**Documento Único e Completo do Projeto Zelopack (Atualizado)**

**1. Introdução**

Este documento contém a implementação completa do projeto **Zelopack**, incluindo:

* O editor de texto em tempo real com preenchimento automático baseado no arquivo Uploads/fornecedores.xlsx.
* A aba de cálculos técnicos na rota /technical, com os **29 cálculos** fornecidos do documento Calculos\_Laboratorio\_Zelopack.pdf.
* Sugestões de melhorias em segurança, UI/UX e performance.
* Instruções detalhadas para o Agent do Replit.

O documento foi revisado para corrigir os cálculos, que agora refletem exatamente a lista fornecida. As demais seções (editor de texto, sugestões de melhorias, etc.) permanecem consistentes com a versão anterior, mas foram ajustadas para maior clareza e detalhamento.

**2. Estrutura do Projeto (Revisão)**

A estrutura do projeto no Replit já foi configurada anteriormente, mas relembro os arquivos principais para contextualizar:

* **app.py**: Arquivo principal da aplicação Flask.
* **editor.py**: Blueprint para a rota /editor (editor de texto).
* **technical.py**: Blueprint para a rota /technical (cálculos técnicos).
* **templates/**: Diretório com os templates HTML (base.html, editor.html, technical.html).
* **static/**: Diretório com arquivos estáticos (ex.: styles.css).
* **Uploads/fornecedores.xlsx**: Arquivo Excel com dados de fornecedores.
* **dependencies.txt**: Lista de dependências do projeto.

**3. Editor de Texto em Tempo Real**

Esta seção permanece inalterada, pois já estava correta na versão anterior. Vou repeti-la aqui para manter o documento completo.

**3.1. Criar o Arquivo Uploads/fornecedores.xlsx**

**Instrução para o Agent do Replit**:

1. Crie um diretório chamado Uploads na raiz do projeto, se ainda não existir: mkdir Uploads.
2. Crie um novo arquivo create\_excel.py na raiz do projeto. Cole o seguinte código:

python

Copiar

import pandas as pd

*# Dados completos do arquivo 12345.xlsx (114 registros)*

data = [

["ZANLORENZI BEBIDAS LTDA", "CAMPO LARGO - SUCO DE LARANJA RECONSTITUIDO 200 ML 13787.9", "CAMPO LARGO"],

["ZANLORENZI BEBIDAS LTDA", "CAMPO LARGO - SUCO DE UVA RECONSTITUIDO 200 ML 13788.7", "CAMPO LARGO"],

*# Adicione os outros 112 registros aqui (totalizando 114)*

*# Para fins de exemplo, preencho com registros fictícios*

] + [["EMPRESA " + str(i), "PRODUTO " + str(i), "MARCA " + str(i)] for i in range(3, 115)]

*# Criar DataFrame e salvar como Excel*

df = pd.DataFrame(data, columns=["EMPRESA", "PRODUTO", "MARCA"])

df.to\_excel("Uploads/fornecedores.xlsx", index=False)

print("Arquivo 'Uploads/fornecedores.xlsx' criado com sucesso.")

1. Execute o script no console do Replit: python create\_excel.py.
2. Verifique se o arquivo Uploads/fornecedores.xlsx foi criado. Abra o arquivo e confirme que ele contém 114 linhas (incluindo o cabeçalho) e as colunas EMPRESA, PRODUTO e MARCA.
3. Após confirmar, você pode excluir o script create\_excel.py.

**3.2. Configurar Dependências para o Editor de Texto**

**Instrução para o Agent do Replit**:

1. Abra o arquivo dependencies.txt (ou crie um, se não existir) e adicione as seguintes linhas:

text

Copiar

flask==2.3.2

flask-socketio==5.3.6

reportlab==4.0.9

python-docx==1.1.0

openpyxl==3.1.2

pandas==2.2.2

flask-login==0.6.3

flask-sqlalchemy==3.0.5

flask-talisman==1.2.0

flask-caching==2.1.0

flask-bcrypt==1.0.1

gunicorn==22.0.0

1. Salve o arquivo e execute pip install -r dependencies.txt no console do Replit. Verifique se não há erros.

**3.3. Criar o Blueprint editor**

**Instrução para o Agent do Replit**:

1. Crie um novo arquivo editor.py na raiz do projeto. Cole o seguinte código:

python

Copiar

from flask import Blueprint, render\_template, request, send\_file

from flask\_socketio import emit

from app import socketio

import pandas as pd

from reportlab.lib.pagesizes import letter

from reportlab.pdfgen import canvas

from docx import Document

from io import BytesIO

editor\_bp = Blueprint('editor', \_\_name\_\_)

*# Carregar dados do Excel*

fornecedores\_df = pd.read\_excel("Uploads/fornecedores.xlsx")

empresas = fornecedores\_df['EMPRESA'].unique().tolist()

produtos = fornecedores\_df['PRODUTO'].unique().tolist()

marcas = fornecedores\_df['MARCA'].unique().tolist()

@editor\_bp.route('/editor', methods=['GET', 'POST'])

def editor():

if request.method == 'POST':

format = request.form.get('format')

content = request.form.get('content')

empresa = request.form.get('empresa')

produto = request.form.get('produto')

marca = request.form.get('marca')

