31	1469
Acurácia			**40,28%**	10659
Média	0,45	0,37	0,39	10659
Média proporcional a classe	0,42	0,40	0,40	10659

Fonte: elaborado pelos autores (2025)

Dado os resultados, destaque a acurácia de 40,28% e algumas emoções com valores de precisão maiores que 0,5, o *Random Forest* mostrou-se mais satisfatório dado seu desempenho e acurácia, portanto foi o modelo com mais relevância para o projeto.

4.2.3 Desenvolvimento do modelo de classificação

A maioria dos *datasets* requer um bom tratamento para que a análise dos dados seja a melhor possível, e isso é ainda mais evidente em conjuntos extensos como o XMIDI. À primeira vista, foi feita uma limpeza de dados nas extremidades do conjunto: músicas com menos de 20 ou mais de 500 *forte-classes* foram removidas, seja por conterem pouquíssimas amostras ou por apresentarem um número excessivamente grande. A decisão foi tomada para manter maior uniformidade dos dados. Posteriormente selecionamos apenas as músicas que possuíam entre 150 e 300 *forte classes* para treino e testes.

As forte classes são representações numéricas de conjuntos de classes de altura que descrevem acordes com base em suas relações intervalares. Optamos por converter os acordes do *dataset* em *forte classes* e salvá-las em formato *CSV* para essa representação, a fim de dar

mais enfoque ao sentido harmônico do acorde do que à sua nomenclatura. Essa ação reduziu a quantidade de acordes a serem analisados pela IA, facilitando o treinamento.

Em seguida, escolhemos quatro classes-alvo para classificação: *happy* (feliz), *sad* (triste), *angry* (irritado) e *romantic* (romântico), uma vez que essas são as emoções mais discrepantes entre si em termos humanos. Por exemplo, inicialmente havia a classe *quiet* (quieto) e *warm* (calma) que, de certo modo, são parecidas. Fazendo isso, os resultados são mais variados e possuem clara diferença entre si. Porém, devido à discrepância na quantidade de músicas em cada uma dessas emoções, também foi necessário uniformizar o número de amostras por classe antes do treinamento.

Após a limpeza de dados, obtivemos um conjunto curado totalizando 8.416 amostras (2.104 por classe) para o treinamento do modelo. A partir desse ponto, a estratégia para alcançar melhor precisão foi construir um *pipeline* progressivo, começando com características musicais simples e adicionando camadas de complexidade apenas quando comprovadamente geraram ganhos reais.

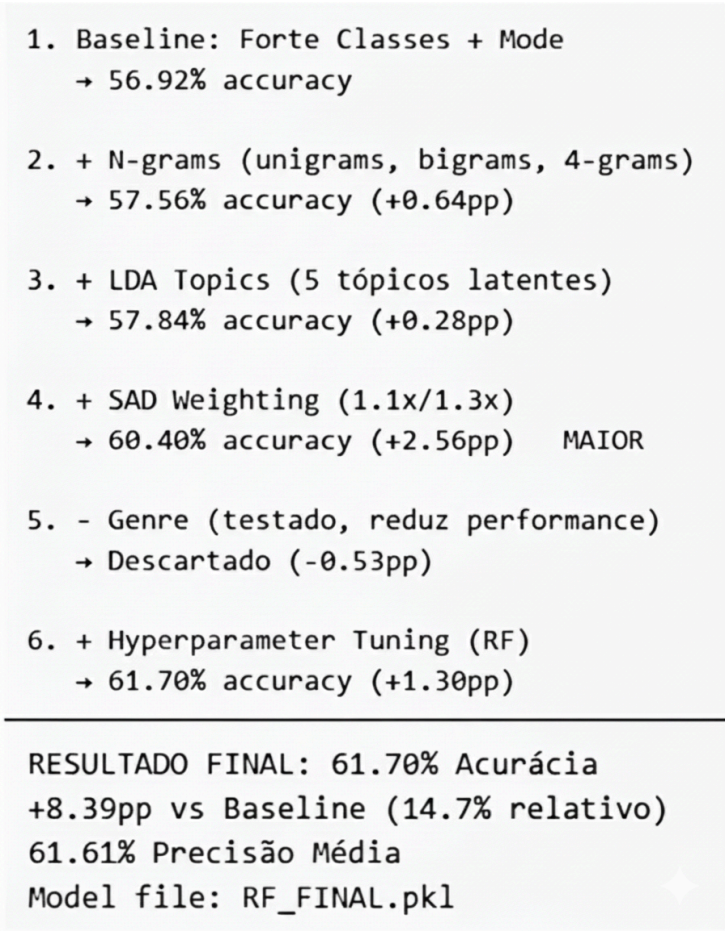
A *baseline* do modelo consistiu em treinar apenas com base na incidência de *forte classes* e no tom (maior ou menor). Em seguida, utilizamos a técnica de *n-grams* com valores de n de um a cinco (referente ao número de acordes agrupados) para identificar as sequências mais comuns em cada classe. Posteriormente, aplicamos a técnica de *Latent Dirichlet Allocation* (LDA) para encontrar padrões estatísticos nas sequências de *forte classes*, análise que o *n-grams* não consegue realizar. Depois, aumentamos o peso das amostras da classe "*sad*" devido à sua menor quantidade e ao alto número de músicas em tom maior nessa categoria, atribuindo um peso adicional às músicas em modo menor, de forma a reforçar a relação entre tom menor e tristeza (escolha comum para compositores).

Também realizamos um treinamento com o gênero musical como *feature*, mas essa abordagem reduziu a precisão. Por fim, configuramos os hiperparâmetros do modelo, definindo 1200 árvores para o *Random Forest* com profundidade máxima de 25 níveis. O resultado final subiu de cerca de 40% nos testes no início do projeto com SVM, *Random Forest* e *Naive Bayes* para aproximadamente 62% com o novo modelo. Foram definidas métricas internas para essa acurácia, são elas:

1. O modelo não poderia ter menos de 50% de precisão, pois isso indicaria mais erros do que acertos, sendo assim insatisfatório;
2. Não poderia ser exatamente 50%, o qual indicaria o puro acaso;
3. Não poderia ser muito próximo de 100%, categorizando *overfitting*.

A Figura 8 apresenta capturas de tela dos resultados de testes do modelo:

Figura 8: Cálculos para acurácia do Random Forest



Fonte: elaborado pelos autores (2025)

4.2.4 Experimento com progressões menores

Após a finalização do modelo de melhor acurácia, foi analisado que as sequências usadas para treinamento eram longas (aproximadamente 150 forteclass). Isso traz um melhor resultado quando usado em músicas, por exemplo, uma vez que são mais acordes que a inteligência artificial é capaz de encontrar padrões. Com base nisso, o mesmo *dataset* balanceado e com quatro classes de classificação foi retrabalhado para que as sequências ficassem com, no máximo, 40 acordes numa mesma linha.

Então, seguindo os mesmos passos, o *dataset* foi criado, balanceado para que as quatro emoções tivessem a mesma quantidade presente e um novo modelo foi treinado com essa configuração, mas o resultado atingido não foi satisfatório, sendo 60%, aproximadamente, de acurácia do modelo. A Tabela 4 contém as informações retiradas dos testes com esse tipo de treinamento.

