

Projeto 1: Pichu



Integrantes:
Camila Faleiros RA: 10395818
Felipe Clé Monteiro RA: 10395521
Matheus do Nascimento RA: 10395894
Pedro Henrique Rocha RA: 10396190

1. Ideação

a) Tema escolhido

- **Nome do Site:** Gerador de Dicas de Bem-Estar
- **Propósito:** Iremos fornecer dicas simples e rápidas para que as pessoas melhorem sua saúde física e mental de forma acessível.

b) Justificativa:

- **Relevância:** As pessoas estão cada vez mais em busca de conteúdos curtos e objetivos que tragam benefícios à saúde e ao bem-estar, mas nem sempre sabem onde encontrá-los.
- **Impacto:** O projeto pode incentivar práticas positivas na vida diária da comunidade, seja em casa, no trabalho ou na escola.

c) Objetivos

1. **Gerar dicas aleatórias** de bem-estar (exercícios, hidratação, mindfulness etc.).
2. **Estimular** o usuário a aplicar pelo menos uma dica por dia.
3. **Facilitar** o acesso a informações básicas de saúde e qualidade de vida.

2. Protótipo

Abaixo, temos um esboço do que imaginamos para o nosso site:



Optamos por fazer um site com somente uma página, para que seja algo fácil de consumir. Nele, o usuário poderá clicar no botão “gerar dicas”, de forma que uma dica aleatória será apresentada em sua tela. Ele também poderá curtir, comentar e compartilhar essa dica.

3. Tipo de Website

O website é um **site interativo**, focado em oferecer dicas de bem-estar para o usuário. Ele foi pensado para ser simples, direto e fácil de usar, funcionando totalmente no navegador, sem precisar de servidores ou bancos de dados externos. Nele teremos:

- **Interatividade no Navegador:**
Toda a ação acontece no seu computador ou celular. Quando você clica em um botão para gerar uma dica ou indicar que gostou, o site atualiza essas informações imediatamente, sem recarregar a página. Isso será feito com JavaScript, que permite que o site "responda" instantaneamente às suas interações.
- **Organização Clara e Semântica:**
O conteúdo será organizado de forma lógica com seções bem definidas (como cabeçalho, área principal e rodapé). Essa estrutura facilita a compreensão do site tanto para os usuários quanto para os mecanismos de busca, ajudando a melhorar a acessibilidade.
- **Design Moderno e Responsivo:**
Usando CSS3, o site adotará um visual contemporâneo e se adapta a diferentes tamanhos de tela. Isso significa que ele fica bem tanto em computadores quanto em dispositivos móveis.
- **Simplicidade e Autonomia:**
Sem depender de dados externos, todas as informações necessárias (como as dicas) são pré-definidas. Essa abordagem torna o site robusto e fácil de manter.

4. Explicação do código:

Para essa parte do projeto, elaboramos o código HTML e CSS, para concretizar o estilo idealizado. Além disso, criamos uma funcionalidade no arquivo JS, onde o programa gera uma dica para o usuário, ao clicar no botão “Gerar dica”.

A seguir, vamos entender um pouco do código HTML:

1. Declaração e Configuração Básica

```
1  <!DOCTYPE html>
2  <html lang="pt-BR">
3  <head>
4  |  <meta charset="UTF-8" />
5  |  <title>Gerador de Dicas de Bem-Estar</title>
6  |  <meta name="viewport" content="width=device-width, initial-scale=1.0">
7
```

- **<!DOCTYPE html>**: Indica que estamos usando HTML5, a versão que foi solicitada em aula.
- **<html lang="pt-BR">**: Abre o documento HTML e define o idioma como português do Brasil.
- **<head>**: Contém informações importantes sobre o site, como a codificação de caracteres e o título.
- **<meta charset="UTF-8" />**: Garante que todos os caracteres (como acentos) serão exibidos corretamente.
- **<title>**: Define o nome do site que aparece na aba do navegador.
- **<meta name="viewport" content="width=device-width, initial-scale=1.0">**: Configura a visualização para dispositivos móveis, garantindo que a largura da página se adapte à largura do dispositivo e proporcionando uma experiência responsiva.

2. Inclusão de Fontes e Arquivos Externos

2.1. Fontes do Google:

```
8  <!-- Fonte do Google Fonts -->
9  <link rel="preconnect" href="https://fonts.googleapis.com">
10 <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
11 <link
12   href="https://fonts.googleapis.com/css2?family=Quicksand:wght@400;600;700&display=swap"
13   rel="stylesheet"
14 >
15
```

- `<link rel="preconnect" ...>`: Essas tags estabelecem uma conexão antecipada com os servidores do Google Fonts, o que melhora o tempo de carregamento da fonte.
- `<link href="https://fonts.googleapis.com/css2?family=Quicksand:...>`: Importa a fonte **Quicksand** com os pesos 400, 600 e 700. Essa fonte será usada para estilizar os textos da página.

2.2. CSS Externo:

```
15
16  <!-- Link para o arquivo CSS externo -->
17  <link rel="stylesheet" href="style.css">
18 </head>
```

- `<link rel="stylesheet" href="style.css">`: Vincula um arquivo CSS externo chamado `style.css`, que contém as regras de estilo para a página. Isso ajuda a separar a estrutura (HTML) da apresentação (CSS).

3. Corpo do documento (body)

```
19 <body>
20  <header>
21    <h1>Gerador de Dicas de Bem-Estar</h1>
22  </header>
23
```

- `<header>`: Define o cabeçalho da página, que geralmente contém o título ou logotipo.
- `<h1>Gerador de Dicas de Bem-Estar</h1>`: Um título principal que informa imediatamente ao usuário o propósito da página.

4. Conteúdo principal

4.1. Área de geração de dicas:

```
24      <main>
25          <section id="gerador-de-dicas" class="gerador-de-dicas">
26              <!-- Botão para gerar dica -->
27              <section id="area-gerar-dica" class="area-gerar-dica">
28                  <button id="btn-gerar-dica">Gerar Dica</button>
29              </section>
30
```

- **<main>**: A tag `<main>` delimita o conteúdo principal do documento, excluindo cabeçalho, rodapé e barras laterais.
- **<section id="gerador-de-dicas" class="gerador-de-dicas">**: Esta seção engloba todo o mecanismo do gerador de dicas. O uso de `id` e `class` possibilita a estilização e manipulação via JavaScript.
- **<section id="area-gerar-dica" class="area-gerar-dica">**: Uma sub-seção destinada a abrigar o botão que, quando clicado, gera uma dica de bem-estar.
- **<button id="btn-gerar-dica">Gerar Dica</button>**: Botão que acionará (via JavaScript) a função de gerar uma nova dica.