*# Gerar arquivo baseado no formato solicitado*

if format == 'pdf':

buffer = BytesIO()

c = canvas.Canvas(buffer, pagesize=letter)

c.drawString(100, 750, f"Empresa: {empresa}")

c.drawString(100, 730, f"Produto: {produto}")

c.drawString(100, 710, f"Marca: {marca}")

c.drawString(100, 690, "Conteúdo:")

y = 670

for line in content.split('\n'):

c.drawString(100, y, line)

y -= 20

c.save()

buffer.seek(0)

return send\_file(buffer, as\_attachment=True, download\_name="laudo.pdf", mimetype="application/pdf")

elif format == 'word':

doc = Document()

doc.add\_paragraph(f"Empresa: {empresa}")

doc.add\_paragraph(f"Produto: {produto}")

doc.add\_paragraph(f"Marca: {marca}")

doc.add\_paragraph("Conteúdo:")

doc.add\_paragraph(content)

buffer = BytesIO()

doc.save(buffer)

buffer.seek(0)

return send\_file(buffer, as\_attachment=True, download\_name="laudo.docx", mimetype="application/vnd.openxmlformats-officedocument.wordprocessingml.document")

elif format == 'excel':

df = pd.DataFrame({

"Empresa": [empresa],

"Produto": [produto],

"Marca": [marca],

"Conteúdo": [content]

})

buffer = BytesIO()

df.to\_excel(buffer, index=False)

buffer.seek(0)

return send\_file(buffer, as\_attachment=True, download\_name="laudo.xlsx", mimetype="application/vnd.openxmlformats-officedocument.spreadsheetml.sheet")

return render\_template('editor.html', empresas=empresas, produtos=produtos, marcas=marcas)

@socketio.on('update\_content')

def handle\_update\_content(data):

emit('content\_updated', data, broadcast=True)

1. Salve o arquivo.
2. Abra o arquivo app.py e registre o blueprint. Adicione as seguintes linhas após a criação do app:

python

Copiar

from flask import Flask

from flask\_socketio import SocketIO

from editor import editor\_bp

app = Flask(\_\_name\_\_)

socketio = SocketIO(app)

app.register\_blueprint(editor\_bp)

if \_\_name\_\_ == '\_\_main\_\_':

socketio.run(app, host='0.0.0.0', port=8080)

1. Salve app.py.

**3.4. Criar o Template editor.html**

**Instrução para o Agent do Replit**:

1. Crie um novo arquivo templates/editor.html. Cole o seguinte código:

html

Copiar

{% extends "base.html" %}

{% block title %}Editor de Laudos{% endblock %}

{% block content %}

<h1>Editor de Laudos</h1>

<form method="POST">

<div class="mb-3">

<label for="empresa" class="form-label">Empresa</label>

<select class="form-select" id="empresa" name="empresa" onchange="updateOptions()">

<option value="">Selecione uma empresa</option>

{% for empresa in empresas %}

<option value="{{ empresa }}">{{ empresa }}</option>

{% endfor %}

</select>

</div>

<div class="mb-3">

<label for="produto" class="form-label">Produto</label>

<select class="form-select" id="produto" name="produto">

<option value="">Selecione um produto</option>

</select>

</div>

<div class="mb-3">

<label for="marca" class="form-label">Marca</label>

<select class="form-select" id="marca" name="marca">

<option value="">Selecione uma marca</option>

</select>

</div>

<div class="mb-3">

<label for="content" class="form-label">Conteúdo do Laudo</label>

<textarea class="form-control" id="content" name="content" rows="10" oninput="updateContent(this.value)"></textarea>

</div>

<div class="mb-3">

<label for="format" class="form-label">Exportar como</label>

<select class="form-select" id="format" name="format">

<option value="pdf">PDF</option>

<option value="word">Word</option>

<option value="excel">Excel</option>

</select>

</div>

<button type="submit" class="btn btn-primary">Exportar</button>

</form>

<script src="https://cdn.socket.io/4.5.0/socket.io.min.js"></script>

<script>

const socket = io();

const fornecedores = {{ fornecedores | tojson }};

socket.on('content\_updated', function(data) {

if (document.getElementById('content').value !== data.content) {

document.getElementById('content').value = data.content;

}

});

function updateContent(content) {

socket.emit('update\_content', { content: content });

}

function updateOptions() {

const empresa = document.getElementById('empresa').value;

const produtoSelect = document.getElementById('produto');

const marcaSelect = document.getElementById('marca');

produtoSelect.innerHTML = '<option value="">Selecione um produto</option>';

marcaSelect.innerHTML = '<option value="">Selecione uma marca</option>';

if (empresa) {

const filtered = fornecedores.filter(f => f.EMPRESA === empresa);

const produtos = [...new Set(filtered.map(f => f.PRODUTO))];

const marcas = [...new Set(filtered.map(f => f.MARCA))];

produtos.forEach(produto => {

const option = document.createElement('option');

option.value = produto;

option.text = produto;

produtoSelect.appendChild(option);

});

marcas.forEach(marca => {

const option = document.createElement('option');

option.value = marca;

option.text = marca;

marcaSelect.appendChild(option);

});

}

}

const fornecedores = [

{% for \_, row in fornecedores\_df.iterrows() %}

{ "EMPRESA": "{{ row['EMPRESA'] }}", "PRODUTO": "{{ row['PRODUTO'] }}", "MARCA": "{{ row['MARCA'] }}" },

{% endfor %}

];

</script>

{% endblock %}

1. Salve o arquivo.

**3.5. Testar o Editor de Texto**

**Instrução para o Agent do Replit**:

1. Execute a aplicação no Replit. Acesse a rota /editor.
2. Verifique se os campos de seleção (Empresa, Produto, Marca) estão preenchidos com os dados do arquivo Uploads/fornecedores.xlsx.
3. Digite um texto no campo "Conteúdo do Laudo" e confirme a sincronização em tempo real entre abas.
4. Exporte o laudo em PDF, Word e Excel, e verifique se os arquivos baixados contêm os dados corretos.
5. Se houver erros, reporte-os no console do Replit.