Tabela 4: Análise de resultado - Sequências curtas de acordes

Análise de resultado - Sequências curtas de acordes				
Emoção	***Precision***	***Recall***	***F1-Score***	***Support***
Irritado (*angry*)	0,72	0,78	0,75	1965
Feliz (*happy*)	0,56	0,52	0,52	1965
Romântica (*romantic*)	0,55	0,59	0,59	1965
Triste (*sad*)	0,55	0,52	0,52	1965
Acurácia			**59,80%**	7860
Média	0,59	0,60	0,59	7860
Média proporcional a classe	0,59	0,60	0,59	7860

Fonte: elaborado pelos autores (2025)

4.3 Transcrição de áudio

Assim como foi resumido no item 3.3.3, o processo de transcrição tornou-se uma funcionalidade sensível e problemática ao longo do projeto. A rotina tinha interferência de ruídos, ou seja, qualquer que seja o som capturado durante um acorde tocado poderia alterar o resultado final. Outro ponto identificado com testes é que, ao transcrever de MP3 para WAV e depois para MIDI, havia uma redução nas ondas, tornando o som mais grave, e quando aplicados filtros de compressão, algumas informações eram perdidas. Baseado nisso, houve a necessidade de aumentar a faixa de Hertz (frequência) antes da conversão (uma oitava, para não perder informações relevantes do que foi tocado), em seguida a aplicação de filtro de *noise reduction* (redução de ruído, do inglês) para a API.

Como explicitado, o primeiro processo é o aumento das notas em uma oitava para não comprometer o tom e deixar mais agudo à fim de uma melhor extração das informações. A Figura 9 ilustra como essa alteração foi feita.

Figura 9: *Pitch shift* do arquivo WAV antes da transcrição

```
def pitch_shift_wav(self, n_steps=12):
    wav_path = self.get_wav_path()
    samples, sr = librosa.load(wav_path, sr=None)
    shifted = librosa.effects.pitch_shift(y=samples, sr=sr, n_steps=n_steps)
    sf.write(wav_path, shifted, sr)
```

Fonte: elaborado pelos autores (2025)

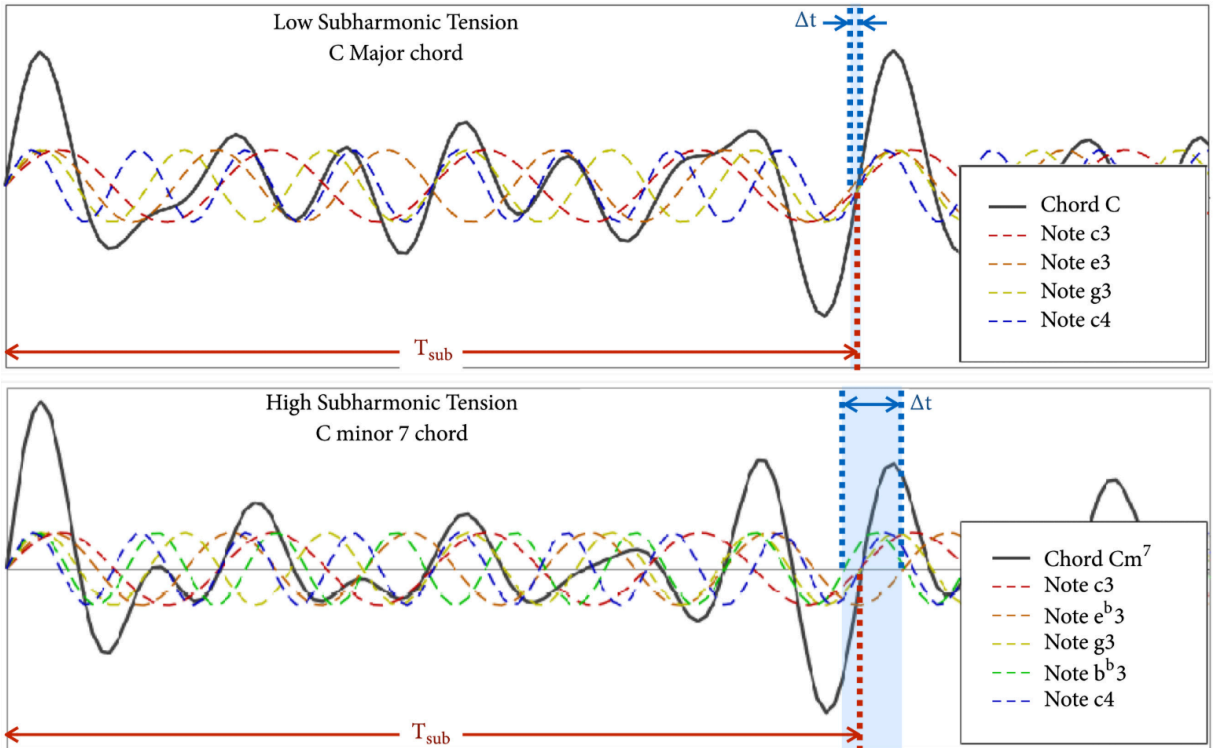
Essa função é a responsável por aumentar uma oitava no arquivo de áudio. Na linha 132, a função *pitch_shift* da biblioteca librosa (LIBROSA DEVELOPERS, 2025) aceita alguns parâmetros usados, são eles:

I. ***y***: É o sinal de áudio original, no caso, um array chamado *samples* com as amostras lidas diretamente do arquivo *.wav;*
II. **sr**: É a taxa de amostragem *(sampling rate*, da sigla em inglês). Nesse uso, ela é capturada, também, do próprio arquivo *.wav*, mas possui um valor em Hertz (como 44.100Hz, por exemplo);
III. **n_steps**: número de semitons de deslocamento no áudio. No caso da função, 12 semitons equivalem à uma oitava completa mais aguda, evitando cair em limitações dos acordes graves e serem removidas sem necessidade.

Após a aplicação dessa função os testes foram refeitos e houve certa melhora, porém ainda havia um certo problema de interferência de algumas notas.

A Figura 10 apresenta uma comparação de tensão sub-harmônica, conceito derivado da música e análise espectral, que indica o grau de estabilidade dos resultados das relações entre frequências que não seguem proporções simples, como acontece com instrumentos iguais ao violão. Em resumo, alta tensão sub-harmônica indica que o conjunto é mais instável, ambíguo e irregular; por sua vez, a baixa tensão sub-harmônica reflete uma organização mais clara e estável.

Figura 10: Comparação de ondas entre dois acordes



Fonte: Adaptado de Chan, P. Y.; Dong, M.; Li, H. (2019).

O gráfico possui curvas que representam as notas que compõem os acordes e são descritas na legenda. O ponto de destaque é o Δt, uma vez que ilustra a pequena diferença criada pela nota destacada em verde (*Note b^b3*). Mesmo uma mínima variação como tal pode alterar significativamente a interpretação do acorde, já que essa é dada justamente pelas relações de frequência e da distribuição dos harmônicos no espectro. Assim, um "desvio" quase imperceptível é capaz de sintetizar um resultado harmônico inesperado e incongruente.

Então, a redução de ruído foi essencial para a limpeza inicial de pequenas interferências no áudio. No trecho do *script* presente na Figura 11 é possível analisar o uso da biblioteca de redução de ruídos.

