

Como desenvolvedor, para criar a \*\*Calculadora de Impostos para Freelancers\*\* com foco em simplicidade e alta demanda, siga este roteiro técnico detalhado:

---

### \*\*1. Arquitetura Técnica (MVP em 7 Dias)\*\*

#### \*\*Stack Recomendada\*\*

| Camada | Tecnologia |

Por quê?



| \*\*Frontend\*\* | React + Vite |

Desenvolvimento rápido, hot-reload e componentes reutilizáveis.

| \*\*Backend\*\* | Node.js + Express + SQLite |

| Leve, sem necessidade de servidor dedicado. SQLite armazena alíquotas. |

| **\*\*Deploy\*\*** | Vercel (frontend) + Render (backend) | Gratuito para pequeno tráfego, CI/CD integrado.

| **\*\*Autenticação\*\*** | Firebase Authentication | Pronto para uso, com planos gratuitos generosos.

---

### ### **2. Estrutura do Projeto**

---

calculadora-impostos/

```
  └── frontend/          # React
      └── src/
          └── components/   # Formulário, Resultados, Explicacoes
              └── hooks/     #
                  └── useCalcularImpostos
                      └── utils/    # formatação de moeda
  └── backend/           # Node.js
      └── routes/
          └── calcular.js  # Lógica de
```

```
cálculo
|   |   └── cidades.js    # Alíquotas por
cidade
|   └── models/
|       └── Imposto.js    # Schema de
dados
└── database/
    └── alíquotas.db    # SQLite com ISS
por cidade/serviço
```
---
```

### ### \*\*3. Fluxo de Desenvolvimento Passo a Passo\*\*

#### #### \*\*Passo 1: Backend (2-3 Dias)\*\*

##### \*\*a. Configurar Servidor Node.js\*\*

```javascript

```
// backend/server.js
```

```
const express = require('express');
```

```
const sqlite3 = require('sqlite3').verbose();
```

```
const app = express();
```

```
app.use(express.json());  
  
// Conectar ao banco de dados  
const db = new sqlite3.Database('./  
database/aliquotas.db');  
  
  
// Rota principal de cálculo  
app.post('/calcular', (req, res) => {  
    const { receita, cidade, servico, despesas  
} = req.body;  
    const iss = calcularISS(receita, cidade,  
servico);  
    const baseIRPF = receita - despesas - iss;  
    const irpf = calcularIRPF(baseIRPF);  
  
  
    res.json({  
        iss: iss.toFixed(2),  
        irpf: irpf.toFixed(2),  
        pisCofins: (receita * 0.0365).toFixed(2),  
        // PIS/Cofins 3,65%  
        total: (iss + irpf + (receita *  
0.0365)).toFixed(2)  
    });  
});
```

```
// Iniciar servidor
app.listen(3000, () =>
  console.log('Backend rodando'));
```
});
```

## **\*\*b. Lógica de Cálculo dos Impostos\*\***

# ```javascript

## // backend/routes/calcular.js

```
function calcularISS(receita, cidade,  
servico) {
```

// Buscar alíquota no banco de dados (ex:  
São Paulo = 5%)

```
const aliquota = db.get(`SELECT aliquota
FROM cidades WHERE nome = ? AND
servico = ?`, [cidade, servico]);
```

```
return receita * (aliquota || 0.05); //
```

Padrão 5% se não encontrar

}

```
function calcularIRPF(base) {
```

## // Tabela progressiva 2024 (simplificada)

```
if (base <= 2112) return 0;
if (base <= 2826.65) return base * 0.075 -
158.40;
if (base <= 3751.05) return base * 0.15 -
370.40;
return base * 0.225 - 651.73; // Até R$ 4.664,68
}
...

```

## \*\*c. Popular Banco de Dados (SQLite)\*\*

```
```sql
-- backend/database/aliquotas.db
CREATE TABLE cidades (
    nome TEXT,
    servico TEXT,
    aliquota REAL
);
```

```
INSERT INTO cidades VALUES
('São Paulo', 'Consultoria', 0.05),
('Rio de Janeiro', 'Design', 0.05),
('Belo Horizonte', 'Programação', 0.04);
```

