

igtí



# RELATÓRIO PROJETO APLICADO

**Instituto de Gestão e Tecnologia da Informação**  
**Relatório do Projeto Aplicado**

# Metrônomo Parametrizável

Gabriel Barbosa de Oliveira

Orientador(a):

Marcos Prochnow

23/08/2022



Gabriel Barbosa de Oliveira

INSTITUTO DE GESTÃO E TECNOLOGIA DA INFORMAÇÃO

RELATÓRIO DO PROJETO APLICADO

# Metrônomo Parametrizável

Relatório de Projeto Aplicado  
desenvolvido para fins de conclusão do  
curso Desenvolvedor Front End.

Orientador (a): Marcos Prochnow

Lorena

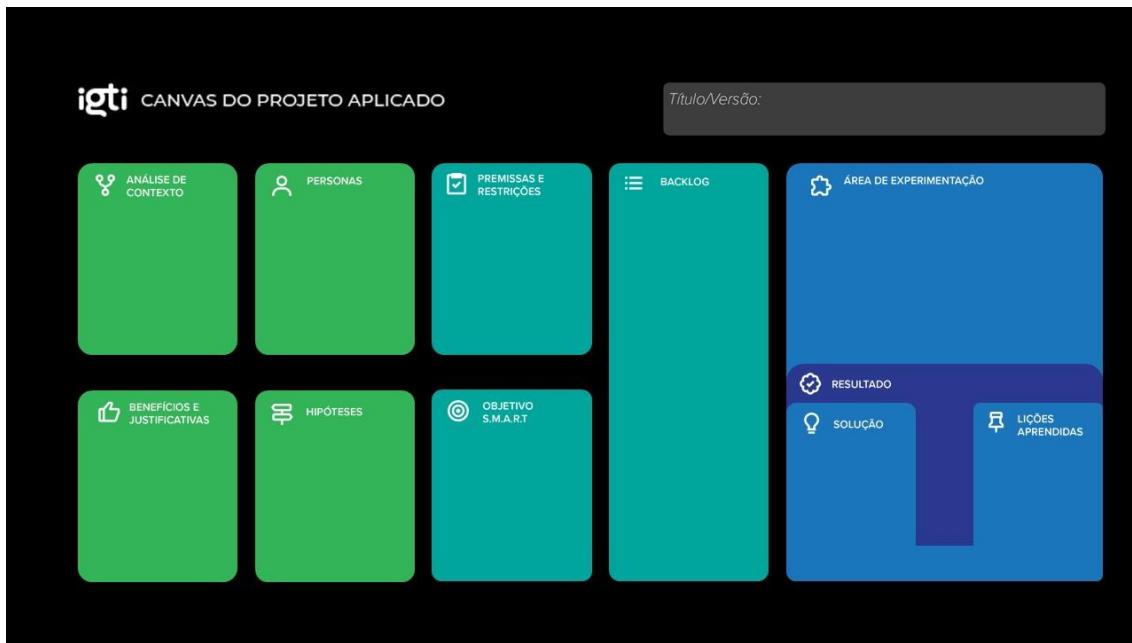
23/08/2022

# Sumário

1. CANVAS do Projeto Aplicado	4
1.1 Desafio	5
1.1.1 Análise de Contexto	5
1.1.2 Personas	8
1.1.3 Benefícios e Justificativas	10
1.1.4 Hipóteses	12
1.2 Solução	14
1.2.1 Objetivo SMART	14
1.2.2 Premissas e Restrições	15
1.2.3 Backlog de Produto	16
2. Área de Experimentação	17
2.1 Sprint 1	17
2.1.1 Solução	17
• Evidência do planejamento:	17
• Evidência da execução de cada requisito:	17
• Evidência dos resultados:	22
2.1.2 Experiências vivenciadas	24
2.2 Sprint 2	25
2.2.1 Solução	25
• Evidência do planejamento:	25
• Evidência da execução de cada requisito:	26
• Evidência dos resultados:	29
2.2.2 Experiências vivenciadas	35
2.3 Sprint 3	36
2.3.1 Solução	36
• Evidência do planejamento:	36
• Evidência da execução de cada requisito:	37
• Evidência dos resultados:	42
2.3.2 Experiências vivenciadas	45
3. Considerações Finais	46
3.1 Resultados	46
3.2 Contribuições	47
3.3 Próximos passos	48

## 1. CANVAS do Projeto Aplicado

Figura conceitual, que representa todas as etapas do Projeto Aplicado.



## 1.1 Desafio

### 1.1.1 Análise de Contexto

Uma das etapas mais difíceis do estudo musical é relacionada a independência rítmica. Para que ela seja atingida com sucesso, o músico necessita conseguir acentuar cada nota em seu respectivo tempo gerando uma variação do ritmo de um compasso musical. O estudante que deseja se aprofundar nesse tipo de refinamento técnico necessita de uma ferramenta muito específica chamada metrônomo.

O metrônomo é capaz de marcar andamentos musicais recorrentes, ou seja, ele é o responsável por determinar a velocidade que um trecho ou uma música completa terá em sua execução. Seu funcionamento é medido em batidas, a primeira batida sempre é mais forte e as subsequentes mais fracas. Porém, ele nem sempre representa o que é tocado pelo músico.

Por exemplo, vamos supor que um compasso x tem quatro batidas e o músico precisa caracterizá-lo como um blues. Neste cenário, a ferramenta convencional não consegue representar o que precisa ser tocado e acentuado. Pois ele está marcando fortemente a primeira batida quando deveria marcar a segunda e a quarta para que caracterizasse o estilo musical em questão. A ideia é criar um web app com um metrônomo personalizável capaz de deixar essas acentuações mais visuais e sonoras para quem usa a ferramenta para estudo e mais didático para quem leciona.

Matriz CSD			
	Certezas	Suposições	Dúvidas
Atores	Estudantes de música necessitam de metrônomo para estudo	Estudantes de música não dão atenção o suficiente para acentuações de notas em seus respectivos tempos	Estudantes de música dariam atenção devida para uma ferramenta de metrônomo personalizável?
	Professores de música precisam ensinar alunos a usar um metrônomo para estudo	Professores de música poderiam utilizar uma ferramenta para refinar a didática e fixar melhor o entendimento dos alunos a respeito do assunto	Professores de música usariam uma ferramenta que foge dos métodos tradicionais de ensino?
	Produtores musicais necessitam de metrônomos para gravação de músicas	Produtores musicais poderiam utilizar uma ferramenta para facilitar gravação de músicas com bandas menos experientes	Uma ferramenta com marcação de tempo diferente ajudaria as bandas a gravarem com mais facilidade?
Cenários	Em um estudo individual, ajudaria a fixar as acentuações em tempos fortes	Ajudaria o estudante de música a conseguir aumentar a velocidade e qualidade de execução mais rapidamente	Existe risco de virar um aprendizado apenas momentâneo?
	Em uma aula de música ajudaria a didática do professor	Ajudaria o aluno a entender e compreender como estudar mais tranquilamente	Existe risco de não haver necessidade de tanta complexidade?
	Em uma gravação musical ajudaria uma banda ou músico a gravar com mais facilidade	O músico conseguiria se sentir mais à vontade com as marcações de tempo mesmo que ainda não tenha experiência gravando.	Existe possibilidade de o músico gravar sem metrônomo?
Regras	A ferramenta necessariamente precisa permitir marcação de	Esse seria o diferencial da ferramenta	Apenas essa funcionalidade é

	acentuação de tempos personalizadas		suficientemente atrativa?
	A ferramenta necessariamente precisa permitir funcionamento padrão de um metrônomo tradicional	Possibilita que todos os perfis de músico utilizem a ferramenta	Os músicos entenderiam esses conceitos facilmente?

