UNIVERSIDADE FEDERAL DE CAMPINA GRANDE
CENTRO DE ENGENHARIA ELÉTRICA E INFORMÁTICA
UNIDADE ACADÊMICA DE SISTEMAS E COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ANDERSON SALES DE MENEZES

CIFRA TUBE:

UMA FERRAMENTA PARA CIFRAGEM DE MÚSICAS UTILIZANDO VÍDEOS DO YOU TUBE

CAMPINA GRANDE - PB

2019

ANDERSON SALES DE MENEZES

CIFRA TUBE:

UMA FERRAMENTA PARA CIFRAGEM DE MÚSICAS UTILIZANDO VÍDEOS DO YOU TUBE

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Orientador: Professor Dr. Nazareno Andrade.

CAMPINA GRANDE - PB

2019

M543c Menezes, Anderson Sales de.
Cifra Tube : uma ferramenta para cifragem de músicas utilizando vídeos do You Tube. / Anderson Sales de Menezes. – 2019.

10 f.

Orientador: Prof. Dr. Nazareno Andrade.
Trabalho de Conclusão de Curso - Artigo (Curso de Bacharelado em Ciência da Computação) - Universidade Federal de Campina Grande; Centro de Engenharia Elétrica e Informática.

1. Cifra musical. 2. Música - cifras. 3. Dados públicos. 4. Aplicação online. 5. Desenvolvimento de aplicação - música. 6. Ferramenta para cifra de músicas. 7. You Tube – vídeos e cifras. I. Andrade, Nazareno. II. Título.

CDU:004(045)

Elaboração da Ficha Catalográfica:

Johnny Rodrigues Barbosa
Bibliotecário-Documentalista
CRB-15/626

ANDERSON SALES DE MENEZES

CIFRA TUBE:

UMA FERRAMENTA PARA CIFRAGEM DE MÚSICAS UTILIZANDO VÍDEOS DO YOU TUBE

Trabalho de Conclusão Curso apresentado ao Curso Bacharelado em Ciência da Computação do Centro de Engenharia Elétrica e Informática da Universidade Federal de Campina Grande, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

BANCA EXAMINADORA:

Professor Dr. Nazareno Andrade
Orientador – UASC/CEEI/UFCG

Professor Dr. Eanes Torres Pereira
Examinador – UASC/CEEI/UFCG

Professor Dr. Tiago Lima Massoni
Examinador – UASC/CEEI/UFCG

Trabalho aprovado em: 02 de julho de 2019.

CAMPINA GRANDE - PB

CifraTube: uma ferramenta para cifragem de músicas utilizando vídeos do YouTube

Anderson Sales de Menezes
Universidade Federal de Campina Grande
Campina Grande, Paraíba, Brasil
anderson.menezes@ccc.ufcg.edu.br

1 Resumo

A cifra é uma notação muito utilizada para aprender músicas, principalmente por músicos amadores. No entanto, não existem ferramentas simples e grátis para criar e acompanhar as cifras de forma interativa junto com a música. Portanto, esse trabalho visa o desenvolvimento de uma ferramenta para cifragem interativa de músicas utilizando vídeos do YouTube, que é uma plataforma popular que permite o acesso à milhões de músicas de forma gratuita. Através dessa ferramenta, os usuários poderão aprender músicas de maneira mais fácil e precisa.

2 Introdução

Cifra e partitura são sistemas de notação musical muito conhecidos e utilizados no processo de aprendizagem de músicas. O primeiro é mais utilizado por músicos iniciantes e o segundo por músicos mais experientes, por ser um sistema mais complexo e que exige mais conhecimento de teoria musical.

Uma das principais diferenças entre essas notações é que a cifra é estritamente harmônica, podendo apenas indicar os acordes que acompanham a voz, que é representada pela letra da canção. Já a partitura é capaz de representar harmonia, melodia, ritmo, expressividade, entre outros, que são elementos musicais igualmente importantes [1].

Diante da grande diferença de capacidade de representação entre esses dois sistemas e da inviabilidade de cifrar músicas completamente instrumentais, fica evidente a necessidade de uma ferramenta que consiga agregar à cifra formas de representar mais elementos, porém sem aumentar a dificuldade de aprendizado do sistema, para que músicos que sabem ler a cifra tradicional possam desfrutar de mais recursos com o conhecimento que ele já tem.