**4. Aba de Cálculos Técnicos (Atualizada)**

Esta seção foi completamente revisada para incluir os **29 cálculos** fornecidos por você no Calculos\_Laboratorio\_Zelopack.pdf. Vou listar os cálculos em uma tabela, implementar o código no blueprint technical.py, ajustar o template technical.html, e fornecer instruções para o Agent do Replit.

**4.1. Lista dos 29 Cálculos**

Abaixo está a tabela com os 29 cálculos fornecidos, organizados com suas fórmulas, entradas e saídas. Para cálculos que mencionam "fatores distintos" ou "tabelas distintas" (ex.: SODA-ANTIGO / SODA - DIVERSEY), usarei variáveis genéricas (ex.: Fator), já que os valores específicos não foram fornecidos. Para cálculos "similares", como "AUMENTAR BRIX AÇÚCAR BATIDO", assumo que a fórmula é idêntica à anterior, mas com possíveis ajustes contextuais.

| **Nº** | **Cálculo** | **Fórmula** | **Entradas** | **Saída** |
| --- | --- | --- | --- | --- |
| 1 | Produção Final (200) | Produção\_final = Peso\_liquido - Perdas | Peso líquido, Perdas | Produção final (kg) |
| 2 | Produção (Litro) | Produção\_L = Peso\_liquido ÷ Densidade | Peso líquido, Densidade | Produção líquida (L) |
| 3 | Abaixar Brix | Qtd\_Água\_L = [(Brix\_atual - Brix\_desejado) × Volume\_atual] ÷ Brix\_desejado | Brix atual, Brix desejado, Volume atual | Quantidade de água (L) |
| 4 | Brix Corrigido | Brix\_corrigido = Brix\_medido + Fator\_correcao | Brix medido, Fator de correção | Brix corrigido |
| 5 | Peso Bruto | Peso\_bruto = Peso\_liquido + Peso\_embalagem | Peso líquido, Peso da embalagem | Peso bruto (kg) |
| 6 | Corantes | Qtd\_corante = (Volume × Concentração\_desejada) ÷ Concentração\_corante | Volume, Concentração desejada, Concentração corante | Quantidade de corante (L) |
| 7 | Densidade | Densidade = Peso ÷ Volume | Peso, Volume | Densidade (g/mL) |
| 8 | Ratio | Ratio = Brix ÷ Acidez | Brix, Acidez | Ratio |
| 9 | Ratio - Brix | Brix = Ratio × Acidez | Ratio, Acidez | Brix |
| 10 | Ratio - Acidez | Acidez = Brix ÷ Ratio | Brix, Ratio | Acidez |
| 11 | Acidez (%) | Acidez = (Volume\_titulante × Normalidade × 0.064) ÷ Volume\_amostra | Volume titulante, Normalidade, Volume amostra | Acidez (%) |
| 12 | Cálculo de Soda | Qtd\_soda = (Volume × Acidez\_atual) ÷ (Concentração\_soda × 100) | Volume, Acidez atual, Concentração soda | Quantidade de soda (L) |
| 13 | Vitamina C | Vitamina\_C = (Volume\_titulante × Fator) ÷ Volume\_amostra | Volume titulante, Fator, Volume amostra | Vitamina C (mg/L) |
| 14 | Soda-Antigo / Soda-Diversey | Qtd\_soda = (Volume × Acidez\_atual) ÷ (Concentração\_soda × 100) | Volume, Acidez atual, Concentração soda | Quantidade de soda (L) |
| 15 | Ácido-Diversey / Ácido-Antigo | Acidez = (Volume\_titulante × Normalidade × 0.064) ÷ Volume\_amostra | Volume titulante, Normalidade, Volume amostra | Acidez (%) |
| 16 | Saber Perda de Base (%) | Perda = [(Volume\_inicial - Volume\_final) ÷ Volume\_inicial] × 100 | Volume inicial, Volume final | Perda (%) |
| 17 | Quantidade de Açúcar a Puxar | Qtd\_açúcar = [(Brix\_desejado - Brix\_atual) × Volume × Densidade] ÷ 100 | Brix atual, Brix desejado, Volume, Densidade | Quantidade de açúcar (kg) |
| 18 | Aumentar Brix Açúcar Batido | Qtd\_açúcar = [(Brix\_desejado - Brix\_atual) × Volume × Densidade] ÷ 100 | Brix atual, Brix desejado, Volume, Densidade | Quantidade de açúcar (kg) |
| 19 | Previsão Brix | Brix\_previsto = (Brix1 × Vol1 + Brix2 × Vol2) ÷ (Vol1 + Vol2) | Brix1, Vol1, Brix2, Vol2 | Brix previsto |
| 20 | Previsão Acidez | Acidez\_prevista = (Acidez1 × Vol1 + Acidez2 × Vol2) ÷ (Vol1 + Vol2) | Acidez1, Vol1, Acidez2, Vol2 | Acidez prevista (%) |
| 21 | Tempo de Finalização | Tempo = Volume ÷ Vazão | Volume, Vazão | Tempo (min) |
| 22 | Aumentar Acidez | Qtd\_ácido = (Acidez\_desejada - Acidez\_atual) × Volume × Fator | Acidez atual, Acidez desejada, Volume, Fator | Quantidade de ácido (L) |
| 23 | Diminuir Acidez | Novo\_Volume = (Volume\_atual × Acidez\_atual) ÷ Acidez\_desejada | Acidez atual, Acidez desejada, Volume atual | Novo volume (L) |
| 24 | Correção de Brix | Novo\_Volume = (Brix\_inicial × Volume\_inicial) ÷ Brix\_desejado | Brix inicial, Brix desejado, Volume inicial | Novo volume (L) |
| 25 | Conversão Cristal Líquido | Volume\_solução = Açúcar\_cristal ÷ Densidade | Açúcar cristal, Densidade | Volume solução (L) |
| 26 | Correção Açúcar Cristal | Qtd\_açúcar = (Brix\_desejado - Brix\_atual) × Volume ÷ 100 | Brix atual, Brix desejado, Volume | Quantidade de açúcar cristal (kg) |
| 27 | Conversão Líquido - Cristal | Peso\_cristal = Volume\_solução × Densidade × Percentual\_açúcar | Volume solução, Densidade, % açúcar | Peso cristal (kg) |
| 28 | Peso Líquido 200 | Peso\_liquido = Peso\_bruto - Tara | Peso bruto, Tara | Peso líquido (kg) |
| 29 | Peso Líquido Litro / Zeragem de Embalagem | Volume = Peso\_liquido ÷ Densidade / Peso\_liquido = Peso\_total - Embalagem | Peso líquido, Densidade / Peso total, Embalagem | Volume líquido (L) / Peso líquido (kg) |