POEMS				
Pessoas	Objetos	Ambiente	Mensagem	Serviços
Estudante de música	Instrumentos musicais, computador, celular. Objetos não são relacionados, mas são utilizados em paralelo	Não se aplica		
Professor de Música	Instrumentos musicais, computador, celular. Objetos não são relacionados, mas são utilizados em paralelo	Não se aplica		
Produtor Musical	Computador	Não se aplica		

### 1.1.2 Personas

**Persona 1: Gabriel Idade: 25 anos**

**O que vê?**

- Trabalha em um mundo majoritariamente corporativo.
- Está sempre ativo em bandas de rock.
- Pensa que precisa melhorar suas habilidades como músico
- Acredita que a tecnologia pode ajudar bastante no processo

**O que ouve?**

- Está sempre ligado a influenciadores de programação e musicais.
- Tem ídolos que estão ligados majoritariamente a música.
- Ouve que é sempre necessário estar em constante evolução

**O que ele fala e faz?**

- Costuma dizer que ama música
- Sempre procura estudar utilizando metrônomo
- Busca sempre estar em constante evolução na parte profissional e musical
- Procura estudar música como Hobbie.
- Ama compor músicas próprias.

**Quais suas dores?**

- Tem medo de passar vergonha nos shows que faz com sua banda
- Tem receio de fracassar em tarefas importantes do trabalho
- Odeia ter que lidar com tarefas maçantes e monótonas

**Quais suas necessidades?**

- Necessita de ferramentas que auxiliem nos estudos diários

**Persona 2: Luciano Idade: 35 anos**

**O que vê?**

- Trabalha como professor de música
- Está ligado inteiramente ao mundo da música pois possui um grupo de jazz

**O que ouve?**

- Sempre está escutando os mais diversos artistas ligados ao jazz
- Apaixonado por guitarra
- Tocar parece muito difícil

**O que ele fala e faz?**

- Costuma dizer que ama música
- Ama compor músicas próprias
- Ama estar em contato com os alunos

**Quais suas dores?**

- Sente-se na necessidade de ter sempre várias didáticas na mão.
- É difícil conciliar aulas e estudo pessoal

**Quais suas necessidades?**

- Necessita de ferramentas que auxiliem nos estudos diários

### 1.1.3 Benefícios e Justificativas

#### Perfil do Cliente

- **Tarefas do Cliente**

Tocar algum instrumento musical

Tocar notas com acentuação correta

- **Dores**

Estresse com acentuações erradas

Não consegue evoluir qualidade e velocidade das notas tocas

Ser visto como um músico ruim

- **Ganhos**

Ter tranquilidade para executar ritmos complexos

Ser visto como um músico competente

Evoluir mais facilmente com acentuação de notas

Evoluir velocidade de execução

#### Proposta de Valor

- **Produtos e Serviços**

Metrônomo personalizável

- **Analgésicos**

Controle de execução no instrumento

Ser visto como um bom músico

- **Criadores de Ganho**

Evolução mais rápida

Menos monotonia na prática do instrumento

ITENS	DETALHAMENTO
<b>OBJETIVOS</b>	<ul style="list-style-type: none"> <li>• Pegar Instrumento Musical</li> <li>• Ligar metrônomo</li> <li>• Selecionar notas a serem acentuadas</li> <li>• Tocar junto com as acentuações selecionadas</li> </ul>
<b>ATIVIDADES</b>	<ul style="list-style-type: none"> <li>• Pegar Instrumento Musical</li> <li>• Ligar metrônomo</li> </ul>
<b>QUESTÕES</b>	<ul style="list-style-type: none"> <li>• Como fazer para corrigir acentuações erradas se a marcação do metrônomo me confunde?</li> </ul>
<b>BARREIRAS</b>	<ul style="list-style-type: none"> <li>• Não existe ferramenta que me permita customizar acentuações para estudar</li> </ul>
<b>AÇÕES DO CLIENTE</b>	<ul style="list-style-type: none"> <li>• Quer evoluir na independência rítmica</li> <li>• Procura métodos de estudo</li> <li>• Estuda</li> </ul>
<b>FUNCIONALIDADES</b>	<ul style="list-style-type: none"> <li>• Marca tempo das batidas</li> <li>• Permite personalização de batidas fortes</li> <li>• Permite personalização de comprimento dos compassos</li> <li>• Emite sons fortes e fracos</li> </ul>
<b>INTERAÇÃO</b>	<ul style="list-style-type: none"> <li>• Ajudar estudantes a conseguir melhorar técnica com seus respectivos instrumentos</li> <li>• Feedback sonoro dos tempos selecionados</li> </ul>
<b>ONDE OCORRE</b>	<ul style="list-style-type: none"> <li>• Sala</li> <li>• Sala de Aula</li> <li>• Quarto</li> <li>• Qualquer ambiente com um dispositivo que possa acessar a internet</li> </ul>
<b>TAREFAS APARENTEIS</b>	<ul style="list-style-type: none"> <li>• Vídeos com músicos utilizando a ferramenta</li> <li>• Demonstração de facilidade de evolução</li> </ul>

## TAREFAS ESCONDIDAS

## PROCESSOS DE SUPORTE

## SAÍDA DESEJÁVEL

- Sistema de gerenciamento de execução de sons
- Sistema de controle de velocidade
- Divulgação via redes sociais da ferramenta
- Implantação na prática com professores de música
- Estimular crescente de parte técnica nos alunos
- Abordar uma nova maneira de estudar
- Buscar excelência em visual
- Buscar excelência em performance
- Ter uma experiência prazerosa e que ajude os estudantes de música

### 1.1.4 Hipóteses

#### Matriz de observações para hipóteses

- **Risco Alto e Alto Valor Percebido**
  - Funcionalidade de acompanhamento de evolução
  - Plataforma de acompanhamento para usuários
  - Login de usuário
- **Risco Baixo e Alto Valor Percebido**
  - Funcionalidade de parametrizar tempos fortes
  - Funcionalidade de parametrizar quantidade de execuções
- **Risco Baixo e Baixo Valor Percebido**
  - Funcionalidade de metrônomo comum
- **Risco Alto e Baixo Valor Percebido**
  - Personalização de usuário

#### Priorização de Ideias

Iniciativas	B	A	S	I	C	O	Soma	Ranking
Criação metrônomo personalizável	4	4	4	5	4	3	24	1
Alteração em alguma plataforma de ensino existente	3	2	2	2	3	3	15	2

Escala	B - Benefícios	A - Abrangência	S - Satisfação	I - Investimentos	C - Cliente	O - Operacionalidade
5	De vital importância	Total (de 70 a 100%)	Muito grande	Pouquíssimo investimento	Nenhum impacto	Muito fácil
4	Significativo	Muito grande (de 40 a 70%)	Grande	Algum investimento	Impacto pequeno	Fácil
3	Razoável ( de 20 a 40%)	Média	Médio investimento	Médio impacto	Média facilidade	
2	Poucos benefícios	Pequena ( de 5 a 20%)	Pequena	Alto investimento	Impacto grande	Difícil
1	Algum benefício	Muito pequena	Quase não é notada	Altíssimo investimento	Impacto muito grande no cliente	Muito difícil

**Fonte:** <https://engenhariaexercicios.com.br/gestao-de-qualidade/matriz-gut-basico-conceito-aplicacao-das-matrices-priorizacao/>.