---

---

#### \*\*Passo 2: Frontend (3-4 Dias)\*\*

\*\*a. Formulário Principal\*\*

```jsx

```
// frontend/src/components/Formulario.js
import { useState } from 'react';
```

```
export default function Formulario() {
  const [dados, setDados] = useState({
    receita: '',
    cidade: 'São Paulo',
    servico: 'Consultoria',
    despesas: ''
  });
}
```

```
const handleChange = (e) => {
  setDados({ ...dados, [e.target.name]: e.target.value });
};
```

```
return (
  <form>
    <input
      name="receita"
      type="number"
      placeholder="Receita Bruta (R$)"
      value={dados.receita}
      onChange={handleChange}
    />

    <select name="cidade"
    value={dados.cidade}
    onChange={handleChange}>
      <option value="São Paulo">São
      Paulo</option>
      <option value="Rio de Janeiro">Rio de
      Janeiro</option>
    </select>

    {/* ... outros campos ... */}

  </form>
);
```

```
}
```

```
...
```

## \*\*b. Hook de Cálculo\*\*

```
```jsx
```

```
// frontend/src/hooks/  
useCalcularImpostos.js  
import { useState } from 'react';
```

```
export default function  
useCalcularImpostos() {  
  const [resultados, setResultados] =  
  useState(null);  
  
  const calcular = async (dados) => {  
    const response = await fetch('http://  
localhost:3000/calcular', {  
      method: 'POST',  
      headers: { 'Content-Type': 'application/  
json' },  
      body: JSON.stringify(dados)  
    });  
    const data = await response.json();
```

```
    setResultados(data);
};

return { resultados, calcular };
}
...

```

## \*\*c. Exibição de Resultados\*\*

```
```jsx
// frontend/src/components/Resultados.js
export default function Resultados({
resultados }) {
if (!resultados) return null;

return (
<div className="resultados">
  <h3>Resumo dos Impostos</h3>
  <p>ISS: R$ {resultados.iss}</p>
  <p>IRPF: R$ {resultados.irpf}</p>
  <p>PIS/Cofins: R$ {resultados.pisCofins}</p>
  <p className="total">Total: R$ {resultados.total}</p>

```

```
<div className="explicacao">
  <p>Como calculamos:</p>
  <ul>
    <li>ISS: 5% sobre a receita (alíquota de São Paulo)</li>
    <li>IRPF: Tabela progressiva sobre (Receita - Despesas - ISS)</li>
  </ul>
</div>
</div>
);
}
```
---
```

#### ### \*\*4. Funcionalidades Essenciais para MVP\*\*

| Prioridade   Funcionalidade | Implementação |
|-----------------------------|---------------|
| Prioridade                  |               |

|                      |   |
|----------------------|---|
| Cálculo de ISS       | Buscar alíquota no<br>SQLite por cidade/serviço           |
| Alta                 |   |
| Cálculo de IRPF      | Aplicar tabela<br>progressiva                             |
| Alta                 |   |
| PIS/Cofins           | 3,65% sobre receita<br>(padrão nacional)                  |
| Média                |   |
| Interface Responsiva | Mobile-first<br>com CSS Grid/Flexbox                      |
| Alta                 |   |
| Validação de Dados   | Impedir valores<br>negativos ou texto em campos numéricos |
| Alta                 |   |

---

### \*\*5. Diferenciais Técnicos para se  
Destacar\*\*

#### \*\*a. Explicações Contextuais\*\*

```jsx

```
// Exemplo de tooltip explicativo
<div className="tooltip">
  <span>?</span>
  <div className="tooltip-content">
    ISS: Imposto sobre Serviços. Varia por
    cidade (ex: 5% em SP).
  </div>
</div>
...
``
```

#### #### \*\*b. Atualização Automática de Alíquotas\*\*