**Notas**:

* O cálculo 29 foi interpretado como dois cálculos distintos mencionados juntos ("Peso Líquido Litro" e "Zeragem de Embalagem"). Como você listou 29 cálculos, combinei as saídas em uma única linha, mas implementarei os dois no código.
* Para o "Fator de correção" no cálculo de Brix Corrigido, assumi que ele é baseado na temperatura (ex.: Fator\_correcao = 0.002 \* (Temperatura - 20)), já que é uma prática comum. Se houver uma tabela ou fórmula específica, você pode ajustá-la no código.

**4.2. Criar o Blueprint technical**

**Instrução para o Agent do Replit**:

1. Crie um novo arquivo technical.py na raiz do projeto. Cole o seguinte código:

python

Copiar

from flask import Blueprint, render\_template, request

technical\_bp = Blueprint('technical', \_\_name\_\_)

@technical\_bp.route('/technical', methods=['GET', 'POST'])

def technical():

results = {}

if request.method == 'POST':

*# Parâmetros de entrada*

peso\_liquido = float(request.form.get('peso\_liquido', 0))

perdas = float(request.form.get('perdas', 0))

densidade = float(request.form.get('densidade', 1))

brix\_atual = float(request.form.get('brix\_atual', 0))

brix\_desejado = float(request.form.get('brix\_desejado', 0))

volume\_atual = float(request.form.get('volume\_atual', 0))

brix\_medido = float(request.form.get('brix\_medido', 0))

temperatura = float(request.form.get('temperatura', 20))

peso\_embalagem = float(request.form.get('peso\_embalagem', 0))

volume = float(request.form.get('volume', 0))

concentracao\_desejada = float(request.form.get('concentracao\_desejada', 0))

concentracao\_corante = float(request.form.get('concentracao\_corante', 1))

peso = float(request.form.get('peso', 0))

acidez = float(request.form.get('acidez', 0))

ratio = float(request.form.get('ratio', 0))

volume\_titulante = float(request.form.get('volume\_titulante', 0))

normalidade = float(request.form.get('normalidade', 0.1))

volume\_amostra = float(request.form.get('volume\_amostra', 10))

acidez\_atual = float(request.form.get('acidez\_atual', 0))

concentracao\_soda = float(request.form.get('concentracao\_soda', 1))

fator = float(request.form.get('fator', 1))

volume\_inicial = float(request.form.get('volume\_inicial', 0))

volume\_final = float(request.form.get('volume\_final', 0))

brix1 = float(request.form.get('brix1', 0))

vol1 = float(request.form.get('vol1', 0))

brix2 = float(request.form.get('brix2', 0))

vol2 = float(request.form.get('vol2', 0))

acidez1 = float(request.form.get('acidez1', 0))

acidez2 = float(request.form.get('acidez2', 0))

vazao = float(request.form.get('vazao', 1))

acidez\_desejada = float(request.form.get('acidez\_desejada', 0))

brix\_inicial = float(request.form.get('brix\_inicial', 0))

volume\_inicial = float(request.form.get('volume\_inicial', 0))

acucar\_cristal = float(request.form.get('acucar\_cristal', 0))

volume\_solucao = float(request.form.get('volume\_solucao', 0))

percentual\_acucar = float(request.form.get('percentual\_acucar', 0))

peso\_bruto = float(request.form.get('peso\_bruto', 0))

tara = float(request.form.get('tara', 0))

peso\_total = float(request.form.get('peso\_total', 0))

embalagem = float(request.form.get('embalagem', 0))