4.2. Exibição da dica gerada

```
30
31          <!-- Exibição da dica -->
32          <section id="exibicao-dica" class="exibicao-dica">
33              <section id="icone-estrela" class="icone-estrela">★</section>
34              <p id="texto-dica">Clique em "Gerar Dica" para começar!</p>
35          </section>
```

- **<section id="exibicao-dica" class="exibicao-dica">**
Área onde a dica gerada será exibida para o usuário.
- **<section id="icone-estrela" class="icone-estrela">★</section>**
Exibe um ícone (neste caso, uma estrela) que pode servir para destacar a dica ou simplesmente para fins decorativos.
- **<p id="texto-dica">Clique em "Gerar Dica" para começar!</p>**
Parágrafo que contém uma mensagem inicial, instruindo o usuário a clicar no botão para iniciar a geração das dicas.

4.3. Ações: Curtir e Compartilhar

```
36
37      <!-- Botões curtir e compartilhar -->
38      <section id="acoes" class="acoes">
39          <section id="botao-like" class="botao-like">
40              <button id="btn-like">Gostei</button>
41          </section>
42          <section id="botao-compartilhar" class="botao-compartilhar">
43              <button id="btn-compartilhar">Compartilhar</button>
44          </section>
45      </section>
```

- **<section id="acoes" class="acoes">**: Uma seção dedicada a ações que o usuário pode realizar após ver a dica.

- **Botão "Gostei"**

Dentro de **<section id="botao-like" class="botao-like">** temos:

- **<button id="btn-like">Gostei</button>**: Botão que, provavelmente, registra que o usuário curtiu a dica.

- **Botão "Compartilhar"**

Dentro de **<section id="botao-compartilhar" class="botao-compartilhar">** temos:

- **<button id="btn-compartilhar">Compartilhar</button>**: Botão que possibilita ao usuário compartilhar a dica, possivelmente via redes sociais ou outros meios.

4.4. Campo para comentário ou dica extra

```
47      <!-- Campo para inserir comentário/dica extra -->
48      <section id="area-dica-extra" class="area-dica-extra">
49          <label for="input-dica-extra">Adicionar comentário ou dica:</label>
50          <textarea id="input-dica-extra" rows="3"></textarea>
51      </section>
```

- **<section id="area-dica-extra" class="area-dica-extra">**: Área destinada para o usuário inserir um comentário ou sugerir uma dica extra.

- **<label for="input-dica-extra">Adicionar comentário ou dica:</label>**: Uma etiqueta associada ao campo de texto, melhorando a usabilidade e acessibilidade.

- **<textarea id="input-dica-extra" rows="3"></textarea>**: Uma caixa de texto multilinha onde o usuário pode digitar seu comentário ou sugestão. O atributo **rows="3"** define que inicialmente serão exibidas 3 linhas de texto.

4.5. Botão para enviar o comentário ou dica extra

```
53 |     <!-- Botão de envio -->
54 |     <section id="enviar-dica-extra" class="enviar-dica-extra">
55 |         <button id="btn-enviar-dica-extra">Enviar</button>
56 |     </section>
57 | </section>
58 | </main>
```

- **<section id="enviar-dica-extra" class="enviar-dica-extra">**: Esta seção contém o botão que envia o comentário ou dica adicional.
- **<button id="btn-enviar-dica-extra">Enviar</button>**: Ao ser clicado, este botão deverá, provavelmente via JavaScript, capturar o conteúdo do campo de texto e realizar uma ação (como exibir, armazenar ou enviar o comentário).

5. Rodapé (Footer)

```
60 |     <footer>
61 |         <p>&copy; 2025 Bem-Estar – Todos os direitos reservados</p>
62 |     </footer>
```

- **<footer>**: O rodapé contém informações que normalmente ficam na parte inferior da página.
- **<p>© 2025 Bem-Estar – Todos os direitos reservados</p>**: Um parágrafo que indica os direitos autorais e o nome da marca ou serviço, informando que todos os direitos estão reservados.

6. Elementos adicionais para interatividade

6.1. Modal para mensagens

```
64    <!-- Modal para exibir mensagens -->
65    <section id="modal-container" class="modal-container">
66        <section id="modal-content" class="modal-content">
67            <span id="modal-close" class="modal-close">&times;</span>
68            <p id="modal-message" class="modal-message"></p>
69        </section>
70    </section>
```

- **<section id="modal-container" class="modal-container">**: Um container que abriga o modal, uma janela pop-up que aparece para exibir mensagens ou alertas.
- **<section id="modal-content" class="modal-content">**: Define o conteúdo interno do modal.
- **×**: Um elemento de fechamento (representado pelo símbolo “×”) que permite ao usuário fechar o modal.
- **<p id="modal-message" class="modal-message"></p>**: Um parágrafo que receberá dinamicamente a mensagem que será exibida no modal.

6.2. Container para balões de comentário

```
72    <!-- Container para os balões de comentário -->
73    <section id="bubble-container" class="bubble-container"></section>
74
```

<section id="bubble-container" class="bubble-container">: Um container vazio destinado a ser preenchido dinamicamente (via JavaScript) com “balões” de comentário. Esses balões podem ser usados para exibir feedbacks ou interações do usuário de forma visual.

7. Inclusão do JavaScript

```
75  |  <!-- Importando o arquivo JavaScript externo (dentro da pasta js) -->
76  |  <script src="js/main.js"></script>
77  |  </body>
78  |  </html>
```

- `<script src="js/main.js"></script>`: Essa linha importa um arquivo JavaScript externo localizado na pasta `js` com o nome `main.js`. É nesse script que contá toda a lógica para:

- Gerar dicas de bem-estar quando o usuário clicar em “Gerar Dica”.
 - Manipular os eventos dos botões “Gostei”, “Compartilhar” e “Enviar”.
 - Controlar a exibição do modal e dos balões de comentário.
- `</body>` e `</html>`: Essas tags fecham o corpo e o documento HTML, respectivamente.