Figura 11: Trecho de redução de ruído

```
samples = nr.reduce_noise(y=samples, sr=sr, prop_decrease=0.7)
```

Fonte: elaborado pelos autores (2025)

A função em questão possui três parâmetros de funcionamento, são eles:

I. ***y***: É o sinal de áudio original, no caso, um array chamado *samples* com as amostras lidas diretamente do arquivo *.wav;*

II. **sr**: É a taxa de amostragem *(sampling rate*, da sigla em inglês). Nesse uso, ela é capturada, também, do próprio arquivo *.wav*, mas possui um valor em Hertz (como 44.100Hz, por exemplo);

III. **prop_decrease**: esse parâmetro controla a quantidade de ruído identificado a ser removido, variado de 0 (nenhuma redução) ou 1 (máxima redução). O valor 0.7 foi determinado por testes manuais, o qual se mostrou importante para uma redução de interferências moderada-forte.

Ao final, para determinação dos acordes tocados, filtros melhorados para diminuir o tempo entre notas tocadas e um valor específico de *0.18* da razão entre notas tocadas pôde melhorar significativamente a detecção de acordes tocados. Esse valor foi atingido por meio de testes manuais até a obtenção de um resultado satisfatório. A Figura 12 mostra um pequeno trecho de código onde a variável *bucket_size* com o valor 0.18 foi determinante para detecção de notas que compunham um acorde:

Figura 12: Intervalo de detecção de notas de um acorde

```
bucket = round(note.start / bucket_size) * bucket_size
notes_by_time.setdefault(bucket, []).append(note.pitch)
```

Fonte: elaborado pelos autores (2025)

4.4 Extração de informações harmônicas do arquivo MIDI e letras das melodias

Os arquivos MIDI possuem informações relevantes e de fácil extração em Python (Python Software Foundation, 2025), especialmente com o uso da biblioteca music21 (Cuthbert *et al*., 2025) e basic-pitch (SPOTIFY AB, 2025). Ambas possuem métodos integrados para análise do arquivo e recolhimento de informações importantes. Por exemplo, a função nomeada de *extract_chord_progression* encontrada no repositório do projeto acessível por meio do link: <https://github.com/PixeLarm12/moodstrings-app/tree/main> é um exemplo da combinação dessas duas bibliotecas para recolhimento de informações. Nesse processo, é interpretado as notas agrupadas por certo período de tempo, identificando um acorde, e já nomeando-o para os padrões musicais conhecidos. Especialmente, no mesmo

repositório, é possível encontrar o arquivo intitulado de *MidiService.py*, onde todos os métodos de captura ou manipulação de arquivos MIDI é feito.

Uma adequação feita nos processos de envio de arquivos foi feita para melhorar a usabilidade. Logo após o *upload* do arquivo de áudio, o tempo (BPM) e progressão é extraída com os métodos anteriores, então são mostradas ao usuário para confirmação. Caso haja necessidade de correção, o usuário o faz por meio da inserção manual tanto da progressão quanto do BPM, em sequência será extraído informações adicionais e enviados para a predição da emoção baseada no novo modelo refinado. Essa etapa permite que, dada qualquer possível incongruência capturada pela transcrição, o usuário possa corrigir manualmente se for cabível à ele. Faixas de áudio são voláteis e sensíveis o suficiente para, mesmo com um método avançado de extração, apresentar problemas técnicos ou parecer errado, mas o fato de um usuário ter usado uma dinâmica alternativa para tocar, um resultado ligeiramente discrepante aparecerá.

Outro ponto importante é a existência de um módulo a parte de transcrição das letras das músicas, caso haja. Essa funcionalidade é liberada em português e inglês e foi feita num container a parte à fim de agregar valor ao produto final. Essa parte roda de forma assíncrona após o envio do arquivo, uma vez que é mais pesada computacionalmente, então o usuário tem a experiência de, no final do processo, já ter informação da letra. Para esse processo, a biblioteca *speech_recognition* (UBERI, 2025).

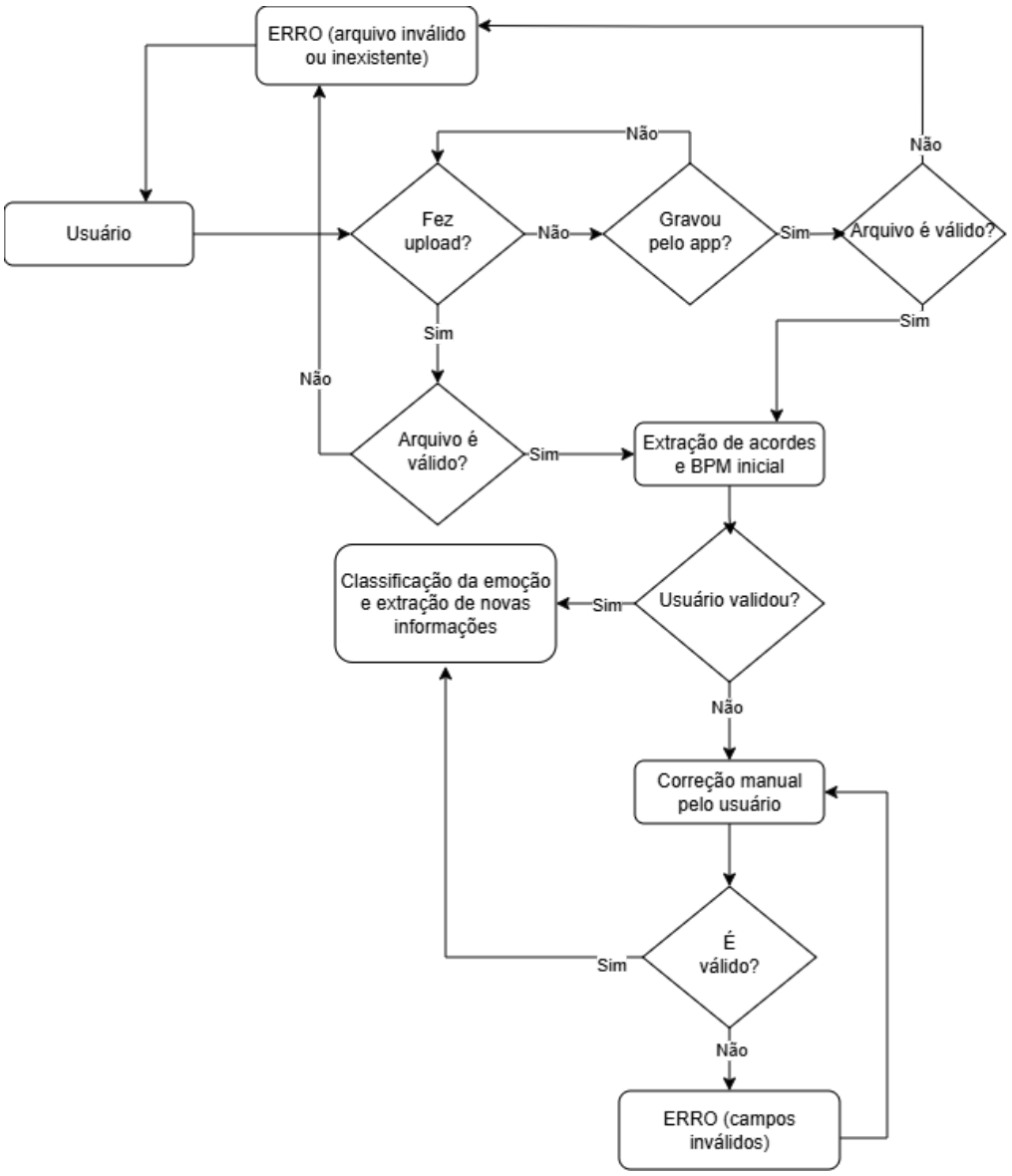
4.5 Fluxo do aplicativo

Com base no artefato desenvolvido, é possível entender o funcionamento de forma simples ao interpretar o fluxograma a seguir que contribui para o total entendimento dos processos que o usuário e arquivos passarão durante o projeto.