## 1.2 Solução

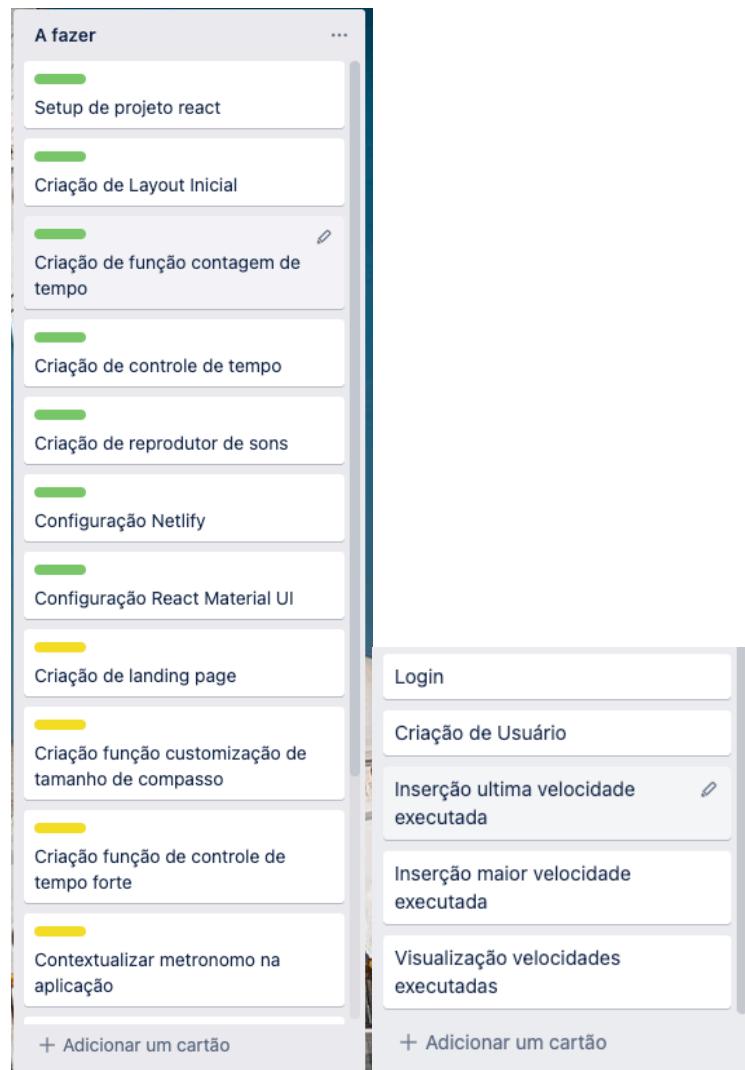
### 1.2.1 Objetivo SMART

Criação de uma web app de um metrônomo personalizável utilizando o período de 2 meses do projeto aplicado do IGTI, possibilitando que estudantes de música consigam praticar acentuações rítmicas mais eficientemente.

## 1.2.2 Premissas e Restrições

Risco Identificado	Impacto Potencial	Ações Preventivas	Ações Corretivas
Baixa Adesão	Alto	Layout e funcionalidades Impecáveis	Demonstração de eficiência por meio de profissionais do ramo
Dificuldade de entendimento do usuário	Moderado	Layout responsivo e textos bem explicativos	Ux bem definido

## 1.2.3 Backlog de Produto



## 2. Área de Experimentação

### 2.1 Sprint 1

O foco da primeira sprint foi na definição da tecnologia a ser utilizada no front-end e o desenvolvimento da função que servirá de base para a construção do metrônomô personalizável.

Considerando o objetivo de fixar o máximo possível de informações lecionadas no decorrer do curso, decidi que sairia da minha zona de conforto e utilizaria uma tecnologia na qual não estou tão habituado a trabalhar. Sendo assim a tecnologia selecionada foi o react.js

Como o foco dessa sprint foram as definições e o desenvolvimento da função base, foram definidas as seguintes ferramentas para auxilio na criação dessa aplicação

- Create-React-App para criação de toda a estrutura da aplicação SPA, foi utilizado o template de typescript
- Material Design (MUI) para estilização de componentes básicos como botões e inputs
- React-icons para ícones
- Howl.js para tratamento de áudio entre plataformas
- Scss para criações personalizadas de estilização de estrutura

#### 2.1.1 Solução

- Evidência do planejamento:

Para a Sprint 1 foram alocadas as seguintes tarefas



- Evidência da execução de cada requisito:

#### Tarefa Setup de Projeto react

Utilizado create-react-app para criar estrutura inicial da aplicação

```

CUSTOM-METRONOME
  > .netlify
  > netlify
  > node_modules
    < CUSTOM-METRONOME>
      "name": "custom-metronome",
      "version": "0.1.0",
      "private": true,
      "dependencies": {
        "@emotion/react": "^11.10.4",
        "@emotion/styled": "^11.10.4",
        "@mui/icons-material": "^5.10.3",
        "@mui/material": "^5.10.5",
        "@netlify/functions": "^1.2.0",
        "@testing-library/jest-dom": "^5.16.5",
        "@testing-library/react": "^13.4.0",
        "@testing-library/user-event": "^13.5.0",
        "@types/jest": "^27.5.2",
        "@types/node": "^16.11.58",
        "@types/react": "^18.0.19",
        "@types/react-dom": "^18.0.6",
        "howler": "^2.2.3",
        "lodash": "4.17.21",
        "react": "^18.2.0",
        "react-dom": "^18.2.0",
        "react-scripts": "5.0.1",
        "sass": "1.54.9",
        "typescript": "^4.8.3",
        "web-vitals": "2.1.4"
      },
      > Depurar
      "scripts": [
        "start": "react-scripts start",
        "build": "react-scripts build",
        "test": "react-scripts test",
        "eject": "react-scripts eject"
      ],
      > "eslintConfig": {...},
      > "browserslist": {
        "production": [...],
        "development": [...]
      },
      "devDependencies": {
        "@types/howler": "^2.2.7",
        "@types/lodash": "4.14.185"
      }
    
```

## Tarefa Criação do Layout

Essa atividade previa criar os componentes necessários para compor o layout base

```

return (
  <>
  <section className="container">
    <section className="metronome">
      <div className="bpm-display">
        <span className="tempo">{metronomeValue}</span>
        <span className="bpm">BPM</span>
      </div>
      <div className="tempo-text">Nice and steady</div>
      <NumberController onClick={this.handleBeatChange} component={
        <div className='slider-container'>
          <Slider
            aria-label="slider"
            defaultValue={60}
            min={20}
            max={280}
            track={false}
            value={metronomeValue}
            valueLabelDisplay="auto"
            onChange={this.changeValue}
          />
        </div>
      } />
    </section>
    <section className="action-button">
      <Button size="large" startIcon={currentPlayStatusComponent} onClick={this.handlePlayStatus}>
        {currentPlayStatusText}
      </Button>
    </section>
    <NumberController onClick={this.handleMeasuresChange} component={
      <div className="beats-number-container">{beatsNumber}</div>
    } />
    <span className="beats-per-measure-text">Beats per measure</span>
  </section>
</section>