Agora, vamos partir para a explicação do código CSS:

1. Reset básico

```
1  /* Reset básico */
2  * {
3      margin: 0;
4      padding: 0;
5      box-sizing: border-box;
6 }
```

- `* { ... }` : O seletor universal (*) aplica as regras a todos os elementos da página.
- `margin: 0; padding: 0;` : Zera as margens e os espaçamentos padrões de todos os elementos, garantindo consistência entre os navegadores.
- `box-sizing: border-box;` : Faz com que a largura e altura de um elemento incluam seu padding e border, facilitando o controle do layout.

2. Estilização do corpo da página

```
8  /* Corpo com gradiente de fundo */
9  body {
10     font-family: 'Quicksand', sans-serif;
11     background: linear-gradient(135deg, #a8edea, #fed6e3);
12     color: #333;
13     min-height: 100vh;
14     display: flex;
15     flex-direction: column;
16     position: relative;
17     overflow-x: hidden;
18 }
```

- `font-family: 'Quicksand', sans-serif;` : Define a fonte principal do corpo, utilizando a fonte importada do Google Fonts e uma fallback para sans-serif.
- `background: linear-gradient(135deg, #a8edea, #fed6e3);` : Cria um fundo com gradiente linear em um ângulo de 135 graus, passando de um tom de azul claro (#a8edea) para um rosa claro (#fed6e3).
- `color: #333;` : Define a cor padrão do texto como um cinza escuro.
- `min-height: 100vh;` : Garante que o corpo ocupe, no mínimo, toda a altura visível da janela (viewport).
- `display: flex; flex-direction: column;` : Utiliza o Flexbox para organizar os elementos internos em uma coluna, permitindo um layout responsivo.
- `position: relative;` : Permite que elementos posicionados de forma absoluta dentro do `body` sejam referenciados em relação a ele.

- `overflow-x: hidden;` : Evita que haja rolagem horizontal, ocultando conteúdos que possam ultrapassar a largura da tela.

3. Cabeçalho

```

20  /* Cabeçalho */
21  header {
22    text-align: center;
23    padding: 1.5rem 1rem;
24  }
25
26  header h1 {
27    font-size: 2rem;
28    font-weight: 700;
29    color: #333;
30  }

```

- `header { ... }`: Centraliza o conteúdo do cabeçalho com `text-align: center` e adiciona espaçamento interno (padding) para dar "respiro" ao conteúdo.
- `header h1 { ... }`: Define o estilo do título principal:
 - `font-size: 2rem;` – Tamanho de fonte grande.
 - `font-weight: 700;` – Peso da fonte bem-marcado (negrito).
 - `color: #333;` – Cor do texto em cinza escuro.

4. Main

```

32  /* Main centralizado */
33  main {
34    flex: 1;
35    display: flex;
36    justify-content: center;
37    align-items: center;
38    padding: 1rem;
39  }

```

- `flex: 1;` : Faz com que o `<main>` ocupe todo o espaço disponível entre o cabeçalho e o rodapé.
- `display: flex; justify-content: center; align-items: center;` : Utiliza Flexbox para centralizar o conteúdo horizontal e verticalmente.
- `padding: 1rem;` : Adiciona espaçamento interno para evitar que o conteúdo fique colado nas bordas.

5. Container principal

```
41  /* Container principal (card) */
42  #gerador-de-dicas {
43    background-color: #fff;
44    max-width: 480px;
45    width: 100%;
46    border-radius: 12px;
47    box-shadow: 0 8px 16px rgba(0, 0, 0, 0.15);
48    padding: 2rem 1.5rem;
49    display: flex;
50    flex-direction: column;
51    align-items: center;
52    animation: fadeIn 0.5s ease-in-out;
53    z-index: 1;
54 }
```

- **background-color: #fff;** : Define o fundo branco para o card.
 - **max-width: 480px; width: 100%;** : Limita a largura máxima em 480 pixels, mas permite que ele ocupe 100% da largura disponível em telas menores.
 - **border-radius: 12px;** : Adiciona cantos arredondados ao container.
 - **box-shadow: 0 8px 16px rgba(0, 0, 0, 0.15);** : Cria uma sombra para dar efeito de profundidade.
 - **padding: 2rem 1.5rem;** : Espaçamento interno generoso para manter o conteúdo afastado das bordas.
 - **display: flex; flex-direction: column; align-items: center;** : Organiza os elementos internos em coluna e centraliza-os.
 - **animation: fadeIn 0.5s ease-in-out;** : Aplica uma animação de entrada suave (definida logo abaixo).
 - **z-index: 1;** : Garante que o card fique acima de outros elementos de fundo.
- A ideia é que, ao clicar no botão "Gerar Dica", uma dica seja exibida no parágrafo reservado. Em etapas futuras, será adicionado o CSS para estilizar a página e o JavaScript para fazer o botão funcionar, além de criarmos um botão de like, que irá contar quantos likes a página teve, uma caixinha de comentários, onde os usuários poderão comentar acerca das dicas, e, por último, um botão de compartilhamento ☺

6. Animação de entrada

```
56  /* Animação suave de entrada */
57  @keyframes fadeIn {
58    from { opacity: 0; transform: translateY(20px); }
59    to { opacity: 1; transform: translateY(0); }
60  }
61
```

- `@keyframes fadeIn { ... }` : Define uma animação chamada fadeIn.
- `from { opacity: 0; transform: translateY(20px); }` : No início, o elemento começa transparente e deslocado 20 pixels para baixo.
- `to { opacity: 1; transform: translateY(0); }` : No final, o elemento se torna totalmente opaco e retorna à sua posição original.

7. Seções de ação e área de comentário

```
62  .area-gerar-dica,
63  .enviar-dica-extra,
64  .area-dica-extra {
65    margin-bottom: 1.5rem;
66    width: 100%;
67    display: flex;
68    justify-content: center;
69  }
```

Seletores agrupados (`.area-gerar-dica, .enviar-dica-extra, .area-dica-extra`) : Aplica as mesmas regras para essas três seções:

- `margin-bottom: 1.5rem;` : Espaço abaixo de cada seção para separação visual.
- `width: 100%;` : Ocupam 100% da largura do container pai.
- `display: flex; justify-content: center;` : Centralizam seu conteúdo usando Flexbox.