A Figura 13 inicia-se com o usuário acessando o navegador e enviando o arquivo. Caso isso não ocorra, é esperado então que ele tenha optado por gravar diretamente da *web*. Espera-se até que tenha um arquivo para análise e, validado-o, passe por uma extração inicial de acordes (transcrição). A partir dessa parte, o usuário espera alguns segundos e pode validar o andamento (BPM) e sequência extraída. Caso falte ou tenha erros, ele será capaz de alterar.

Assim que pronto, seja corrigido ou confirmado, a sequência de acordes é enviada para a classificação de emoções do modelo treinado e, assim que obter resposta da API, o resultado é mostrado para o usuário.

Figura 13: Fluxograma simplificado do funcionamento Moodstrings

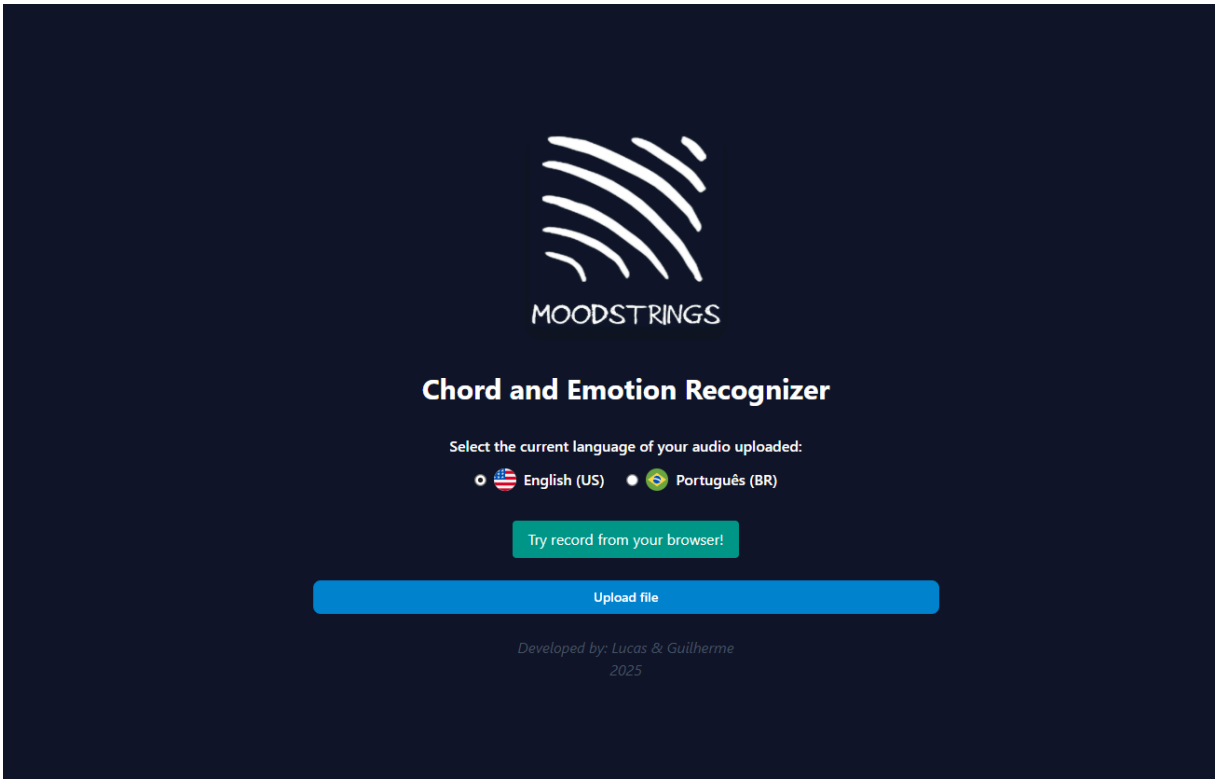


Fonte: elaborado pelos autores (2025)

4.6 Interface e interação com o usuário

O sistema ficou simples, com apenas três variações que seguem o mesmo padrão *single page application* (SPA). Esse termo se refere à uma página HTML construída em uma só tela e, conforme interação com o usuário, sobrescreve o conteúdo ao invés de carregar ou redirecionar o indivíduo. As figuras a seguir mostram como o *frontend* ficou definido.

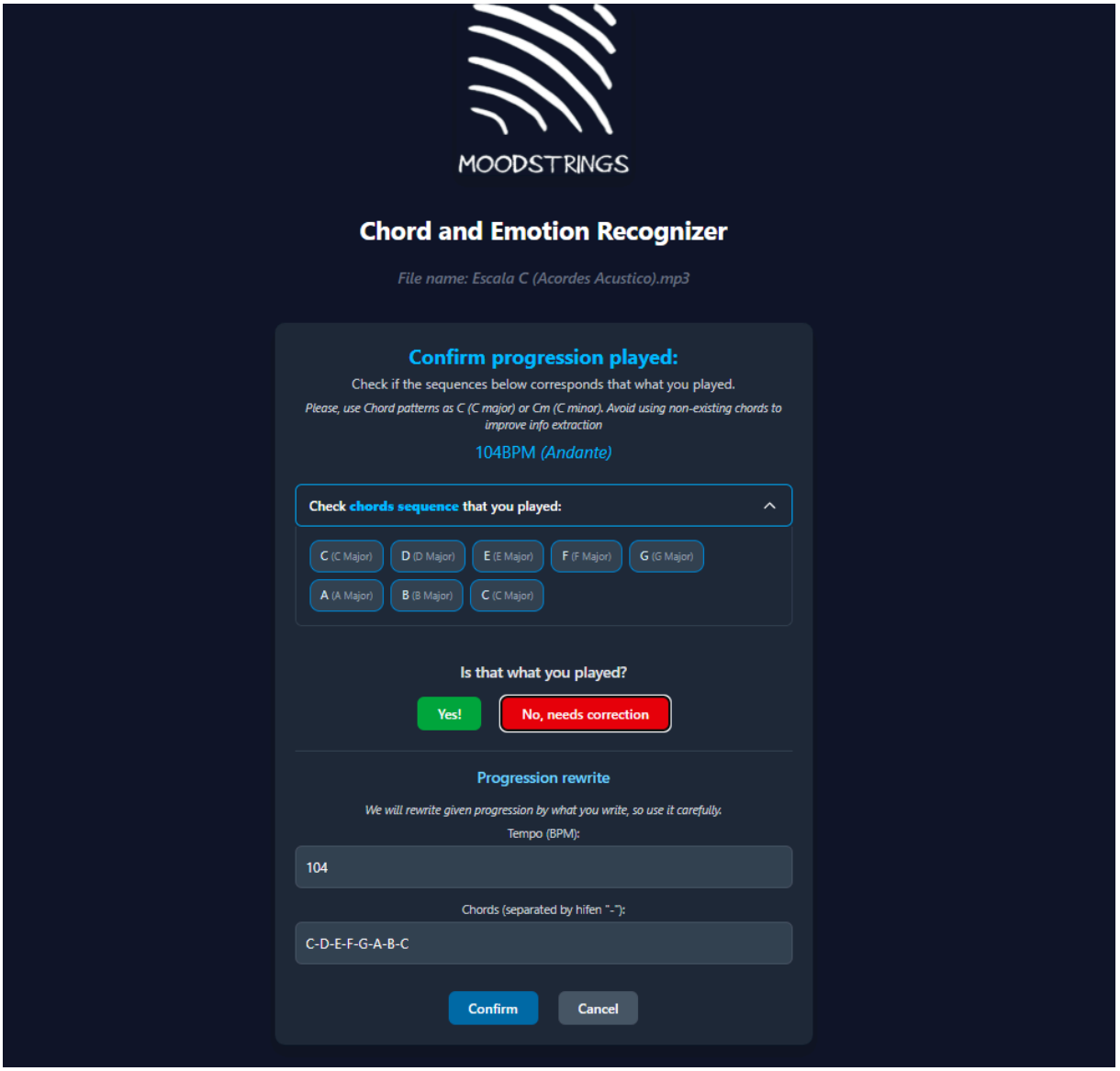
Figura 14: Página inicial do projeto



Fonte: elaborado pelos autores (2025)