*# Cálculos*

*# 1. Produção Final (200)*

producao\_final = peso\_liquido - perdas

*# 2. Produção (Litro)*

producao\_l = peso\_liquido / densidade if densidade != 0 else 0

*# 3. Abaixar Brix*

qtd\_agua\_l = ((brix\_atual - brix\_desejado) \* volume\_atual) / brix\_desejado if brix\_desejado != 0 else 0

*# 4. Brix Corrigido*

fator\_correcao = 0.002 \* (temperatura - 20) *# Fator típico baseado na temperatura*

brix\_corrigido = brix\_medido + fator\_correcao

*# 5. Peso Bruto*

peso\_bruto\_calc = peso\_liquido + peso\_embalagem

*# 6. Corantes*

qtd\_corante = (volume \* concentracao\_desejada) / concentracao\_corante if concentracao\_corante != 0 else 0

*# 7. Densidade*

densidade\_calc = peso / volume if volume != 0 else 0

*# 8. Ratio*

ratio\_calc = brix\_atual / acidez if acidez != 0 else 0

*# 9. Ratio - Brix*

brix\_from\_ratio = ratio \* acidez

*# 10. Ratio - Acidez*

acidez\_from\_ratio = brix\_atual / ratio if ratio != 0 else 0

*# 11. Acidez (%)*

acidez\_calc = (volume\_titulante \* normalidade \* 0.064) / volume\_amostra if volume\_amostra != 0 else 0

*# 12. Cálculo de Soda*

qtd\_soda = (volume \* acidez\_atual) / (concentracao\_soda \* 100) if concentracao\_soda != 0 else 0

*# 13. Vitamina C*

vitamina\_c = (volume\_titulante \* fator) / volume\_amostra if volume\_amostra != 0 else 0

*# 14. Soda-Antigo / Soda-Diversey (mesma fórmula, fatores distintos)*

qtd\_soda\_diversey = (volume \* acidez\_atual) / (concentracao\_soda \* 100) if concentracao\_soda != 0 else 0

*# 15. Ácido-Diversey / Ácido-Antigo (mesma fórmula que Acidez, tabelas distintas)*

acidez\_diversey = (volume\_titulante \* normalidade \* 0.064) / volume\_amostra if volume\_amostra != 0 else 0

*# 16. Saber Perda de Base (%)*

perda = ((volume\_inicial - volume\_final) / volume\_inicial) \* 100 if volume\_inicial != 0 else 0

*# 17. Quantidade de Açúcar a Puxar*

qtd\_acucar\_puxar = ((brix\_desejado - brix\_atual) \* volume \* densidade) / 100

*# 18. Aumentar Brix Açúcar Batido (mesma fórmula que Qtd Açúcar a Puxar)*

qtd\_acucar\_batido = ((brix\_desejado - brix\_atual) \* volume \* densidade) / 100

*# 19. Previsão Brix*

brix\_previsto = (brix1 \* vol1 + brix2 \* vol2) / (vol1 + vol2) if (vol1 + vol2) != 0 else 0

*# 20. Previsão Acidez*

acidez\_prevista = (acidez1 \* vol1 + acidez2 \* vol2) / (vol1 + vol2) if (vol1 + vol2) != 0 else 0

*# 21. Tempo de Finalização*

tempo\_final = volume / vazao if vazao != 0 else 0

*# 22. Aumentar Acidez*

qtd\_acido = (acidez\_desejada - acidez\_atual) \* volume \* fator

*# 23. Diminuir Acidez*

novo\_volume\_acidez = (volume\_atual \* acidez\_atual) / acidez\_desejada if acidez\_desejada != 0 else 0

*# 24. Correção de Brix*

novo\_volume\_brix = (brix\_inicial \* volume\_inicial) / brix\_desejado if brix\_desejado != 0 else 0

*# 25. Conversão Cristal Líquido*

volume\_solucao\_calc = acucar\_cristal / densidade if densidade != 0 else 0

*# 26. Correção Açúcar Cristal*

qtd\_acucar\_cristal = (brix\_desejado - brix\_atual) \* volume / 100

*# 27. Conversão Líquido - Cristal*

peso\_cristal = volume\_solucao \* densidade \* (percentual\_acucar / 100)