O objetivo deste trabalho é desenvolver uma ferramenta que consiste em um site que utiliza o mesmo princípio da cifra, porém substituindo a letra da música por um vídeo do YouTube — que é uma das plataformas mais usadas para ouvir música [2] e seu player possui uma API simples e fácil de usar [3].

Atualmente, as soluções mais similares a esta proposta são baseadas em ferramentas de detecção automática do andamento da música e acordes. No entanto, elas não funcionam tão bem, não são gratuitas e algumas não dão a opção de o usuário criar a própria cifra de forma manual. Os sites mais populares desse tipo são o Chordify [4] e o Yalp [5]. Ambos são pagos, mas possuem uma versão grátis com recursos limitados.

3 Solução

3.1 Descrição

O CifraTube é uma ferramenta online que permite a cifragem de músicas, de forma interativa, utilizando vídeos do YouTube. O site traz vantagens para todos os tipos de músicos: do casual ao profissional, do iniciante ao experiente.

Os músicos mais experientes, por exemplo, que costumam fazer suas próprias cifras, podem anotar acordes e outras observações (Figura 1(e)) na barra de reprodução do vídeo. Cada anotação fica registrada no instante determinado (Figura 1(d)), de forma que ao reproduzir o vídeo novamente, as notas aparecem em destaque na tela (Figura 1(a, b)). De modo semelhante a sites populares como Cifra Club [6] e Cifras.com.br [7], a cifra criada pode ser salva junto com o link do vídeo utilizado, para ser acessada posteriormente e pode ser reproduzida por outros músicos que eventualmente desejem aprender a mesma música. A ferramenta também facilita a identificação do momento que cada acorde deve ser tocado, de forma mais precisa, sem depender da letra da música e sem precisar conhecê-la previamente, como acontece com as cifras tradicionais.

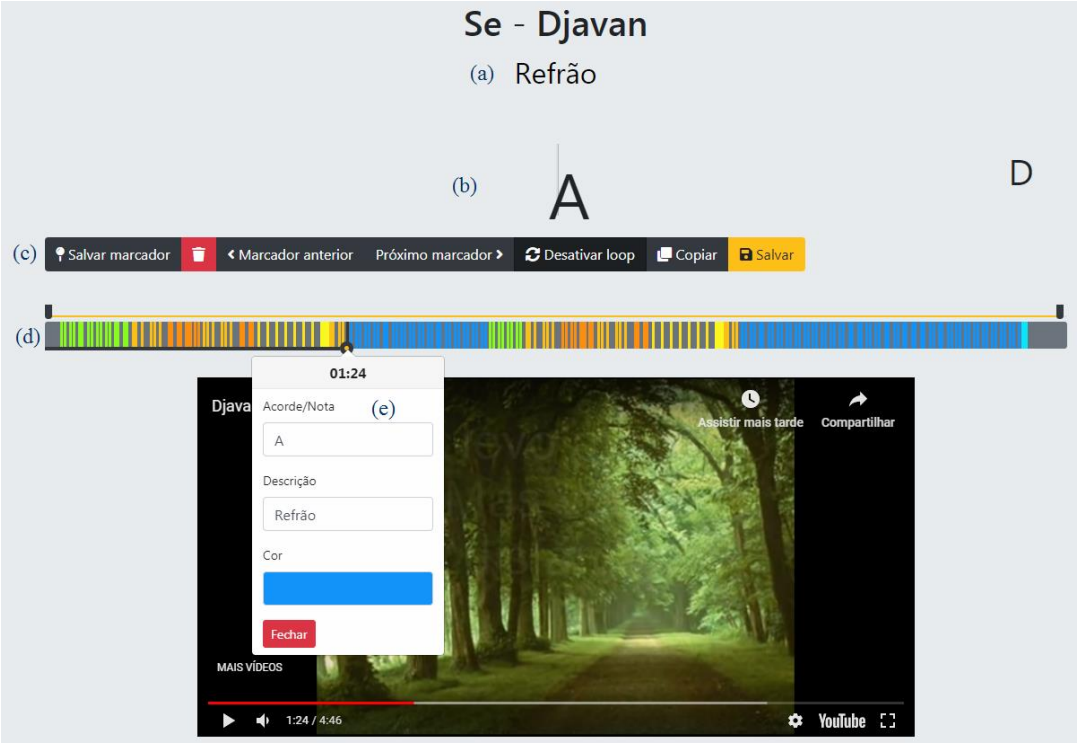

Figura 1 - Tela principal do CifraTube na visão do usuário que está cifrando a música. (a) Descrição do marcador; (b) barra de acordes; (c) menu de funcionalidades; (d) barra de reprodução e marcadores; (e) editor de marcador