A Figura 14 ilustra o início, onde o usuário tem contato com a tela minimalista com o logotipo do projeto e as opções de “Envio de arquivo” (que segue o padrão português do computador do usuário) ou selecionar a opção de “Gravação via navegador”. Além disso, os idiomas são pré-requisitos para que o módulo de reconhecimento de letras de música saiba qual é a língua que ele deve interpretar.

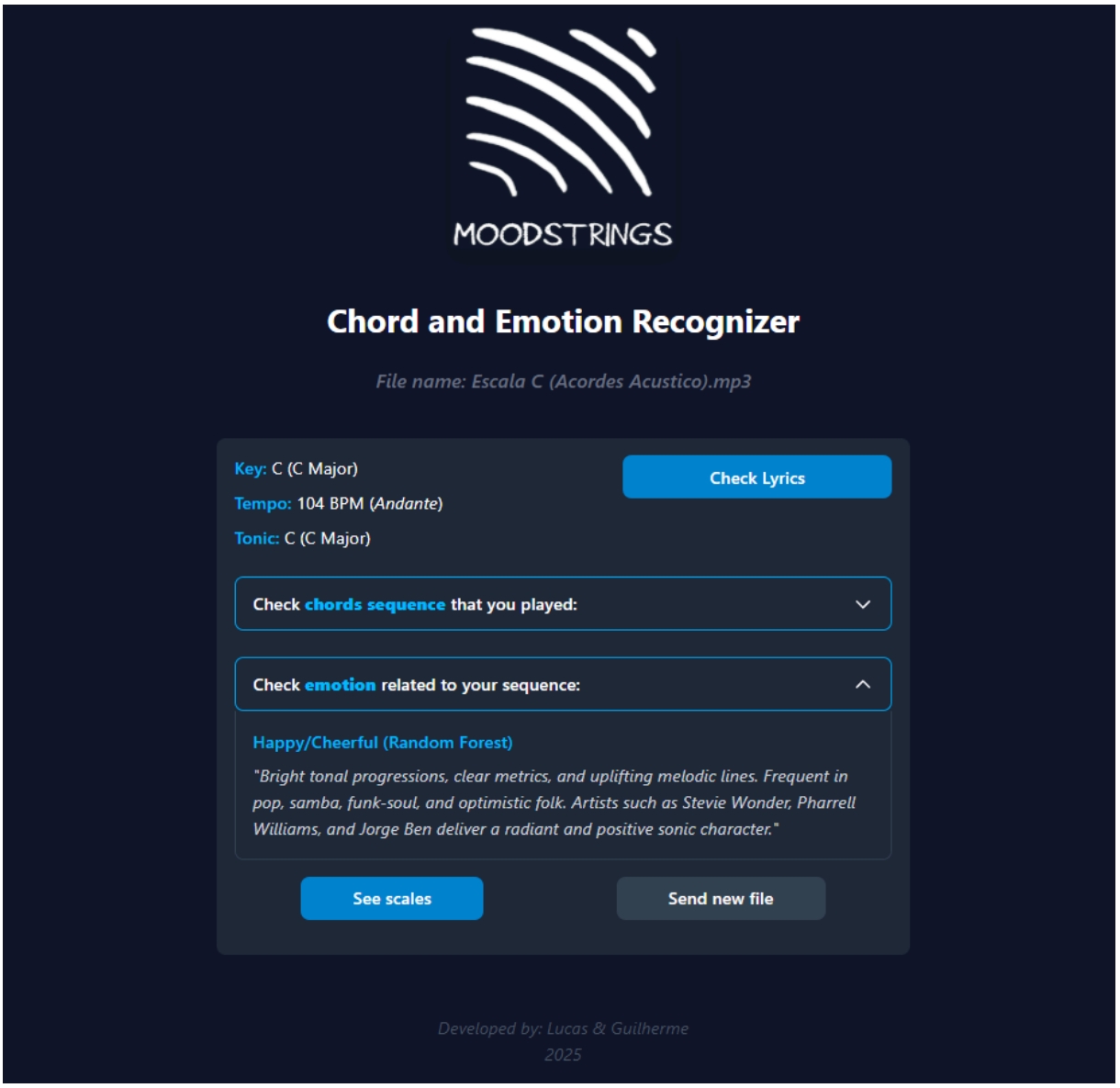
Figura 15: Apresentação dos acordes reconhecidos do áudio do usuário



Fonte: elaborado pelos autores (2025)

Seguindo, a Figura 15, segue com a apresentação do resultado da extração. Ela aparece logo após o envio ou gravação do arquivo via navegador, o usuário tem contato com essa tela que reconheceu aquilo que ele tocou, apresentado como “Sequência de acordes” (em inglês, *chord sequence*). Assim ele tem a opção de confirmar ou corrigir.

Figura 16: Página final - emoção



Fonte: elaborado pelos autores (2025)

Por fim, após a segunda tela apresentada, o usuário vê a Figura 16. Aqui contém informações relevantes sobre o que foi tocado, bem como é possível pegar informações sobre os acordes tocados (como função harmônica da escala e notas que o compõem), bem como a emoção presente. É possível também ver as escalas (botão *see scales*) para ver a atual e a relativa ou selecionar a opção de enviar outro arquivo. Por fim, também há o botão *Check lyrics* para abrir informações extraídas da letra da música enviada para análise.

É característico os tons azuis para a paleta de cores do projeto, bem como seu padrão escrito em inglês (comum em ferramentas). Isso foi determinado arbitrariamente entre os

autores, exceção do idioma que é necessário manter o que é encontrado em outras ferramentas (alguns nomes de acordes podem variar do inglês e português, por exemplo).

4.7 Processos de validação

Como grande importância do projeto, principalmente para a validação da sua coerência com o que foi planejado, usou-se alguns processos para avaliação dos sistema, são eles:

4.7.1 Validações manuais

A primeira e mais usada validação foi a manual. Desse modo os autores conseguiram, por tentativa e erro e simulação de usuário, encontrar defeitos e brechas que poderiam interferir se não fossem corrigidas. O teste detectou inconsistências como na nomeação dos acordes e suas funções, campos sem validação e mensagem de erros não tratados. Assim, foi possível melhorar a qualidade e usabilidade geral do projeto.

4.7.2 Validação da análise harmônica

Algo próximo da análise anterior, as informações melódicas extraídas foram comparadas manualmente por músicos, como professores, guitarristas e conhecidos. Por meio dessa validação teórica, algumas incongruências em escalas e escalas relativas, acordes nomeados erroneamente e interpretação das emoções classificadas foram corrigidas.