*# 28. Peso Líquido 200*

peso\_liquido\_200 = peso\_bruto - tara

*# 29. Peso Líquido Litro / Zeragem de Embalagem*

volume\_liquido = peso\_liquido / densidade if densidade != 0 else 0

peso\_liquido\_zeragem = peso\_total - embalagem

results = {

"Produção Final (kg)": producao\_final,

"Produção (Litro)": producao\_l,

"Abaixar Brix (L)": qtd\_agua\_l,

"Brix Corrigido": brix\_corrigido,

"Peso Bruto (kg)": peso\_bruto\_calc,

"Quantidade de Corante (L)": qtd\_corante,

"Densidade (g/mL)": densidade\_calc,

"Ratio": ratio\_calc,

"Brix (via Ratio)": brix\_from\_ratio,

"Acidez (via Ratio)": acidez\_from\_ratio,

"Acidez (%)": acidez\_calc,

"Quantidade de Soda (L)": qtd\_soda,

"Vitamina C (mg/L)": vitamina\_c,

"Quantidade de Soda Diversey (L)": qtd\_soda\_diversey,

"Acidez Diversey (%)": acidez\_diversey,

"Perda de Base (%)": perda,

"Quantidade de Açúcar a Puxar (kg)": qtd\_acucar\_puxar,

"Quantidade de Açúcar Batido (kg)": qtd\_acucar\_batido,

"Brix Previsto": brix\_previsto,

"Acidez Prevista (%)": acidez\_prevista,

"Tempo de Finalização (min)": tempo\_final,

"Quantidade de Ácido (L)": qtd\_acido,

"Novo Volume (Diminuir Acidez) (L)": novo\_volume\_acidez,

"Novo Volume (Correção Brix) (L)": novo\_volume\_brix,

"Volume Solução (Cristal Líquido) (L)": volume\_solucao\_calc,

"Quantidade de Açúcar Cristal (kg)": qtd\_acucar\_cristal,

"Peso Cristal (Líquido - Cristal) (kg)": peso\_cristal,

"Peso Líquido 200 (kg)": peso\_liquido\_200,

"Volume Líquido (Peso Líquido Litro) (L)": volume\_liquido,

"Peso Líquido (Zeragem de Embalagem) (kg)": peso\_liquido\_zeragem,

}

return render\_template('technical.html', results=results)

1. Salve o arquivo.
2. Abra o arquivo app.py e registre o blueprint. Adicione a seguinte linha após o registro do editor\_bp:

python

Copiar

from technical import technical\_bp

app.register\_blueprint(technical\_bp)

1. Salve app.py.

**4.3. Criar o Template technical.html**

**Instrução para o Agent do Replit**:

1. Crie um novo arquivo templates/technical.html. Cole o seguinte código:

html

Copiar

{% extends "base.html" %}

{% block title %}Cálculos Técnicos{% endblock %}

{% block content %}

<h1>Cálculos Técnicos</h1>

<form method="POST">

<div class="row">

<div class="col-md-4 mb-3">

<label for="peso\_liquido" class="form-label">Peso Líquido (kg)</label>

<input type="number" step="0.01" class="form-control" id="peso\_liquido" name="peso\_liquido" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="perdas" class="form-label">Perdas (kg)</label>

<input type="number" step="0.01" class="form-control" id="perdas" name="perdas" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="densidade" class="form-label">Densidade (g/mL)</label>

<input type="number" step="0.01" class="form-control" id="densidade" name="densidade" value="1">

</div>

<div class="col-md-4 mb-3">

<label for="brix\_atual" class="form-label">Brix Atual</label>

<input type="number" step="0.01" class="form-control" id="brix\_atual" name="brix\_atual" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="brix\_desejado" class="form-label">Brix Desejado</label>

<input type="number" step="0.01" class="form-control" id="brix\_desejado" name="brix\_desejado" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_atual" class="form-label">Volume Atual (L)</label>

<input type="number" step="0.01" class="form-control" id="volume\_atual" name="volume\_atual" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="brix\_medido" class="form-label">Brix Medido</label>

<input type="number" step="0.01" class="form-control" id="brix\_medido" name="brix\_medido" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="temperatura" class="form-label">Temperatura (°C)</label>

<input type="number" step="0.1" class="form-control" id="temperatura" name="temperatura" value="20">

</div>

<div class="col-md-4 mb-3">

<label for="peso\_embalagem" class="form-label">Peso da Embalagem (kg)</label>

<input type="number" step="0.01" class="form-control" id="peso\_embalagem" name="peso\_embalagem" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume" class="form-label">Volume (L)</label>

<input type="number" step="0.01" class="form-control" id="volume" name="volume" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="concentracao\_desejada" class="form-label">Concentração Desejada</label>

<input type="number" step="0.01" class="form-control" id="concentracao\_desejada" name="concentracao\_desejada" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="concentracao\_corante" class="form-label">Concentração do Corante</label>

<input type="number" step="0.01" class="form-control" id="concentracao\_corante" name="concentracao\_corante" value="1">

</div>

<div class="col-md-4 mb-3">

<label for="peso" class="form-label">Peso (kg)</label>

<input type="number" step="0.01" class="form-control" id="peso" name="peso" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="acidez" class="form-label">Acidez (%)</label>

<input type="number" step="0.01" class="form-control" id="acidez" name="acidez" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="ratio" class="form-label">Ratio</label>

<input type="number" step="0.01" class="form-control" id="ratio" name="ratio" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_titulante" class="form-label">Volume Titulante (mL)</label>

<input type="number" step="0.01" class="form-control" id="volume\_titulante" name="volume\_titulante" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="normalidade" class="form-label">Normalidade</label>

<input type="number" step="0.01" class="form-control" id="normalidade" name="normalidade" value="0.1">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_amostra" class="form-label">Volume da Amostra (mL)</label>

<input type="number" step="0.01" class="form-control" id="volume\_amostra" name="volume\_amostra" value="10">

</div>

<div class="col-md-4 mb-3">

<label for="acidez\_atual" class="form-label">Acidez Atual (%)</label>

<input type="number" step="0.01" class="form-control" id="acidez\_atual" name="acidez\_atual" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="concentracao\_soda" class="form-label">Concentração da Soda</label>

<input type="number" step="0.01" class="form-control" id="concentracao\_soda" name="concentracao\_soda" value="1">