```
```javascript
// backend/routes/atualizar.js
const axios = require('axios');

// Buscar novas alíquotas em APIs governamentais
const atualizarAliquotas = async () => {
  const response = await axios.get('https://
api.gov.br/aliquotas');
  // Atualizar banco de dados
  db.exec('UPDATE cidades SET aliquota =
```

```
? WHERE nome = ?',[novaAliquota,  
cidade]);  
};  
---
```

## #### \*\*c. Exportação de Relatório (PDF)\*\*

```
```javascript  
// backend/routes/relatorio.js  
const PDFDocument = require('pdfkit');  
  
app.post('/relatorio', (req, res) => {  
  const doc = new PDFDocument();  
  doc.text(`Relatório de Impostos - ${new  
Date().toLocaleDateString()}`);  
  doc.text(`Total a pagar: R$ ${  
req.body.total}`);  
  doc.pipe(res);  
  doc.end();  
});  
---
```

---

## ### \*\*6. Deploy e Monitoramento\*\*

### #### \*\*Passo a Passo\*\*

#### 1. \*\*Frontend (Vercel)\*\*:

```
```bash
```

```
cd frontend
```

```
npm run build
```

```
vercel --prod
```

```
```
```

#### 2. \*\*Backend (Render)\*\*:

- Conectar repositório GitHub ao Render.

- Configurar start command: `node server.js`.

#### 3. \*\*Banco de Dados\*\*:

- Usar SQLite (arquivo único) ou migrar para PostgreSQL se precisar de escala.

#### 4. \*\*Monitoramento\*\*:

- Adicionar Sentry para erros:

```
```javascript
```

```
import * as Sentry from "@sentry/node";
```

```
Sentry.init({ dsn: "SEU_DSN" });
```

...

---

## ### \*\*7. Estratégia Pós-Lançamento\*\*

Ação	Impacto
----- -----	
-----	
**Coletar Feedback** "Isso foi útil?" com campo de texto para sugestões.	Adicionar botão
**Análise de Uso**	Rastrear cálculos mais comuns (ex.: consultoria em SP = 70% dos usos).
**Expansão de Impostos**	Adicionar CSLL, IPI e ICMS conforme demanda dos usuários.
**Parcerias**	Integrar com plataformas de pagamento (ex.: Gerencianet).

---

### ### \*\*8. Custos e Timeline\*\*

Fase	Custo Estimado
Timeline	
----- ----- -----	
Desenvolvimento MVP   R\$ 0 (usando tecnologias gratuitas)   7 dias	
Hospedagem Inicial   R\$ 0 (planos gratuitos Vercel/Render)   Contínuo	
Domínio Próprio   R\$ 30/ano	
Opcional	
Ferramentas de Suporte   R\$ 0 (Sentry, Firebase)   Contínuo	

---

### ### \*\*9. Exemplo de Resultado Final\*\*

#### \*\*Interface\*\*:

...

[ Receita Bruta ] R\$ 5.000,00

[ Cidade ] São Paulo ▼

[ Serviço ] Consultoria ▼  
[ Despesas ] R\$ 1.000,00

[ Calcular Impostos ]

--- RESULTADOS ---

- ISS: R\$ 250,00 (5%)
  - IRPF: R\$ 193,27 (tabela progressiva)
  - PIS/Cofins: R\$ 182,50 (3,65%)
- Total: R\$ 625,77

? Por que esse valor?

Explicação detalhada dos cálculos...

---

### \*\*Conclusão para Desenvolvedores\*\*

- \*\*Viabilidade Técnica\*\*: MVP em 7 dias com stack simples.
- \*\*Diferencial\*\*: Explicações transparentes + atualização automática de alíquotas.

- **\*\*Escala\*\***: Comece com SQLite e migre para PostgreSQL se passar de 10k usuários.
- **\*\*Monetização\*\***: Premium a partir de R\$ 15/mês para relatórios e lembretes.

**\*\*Próximo Passo\*\***: Crie um repositório no GitHub, implemente o backend primeiro e depois o frontend. Teste com amigos freelancers antes do lançamento!