4.8 Resultado final e comparações com outros projetos

No desenvolvimento, é necessário a comparação com projetos similares para parametrizar corretamente os valores encontrados. Dessa forma, houve análise com o *software* Samplab (SAMPLAB, 2025) para os resultados da extração dos acordes e para as emoções extraídas, um estudo intitulado de *Audio-based Deep Music Emotion Recognition* (LIU; HAN; MA; GUO, 2018) foi usado em paralelo com os resultados da presente pesquisa.

Também foi feita uma última análise com o modelo de classificação construído ao longo da pesquisa. Com a Tabela 5 é possível notar uma melhoria final e extração do máximo potencial do método *Random Forest* abordado durante o desenvolvimento. Assim, obteve-se uma acurácia maior de 61%.

Tabela 5: Análise de resultado do modelo final

Análise de resultado - Modelo final				
Emoção	***Precision***	***Recall***	***F1-Score***	***Support***
Irritado (*angry*)	0,78	0,77	0,78	316
Feliz (*happy*)	0,57	0,55	0,56	316
Romântica (*romantic*)	0,55	0,56	0,56	316
Triste (*sad*)	0,55	0,57	0,56	316
Acurácia			**61,28%**	1264
Média	0,61	0,61	0,61	1264
Média proporcional a classe	0,61	0,61	0,61	1264

Fonte: elaborado pelos autores (2025)

4.8.1 Comparação de reconhecimento de acordes

Para comparação da precisão dos áudios tocados gravados, o *software web* Samplab (SAMPLAB, 2025) foi utilizado, mas com ressalvas:

1. O programa possui versão gratuita limitada, então apenas 10 segundos foram liberados para teste. Isso não tem uma influência no resultado final, já que os áudios comparados possuem praticamente a mesma duração;
2. O sistema de comparação utiliza inteligência artificial para detecção dos acordes, já a presente pesquisa faz isso de maneira de análise de áudio e captação de acordes por intervalo entre notas na faixa gravada.

Então, veja a Tabela 6 que contém os resultados das comparações feitas entre Moodstrings e Samplab (SAMPLAB, 2025). A primeira coluna contém o que foi realmente tocado, da mesma forma os resultados apresentados são os acordes reconhecidos. A nomenclatura universal foi utilizada, então pode haver discordância com a língua falada, como por exemplo o acorde *Amaj7* que pode ser interpretado *A7M* em alguns lugares (*lá maior com sétima maior*). Outro ponto é a presença da sigla "*sus*", que significa *suspenso*. No caso, essa escrita indica que a terça que forma o acorde foi trocada pela segunda (ex: *Csus2*) ou quarta (ex: *Csus4*).

Tabela 6: Comparação de reconhecimento de acordes

Comparação de reconhecimento de acordes		
Sequência gravada	**Acordes reconhecidos**	
	Moodstrings	**Samplab**
C,D,E,F,G,A,B,C	C,D,E,F,G,A,B,C	Em,Dm,Asus4,Em,Esus4,F,G
Am-Dm-Em-Fm	Am,Dm,Em,Em	Am,Dm,Em,Csus4
A7M-Bm7-E7-E7-Bm7-A7M	Amaj7,E7,E7,B,A,Amaj7,Amaj7	Amaj7,Bm,Asus4,E7,E7
G,C,E,F#,F#,E,C,G	G,G,C,E,F#,F#,E,C,G,G	G,Am,C,Esus4,F#m

Fonte: elaborado pelos autores (2025)

Assim, é possível afirmar que, levando em consideração que a rotina de reconhecimento de acordes trabalha exclusivamente com análise de intervalo de notas de uma faixa de áudio em contraposto com um sistema que usa IA, essa funcionalidade provou-se suficiente e satisfatória para o estudo. Os áudios foram gravados com um microfone dinâmico, ou seja, que captou apenas o instrumento (violão), com afinação padrão, ambiente silencioso e tocados sequencialmente sem dinâmicas no ritmo.

4.8.2 Comparação da inteligência artificial de classificação de emoção

É possível traçar um paralelo entre os modelos de classificação de emoções do *Moodstrings* e o modelo apresentado no paper *Audio-based Deep Music Emotion Recognition* (LIU; HAN; MA; GUO, 2018). Na pesquisa em questão, os autores adotaram uma abordagem distinta para alcançar um classificador emocional satisfatório. Para isso, foi desenhada uma CNN (*Convolutional Neural Network*) capaz de receber o espectrograma de uma música como entrada e, a partir dele, realizar a predição das emoções.

Ao comparar os resultados obtidos pelo *Moodstrings* com os testes descritos no estudo citado, é possível observar na Tabela 7 que a precisão do modelo desenvolvido neste trabalho foi superior àquela alcançada pelos experimentos utilizando SVM. No entanto, também se torna evidente que o uso de um algoritmo mais robusto, como a CNN em vez de *Random Forest*, proporcionaria resultados ainda melhores.

Tabela 7: Testes CNN versus SVM (*Audio-based Deep Music Emotion Recognition*)

Model	acc 1	acc 2	acc 3	acc 4	acc 5	acc6	acc7	acc8	acc9	acc10	Average
CNN	0.707	0.683	0.659	0.756	0.756	0.732	0.780	0.732	0.683	0.756	0.724
SVM	0.370	0.345	0.382	0.403	0.385	0.390	0.400	0.394	0.399	0.386	0.385

Fonte: adaptado de LIU et al., 2018.

4.9 Limitações técnicas encontradas

Apesar do planejamento inicial ter sido fundamentado e bem congruente, o desenvolvimento passou por empecilhos técnicos que foram superados. O primeiro ponto trata-se da transcrição de áudio.

Esse procedimento é técnico e bem delicado. Nos testes iniciais, qualquer que fosse o ruído presente na hora da gravação (como buzinas na rua, portas batendo ou conversas altas) compactuava para resultados insatisfatórios como ausência de notas na extração que foram, de fato, tocadas, ou até entendimento errado das mesmas. Após intensos períodos de testes manuais e estudos, esse problema foi, em sua maioria, resolvido. Eventuais sons externos altos podem ainda ser comprometedores, mas em situações normais, o processo de transcrição é satisfatório.

O segundo destaque é quanto à inteligência artificial, especialmente na criação de um *dataset* conciso. O conjunto de dados iniciais escolhido possuía centenas de milhares de itens, o que tornou etapas de extração de informações algo de grande peso computacional. Isso impactou diretamente no tempo de treinamento dos modelos, tempo de execução e precisão dos resultados. As técnicas aplicadas para superação desses pontos já foram descritas nos tópicos anteriores, o que foi de grande valia para a pesquisa.

Por fim, um desafio enfrentado foi a análise dos arquivos midi, tendo em vista que as bibliotecas utilizadas para extração de dados do *dataset* XMIDI não são completamente precisas e, por isso, inconsistências ocorreram no treinamento do modelo de classificação.

5 MELHORIAS FUTURAS

5.1 Transcrição de áudio

Considerando o desenvolvimento, riscos e a complexidade envolvidos nesta funcionalidade, a ampliação do suporte a outros formatos de áudio constitui uma melhoria relevante. Extensões comuns como *.wav*, *.opus* e *.wma*, frequentemente utilizadas em gravações de celulares e aplicativos de mensagens como o WhatsApp, poderiam aumentar a acessibilidade do sistema para usuários iniciantes ou com menor familiaridade tecnológica. Assim, o alcance da aplicação se ampliaria, contribuindo para a disseminação do interesse musical, um dos objetivos fundamentais estabelecidos neste projeto.