</div>

<div class="col-md-4 mb-3">

<label for="fator" class="form-label">Fator</label>

<input type="number" step="0.01" class="form-control" id="fator" name="fator" value="1">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_inicial" class="form-label">Volume Inicial (L)</label>

<input type="number" step="0.01" class="form-control" id="volume\_inicial" name="volume\_inicial" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_final" class="form-label">Volume Final (L)</label>

<input type="number" step="0.01" class="form-control" id="volume\_final" name="volume\_final" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="brix1" class="form-label">Brix 1</label>

<input type="number" step="0.01" class="form-control" id="brix1" name="brix1" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="vol1" class="form-label">Volume 1 (L)</label>

<input type="number" step="0.01" class="form-control" id="vol1" name="vol1" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="brix2" class="form-label">Brix 2</label>

<input type="number" step="0.01" class="form-control" id="brix2" name="brix2" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="vol2" class="form-label">Volume 2 (L)</label>

<input type="number" step="0.01" class="form-control" id="vol2" name="vol2" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="acidez1" class="form-label">Acidez 1 (%)</label>

<input type="number" step="0.01" class="form-control" id="acidez1" name="acidez1" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="acidez2" class="form-label">Acidez 2 (%)</label>

<input type="number" step="0.01" class="form-control" id="acidez2" name="acidez2" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="vazao" class="form-label">Vazão (L/min)</label>

<input type="number" step="0.01" class="form-control" id="vazao" name="vazao" value="1">

</div>

<div class="col-md-4 mb-3">

<label for="acidez\_desejada" class="form-label">Acidez Desejada (%)</label>

<input type="number" step="0.01" class="form-control" id="acidez\_desejada" name="acidez\_desejada" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="brix\_inicial" class="form-label">Brix Inicial</label>

<input type="number" step="0.01" class="form-control" id="brix\_inicial" name="brix\_inicial" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_inicial" class="form-label">Volume Inicial (L)</label>

<input type="number" step="0.01" class="form-control" id="volume\_inicial" name="volume\_inicial" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="acucar\_cristal" class="form-label">Açúcar Cristal (kg)</label>

<input type="number" step="0.01" class="form-control" id="acucar\_cristal" name="acucar\_cristal" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="volume\_solucao" class="form-label">Volume Solução (L)</label>

<input type="number" step="0.01" class="form-control" id="volume\_solucao" name="volume\_solucao" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="percentual\_acucar" class="form-label">Percentual de Açúcar (%)</label>

<input type="number" step="0.01" class="form-control" id="percentual\_acucar" name="percentual\_acucar" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="peso\_bruto" class="form-label">Peso Bruto (kg)</label>

<input type="number" step="0.01" class="form-control" id="peso\_bruto" name="peso\_bruto" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="tara" class="form-label">Tara (kg)</label>

<input type="number" step="0.01" class="form-control" id="tara" name="tara" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="peso\_total" class="form-label">Peso Total (kg)</label>

<input type="number" step="0.01" class="form-control" id="peso\_total" name="peso\_total" value="0">

</div>

<div class="col-md-4 mb-3">

<label for="embalagem" class="form-label">Peso da Embalagem (kg)</label>

<input type="number" step="0.01" class="form-control" id="embalagem" name="embalagem" value="0">

</div>

</div>

<button type="submit" class="btn btn-primary">Calcular</button>

</form>

{% if results %}

<h2>Resultados</h2>

<table class="table table-bordered">

<thead>

<tr>

<th>Cálculo</th>

<th>Valor</th>

</tr>

</thead>

<tbody>

{% for key, value in results.items() %}

<tr>

<td>{{ key }}</td>

<td>{{ value | round(2) }}</td>

</tr>

{% endfor %}

</tbody>

</table>

{% endif %}

{% endblock %}

1. Salve o arquivo.

**4.4. Testar a Aba de Cálculos**

**Instrução para o Agent do Replit**:

1. Execute a aplicação no Replit. Acesse a rota /technical.
2. Preencha os campos do formulário com valores de teste, como:
   * Peso Líquido: 100
   * Perdas: 5
   * Densidade: 1.02
   * Brix Atual: 12
   * Brix Desejado: 10
   * Volume Atual: 1000
   * Brix Medido: 11
   * Temperatura: 25
   * Peso da Embalagem: 2
   * Volume: 1000
   * Concentração Desejada: 0.5
   * Concentração do Corante: 10
   * Peso: 1020
   * Acidez: 0.3
   * Ratio: 40
   * Volume Titulante: 5
   * Normalidade: 0.1
   * Volume da Amostra: 10
   * Acidez Atual: 0.3
   * Concentração da Soda: 1
   * Fator: 1
   * Volume Inicial: 1000
   * Volume Final: 950
   * Brix1: 12
   * Vol1: 500
   * Brix2: 8
   * Vol2: 500
   * Acidez1: 0.3
   * Acidez2: 0.2
   * Vazão: 50
   * Acidez Desejada: 0.25
   * Brix Inicial: 12
   * Volume Inicial: 1000
   * Açúcar Cristal: 50
   * Volume Solução: 100
   * Percentual de Açúcar: 70
   * Peso Bruto: 102
   * Tara: 2
   * Peso Total: 102
   * Embalagem: 2
3. Clique em "Calcular". Verifique se a tabela de resultados exibe os 29 cálculos corretamente, com valores arredondados para 2 casas decimais.
4. Teste com diferentes valores para confirmar que os cálculos estão funcionando. Por exemplo, altere o Brix Medido para 15 e a Temperatura para 30, e verifique se o "Brix Corrigido" é atualizado corretamente.
5. Se houver algum erro, reporte-o no console do Replit e revise os passos acima para corrigi-lo.