8. Container para ações

```
71  /* Container para os botões separados (AÇÕES) */
72  .acoes {
73    width: 100%;
74    display: flex;
75    justify-content: space-between;
76    gap: 1rem;
77    margin-bottom: 1.5rem;
78 }
```

- **display: flex; justify-content: space-between;** : Organiza os botões de ação (como "Gostei" e "Compartilhar") de forma que fiquem separados e alinhados.
- **gap: 1rem;** : Adiciona um espaço de 1rem entre os botões.
- **margin-bottom: 1.5rem;** : Espaçamento abaixo do container, separando-o de outros elementos.

9. Container de cada botão

```
80  /* Cada botão em seu container */
81  .botao-like,
82  .botao-compartilhar {
83    flex: 1;
84    display: flex;
85    justify-content: center;
86 }
```

- **flex: 1;** : Faz com que cada container ocupe igualmente o espaço disponível.
- **display: flex; justify-content: center;** : Centraliza o botão dentro do seu container.

10. Estilização dos botões gerais

```
88  /* Botões gerais */
89  button {
90    background-color: #ff8fab;
91    color: #fff;
92    border: none;
93    border-radius: 6px;
94    padding: 0.75rem 1.5rem;
95    font-size: 1rem;
96    font-weight: 600;
97    cursor: pointer;
98    transition: background-color 0.3s, transform 0.3s;
99    width: 100%;
100 }
101
```

- **background-color: #ff8fab;** : Define uma cor de fundo rosa para os botões.
- **color: #fff;** : Define a cor do texto como branco.
- **border: none;** : Remove quaisquer bordas padrões.
- **border-radius: 6px;** : Arredonda levemente os cantos dos botões.
- **padding: 0.75rem 1.5rem;** : Espaçamento interno para deixar o botão mais “clicável” e com bom tamanho.
- **font-size: 1rem; font-weight: 600;** : Define o tamanho e a espessura da fonte.
- **cursor: pointer;** : Altera o cursor para indicar que o elemento é clicável.
- **transition: background-color 0.3s, transform 0.3s;** : Adiciona uma transição suave para mudanças de cor e posição durante interações.
- **width: 100%;** : Garante que o botão ocupe toda a largura do seu container.

10.1. Efeitos de Hover e Active para os botões

```
102  button:hover {  
103      background-color: #f76790;  
104      transform: translateY(-2px);  
105  }  
106  
107  button:active {  
108      transform: translateY(0);  
109  }  
110
```

- **button:hover { ... }** : Quando o usuário passa o mouse sobre o botão:
 - A cor de fundo muda para um tom um pouco mais escuro (#f76790).
 - O botão se desloca ligeiramente para cima (`translateY(-2px)`), criando um efeito de "elevação".
- **button:active { ... }** : Ao clicar (estado ativo), o botão retorna à posição original, removendo o deslocamento.

11. Exibição da dica

```
111  /* Exibição da dica */  
112  .exibicao-dica {  
113      text-align: center;  
114      margin-bottom: 1.5rem;  
115      width: 100%;  
116  }
```

- **text-align: center;** : Centraliza o texto dentro da área de exibição.
- **margin-bottom: 1.5rem;** : Espaçamento inferior para separar de outros elementos.
- **width: 100%;** : Ocupa toda a largura disponível do container pai.

12. Detalhe (estrela) e texto da dica

```
118  /* Ícone ou estrela no centro */
119  .icone-estrela {
120    font-size: 2rem;
121    margin-bottom: 0.5rem;
122    color: #ff8fab;
123  }
124
125  #texto-dica {
126    font-size: 1.1rem;
127    font-weight: 600;
128    color: #444;
129    min-height: 2.5em;
130  }
```

- `.icone-estrela { ... }` : Define o estilo do ícone (nesse caso, a estrela):
 - `font-size: 2rem;` – Tamanho grande para chamar a atenção.
 - `margin-bottom: 0.5rem;` – Espaço abaixo para separar do texto.
 - `color: #ff8fab;` – Cor rosa combinando com os botões.
- `#texto-dica { ... }` : Estiliza o parágrafo onde a dica é exibida:
 - `font-size: 1.1rem;` – Tamanho de fonte um pouco maior que o padrão.
 - `font-weight: 600;` – Peso médio para dar destaque.
 - `color: #444;` – Cor de texto em tom de cinza.
 - `min-height: 2.5em;` – Garante uma altura mínima para que o espaço fique reservado mesmo se a dica estiver vazia ou curta.

13. Área para comentários ou dica extra

```
132     .area-dica-extra label {  
133         margin-bottom: 0.5rem;  
134         font-weight: 600;  
135         color: #666;  
136     }  
137  
138     .area-dica-extra textarea {  
139         resize: none;  
140         border-radius: 6px;  
141         border: 1px solid #ccc;  
142         padding: 0.5rem;  
143         font-size: 1rem;  
144         font-family: 'Quicksand', sans-serif;  
145     }  
146
```

- `.area-dica-extra label { ... }` : Estiliza o rótulo do campo de texto:

- `margin-bottom: 0.5rem;` – Espaço abaixo do label.
- `font-weight: 600;` – Peso da fonte para destacá-lo.
- `color: #666;` – Cor de texto em um cinza intermediário.

- `.area-dica-extra textarea { ... }` : Define o estilo da área de texto:

- `resize: none;` – Impede que o usuário redimensione a caixa.
- `border-radius: 6px;` – Cantos arredondados.
- `border: 1px solid #ccc;` – Borda leve e em tom de cinza.
- `padding: 0.5rem;` – Espaçamento interno para conforto na digitação.
- `font-size: 1rem; font-family: 'Quicksand', sans-serif;` – Garante consistência com o restante da página.