5.2 Aumento da precisão

Por mais que o atual percentual de acurácia do modelo desenvolvido foi satisfatório, a aplicação de técnicas mais avançadas de processamento de dados e refinamento de inteligência artificial para que fossem alcançados patamares próximos dos 80 a 85% de precisão seriam ideais. Isso não descredibiliza a pesquisa efetuada, por outro lado complementaria todos os processos transpassados para a conclusão do mesmo.

5.3 Classificação do gênero

Seguindo os mesmos processos para a identificação de emoções a partir de elementos harmônicos, a implementação de um módulo para predição do gênero musical representaria uma extensão natural do projeto. O próprio conjunto de dados utilizado para a emoção possui essa informação. Tal aprimoramento adicionaria uma camada adicional de complexidade e valor analítico, incentivando ainda mais estudos sobre padrões harmônicos e estruturais característicos de diferentes estilos musicais, nos quais a inteligência artificial torna-se uma ferramenta determinante.

5.4 Padrões de design

Embora a identidade visual atual tenha atendido aos requisitos do projeto, uma atualização das paletas de cores, padrões de *frontend* e diretrizes de usabilidade em conformidade com tendências recentes poderiam aprimorar a experiência do usuário. Essa modernização favorece maior aderência às expectativas das novas gerações e fortaleceria o apelo visual da plataforma.

5.5 Partituras

Algumas funcionalidades adicionais foram identificadas como potenciais incrementos para o projeto. Um dos pontos promissores é a possibilidade de disponibilizar o *download* da progressão harmônica em formato de partitura, ampliando sua utilidade para músicos e estudantes. De modo complementar, também seria viável permitir a importação de partituras para análise, possibilitando a classificação de emoções com base nas informações extraídas da escrita musical com um algoritmo robusto de transcrição de arquivo e interpretação musical.

5.6 Incorporar outras abordagens

Ao longo do projeto, foram encontradas, por meio de pesquisa, outras iniciativas com objetivos semelhantes, embora fundamentadas em bases bibliográficas diferentes. Entre elas, destacam-se trabalhos baseados no Modelo Circumplexo das Emoções proposto por James A. Russell (1979, 1980), que utiliza indicativos análogos às emoções selecionadas neste projeto. Abordagens como essa poderão, futuramente, ter seus elementos incorporados ao projeto, e assim possibilitar a colheita de novos resultados e aprimoramentos para a classificação. Como exemplo de modelos que utilizam essa abordagem, podemos citar o modelo Music2Emo, proposto por Kang e Herremans (2025), que combina o Modelo Circumplexo das Emoções com estratégias de aprendizado de máquina.

6 CONCLUSÃO

É evidente que o desenvolvimento de uma aplicação integrada a um modelo de inteligência artificial bem treinado pode representar uma solução significativa para os desafios enfrentados por estudantes e profissionais no estudo e na compreensão da teoria musical.

Sob a ótica acadêmica, a criação de uma ferramenta desse tipo tem o potencial de estimular novas pesquisas e estudos que contribuam para o avanço da inteligência artificial e para seu entendimento no contexto musical, também como para a musicoterapia (BENENZON, 1989).

Sob a ótica social, recursos como este constituem um ponto de partida relevante para facilitar e ampliar a democratização do acesso ao ensino musical.

Ao longo do projeto, observou-se que a combinação de técnicas de ciência de dados e de inteligência artificial, aliada a uma pesquisa intensa e ao entendimento do tema, foi essencial para conduzir o trabalho de forma mais eficiente e alcançar os melhores resultados possíveis, dentro das limitações, concluindo com um modelo treinado com aproximadamente 62% de precisão.

Ademais, ressalta-se que, embora os resultados tenham sido satisfatórios, a ampliação do conjunto de dados, adoção de modelos mais avançados de classificação e aplicação de uma ferramenta mais sensível para interpretação harmônica são claras oportunidades de aprimoramento. Do ponto de vista científico, o presente estudo complementa pesquisas existentes e é uma referência clara quanto a integração de inteligência artificial no campo da música, fornecendo processos replicáveis e contribuindo para um campo ainda pouco explorado. Entretanto, destaca-se a importância do uso responsável da tecnologia, que deve atuar como suporte ao aprendizado musical, preservando o caráter artístico e subjetivo da música.

Diante do exposto, os resultados alcançados demonstram que a integração entre tecnologia e música abre caminhos promissores para a ciência, a sociedade e a educação, ao mesmo tempo em que aponta para possíveis aprimoramentos capazes de maximizar seu impacto.

REFERÊNCIAS

ANTHEMSCORE. **AnthemScore**: automatic music transcription software. Versão 4. [S. l.]: Lunaverus, 2025. Disponível em: https://www.lunaverus.com/. Acesso em: abr. 2025.

APPTUTS. **Sites que utilizam inteligência artificial para criar músicas**. [S. l.], 2025. Disponível em: https://www.apptuts.net/tutorial/informatica/sites-inteligencia-artificial-criam-musicas/. Acesso em: 15 fev. 2025.

CELEMONY. **Melodyne**. Munique: Celemony Software GmbH, 2025. Disponível em: https://www.celemony.com/. Acesso em: abr. 2025.

CHAN, Paul Yaozhu; DONG, Minghui; LI, Haizhou. The Science of Harmony: A Psychophysical Basis for Perceptual Tensions and Resolutions in Music. **Research**, Washington D.C., v. 2019, Article ID 2369041, 2019. Disponível em: https://doi.org/10.34133/2019/2369041. Acesso em: 03 nov. 2025.

CHORD AI. **Chord AI**: real-time chord recognition from audio. [S. l.], 2025. Disponível em: https://chordai.net/. Acesso em: abr. 2025.

CHORDWATCH. **Chordwatch**: real-time MIDI chord analyzer. [S. l.], 2025. Disponível em: https://www.chordwatch.com/. Acesso em: abr. 2025.

DEAN, Roger T. *et al*. **The influence of tempo and harmonic complexity on emotional responses to music**. 2023. Preprint (Psychology). Disponível em: https://osf.io/preprints/psyarxiv/kdqgx_v1. Acesso em: 06 abr. 2025.

DIGITALOCEAN. **S.O.L.I.D**: The First Five Principles of Object Oriented Design. [S. l.], 2025. Disponível em: https://www.digitalocean.com/community/conceptual-articles/s-o-l-i-d-the-first-five-principles-of-object-oriented-design. Acesso em: 1 jul. 2025.

DOCKER INC. **Docker**: empowering app development for developers. Palo Alto: Docker Inc., 2025. Disponível em: https://www.docker.com. Acesso em: 25 jun. 2025.

GITHUB. **GitHub**. São Francisco: GitHub Inc., 2025. Disponível em: https://github.com. Acesso em: 25 jun. 2025.

GOVER, Matan; ZEWI, Oded. Music translation: generating piano arrangements in different playing levels. In: INTERNATIONAL SOCIETY FOR MUSIC INFORMATION RETRIEVAL CONFERENCE (ISMIR), 23., 2022, Bengaluru. **Proceedings** [...]. Bengaluru: ISMIR, 2022. Disponível em: https://archives.ismir.net/ismir2022/paper/000003.pdf. Acesso em: 15 fev. 2025.