```

```

* {
  margin: 0;
  padding: 0;
}

.container {
  display: flex;
  justify-content: center;
  align-items: center;
  height: 100%;
}

.metronome {
  display: flex;
  flex-direction: column;
  width: 50vw;
  height: 250px;
  justify-content: space-between;
}

.bpm-display {
  width: 100%;
  text-align: center;
  color: #1976d2;
  font-weight: bold;
}

.bpm-display .tempo {
  font-size: 5em;
}

.tempo-text {
  font-size: 15px;
  text-transform: uppercase;
  text-align: center;
}

```

```

.tempo-settings {
  display: grid;
  grid-template-columns: 30px auto 30px;
  grid-gap: 16px;
  justify-items: center;
}

.tempo-settings .adjust-tempo-btn {
  width: 30px;
  height: 30px;
  font-size: 40px;
  border-radius: 50%;
  border: 1px solid #ddd;
  text-align: center;
  cursor: pointer;
}

.tempo-settings .adjust-tempo-btn:hover {
  background: #1976d2;
  color: #fff;
}

.tempo-settings .decrease-tempo {
  line-height: 25px;
}

.tempo-settings .increase-tempo {
  line-height: 32px;
}

```

## Tarefa Criação de função contagem de tempo

Essa função é uma implementação de um timer que é acionado de tempos em tempos

```

Complexity is 7 It's time to do something...
function Timer(callback, timeInterval, options) {
  this.timeInterval = timeInterval;

  this.start = () => {
    this.expected = Date.now() + this.timeInterval;
    this.timeout = null;

    if (options.immediate) {
      callback();
    }

    this.timeout = setTimeout(this.round, this.timeInterval);
    console.log('Timer Started');
  }

  this.stop = () => {
    clearTimeout(this.timeout);
    console.log('Timer Stopped');
  }
}

Complexity is 3 Everything is cool!
this.round = () => {
  console.log('timeout', this.timeout);
  let drift = Date.now() - this.expected;
  if (drift > this.timeInterval) {
    if (options.errorCallback) {
      options.errorCallback();
    }
  }
  callback();
  this.expected += this.timeInterval;
  console.log('Drift:', drift);
  console.log('Next round time interval:', this.timeInterval - drift);
  this.timeout = setTimeout(this.round, this.timeInterval - drift);
}

export default Timer;

```

## Tarefa Criação da função controle de tempo

São os handles utilizados para decrementar e adicionar velocidade ao reproduutor de áudio

```
Complexity is 3 Everything is cool!
handleBeatChange = (clickedOption: string) => { █

    let newValue = this.state.metronomeValue;
    clickedOption == "add" ? newValue++ : newValue--;
    if (this.checkIfTimeNumberIsValid(newValue))
        this.setNewMetronomeValue(newValue)
}

Complexity is 4 Everything is cool!
checkIfTimeNumberIsValid = (value: number) => { █
    return value >= 20 && value <= 280 ? true : false;
}
```

```
Complexity is 3 Everything is cool!
changeValue = (event: any, value: any) => { █
    this.setNewMetronomeValue(value);
};

setNewMetronomeValue = (value: number) => {
    this.setState({
        metronomeValue: value
    });
    this.metronomeInstance.timeInterval = 60000 / this.state.metronomeValue;
}
```

## Tarefa Criação de reproduutor de sons

Essa função utiliza a biblioteca Howl js para reproduzir sons

```
private click1: Howl = new Howl({
    src: require('../click1.mp3')
});
private click2: Howl = new Howl({
    src: require('../click2.mp3')
});
```

```

Complexity is 4 Everything is cool!
handleMetronomeClick = () => {
  let { beatsNumber } = this.state;
  let newCount = this.state.count;
  console.log(newCount);
  if (newCount === beatsNumber) {
    newCount = 0;
  }
  if (newCount === 0) {
    this.click1.play();
  } else {
    this.click2.play();
  }
  newCount++;
  this.setState({
    count: newCount,
  });
}

```

### Tarefa Configuração React Material UI

Instalação e execução da biblioteca afim de agilizar a criação de alguns componentes

```

<Slider
  aria-label="slider"
  defaultValue={60}
  min={20}
  max={280}
  track={false}
  value={metronomeValue}
  valueLabelDisplay="auto"
  onChange={this.changeValue}
/>

```

### Tarefa Configuração Netlify

Criação e configuração de deploy para aplicação react



## Builds →

**custom-metronome** Completed

Production: master@[813411b](https://813411b.netlify.app/)



**custom-metronome** Completed

Deploy Preview #6: develop@[84c36db](https://84c36db.netlify.app/)



**custom-metronome** Completed

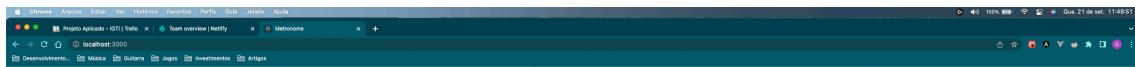
Deploy Preview #5: develop@HEAD

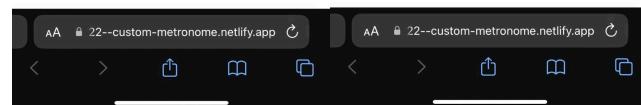
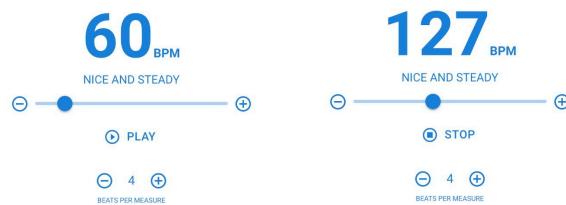
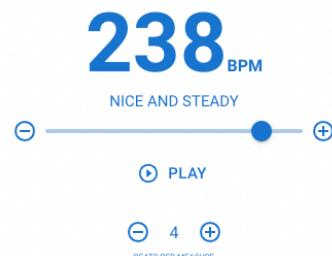
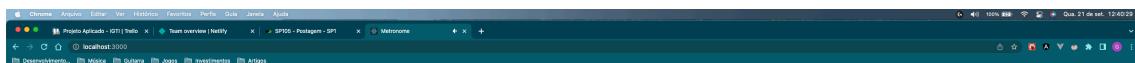


A aplicação se encontra online na seguinte url

<https://63286f56bfe1f032f64fa222--custom-metronome.netlify.app/>

- Evidência dos resultados:





```

0
Timer Started
timeout 11
1
Drift: 5
Next round time interval: 995
timeout 19
2
Drift: 2
Next round time interval: 998
timeout 24
3
Drift: 6
Next round time interval: 994
timeout 29
4
Drift: 5
Next round time interval: 995
timeout 34
1
Drift: 5
Next round time interval: 995
timeout 39
2
Drift: 5
Next round time interval: 995
timeout 44
3
Drift: 5
Next round time interval: 995
timeout 49
4
Drift: 5
Next round time interval: 995
timeout 54

```

## 2.1.2 Experiências vivenciadas

Durante a sprint 1 foram aprendidas as seguintes lições

- Para trabalhar com services em react é necessário que o controle de estado seja feito através de uma class componente em vez de uma function. Os components em forma de função se reconstroem a cada iteração dificultando muito a administração de instancias de classes externas.
- O autoplay da api de áudio do html5 é nativamente bloqueado por navegadores mobile sendo necessário intervir e forçar a execução dos áudios. Para tal feito foi utilizada a biblioteca Howl.js. Essa biblioteca é altamente conceituada para esse tipo de suporte sendo utilizada por grandes players do mercado. Tais como Google e Disney.
- Netlify é uma excelente ferramenta para deploy gratuito de aplicações front end. Através da configuração da mesma e da integração com o git. Foi possível montar um ambiente de CD (contínuos deployment) e notar o problema de autoplay em navegadores mobile.
- Foi necessário remanejar alguns backlogs inicialmente planejados no backlog de produto.