14. Rodapé

```
147  /* Rodapé */  
148  footer {  
149    text-align: center;  
150    padding: 1rem;  
151    font-size: 0.9rem;  
152    color: #555;  
153 }
```

- **text-align: center;** : Centraliza o conteúdo do rodapé.
- **padding: 1rem;** : Espaçamento interno que deixa o conteúdo "respirar".
- **font-size: 0.9rem;** : Tamanho de fonte ligeiramente menor, comum para informações de rodapé.
- **color: #555;** : Cor de texto em um cinza médio, sutil e discreto.

15. Responsividade – Media Query

```
155  @media (max-width: 600px) {  
156    #gerador-de-dicas {  
157      margin: 0 1rem;  
158      padding: 1.5rem 1rem;  
159    }  
160  
161    header h1 {  
162      font-size: 1.5rem;  
163    }  
164  
165    .acoes {  
166      flex-direction: column;  
167    }  
168  
169    .botao-like,  
170    .botao-compartilhar {  
171      width: 100%;  
172    }  
173  }
```

- **@media (max-width: 600px) { ... } :** Aplica estilos específicos para telas com largura de até 600 pixels, melhorando a experiência em dispositivos móveis.

- **Ajustes no #gerador-de-dicas:**

- **margin: 0 1rem;** – Adiciona margens laterais para evitar que o card fique colado nas bordas.
- **padding: 1.5rem 1rem;** – Reduz o espaçamento interno para telas menores.
- **header h1 { font-size: 1.5rem; } :** Diminui o tamanho do título para caber melhor em telas pequenas.
- **.acoes { flex-direction: column; } :** Altera a disposição dos botões de ação de uma linha para uma coluna, facilitando a interação em dispositivos móveis.
- **.botao-like, .botao-compartilhar { width: 100%; } :** Garante que cada botão ocupe 100% da largura disponível quando estiverem em coluna.

16. Estilização da modal

```
175  /* Estilização da Modal */
176  #modal-container {
177    position: fixed;
178    top: 0;
179    left: 0;
180    width: 100%;
181    height: 100%;
182    background: □rgba(0, 0, 0, 0.5);
183    display: none;
184    align-items: center;
185    justify-content: center;
186    z-index: 1000;
187  }
188
189 #modal-content {
190   background: ■#fff;
191   border-radius: 12px;
192   padding: 1.5rem;
193   box-shadow: 0 8px 16px □rgba(0, 0, 0, 0.25);
194   max-width: 400px;
195   width: 90%;
196   text-align: center;
197   position: relative;
198 }
199
200 #modal-close {
201   position: absolute;
202   top: 10px;
203   right: 15px;
204   font-size: 1.5rem;
205   cursor: pointer;
206   color: ■#888;
207 }
208
209 #modal-message {
210   margin-top: 1rem;
211   font-size: 1.1rem;
212   color: □#444;
213 }
```

- **#modal-container { ... } :** Este container cobre toda a tela:
 - **position: fixed; top: 0; left: 0; width: 100%; height: 100%;** – Garante que o modal ocupe a tela inteira.
 - **background: rgba(0, 0, 0, 0.5);** – Aplica um fundo semi-transparente escuro, criando um efeito de "escurecimento" para destacar o modal.
 - **display: none;** – Inicialmente oculto; será exibido via JavaScript quando necessário.
 - **align-items: center; justify-content: center;** – Centraliza o conteúdo do modal.
 - **z-index: 1000;** – Garante que o modal fique acima de outros elementos.
- **#modal-content { ... } :** Define o estilo da caixa interna do modal:

- `background: #fff;` – Fundo branco para o conteúdo.
 - `border-radius: 12px;` – Cantos arredondados.
 - `padding: 1.5rem;` – Espaçamento interno confortável.
 - `box-shadow: 0 8px 16px rgba(0, 0, 0, 0.25);` – Sombra para dar profundidade.
 - `max-width: 400px; width: 90%;` – Limita a largura para não ultrapassar 400px, mas permite 90% da largura em telas pequenas.
 - `text-align: center;` – Centraliza o texto.
 - `position: relative;` – Permite posicionar elementos internos de forma absoluta (como o botão de fechar).
-
- `#modal-close { ... }` : Estiliza o botão de fechar do modal:
 - `position: absolute; top: 10px; right: 15px;` – Posicionado no canto superior direito.
 - `font-size: 1.5rem;` – Tamanho de fonte que indica interatividade.
 - `cursor: pointer;` – Indica que o elemento é clicável.
 - `color: #888;` – Cor em tom de cinza para não chamar muita atenção.
-
- `#modal-message { ... }` : Estiliza a mensagem dentro do modal:
 - `margin-top: 1rem;` – Espaço acima da mensagem.
 - `font-size: 1.1rem;` – Tamanho de fonte confortável para leitura.
 - `color: #444;` – Cor de texto em cinza escuro.

17. Container para balões de comentário

```
215  /* Container para os comentários em balões */  
216  #bubble-container {  
217      position: fixed;  
218      top: 0;  
219      left: 0;  
220      width: 100%;  
221      height: 100%;  
222      pointer-events: none;  
223      z-index: 500;  
224 }
```

#bubble-container { ... } : Cria um container que cobre toda a tela para exibir balões de comentário:

- **position: fixed;** : Fixa o container na tela.
- **pointer-events: none;** : Permite que cliques passem por ele, não interferindo na usabilidade.
- **z-index: 500;** : Posiciona-o abaixo do modal, mas acima do conteúdo principal.