Para deixar o processo de cifragem mais simples e prático, algumas funcionalidades são essenciais e foram implementadas (Figura 2):

- Manipular e navegar pelos marcadores com o teclado: praticidade de o usuário não precisar mexer no mouse após digitar um acorde, além de evitar cliques em marcadores errados.
- Copiar e colar: a maioria das músicas possuem trechos com os mesmos acordes e seria trabalhoso e redundante anotar os mesmos acordes várias vezes.
- Loop de uma região: quando um músico está tirando uma música "de ouvido", geralmente é necessário ouvir um mesmo trecho repetidamente.
- Marcadores coloridos: o usuário tem a opção de escolher a cor de cada marcador, facilitando a identificação de trechos da música, como introdução e refrão.

Apesar de ser uma ferramenta totalmente gratuita, é necessário fazer um cadastro no site para criar novas cifras. Assim, somente o usuário que as criou pode editá-las.

3.2 Arquitetura

Por se tratar de uma aplicação web com persistência de dados, esse projeto é separado em front-end e back-end. O front-end é a parte da aplicação que roda no browser, é a interface que permite o usuário interagir com o sistema. Já o back-end é a parte do servidor, que é responsável pelo gerenciamento dos dados: manipular, salvar, recuperar, servir, etc.

3.2.1 Tecnologias do back-end

No back-end foi utilizado Node.js com o framework Express, para o servidor, e MongoDB para a persistência dos dados. O Node.js é uma tecnologia que facilita o desenvolvimento de APIs REST, por ser relativamente simples de configurar. Como seu código é escrito em JavaScript, assim como o front-end, o processo torna-se mais agradável e eficaz para o desenvolvedor, que não precisa ficar transicionando entre diferentes linguagens de programação. O MongoDB é um banco de dados NoSQL

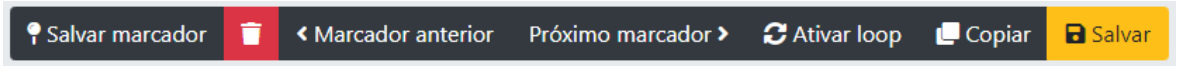

Figura 2 - Menu de funcionalidades

orientado a documentos, cujos dados são representados no formato JSON. Assim como o Node.js, é simples de instalar e implementar. A maior vantagem dessas tecnologias é que elas são de código aberto, não sendo necessário pagar para utilizá-las.

3.2.2 Estrutura do back-end

O servidor dessa aplicação fornece uma API REST, que é consumida pelo lado do cliente.

Como é possível observar na Figura 3, o código é organizado em cinco categorias principais: controllers, helpers, models e routers.

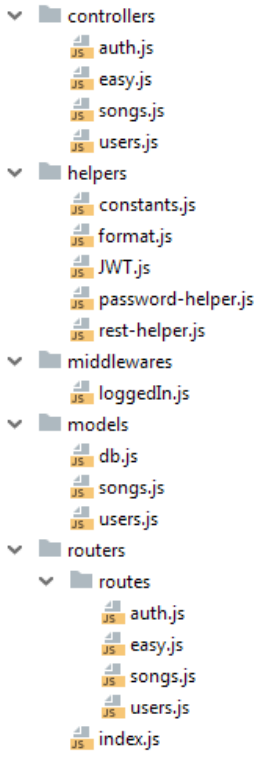

Figura 3 - Estrutura de diretórios do servidor

Os controllers são responsáveis pela parte lógica do servidor e processam as chamadas HTTP. Possuem funções que são executadas de acordo com a rota do servidor que foi acessada. No arquivo auth.js são processadas as chamadas de login, users.js lida com os cadastros de usuários e songs.js processa as chamadas referentes às cifras, como salvar, atualizar, listar.

Os helpers, como o próprio nome sugere, possuem funções auxiliares. O arquivo password-helper.js, por exemplo, possui uma função que criptografa a senha do usuário antes de salvar no banco de dados. Já em rest-helper.js, existe uma função que converte objetos JavaScript para JSON e envia como resposta para o cliente.