## 2.2 Sprint 2

O foco da segunda sprint foi na contextualização do produto em formato de aplicação. Sendo assim, foi utilizado react router para fazer transição de navegação entre as páginas.

Nessa sprint foram criados componentes para representar uma landing page e para inserir o metrônomo dentro de um fluxo de usuário. Durante a sprint houve necessidade de diversas tratativas para atender o formato de diferentes dispositivos.

Foram utilizadas as seguintes ferramentas nessa segunda sprint

- React Router 6.4.1
- Scss para estilização de estruturas e componentes personalizados
- React Material Drawer

### 2.2.1 Solução

- Evidência do planejamento:

Para a sprint 2 foram planejadas as seguintes tarefas:



- Evidência da execução de cada requisito:

### Tarefa criação de landing page

Para essa tarefa foi feita um componente composto de partials para a apresentação da ferramenta

```

1 import React, { Component } from 'react'
2 import Branding from '../partials/Branding/Branding'
3 import Footer from '../partials/Footer/Footer'
4 import UseItEverywhere from '../partials/UseItEverywhere/UseItEverywhere'
5 import HeaderMenu from '../shared/partials/HeaderMenu/HeaderMenu'
6 import './Landing.scss'
Complexity is 7 It's time to do something...
7 export default class Landing extends Component { █
    Complexity is 7 It's time to do something...
8     render() { █
9         return (
10             <>
11                 <HeaderMenu />
12                 <div className='landing-container'>
13                     <Branding />
14                     <UseItEverywhere />
15                     <Footer />
16                 </div>
17             </>
18         )
19     }
20 }

```

```

export default class Branding extends Component<{}, any> { █
Complexity is 13 You must be kidding
render() { █
    const imgUrl: string = require('../assets/img/example.jpeg');
    return (
        <>
            <section className='presentation-container'>
                <h1>Evolu sua prática com <br /><span id="brandingWord"></span></h1>
                <h3 className='sub-text'>O metronomo mais versátil. <br />Acessível diretamente do seu navegador </h3>
                <div className='branding-actions'>
                    <ThemeProvider theme={newTheme}>
                        <Button variant="contained" color="primary" component={Link} to="/metronome/">Comece aqui</Button>
                    </ThemeProvider>
                </div>
                <section className='branding-image'>
                    <img src={imgUrl} alt="app photo" />
                </section>
            </section>
        </>
    )
}

```

```

import React, { Component } from 'react'
import './UseItEverywhere.scss'
Complexity is 16 You must be kidding
export default class UseItEverywhere extends Component {
    Complexity is 16 You must be kidding
    render() {
        return (
            <section className='use-it-everywhere-container'>
                <header>
                    <h2>Use onde quiser</h2>
                    <h3>Tenha liberdade e portabilidade para seus estudos</h3>
                </header>
                <section className='devices-container'>
                    <div>
                        <img src={require("../assets/img/phone.png")} alt="Phone Web App picture" />
                        <h3>Celular</h3>
                    </div>
                    <div>
                        <img src={require("../assets/img/computer.png")} alt="Computer Web App picture" />
                        <h3>Computador</h3>
                    </div>
                    <div>
                        <img src={require("../assets/img/tablet.png")} alt="Tablet Web App picture" />
                        <h3>Tablet</h3>
                    </div>
                </section>
            </section>
        )
    }
}

```

```

import React, { Component } from 'react'
import './Footer.scss';
Complexity is 6 It's time to do something...
export default class Footer extends Component {
    Complexity is 6 It's time to do something...
    render() {
        return (
            <footer className='footer-container'>
                <hr />
                <div className='footer-copyright'>
                    <span>© Copyright 2022. Todos os direitos reservados.</span>
                </div>
            </footer>
        )
    }
}

```

## Tarefa criação de função para controle de tamanho de compasso

Teve como objetivo determinar quantas batidas um compasso deve ter.

```

Complexity is 4 Everything is cool!
handleMeasuresChange = (clickedOption: string) => {
    let newValue = this.state.beatsNumber;
    let newPulses: Array<IPulseControllerControlObject> = [];
    if (clickedOption === "add") {
        newValue++;
        newPulses = this.addNewPulse();
    } else {
        newValue--;
        newPulses = this.removeLastPulse();
    };
    if (this.checkIfBeatNumberIsValid(newValue)) {
        this.setState({
            beatsNumber: newValue,
            count: 0,
            currentPulses: newPulses
        })
    }
}

```

## Tarefa criação de função de controle de tempo forte

Teve como objetivo fazer controle de acentuações de notas.

```

addNewPulse(): Array<IPulseControllerControlObject> {
    const { currentPulses } = this.state;
    currentPulses.push({
        id: (currentPulses.length + 2).toString(),
        isActive: false,
        position: currentPulses.length
    })
    return currentPulses
}

removeLastPulse(): Array<IPulseControllerControlObject> {
    const { currentPulses } = this.state;
    currentPulses.pop();
    return currentPulses;
}

import React, { Component } from 'react';
import { IPulseControllerControlObject } from '../../shared/interfaces/props/IPulseControllerControlObject';
import { IPulseControllerProps } from '../../shared/interfaces/props/IPulseControllerProps';
import { IPulseControllerState } from '../../shared/interfaces/states/IPulseController';
// import './PulseController.scss';
Complexity is 0 Everything is cool!
export default class PulseController extends Component<IPulseControllerProps, IPulseControllerState> {
    private dots: Array<JSX.Element> = [];
    constructor(props: IPulseControllerProps) {
        super(props);
        this.state = {
            beatsNumber: 4,
        };
        this.mountDots();
    }
    componentWillMount() {
        this.dots = [];
        this.mountDots();
    }
    Complexity is 5 Everything is cool!
    mountDots() {
        this.props.pulses.forEach((element, index) => {
            const classActive: string = element.isActive ? "active" : "";
            this.dots.push(<span onClick={() => this.handleClick(element, index)} className={dot ${classActive}} key={element.position} id={element.position.toString()}></span>);
        });
    }
}