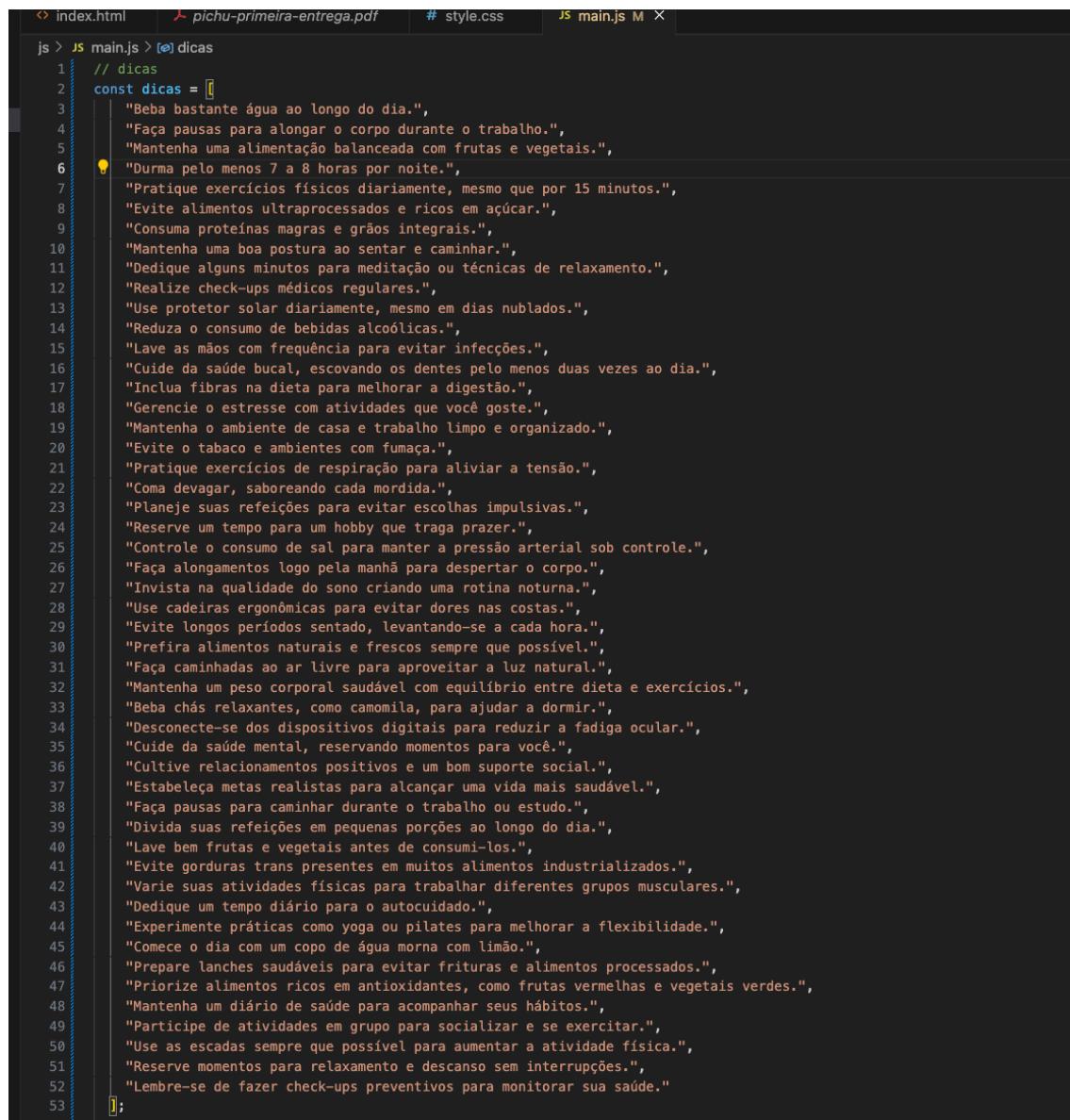
18. Estilização dos balões de comentário

```
226  /* Estilo dos balões de comentário */
227  .comment-bubble {
228    position: absolute;
229    background: #fff;
230    border: 2px solid #ff8fab;
231    border-radius: 20px;
232    padding: 1.5rem 2rem;
233    box-shadow: 0 4px 8px rgba(0,0,0,0.2);
234    font-size: 1.9rem;
235    color: #444;
236    max-width: 400px;
237    opacity: 0.9;
238 }
```

- `.comment-bubble { ... } :` Define o estilo dos balões que podem aparecer para exibir comentários:
- `position: absolute;`: Permite posicionar os balões livremente na tela.
- `background: #fff;`: Fundo branco para contraste.
- `border: 2px solid #ff8fab;`: Borda de 2px em tom rosa, para destacar o balão.
- `border-radius: 20px;`: Bordas bem arredondadas, conferindo um visual de "balão".
- `padding: 1.5rem 2rem;`: Espaçamento interno para que o conteúdo não fique colado às bordas.
- `box-shadow: 0 4px 8px rgba(0,0,0,0.2);`: Sombra suíl para dar profundidade.
- `font-size: 1.9rem;`: Tamanho de fonte grande, possivelmente para destacar o comentário.
- `color: #444;`: Cor do texto em cinza escuro.
- `max-width: 400px;`: Limita a largura dos balões para manter a legibilidade.
- `opacity: 0.9;`: Um toque de transparência para que o balão não fique totalmente opaco.

Por fim, vamos entender um pouco do arquivo JS que já foi desenvolvido até então:

1. Array com dicas para sorteio aleatório



```
// dicas
const dicas = [
    "Beba bastante água ao longo do dia.",
    "Faça pausas para alongar o corpo durante o trabalho.",
    "Mantenha uma alimentação balanceada com frutas e vegetais.",
    "Durma pelo menos 7 a 8 horas por noite.",
    "Pratique exercícios físicos diariamente, mesmo que por 15 minutos.",
    "Evite alimentos ultraprocessados e ricos em açúcar.",
    "Consuma proteínas magras e grãos integrais.",
    "Mantenha uma boa postura ao sentar e caminhar.",
    "Dedique alguns minutos para meditação ou técnicas de relaxamento.",
    "Realize check-ups médicos regulares.",
    "Use protetor solar diariamente, mesmo em dias nublados.",
    "Reduza o consumo de bebidas alcoólicas.",
    "Lave as mãos com frequência para evitar infecções.",
    "Cuide da saúde bucal, escovando os dentes pelo menos duas vezes ao dia.",
    "Inclua fibras na dieta para melhorar a digestão.",
    "Gerencie o estresse com atividades que você goste.",
    "Mantenha o ambiente de casa e trabalho limpo e organizado.",
    "Evite o tabaco e ambientes com fumaça.",
    "Pratique exercícios de respiração para aliviar a tensão.",
    "Coma devagar, saboreando cada mordida.",
    "Planeje suas refeições para evitar escolhas impulsivas.",
    "Reserve um tempo para um hobby que traga prazer.",
    "Controle o consumo de sal para manter a pressão arterial sob controle.",
    "Faça alongamentos logo pela manhã para despertar o corpo.",
    "Invista na qualidade do sono criando uma rotina noturna.",
    "Use cadeiras ergonômicas para evitar dores nas costas.",
    "Evite longos períodos sentado, levantando-se a cada hora.",
    "Prefira alimentos naturais e frescos sempre que possível.",
    "Faça caminhadas ao ar livre para aproveitar a luz natural.",
    "Mantenha um peso corporal saudável com equilíbrio entre dieta e exercícios.",
    "Beba chás relaxantes, como camomila, para ajudar a dormir.",
    "Desconecte-se dos dispositivos digitais para reduzir a fadiga ocular.",
    "Cuide da saúde mental, reservando momentos para você.",
    "Cultive relacionamentos positivos e um bom suporte social.",
    "Estabeleça metas realistas para alcançar uma vida mais saudável.",
    "Faça pausas para caminhar durante o trabalho ou estudo.",
    "Divida suas refeições em pequenas porções ao longo do dia.",
    "Lave bem frutas e vegetais antes de consumi-los.",
    "Evite gorduras trans presentes em muitos alimentos industrializados.",
    "Varie suas atividades físicas para trabalhar diferentes grupos musculares.",
    "Dedique um tempo diário para o autocuidado.",
    "Experimente práticas como yoga ou pilates para melhorar a flexibilidade.",
    "Comece o dia com um copo de água morna com limão.",
    "Prepare lanches saudáveis para evitar frituras e alimentos processados.",
    "Priorize alimentos ricos em antioxidantes, como frutas vermelhas e vegetais verdes.",
    "Mantenha um diário de saúde para acompanhar seus hábitos.",
    "Participe de atividades em grupo para socializar e se exercitar.",
    "Use as escadas sempre que possível para aumentar a atividade física.",
    "Reserve momentos para relaxamento e descanso sem interrupções.",
    "Lembre-se de fazer check-ups preventivos para monitorar sua saúde."
];
```

- `const dicas = [...];`: Cria uma lista (array) com várias dicas de saúde. Cada item entre aspas representa uma dica específica, armazenada como uma string.

2. Acesso ao HTML para interagir com ele via JS

```
57 // pegando os elementos da pagina
58 const btnGerarDica = document.getElementById("btn-gerar-dica");
59 const textoDica = document.getElementById("texto-dica");
60 const btnLike = document.getElementById("btn-like");
61 const btnCompartilhar = document.getElementById("btn-compartilhar");
62 const btnEnviarDicaExtra = document.getElementById("btn-enviar-dica-extra");
63 const inputDicaExtra = document.getElementById("input-dica-extra");
64 const modalContainer = document.getElementById("modal-container");
65 const modalMessage = document.getElementById("modal-message");
66 const modalClose = document.getElementById("modal-close");
67
```

- Esses comandos selecionam elementos HTML da interface da página, permitindo que o JavaScript interaja com eles.
- `document.getElementById("id")`: busca um elemento pelo seu atributo id.
- Cada variável criada faz referência a um botão, campo de entrada ou área de exibição na interface:
 - Botões: para gerar dicas, curtir, compartilhar e enviar comentário.
 - Campos de texto: para mostrar a dica e receber input do usuário.
 - Modal: janela que exibe mensagens temporárias para o usuário.
 - Container de balões: exibe visualmente os comentários enviados.

3. Gera uma dica aleatória quando clicado

```
59 // gerando uma dica aleatória
60 btnGerarDica.addEventListener("click", function() {
61   const indiceAleatorio = Math.floor(Math.random() * dicas.length);
62   textoDica.textContent = dicas[indiceAleatorio];
63 });
64
```

- `btnGerarDica.addEventListener("click", function() { ... })` : Adiciona um ouvinte de evento que detecta quando o botão é clicado. Dentro da função, definimos o que acontece quando esse clique ocorre.
- `const indiceAleatorio = Math.floor(Math.random() * dicas.length);` : Gera um número aleatório inteiro entre 0 e o total de dicas.
 - `Math.random()` : gera um número decimal aleatório entre 0 e 1.
 - `Math.floor()` : arredonda esse número para baixo, garantindo um índice válido.
- `textoDica.textContent = dicas[indiceAleatorio];` : Altera o conteúdo de texto do elemento `textoDica`, exibindo a dica escolhida aleatoriamente.

4. Impede o envio de comentários vazios

```
76 // exibir modal - por favor, escreva algo antes de enviar
77 function showModal(message) {
78   modalMessage.textContent = message;
79   modalContainer.style.display = "flex";
80 }
```

Define uma função que exibe a modal com a mensagem recebida como argumento.

- `modalMessage.textContent = message;`; insere o texto na modal.
- `modalContainer.style.display = "flex";`; torna a modal visível.

5. Ocultar a modal da tela

```
82 // ocultar modal
83 function hideModal() {
84   modalContainer.style.display = "none";
85 }
```

Define uma função que oculta a modal da tela.

- `modalContainer.style.display = "none";`; esconde a janela modal.

6. Oculta modal qual clica no 'x'

```
87 // fechar a modal - 'x'
88 modalClose.addEventListener("click", hideModal);
89 modalContainer.addEventListener("click", function(e) {
90   if (e.target === modalContainer) {
91     hideModal();
92   }
93 });
94
```

• Adiciona um evento para fechar a modal ao clicar no botão "X".

• Fecha a modal se o usuário clicar fora da caixa de conteúdo.