Os middlewares também são responsáveis pela parte lógica do servidor. Antes de uma requisição chegar em um controller, ela pode passar por um middleware, que verifica se o usuário está logado no sistema, por exemplo, como no caso do arquivo loggedIn.js.

Os models são responsáveis por definir o esquema de cada modelo que será usado no banco de dados e alguns comportamentos específicos do modelo. Cada atributo do modelo e seu respectivo tipo de dado são determinados aqui. O modelo de usuário possui os atributos *nome*, *senha*, *email* e *data de criação*, e os tipos são *String*, *String*, *String* e *Date*, respectivamente, como mostra a Figura 4. Para facilitar a comunicação do servidor com o MongoDB, foi utilizada a biblioteca mongoose.

Nos routers é onde são definidos os endpoints do servidor e cada rota aponta para uma função de um controller. Também é aqui que são determinadas as rotas protegidas, através dos middlewares, como se pode ver na Figura 5.

```
const userSchema = Schema({
    name: {
        type: String,
        required: true
    },
    password: {
        type: String,
        select: false
    },
    email: {
        type: String,
        required: true,
        unique: true
    },
    createdDate: {
        type: Date,
        required: true,
        default: new Date()
    }
}, { collection: 'user',
    toObject: {
        transform: function (doc, ret) {
            delete ret.id;
            delete ret.password;
        },
        getters: true
    }
});
```

Figura 4 - Esquema do modelo de usuário

3.2.3 Autenticação de usuários

Como as APIs REST são stateless, ou seja, não guardam estado e não possui sessões, como outros tipos de servidores, a autenticação de usuários aqui é feita através de tokens JWT. Ao fazer login, o cliente recebe um token de autenticação gerado do lado do servidor e o guarda. Toda chamada que for feita do lado do cliente, em nome do

```
var express = require('express');
var usersRouter = express.Router();
var usersController = require('../../controllers/users');
var loggedInMW = require('../../middlewares/loggedIn');

usersRouter.post('/register', usersController.registerUser);
usersRouter.put('/:id', loggedInMW.verifyJWToken, usersController.updateUser);
```

Figura 5 - Código que define os endpoints para cadastro e atualização de usuários. Na linha 10, o endpoint de atualização é protegido pelo middleware loggedInMW, que verifica se o usuário está logado antes de chamar a função updateUser no controller de usuários

usuário logado, deve enviar esse token no cabeçalho Authorization, para que ele seja validado e para que seja possível identificar quem está logado antes do servidor prosseguir com o processamento da chamada. Cada token carrega informações criptografadas, como o usuário que é seu "dono" e o seu tempo de validade. Para facilitar essa implementação, a biblioteca jsonwebtoken foi utilizada.

3.2.4 Tecnologia do front-end

A tecnologia escolhida para ser utilizada do lado do cliente foi o framework Angular 7, que é escrito em TypeScript, uma linguagem muito parecida com JavaScript, porém tipada e com alguns recursos adicionais. Além de ter uma documentação bastante detalhada, possui uma grande quantidade de componentes feitos pela comunidade, o que agiliza consideravelmente a implementação de uma solução. Uma das suas maiores vantagens é que, diferente de outros frameworks como jQuery e React, o desenvolvedor não precisa misturar código de lógica com código HTML, contribuindo para que o código TypeScript final seja legível.

3.2.5 Estrutura do front-end

A Figura 6 mostra como é organizado o código do lado do cliente.

Nas pastas components, home, login e register, ficam os HTMLs, CSSs e a maior parte da lógica de todas as páginas do sistema.

Em directives ficam as diretivas do Angular, que permitem o desenvolvedor associar determinados comportamentos a elementos HTML.

No front-end também existem páginas protegidas que só podem ser acessadas se o usuário estiver logado. Isso é feito através dos guards.

Os helpers são códigos que interceptam todas as chamadas HTTP feitas e acrescentam o token de autenticação no cabeçalho, caso o usuário esteja logado, além de verificar erros nas respostas do servidor.

Os models, assim como no back-end, definem os modelos que serão utilizados na aplicação e possuem os mesmos atributos dos modelos do servidor. Essa é um das principais diferenças do Angular para o Angular.js: ao fazer uma chamada que deve ter como resposta determinado modelo, já se sabe exatamente quais os atributos do objeto que será recebido.

Os services são responsáveis pelas chamadas ao servidor.