```

## Tarefa contextualizar metrônomo na aplicação

Teve como objetivo contextualizar o produto dentro de uma aplicação com fluxo definido

```
<HeaderMenu />
<section className="container">
  <section className="metronome">
    <section>
      <div>
        <div className="bpm-display">
          <span className="tempo">{metronomeValue}</span>
          <span className="bpm">BPM</span>
        </div>
        <section className='pulse-controller-container'>
          <PulseController changed={this.handlePulseIntensityChange} pulses={this.state.currentPulses} />
        </section>
        <NumberController onClick={this.handleBeatChange} component={

          <div className="slider-container">

            <Slider
              aria-label="slider"
              defaultValue={60}
              min={20}
              max={200}
              track={false}
              value={metronomeValue}
              valueLabelDisplay="auto"
              onChange={this.changeValue}
            />
          </div>
        } />
      </div>
      <section className="action-button">
        <Button size="large" startIcon={currentPlayStatusComponent} onClick={this.handlePlayStatus}>
          {currentPlayStatusText}
        </Button>
      </section>

      <NumberController disableLeft={({this.state.beatsNumber == 2})} disableRight={({this.state.beatsNumber == 12})} onClick={this.handleMeasuresChange} component={

        <div className="beats-number-container">{beatsNumber}</div>
      } />
      <div className="beats-per-measure-text">
        <span>Batidas por Compaõo</span>
      </div>
    </div>
    <div className="sub-text">
      <p><b>A-Zul</b></p> = Nota com acentuaçao</div>
    </div>
  </section>
</section>
```

```
        <div className='sub-text'>
            <p><span><b>Azul</b></span> = Nota com acentuação</p>
            <p><b>Cinza</b></p> = Nota sem acentuação</p>
            <p>Selecione a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's</p>
        </div>
    </section>

    <section className='application-mode'>
        <Playlist />
    </section>
</section>

<Footer />
</section>
</>
}
}
```

- Evidência dos resultados:

CUSTOM METRONOME

# Evolua sua prática com facilidade

O metronomo mais versátil.  
Acessível diretamente do seu navegador

[COMEÇAR AQUI](#)



CUSTOM PETRÓLEO

LOGIN

**Use onde quiser**  
Tenha liberdade e portabilidade para seus estudos

 Celular

 Computador

 Tablet

© Copyright 2022. Todos os direitos reservados.

CUSTOM PETRÓLEO

LOGIN

**60 BPM**

PLAY

- 4 +

BATIDAS POR COMPASSO

Azul = Nota com acentuação  
Cinza = Nota sem acentuação

Seleciona a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

Listas de Reprodução

- Abecedário Rítmico
- Lista Personalizada
- Adicionar Nova Lista

© Copyright 2022. Todos os direitos reservados.

# 60 BPM

PLAY

- 4 +

BATIDAS POR COMPASSO

**Azul** = Nota com acentuação  
**Cinza** = Nota sem acentuação

Seleciona a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

# Evolua sua prática com **facilidade**

O metronomo mais versátil.  
Acessível diretamente do seu  
navegador

[COMECE AQUI](#)



## Use onde quiser

Tenha liberdade e  
portabilidade para seus  
estudos



Celular



Computador



Celular



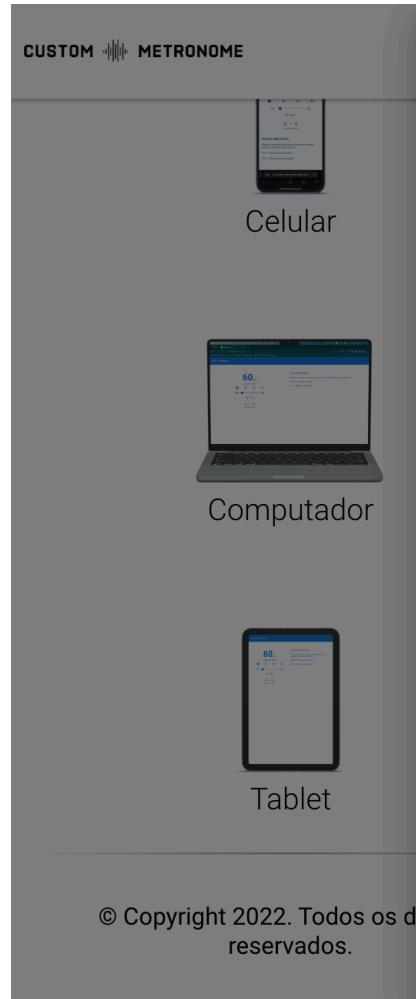
Computador



Tablet

---

© Copyright 2022. Todos os direitos reservados.



**60** BPM



PLAY

− 4 +

BATIDAS POR COMPASSO

**Azul** = Nota com acentuação

**Cinza** = Nota sem acentuação

Selecione a batida que gostaria de acentuar  
clickando no círculo abaixo dos bpm's

Listas de Reprodução

▶ Abecedário Rítmico

PLAY

− 4 +

BATIDAS POR COMPASSO

**Azul** = Nota com acentuação

**Cinza** = Nota sem acentuação

Selecione a batida que gostaria de acentuar  
clickando no círculo abaixo dos bpm's

Listas de Reprodução

▶ Abecedário Rítmico

▶ Lista Personalizada

⊕ Adicionar Nova Lista

## 2.2.2 Experiências vivenciadas

Durante a sprint 2 foram aprendidas as seguintes lições:

- É possível criar um loop de interações apenas com pseudo elementos de css sem necessidade de interação com Javascript
- Foi utilizado react router para fazer gestão de rotas da aplicação
- O strict mode do react faz com que o componente seja instanciado duas vezes e a administração de estado da aplicação não é resetada quando isso acontece
- Componentização de trechos de código para componentes parciais (partials) facilita na reutilização de código.
- Conceito de responsabilidade única de métodos se torna muito útil no react.

## 2.3 Sprint 3

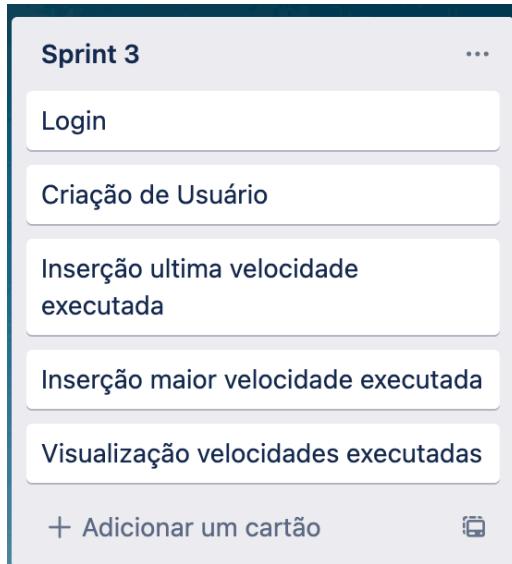
O foco da sprint 3 foi realizar contextualização de dados de usuário. Sendo assim, foram criados fluxos de login, registro de usuário e informações de execuções afim de ilustrar melhor o processo de evolução.

As implementações de Apis foram todas mockadas pelo json-server e foram utilizadas as seguintes ferramentas no desenvolvimento da sprint 3.