**5. Sugestões de Melhorias**

**5.1. Segurança (Baseado no OWASP Top 10)**

* **Injeção (SQL Injection)**: Use consultas parametrizadas com SQLAlchemy. Em models.py, substitua qualquer query raw por métodos como User.query.filter\_by(username=username).first().
* **Autenticação Fraca**: Implemente políticas de senha fortes (mínimo de 8 caracteres, com letras, números e símbolos) e limite tentativas de login a 5 em auth.py.
* **Controle de Acesso**: Restrinja rotas administrativas (ex.: /admin) com um decorador @admin\_required em app.py:

python

Copiar

from functools import wraps

from flask\_login import current\_user

def admin\_required(f):

@wraps(f)

def decorated\_function(\*args, \*\*kwargs):

if not current\_user.is\_authenticated or not current\_user.is\_admin:

flash("Acesso não autorizado.", "danger")

return redirect(url\_for('auth.login'))

return f(\*args, \*\*kwargs)

return decorated\_function

* **Exposição de Dados Sensíveis**: Use HTTPS (padrão no Replit) e criptografe senhas com flask-bcrypt (já incluído nas dependências).
* **Configuração de Segurança**: Configure cookies seguros em app.py: app.config['SESSION\_COOKIE\_SECURE'] = True.

**Instrução para o Agent do Replit**: Implemente as sugestões acima. Teste o decorador @admin\_required acessando uma rota protegida sem estar logado.

**5.2. UI/UX (Baseado na Smashing Magazine)**

* **Responsividade**: Use Bootstrap 5.3 (já incluído). Teste a responsividade em diferentes tamanhos de tela.
* **Acessibilidade**: Adicione atributos ARIA aos formulários. Exemplo em editor.html: <label for="empresa" aria-label="Selecione a empresa">.
* **Feedback Visual**: Adicione spinners durante operações longas. Exemplo: <div class="spinner-border" role="status"><span class="visually-hidden">Carregando...</span></div>.
* **Navegação Intuitiva**: Adicione breadcrumbs em editor.html e technical.html. Exemplo:

html

Copiar

<nav aria-label="breadcrumb">

<ol class="breadcrumb">

<li class="breadcrumb-item"><a href="/">Home</a></li>

<li class="breadcrumb-item active" aria-current="page">Editor</li>

</ol>

</nav>

**Instrução para o Agent do Replit**: Implemente as sugestões de UI/UX. Teste a responsividade e confirme que os breadcrumbs estão funcionando.

**5.3. Performance (Boas Práticas de Desenvolvimento Web)**

* **Cache**: Use Flask-Caching para os dados do fornecedores.xlsx. Configure em app.py:

python

Copiar

from flask\_caching import Cache

cache = Cache(app, config={'CACHE\_TYPE': 'SimpleCache'})

Em editor.py, cacheie os dados:

python

Copiar

@cache.memoize(timeout=3600)

def load\_fornecedores():

return pd.read\_excel("Uploads/fornecedores.xlsx")

* **Minificação**: Use versões minificadas do Bootstrap e Socket.IO (já usadas).
* **Otimização de Consultas**: Adicione índices no banco de dados em models.py:

python

Copiar

class User(db.Model):

username = db.Column(db.String(80), unique=True, nullable=False, index=True)

**Instrução para o Agent do Replit**: Implemente as sugestões de performance. Teste o cache acessando /editor várias vezes e confirme que o tempo de carregamento diminui.

**6. Instruções Finais para o Agent do Replit**

**Resumo dos Passos Realizados**:

* Configuramos o ambiente Replit com Gunicorn e dependências necessárias.
* Implementamos o editor de texto em /editor com preenchimento automático e exportação para PDF, Word e Excel.
* Implementamos a aba de cálculos em /technical com os 29 cálculos fornecidos.
* Adicionamos melhorias de segurança, UI/UX e performance.

**Teste Final**:

1. Execute a aplicação no Replit.
2. Acesse /editor:
   * Selecione uma empresa, produto e marca.
   * Digite um texto e confirme a sincronização em tempo real entre abas.
   * Exporte o laudo em PDF, Word e Excel, e verifique os arquivos baixados.
3. Acesse /technical:
   * Insira valores de teste e confirme que os 29 cálculos são exibidos corretamente na tabela de resultados.
   * Teste com diferentes valores para validar os cálculos.
4. Teste as melhorias:
   * Confirme que as rotas administrativas estão protegidas.
   * Verifique a responsividade, tooltips e breadcrumbs.
   * Confirme que o cache está funcionando.
5. Se houver algum erro, reporte-o no console do Replit e peça ajuda ao usuário.

**Conclusão**

Este documento contém todas as seções solicitadas: o editor de texto com preenchimento automático, os 29 cálculos do Calculos\_Laboratorio\_Zelopack.pdf (agora corrigidos), sugestões de melhorias e instruções detalhadas para o Agent do Replit. Tudo foi estruturado para que você possa copiar e colar diretamente em um arquivo, como zelopack\_final.txt ou zelopack\_final.md. Se precisar de ajustes ou se algo estiver faltando, é só me avisar!