No arquivo app.module.ts é onde todas as dependências da aplicação são injetadas e no app-routing.module.ts é onde as rotas são definidas. Cada rota aponta para uma página do sistema. Os arquivos app.component são relativos à estrutura principal da aplicação, considerando que o Angular é um framework que adota princípios SPA (aplicação de página única), fazendo com que apenas parte da página seja recarregada conforme a interação do usuário, tornando a experiência mais agradável e fluida.

Em assets ficam os recursos de imagens que são utilizados pela aplicação e em environments ficam os arquivos de configuração para cada ambiente em que o sistema será executado: produção e desenvolvimento.

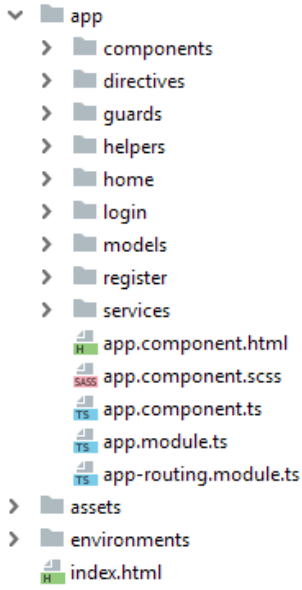

Figura 6 - Estrutura de diretórios do cliente

3.2.6 API do player incorporado do YouTube

Esse projeto é possível graças a API que o reprodutor de vídeos do YouTube fornece [3]. O módulo ngx-youtube-player encapsula essa API e facilita o uso com Angular.

Através dela é possível carregar os vídeos que o usuário deseja utilizar para fazer a cifra, obter informações do vídeo que está reproduzindo, como o tempo decorrido em segundos desde o início da reprodução, bem como manipular propriedades, como por exemplo, ir para um determinado instante do vídeo de forma automática.

3.2.7 Lógica da aplicação

Para criar uma cifra, o usário preenche um pequeno formulário, indicando a URL do vídeo que ele quer cifrar, o título da música e o artista. Esses dados ficam salvos em um objeto do tipo Song, que é um modelo, e seus atributos podem ser vistos na Figura 7.

O objeto *markers* funciona como um mapa que guarda todos os marcadores e suas chaves são o instante do vídeo, em segundos, com uma casa decimal, que o marcador foi adicionado pelo usuário. Cada marcador, por sua vez, guarda as seguintes informações: sua cor, a nota que deve ser tocada naquele instante e uma descrição, que é opcional.

A API provê uma função que detecta quando o estado do player muda, então sempre que o vídeo é reproduzido, uma função da aplicação é executada a cada 100 milisegundos, através do método setInterval() de JavaScript, para checar se há algum marcador naquele instante e atualizar os marcadores e os dados que devem ser mostrados na tela. Esse tempo é rápido o suficiente para mostrar com precisão o momento exato em que a nota deve ser tocada durante a música, sem demandar muito processamento do dispositivo do usuário. Além disso, o método getCurrentTime() do player retorna um número com várias casas decimais, então é necessário arredondar para uma casa decimal, possibilitando o acesso aos marcadores no mapa. Por exemplo: se o instante retornado pelo player for 5,125000, será verificado se existe um marcador com a chave 5,1. Como mostra a Figura 8, as chaves dos marcadores são números com uma casa decimal.

```
export class Song {
  id: string;
  markers: any;
  author: User;
  title: string;
  artist: string;
  videoTitle: string;
  videoUrl: string;
  videoId: string;
  views: number;
  createdDate: Date;
}
```

Figura 7 - Atributos do modelo Song

Figura 8 - Exemplo de objeto markers

A posição dos marcadores na barra de reprodução é calculada por uma regra de três simples. Utiliza-se o tamanho da barra, a duração do vídeo e o instante em que o marcador foi adicionado. Através do Document Object Model (DOM), é possível descobrir o tamanho da barra, que é um elemento *div* no código HTML. Já a duração do vídeo é fornecida pela API do player. Cada marcador também é um *div* com posição absoluta e o resultado do cálculo define o valor da propriedade *left*. Usando os dados da Figura 9 como exemplo, o cálculo seria:

$$\frac{360}{100} = \frac{720}{x}$$
$$360x = 7200$$
$$x = 200$$

Logo, o marcador do instante 100s estaria a 200 pixels de distância do início da barra de reprodução.

Um cálculo semelhante é feito para achar o instante a partir de um clique na barra, considerando que um evento JavaScript fornece o valor relativo em pixels de onde ocorreu o clique.