- Json server
- Json auth server
- Material React UI
- Chart Js
- React Chart JS 2
- UUID
- Axios

### 2.3.1 Solução

- Evidência do planejamento:



- Evidência da execução de cada requisito:

### Tarefa Login

Teve como foco proporcionar que um usuário já cadastrado acesse a aplicação com privilégios de salvar definitivamente informações de execução.

```
Complexity is 28 Bloody hell...
function App() { 
  const [user, setUser] = useState<IUser | null>(checkLoggedUserFromSessionStorage());
}

Complexity is 5 Everything is cool!
useEffect(() => {
  try {
    const {authenticatedUser} = (authenticateUser() as any);
    if (authenticatedUser) {
      setUser(authenticatedUser)
      sessionStorage.setItem("user", JSON.stringify(authenticatedUser));
    }
  } catch {
    onSignOut();
  }
}, []);
```

```
Complexity is 3 Everything is cool!
function checkLoggedUserFromSessionStorage() { 
  const user = sessionStorage.getItem("user");
  return user ? JSON.parse(user) : null;
}

async function authenticateUser() {
  return await new BackendService().create("/login", user);
}

function onSignOut() {
  setUser(null);
}
```

```
return (
  <authContext.Provider value={{ user, onSignOut }}>
    <Router>
      <Routes>
        <Route path='/' element={<Landing />} />
        <Route path='/metronome' element={<Metronome user={user} />} />
        <Route path='/login' element={<Authenticator cardState='login' onSignIn={setUser} />} />
        <Route path='/new-user' element={<Authenticator cardState='new-user' />} />
        <Route
          path="/new-playlist"
          element={
            <RequireAuth>
              <PlaylistForm />
            </RequireAuth>
          }
        />
      </Routes>
    </Router>
    <ToastContainer />
  </authContext.Provider>
);
```

```

export const authContext = React.createContext<IAuthContext>({
  user: {
    name: "Anônimo",
    email: "",
  },
  onSignOut: () => {},
});

export function useAuthContext() {
  return useContext(authContext);
}

```

```

return (
  <FormControl fullWidth sx={{ display: "grid", gridTemplateColumns: "1fr", gridColumnGap: 16 }} variant="standard">
    <TextField id="email" label="E-mail" variant="standard" value={email} onChange={(event) => {
      setEmail(event.target.value)
    }} />
    <TextField id="password" label="Senha" variant="standard" type="password" value={password} onChange={(event) => {
      setPassword(event.target.value)
    }} />
    <footer className="authenticator-actions">
      <Button size="small" onClick={() => submit()}>Enviar</Button>
    </footer>
  </FormControl>
)

```

## Tarefa Criação de Usuário

Teve como foco proporcionar a criação de um novo usuário

```

return (
  <FormControl fullWidth sx={{ display: "grid", gridTemplateColumns: "1fr", gridColumnGap: 16 }} variant="standard">
    <TextField id="name" label="Nome" variant="standard" value={name} onChange={(event) => {
      handleNameChange(event);
    }} />
    <TextField id="email" label="E-mail" variant="standard" value={email} onChange={(event) => {
      handleEmailChange(event);
    }} />
    <TextField id="password" label="Senha" variant="standard" type="password" value={password} onChange={(event) => {
      handlePasswordChange(event);
    }} />
    <footer className="authenticator-actions">
      <Button size="small" onClick={() => submit()}>Enviar</Button>
    </footer>
  </FormControl>
)

```

```

async function postNewUser() {
    try {
        const data = await backendService.create("/users", { name, email, password });
        createNewUserData(data)
    } catch (error) {
        notifyError("Erro ao criar novo usuário !");
    }
}

Complexity is 4 Everything is cool!
async function createNewUserData(data: any) {
    try {
        await backendService.create("/data", {
            id: uuidv4(),
            userId: data.user.id,
            "lastVelocityUsed": 60,
            "velocities": [
                60
            ]
        })

        notifySuccess("Usuário criado com sucesso !");
        navigate("/login");
    } catch {
        notifyError("Erro ao criar dados do usuário !");
    }
}

```

### Tarefa Inserção última velocidade executada

Teve foco em inserir uma informação da última velocidade do metrônomo que foi executada pelo usuário

```

Complexity is 3 Everything is cool!
async putUserData() {

    const userData = {
        "userId": this.getUserSession().id,
        "lastVelocityUsed": this.state.lastVelocityUsed,
        "velocities": this.state.velocities
    }

    try {
        const data = await this.backendService.update(`/data/${this.state.dataId}`, userData);
        new ToastrService().notifySuccess("Dados salvos com sucesso");
    } catch {
        new ToastrService().notifySuccess("Erro ao salvar dados");
    }

    this.mapUserData();
}

mapNewLastVelocity() {
    this.setState({ lastVelocityUsed: this.state.metronomeValue })
}

```

## Tarefa Inserção Maior velocidade executada

Teve como foco salvar qual foi a maior velocidade executada pelo usuário

```
Complexity is 3 Everything is cool!
async |putUserData() { █

    const userData = {
        "userId": this.getUserSession().id,
        "lastVelocityUsed": this.state.lastVelocityUsed,
        "velocities": this.state.velocities
    }

    try {
        const data = await this.backendService.update(`/data/${this.state.dataId}`, userData);
        new ToastrService().notifySuccess("Dados salvos com sucesso");
    } catch {
        new ToastrService().notifySuccess("Erro ao salvar dados");
    }

    this.mapUserData();
}
```

```
addNewVelocity() {
    const newVelocities = this.state.velocities;
    newVelocities.push(this.state.metronomeValue);
    this.setState({ velocities: newVelocities })
}
```

```
<Typography color="text.secondary" component="div" variant="h5">
    {Math.max(...this.state.velocities)} bpm
</Typography>
```

## Tarefa visualização de velocidades executadas

Teve como foco demonstrar ao usuário o caminho percorrido até a ultima velocidade executada

```
<Card variant="outlined" id="graph">
    <CardContent sx={{ flex: '1 0 auto', paddingBottom: 0 }}>
        <Line data={this.state.chartData} />
    </CardContent>
    <div className='card-footer'>
        {this.getUserSession() ? "" : <small>Faça Login para salvar seu progresso definitivamente!</small>}
    </div>
</Card>
```

```

chartData: {
    labels: [""],
    datasets: [
        {
            fill: false,
            label: 'Velocidades Utilizadas',
            backgroundColor: '#1976d2',
            borderColor: '#1976d2',
            data: [60]
        }
    ]
}

```

Complexity is 5 Everything is cool!

```

async getUserData() {
    try {
        const data = await this.backendService.read(`/data?userId=${this.getUserSession().id}`);
        const newChartData = this.state.chartData;
        newChartData.labels = data[0].velocities.map(() => { return "" });
        newChartData.datasets[0].data = data[0].velocities;
        const datasetsCopy = this.state.chartData.datasets.slice(0);
        console.log(newChartData)

        this.setState({
            dataId: data[0].id,
            velocities: data[0].velocities,
            lastVelocityUsed: data[0].lastVelocityUsed,
            chartData: Object.assign(newChartData, this.state.chartData, {
                datasets: datasetsCopy
            })
        })
    } catch {
        new ToastrService().notifyError("Erro ao buscar dados");
    }
}

```

Complexity is 3 Everything is cool!