HUMPHREY, Eric J.; BELLO, Juan P.; LECUN, Yann. Feature learning and deep architectures: new directions for music informatics. **Journal of Intelligent Information Systems**, v. 41, n. 3, p. 461-481, 2013.

JUSLIN, Patrik N.; SLOBODA, John A. Music and emotion: theory and research. **Psychology of Music**, v. 29, n. 1, p. 70–81, 2001. Disponível em: https://doi.org/10.1080/08098130109478028. Acesso em: 6 abr. 2025.

KANG, Jaeyong; HERREMANS, Dorien. **Towards unified music emotion recognition across dimensional and categorical models**. 2025. arXiv preprint arXiv:2502.03979. Disponível em: https://arxiv.org/abs/2502.03979. Acesso em: 18 nov. 2025.

LEE, Chuan-Chung; AGRES, Kat; HERREMANS, Dorien. Composing music using genetic algorithms and other evolutionary techniques. *In*: INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE (ICAI), 2013, Las Vegas. **Proceedings** [...]. Las Vegas: CSREA Press, 2013. p. 1–7. Disponível em: https://worldcomp-proceedings.com/proc/p2013/ICA2548.pdf. Acesso em: 6 abr. 2025.

LIBROSA DEVELOPERS. **Librosa**: audio and music processing in Python. Versão 0.10.0. [S. l.], 2025. Disponível em: https://librosa.org. Acesso em: 06 ago. 2025.

LIU, T.; HAN, L.; MA, L.; GUO, D. Audio-based deep music emotion recognition. *AIP Conference Proceedings*, v. 1967, p. 040021, 2018. Disponível em: https://doi.org/10.1063/1.5039095. Acesso em: 24 nov. 2025.

MICROSOFT RESEARCH. **Tutorial on AI Music Composition**. In: **ACM MULTIMEDIA CONFERENCE**, 2021. Disponível em: https://www.microsoft.com/en-us/research/uploads/prod/2021/10/Tutorial-on-AI-Music-Composition-@ACM-MM-2021.pdf. Acesso em: 6 mar. 2025.

MIDO DEVELOPERS. **Mido**: MIDI Objects for Python. Disponível em: https://mido.readthedocs.io. Acesso em: 25 jun. 2025.

PYTHON SOFTWARE FOUNDATION. **Python**. Versão 3.11. [S. l.], 2025. Disponível em: https://www.python.org. Acesso em: 25 jun. 2025.

RAFFEL, Colin. **pretty_midi**: utilities and data structures for handling MIDI data. Disponível em: https://github.com/craffel/pretty-midi. Acesso em: 25 jun. 2025.

RAMOS DOMINGUES, Lucas. **moodstrings-app**. 2025. Repositório de código. Disponível em: https://github.com/PixeLarm12/moodstrings-app. Acesso em: 03 nov. 2025.

ROBERT, James. **Pydub**: manipulate audio with a simple and easy high level interface. [S. l.], 2025. Disponível em: https://pypi.org/project/pydub/. Acesso em: 06 ago. 2025.

RUSSELL, James A. A circumplex model of affect. **Journal of Personality and Social Psychology**, v. 39, n. 6, p. 1161-1178, 1980.

SAINBURG, Tim. **Noisereduce**: noise reduction using spectral gating in Python. [S. l.], 2025. Disponível em: https://pypi.org/project/noisereduce/. Acesso em: 06 ago. 2025.

SCIKIT-LEARN. **Scikit-learn**: machine learning in Python. [S. l.], 2025. Disponível em: https://scikit-learn.org/stable/. Acesso em: 25 jun. 2025.

SILVA, Júlio Corrêa Barros. **Inteligência artificial e música**. 2023. Trabalho de Conclusão de Curso (Graduação) – Universidade de São Paulo, São Paulo, 2023. Disponível em: https://bdta.abcd.usp.br/directbitstream/b4caa262-3f96-4f91-a0d8-9c8cb05e751b/tc5022-Julio-Silva-Inteligencia.pdf. Acesso em: 9 mar. 2025.

SONGSTERR. **Songsterr Tabs with Rhythm**. [S. l.], 2025. Disponível em: https://www.songsterr.com/. Acesso em: abr. 2025.

SONIC VISUALISER. **Sonic Visualiser**: a tool for viewing and analysing audio files. Londres: Queen Mary, University of London, 2025. Disponível em: https://sonicvisualiser.org/. Acesso em: abr. 2025.

SPOTIFY AB. **Basic-pitch**: a lightweight yet powerful audio-to-MIDI converter with pitch bend detection. Estocolmo: Spotify, 2025. Disponível em: https://github.com/spotify/basic-pitch. Acesso em: 06 ago. 2025.

STURM, Bob L. *et al*. **Music transcription modelling and composition using deep learning**. arXiv preprint arXiv:1604.08723, 2016. Disponível em: https://arxiv.org/abs/1604.08723. Acesso em: 3 mar. 2025.

STURM, Bob L. T.; BEN-TAL, Oded. **AI Music 2020**: Paper 31. [S. l.], 2020. Disponível em: https://boblsturm.github.io/aimusic2020/papers/CSMC__MuMe_2020_paper_31.pdf. Acesso em: 9 mar. 2025.

TAILWIND LABS. **Tailwind CSS**: a utility-first CSS framework. [S. l.], 2025. Disponível em: https://tailwindcss.com. Acesso em: 06 ago. 2025.

TIANGOLO. **FastAPI**: fast (high-performance), web framework for building APIs with Python 3.6+. Disponível em: https://fastapi.tiangolo.com. Acesso em: 25 jun. 2025.

TRELLO. **Trello**. [S. l.]: Atlassian, 2025. Disponível em: https://trello.com/pt-BR. Acesso em: 25 jun. 2025.

VUE.JS. **The Progressive JavaScript Framework**. [S. l.], 2025. Disponível em: https://vuejs.org. Acesso em: 25 jun. 2025.

XMUSIC-PROJECT. **XMIDI_Dataset**: a large-scale symbolic music dataset with emotion and genre labels. Disponível em: https://github.com/xmusic-project/XMIDI_Dataset. Acesso em: 01 jul. 2025.

ZAHLER, Noel. Artificial Intelligence and Music: Implementing an Interactive Computer Performer. **Computer Music Journal**, v. 15, n. 2, p. 38-46, 1991. Disponível em: https://www.researchgate.net/publication/200806044. Acesso em: 9 mar. 2025.

ZHANG, Hangbo *et al*. **MusicBERT**: symbolic music understanding with large-scale pre-training. Redmond: Microsoft Research, 2022. Disponível em: https://microsoft.github.io/muzic/musicbert/. Acesso em: 06 abr. 2025.

SAMPLAB. **Chord Detection**. [S. l.], 2025. Disponível em: https://samplab.com/chord-detection. Acesso em: 24 nov. 2025.

BENENZON, Rolando O. **Teoria da musicoterapia**: contribuição ao conhecimento do contexto não-verbal. São Paulo: Summus Editorial, 1989. Disponível em: https://books.google.com.br/books?hl=pt-BR&lr=&id=wuwi-in0CK0C&oi=fnd&pg=PA11&dq=musicoterapia&ots=o2cKlgPYeZ&sig=rc9Xdo1G3Qtr4K8NTUIe1zoJVy4&redir_esc=y#v=onepage&q&f=false. Acesso em: 08 dez. 2025.

UBERI. **speech_recognition**. Disponível em: <https://github.com/Uberi/speech_recognition\>. Acesso em: 08 dez. 2025.