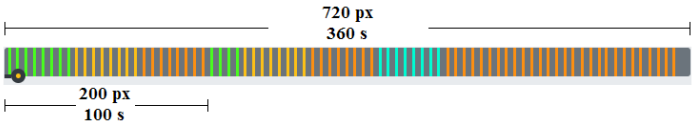

Figura 9 - Barra de reprodução de tamanho 720 pixels para vídeo com duração de 360 segundos

Como explicado anteriormente, a cada 100 milissegundos os dados na tela são atualizados. Uma das funções

executadas nesse intervalo é a *updateVisibleMarkers*. Essa função resulta em uma lista com os marcadores cujos acordes deverão aparecer na tela, além de definir o tamanho e a posição das notas na barra de acordes. São considerados os marcadores que se encontram num intervalo de mais ou menos 2 segundos do tempo de reprodução atual. Considerando que B é o tamanho da barra, I é o instante do marcador e A é o tempo de reprodução atual, a seguinte fórmula é utilizada para calcular a posição dos acordes na barra:

$$\frac{B(I - A + 2)}{2 - (B/2)}$$

Já o tamanho da fonte é calculada de acordo com o instante do marcador. Se o instante for menor do que o tempo de reprodução atual, a fórmula 1 é utilizada. Caso contrário, a 2 é utilizada. Assim, quando o acorde está no centro da barra, o tamanho deve ser 4,5 rem e quando está nas extremidades deve ser 0 rem.

$$\frac{4{,}5(I - A + 2)}{2} \quad (1)$$

$$\frac{4{,}5|I - A - 2|}{2} \quad (2)$$

3.2.8 Ambiente de produção

O servidor da aplicação é hospedado na Amazon Web Services (AWS) através do serviço Elastic Beanstalk, que gerencia toda a parte da infraestrutura automaticamente. Essa plataforma foi escolhida principalmente pela facilidade de fazer o *deploy*.

Para hospedar o banco de dados, foi escolhido o MongoDB Atlas, pois é o serviço oficial do MongoDB e possui um plano grátis. Apesar de possuir um limite de 512MB de armazenamento, é suficiente para este projeto. Além disso, também possui gerenciamento automático da infraestrutura e provê réplicas do banco, aumentando a disponibilidade e a tolerância a falhas.

Também foi adquirido o domínio **cifratube.com**, para que a aplicação seja acessada de forma fácil. O serviço de DNS utilizado foi o Amazon Route 53, já que se encontra dentro do mesmo ambiente da plataforma da AWS.

4 Experiência

4.1 Processo de desenvolvimento

Inicialmente, foi decidido quais tecnologias seriam utilizadas no projeto. Essas escolhas foram baseadas, principalmente, na experiência de outros trabalhos. Após isso, foi feita uma análise de requisitos, onde foram definidas as funcionalidades principais do sistema. Para cada funcionalidade, foi realizado um estudo para determinar a viabilidade técnica, considerando as tecnologias escolhidas. Em seguida, foi implementado um protótipo, como mostra a Figura 10, com o objetivo de concretizar a exequibilidade do projeto. Ele possuía apenas uma tela com as funcionalidades básicas: adicionar e editar marcadores, mostrar os acordes em destaque na tela, enquanto o vídeo do YouTube é reproduzido, e salvar a cifra em um arquivo, já que ainda não havia uma estrutura de back-end.

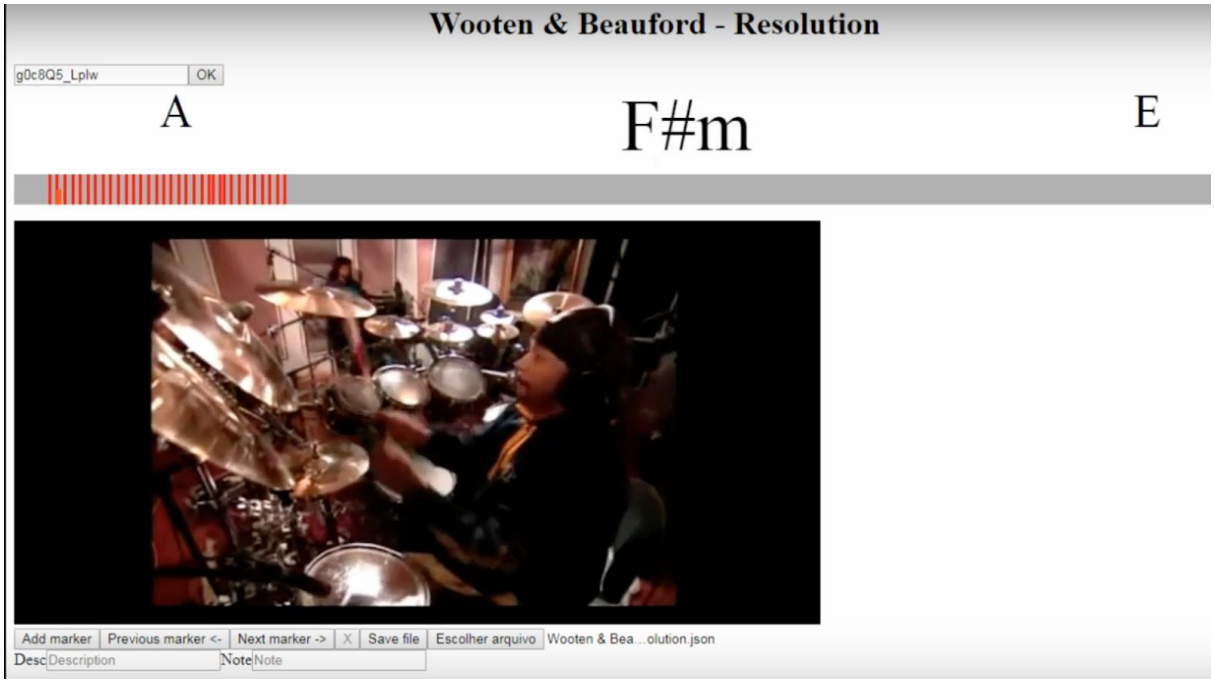

Figura 10 - Única tela do protótipo do CifraTube

De forma iterativa e incremental, o protótipo foi sendo aprimorado e novas funcionalidades foram sendo implementadas. Sempre que um novo recurso era implementado, eram realizados testes e correções de bugs.

4.2 Desafios

A criação de uma interface simples, fácil de usar e visualmente agradável, foi a parte mais desafiadora do projeto. Por se tratar de um produto cujos potenciais usuários já utilizam outras plataformas consolidadas no mercado, o CifraTube precisa ser prático, além de inovador, para atrair usuários.

Além disso, a implementação da barra de acordes com animação não foi uma tarefa fácil. Até chegar nas fórmulas matemáticas utilizadas para determinar a posição e o tamanho da fonte de cada acorde na barra, foi necessário fazer vários testes e ajustes, para que o comportamento esperado fosse alcançado.

4.3 Trabalhos futuros

Trabalhos futuros podem estender o CifraTube para competir com os sites de cifras já existentes, através da implementação de mais algumas funcionalidades:

- Dicionário de acordes: necessário para que músicos menos experientes possam tocar músicas com acordes que ele ainda não conhece;
- Transposição do tom da música;
- Mover marcadores;
- Dar a opção de o usuário fazer/ver a cifra tradicional, com a letra da música;
- Desfazer ações (Ctrl + Z).

Também deverão ser feitas algumas melhorias no aspecto visual do site, para aprimorar a experiência do usuário.

5 Referências

[1] Fortes, Fabrício Pires. Pensamento simbólico e notação musical. Dissertação de mestrado, Universidade Federal de Santa Maria, 2009. Disponível em: https://repositorio.ufsm.br/bitstream/handle/1/9084/FORTES%2C%20FABRICIO%20PIRES.pdf.

[2] More music is played on youtube than on spotify, apple music and every other audio streaming platform combined. Disponível em https://www.musicbusinessworldwide.com/more-music-is-played-on-youtube-than-on-spotify-apple-music-and-every-audio-streaming-platform-combined/.

[3] YouTube Player API Reference for iframe Embeds. Disponível em https://developers.google.com/youtube/iframe_api_reference?csw=1.

[4] Chordify: Instant chords for any song. Disponível em https://chordify.net/.

[5] Yalp - Chords for any song. Disponível em https://www.yalp.io/.

[6] Cifra Club - seu site de cifras e tablaturas. Disponível em https://www.cifraclub.com.br/.

[7] CIFRAS.COM.BR | Site de cifras e tablaturas. Disponível em https://www.cifras.com.br/