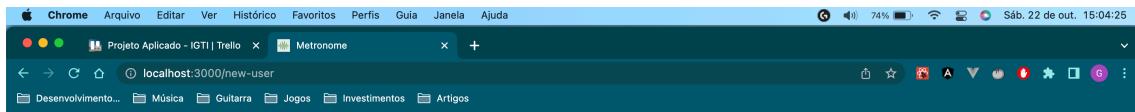
```

generateNewGraph() {
    const newChartData = this.state.chartData;
    const { velocities } = this.state;
    newChartData.labels = velocities.map(() => { return "" });
    newChartData.datasets[0].data = velocities;
    const datasetsCopy = this.state.chartData.datasets.slice(0);

    this.setState({
        chartData: Object.assign(newChartData, this.state.chartData, {
            datasets: datasetsCopy
        })
    })
}

```

- Evidência dos resultados:



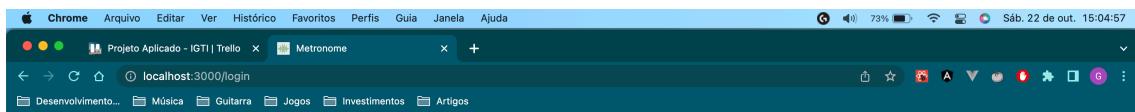
CUSTOM METRONOME

Nome

E-mail

Senha

[ENVIAR](#)



CUSTOM METRONOME

E-mail

Senha

[ENVIAR](#)

Chrome Arquivo Editar Ver Histórico Favoritos Perfil Guia Janela Ajuda

localhost:3000/metronome

CUSTOM METRONOME TESTE

**60 BPM**

PLAY

- 4 + BATIDAS POR COMPASSO

Azul = Nota com acentuação  
Cinza = Nota sem acentuação  
Selecione a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

Última execução: 62 bpm's | Maior velocidade atingida: 100 bpm's

Velocidades Utilizadas

Velocidade (bpm)	Quantidade (Velocidades Utilizadas)
60	60
62	100
64	65
66	55
68	60
70	60
72	60
74	60
76	60
78	60
80	60
82	60
84	60
86	60
88	60
90	60
92	60
94	60
96	60
98	60
100	60

Chrome Arquivo Editar Ver Histórico Favoritos Perfil Guia Janela Ajuda

localhost:3000/metronome

CUSTOM METRONOME TESTE Sair

**60 BPM**

PLAY

- 4 + BATIDAS POR COMPASSO

Azul = Nota com acentuação  
Cinza = Nota sem acentuação  
Selecione a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

Última execução: 62 bpm's | Maior velocidade atingida: 100 bpm's

Velocidades Utilizadas

Velocidade (bpm)	Quantidade (Velocidades Utilizadas)
60	60
62	100
64	65
66	55
68	60
70	60
72	60
74	60
76	60
78	60
80	60
82	60
84	60
86	60
88	60
90	60
92	60
94	60
96	60
98	60
100	60

© Copyright 2022. Todos os direitos reservados.

CUSTOM METRONOME



**60** BPM



PLAY

- 4 +

BATIDAS POR COMPASSO

Azul = Nota com acentuação

Cinza = Nota sem acentuação

Selecione a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

### Última execução

62 bpm's

### Maior velocidade atingida

100 bpm's

CUSTOM METRONOME



Azul = Nota com acentuação

Cinza = Nota sem acentuação

Selecione a batida que gostaria de acentuar clickando no círculo abaixo dos bpm's

### Última execução

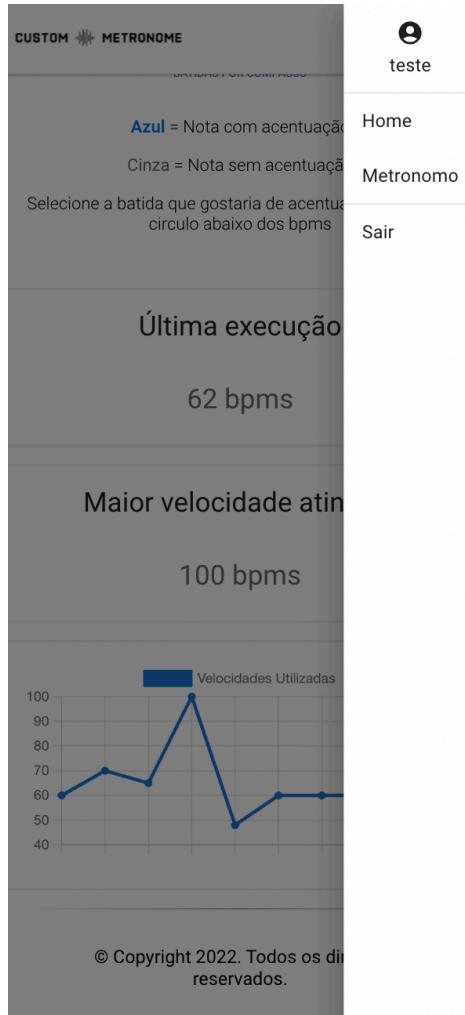
62 bpm's

### Maior velocidade atingida

100 bpm's



© Copyright 2022. Todos os direitos reservados.



### 2.3.2 Experiências vivenciadas

- React Class Components são incompatíveis com react hooks
- Json Server já exporta uma api completa baseada nos objetos do db
- Axios já abstrai boa parte da tratativa de promises
- React Charts 2 abstrai o Chart JS em forma de componentes, possibilitando maior facilidade para implementá-los
- Mesmo com auth context ainda é necessário fazer gestão do estado do usuário

## 3. Considerações Finais

### 3.1 Resultados

Principais resultados alcançados pelo projeto:

- Implementação de Mvp utilizando tecnologia de front end moderna (React)
- Dor sanada pelo Mvp
- Muitas possibilidades de evolução do app
- Fixação do conteúdo de react

Pontos positivos:

- Aprendizado sobre react
- Aprendizado de suporte a áudio api dos browsers
- Aprendizado de trabalho com json server
- Projeto adequado para consumo em diversos tamanhos de dispositivos
- Implementação da principal feature proposta
- Sensação de dever cumprido

Pontos Negativos:

- Devido ao tempo disponível para implementação não foi possível dar mais opções de personalização para um usuário logado
- A aplicação funciona apenas com base em mock do json server
- Devido ao tempo disponível para implementação não foi possível implementar testes unitários

Dificuldades enfrentadas:

- Suporte a api de áudio dos navegadores
- Utilizar conceito de service no react
- Administração do estado de um usuário logado

Lições aprendidas:

- Existem funcionalidades que podem complementar o mvp tais como uma lista de reprodução de batidas para o metrônomo.
- Api de áudio nativa dos browsers é instável e varia o comportamento de acordo com o dispositivo em que está sendo executada. É necessário tratar para que não haja surpresas na hora de iniciar um áudio. No caso desse projeto foi utilizada a biblioteca Howler.js para abstrair e tratar essas condições.
- A implementação de services no react se comporta totalmente diferente do que em outros frameworks de front-end tais como o angular. Como o react é baseado em estado, a instancia será recriada a cada vez que o componente necessitar ser remontado, perdendo assim o controle sobre a execução atual. Sendo assim, é necessário que a implementação exclua a atualização do serviço, a instancia do mesmo tem que ser imutável ao estado do componente. No caso desse projeto foi utilizado o react class componente para melhor administrar esse quesito
- Estado de usuário logado precisa ser administrado junto do session storage
- A estruturação do react força o implementador a pensar de forma que o código necessariamente seja abstraído e genérico para reaproveitamento.
- O estudo deve continuar sendo constante pois as tecnologias estão sempre em constante atualização

### 3.2 Contribuições

Destaco as seguintes contribuições:

- O MVP atingiu o objetivo de permitir com que o usuário consiga personalizar o funcionamento de um metrônomo convencional
- O MVP permite que novas funcionalidades sejam implementadas mais rapidamente
- Alunos, professores e praticantes de instrumentos já podem utilizar a funcionalidade para exercitar a independência rítmica.

### 3.3 Próximos passos

Ainda há muito a ser feito, como por exemplo:

- Implementação de um back-end sem mocks
- Integração com banco de dados
- Implementação de esqueci minha senha
- Funcionalidade de personalização de usuário
- Funcionalidade playlist de reprodução