7. Adiciona comentário em formato de balão

```
95 // função para adicionar o comentário em balão
96 function addBubbleComment(text) {
97   const bubble = document.createElement("div");
98   bubble.classList.add("comment-bubble");
99   bubble.textContent = text;
100
101  // balão - lateral esquerda ou direita
102  const side = Math.random() < 0.5 ? "left" : "right";
103  // posição vertical aleatória
104  const topPosition = Math.floor(Math.random() * 70) + 10;
105
106  if (side === "left") {
107    bubble.style.left = "10px";
108  } else {
109    bubble.style.right = "10px";
110  }
111  bubble.style.top = topPosition + "%";
112
113  // adicionar o balão no container (visível até o final da sessão)
114  bubbleContainer.appendChild(bubble);
115 }
```

- `function addBubbleComment(text) { ... }` : Define uma função que recebe um texto e cria um balão de comentário na tela com esse conteúdo.
- `const bubble = document.createElement("div")` : Cria um novo elemento `<div>` que representará o balão de comentário.
- `bubble.classList.add("comment-bubble")` : Adiciona a classe CSS "`comment-bubble`" ao elemento, aplicando o estilo visual de balão.
- `bubble.textContent = text`: Define o conteúdo do balão como sendo o texto passado à função.
- `const side = Math.random() < 0.5 ? "left" : "right"`: Escolhe aleatoriamente se o balão aparecerá do lado esquerdo ou direito da tela.
 - `Math.random() < 0.5`: retorna `true` ou `false` com 50% de chance para cada lado.
- `const topPosition = Math.floor(Math.random() * 70) + 10`: Define uma posição vertical aleatória entre 10% e 80% da altura da tela.
 - `Math.random() * 70`: gera um número entre 0 e 70.
 - `+ 10`: garante que a posição nunca fique muito colada no topo.

```
• if (side === "left") { bubble.style.left = "10px"; } else { bubble.style.right = "10px"; }: Define a posição horizontal do balão com base na variável side.
```

Se for "`left`", alinha o balão à esquerda; caso contrário, alinha à direita.

```
• bubble.style.top = topPosition + "%": Define a posição vertical do balão com base no número aleatório calculado antes.
```

```
• bubbleContainer.appendChild(bubble): Adiciona o balão recém-criado ao contêiner bubbleContainer, tornando-o visível na tela.
```

8. Função para adição de comentários

```
117 // comentar
118 btnEnviarDicaExtra.addEventListener("click", function() {
119   const textoExtra = inputDicaExtra.value.trim();
120   if (textoExtra) {
121     showModal(`Seu comentário/dica extra: ${textoExtra}`);
122     addBubbleComment(textoExtra);
123     inputDicaExtra.value = "";
124   } else {
125     showModal("Por favor, escreva algo antes de enviar.");
126   }
127 });
128
```

```
• btnEnviarDicaExtra.addEventListener("click", function() { ... }): Adiciona um ouvinte ao botão de envio de dica extra.
```

Quando clicado, executa a função que lê o que foi digitado e valida o conteúdo.

```
• const textoExtra = inputDicaExtra.value.trim(): Obtém o texto digitado no campo de entrada e remove espaços extras nas bordas.
```

```
• if (textoExtra) { ... }: Verifica se o usuário realmente digitou algo.
```

- `showModal(...):` exibe a mensagem com o conteúdo enviado.

- `addBubbleComment(textoExtra):` cria um balão na tela com o comentário.

- `inputDicaExtra.value = "":` limpa o campo de entrada após o envio.

```
• else { showModal("Por favor, escreva algo antes de enviar."); }
```

Se o campo estiver vazio, mostra uma mensagem pedindo que o usuário escreva algo antes de tentar enviar.

9. Função para dar like

```
129 // like
130 btnLike.addEventListener("click", function() {
131   likes++;
132   showModal(`Você curtiu esta dica! Total de likes: ${likes}`);
133 });
134
```

- Detecta o clique no botão de curtir.
- **likes++**: incrementa o contador.
- **showModal(...)**: exibe uma mensagem com o número atualizado de curtidas.

10. Função de compartilhar – copia o link da página

```
135 // compartilhar - copia o link
136 btnCompartilhar.addEventListener("click", function () {
137   const url = window.location.href;
138   if (navigator.clipboard && window.isSecureContext) {
139     // copia com a API moderna
140     navigator.clipboard.writeText(url).then(() => {
141       showModal("Link copiado com sucesso!");
142     }).catch(() => {
143       showModal("Erro ao copiar o link. Tente novamente.");
144     });
145   } else {
146     // fallback para navegadores mais antigos
147     const textArea = document.createElement("textare");
148     textArea.value = url;
149     document.body.appendChild(textArea);
150     textArea.focus();
151     textArea.select();
152
153     try {
154       const sucesso = document.execCommand("copy");
155       showModal(sucesso ? "Link copiado com sucesso!" : "Não foi possível copiar o link.");
156     } catch (err) {
157       showModal("Erro ao copiar o link.");
158     }
159
160     document.body.removeChild(textArea);
161   }
162 });

163
```

- **btnCompartilhar.addEventListener("click", function () { ... })**: Adiciona um ouvinte de evento ao botão de compartilhar.
Quando clicado, executa a função que tenta copiar o link da página para a área de transferência.
- **const url = window.location.href**: Obtém a URL atual da página, que será copiada.

- `if (navigator.clipboard && window.isSecureContext) { ... }:` Verifica se o navegador tem suporte à API moderna `clipboard` e se a página está em um contexto seguro (`https`).

Isso garante que a cópia via `navigator.clipboard.writeText()` funcione.

- `navigator.clipboard.writeText(url).then(() => { ... }).catch(() => { ... }):`

Tenta copiar o link da página.

- `.then(...):` se a cópia for bem-sucedida, exibe uma modal com a mensagem "`Link copiado com sucesso!`".

- `.catch(...):` se der erro, exibe "`Erro ao copiar o link. Tente novamente.`".

- `else { ... }:` Caso o navegador não tenha suporte à API moderna, usa um método alternativo (fallback).

- `const textArea = document.createElement("textArea"):` Cria dinamicamente um campo `<textArea>` para simular a cópia.

- `textArea.value = url:` Define o valor do campo como sendo a URL da página.

- `document.body.appendChild(textArea):` Adiciona esse campo no corpo da página (fora da visão do usuário).

- `textArea.focus(); textArea.select():` Foca e seleciona o texto da URL automaticamente.

- `document.execCommand("copy"):` Executa o comando que copia o conteúdo selecionado para a área de transferência.

- `showModal(...):` Exibe uma mensagem informando se a cópia foi bem-sucedida ou não.

- `document.body.removeChild(textArea):` Remove o campo `<textArea>` da página após a cópia.

Agora, temos todas as funcionalidades do nosso site: gerar uma dica de bem estar, curtir e compartilhar a página e, por fim, deixar um comentário, que ficará em forma de balão em algum lugar aleatório da página quando enviado.

E ele ficou assim:

