Contos de Festa

Web Mobile

Matteo Domiciano Varnier -10390247
Fernando Pegoraro Bilia – 10402097
Rodrigo Barbosa – 10748173

São Paulo, SP

Sumário

Motivação do projeto

No Conto de Festas, acreditamos que cada celebração merece ser única e inesquecível. A ideia de criar este espaço surgiu para facilitar a forma de apresentar nossos trabalhos e tornar a escolha dos clientes ainda mais prática. Assim, reunimos em um só lugar todos os nossos kits de festas completos, prontos para transformar momentos especiais em memórias duradouras.

Nosso compromisso é tornar cada festa prática, divertida e visualmente incrível, cuidando de todos os detalhes para que você possa aproveitar o momento sem preocupações. Com o Conto de Festas, transformar sonhos em realidade nunca foi tão fácil!

Caráter extensionistas

O Conto de Festas nasce como uma iniciativa de extensão que conecta o conhecimento acadêmico (tecnologia, design e gestão) às necessidades reais do mercado local de festas. O projeto promove a aplicação prática do saber por meio do desenvolvimento de uma plataforma que organiza o portfólio, padroniza a apresentação de kits e facilita o atendimento, ampliando a visibilidade digital e a profissionalização do negócio.

Com isso, o site gera impacto social ao fortalecer um empreendimento local, melhorar processos (catálogo, orçamento e comunicação) e tornar o serviço mais acessível aos clientes. Ao mesmo tempo, oferece aos estudantes um ambiente de aprendizagem situada, envolvendo análise de requisitos, usabilidade, métricas de atendimento e melhoria contínua, em diálogo direto com a comunidade.

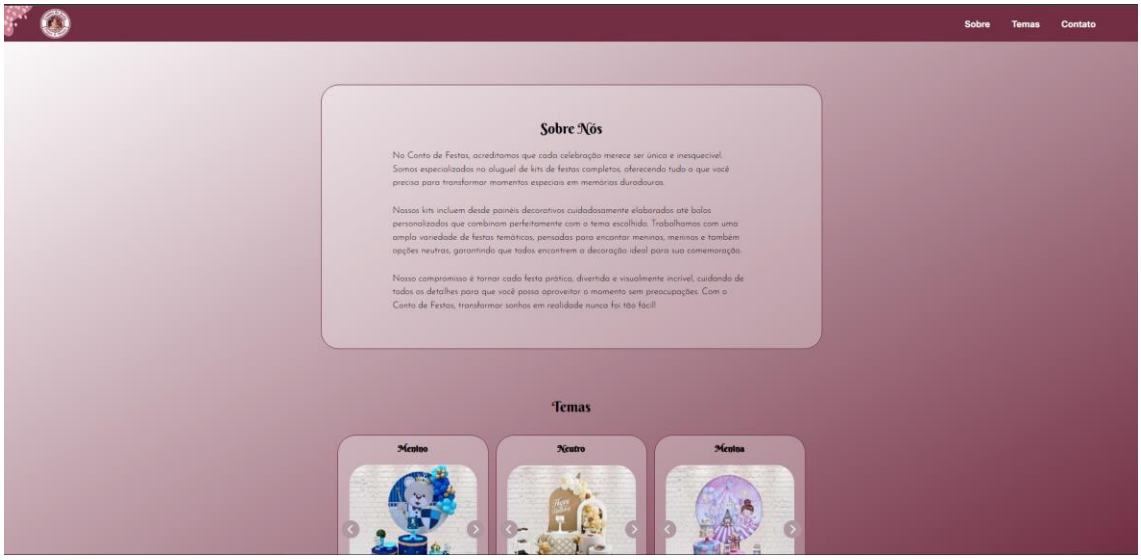
Protótipo

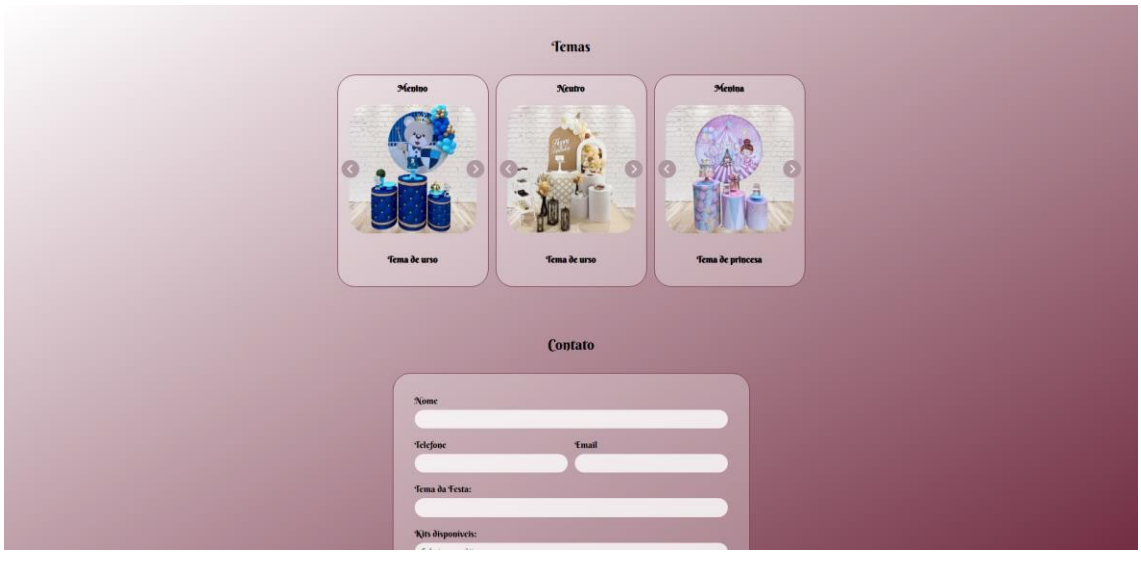
A página principal reúne todas as funcionalidades essenciais do site, com uma navegação simples e intuitiva através do menu no cabeçalho.

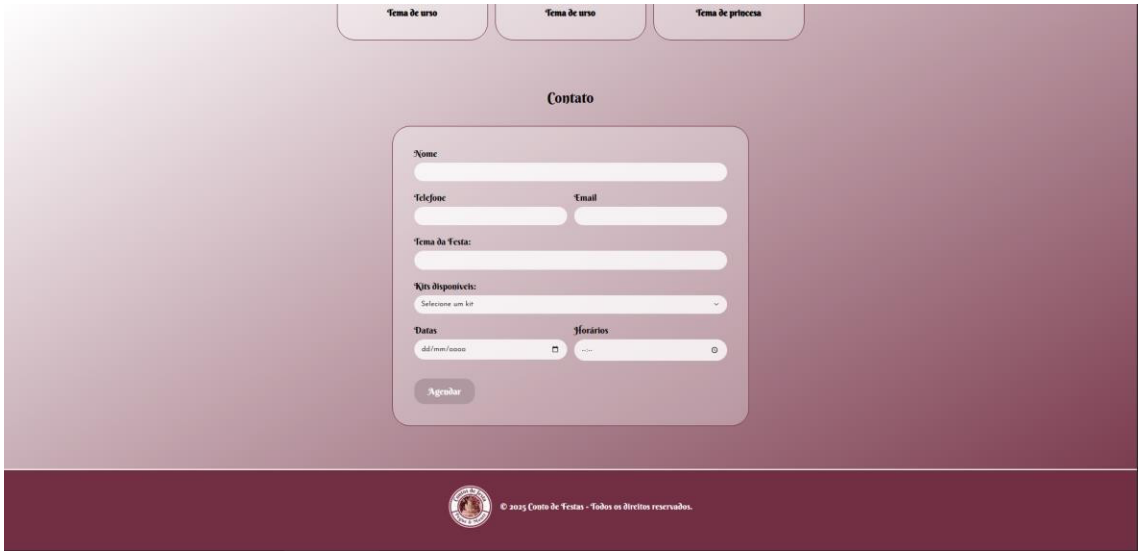
- Sobre Nós – espaço dedicado a apresentar a empresa, sua história e o propósito do Conto de Festas.

- Temas – galeria que exibe os kits de festas já realizados, permitindo ao cliente conhecer as opções disponíveis e se inspirar para sua comemoração.

- Contato – seção com um formulário prático, onde o cliente pode preencher seus dados, escolher o tema desejado, o kit desejado e solicitar o agendamento de forma rápida e direta.







Código

O projeto foi desenvolvido utilizando as seguintes tecnologias:

- **HTML5**
- **CSS3**
- **JavaScript**

Estrutura do Código

HTML

Trecho 1:

```
<!DOCTYPE html>
<html lang="pt-BR">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-
width, initial-scale=1.0" />
    <title>Conto de Festas</title>
    <link rel="stylesheet" href="styles/index.css"
/>

    <link rel="preconnect"
href="https://fonts.googleapis.com" />
    <link rel="preconnect"
href="https://fonts.gstatic.com" crossorigin />
    <link

      href="https://fonts.googleapis.com/css2?famil
      y=Berkshire+Swash&family=Inter:ital,opsz,wght
      @0,14..32,100..900;1,14..32,100..900&family=J
      osefin+Sans:ital,wght@0,100..700;1,100..700&d
      isplay=swap"
            rel="stylesheet"
          />
  </head>
```

Nesse trecho, iniciamos a estrutura do documento HTML. No elemento <head>, que representa o cabeçalho da página, definimos algumas configurações importantes:

- O título da aba do navegador será **"Conto de Festas"**.
- Vinculamos a folha de estilo index.css, responsável pela aparência do site.
- Fazemos a conexão com o serviço do Google Fonts, importando as fontes que serão utilizadas na estilização dos textos da página.

Trecho 2:

```
<body>
    <header class="container flex">
      <img
        src="assets/header_image.png"
        alt="Imagem do header"
        class="header-image"
      />

      <article class="flex">
        <img src="assets/logo.png" alt="Logo" />
      </article>

      <nav class="nav-links flex">
        <a href="#about">Sobre</a>
        <a href="#themes">Temas</a>
        <a href="#submit">Contato</a>
      </nav>
    </header>
```

Nesse trecho, estamos iniciando o corpo do site **<body>,** iniciamos com o titulo da pagina

- Foi inserido uma imagem para decorar o cabeçalho
- Inserido a logo do site
- Foi criado um navegador para com três menus, aonde ira direcionar o usuário no site

Trecho 3:

```
<section id="about" class="container">
  <article>
    <h1>Sobre Nós</h1>
    <p>
      No Conto de Festas, acreditamos que cada
      celebração merece ser única e inesquecível. Somos
      especializados no aluguel de kits de festas
      completos, oferecendo tudo o que você precisa para
      transformar momentos especiais em memórias
      duradouras.
    </p>

    <p>
      Nossos kits incluem desde painéis decorativos
      cuidadosamente elaborados até bolos personalizados
      que combinam perfeitamente com o tema escolhido.
      Trabalhamos com uma ampla variedade de
      festastemáticas, pensadas para encantar meninas,
      meninos e também opções neutras, garantindo que
      todos encontrem a decoração ideal para sua
      comemoração.
    </p>

    <p>
      Nosso compromisso é tornar cada festa prática,
      divertida e visualmente incrível, cuidando de
      todos os detalhes para que você possa aproveitar o
      momento sem preocupações. Com o Conto de Festas,
      transformar sonhos em realidade nunca foi tão
      fácil!
    </p>
  </article>
</section>
```

Nesse trecho, criamos a seção **Sobre Nós**, que apresenta de forma clara a essência da empresa, sua proposta de valor e os serviços oferecidos.
A tag <section> é utilizada para delimitar o bloco, enquanto <article> organiza o conteúdo em parágrafos estruturados.

Trecho 4:

```
<section id="themes" class="container">
      <h1>Temas</h1>
      <article class="boxs">
        <article class="box" data-theme="boy">
          <h2>Menino</h2>
          <figure>
            <button class="btn btn-left">
              <img src="assets/icons/arrow-left.svg"
alt="Seta para esquerda" />
            </button>
            <img src="assets/boy_theme.svg" alt="Tema de
menino" />
            <button class="btn btn-right">
              <img src="assets/icons/arrow-right.svg"
alt="Seta para direita" />
            </button>
          </figure>
          <h3>Tema de urso</h3>
        </article>

        <article class="box" data-theme="neutral">
          <h2>Neutro</h2>
          <figure>
            <button class="btn btn-left">
              <img src="assets/icons/arrow-left.svg"
alt="Seta para esquerda" />
            </button>
            <img src="assets/neutral_theme.svg" alt="Tema
neutro" />
            <button class="btn btn-right">
              <img src="assets/icons/arrow-right.svg"
alt="Seta para direita" />
            </button>
          </figure>
          <h3>Tema de urso</h3>
        </article>

        <article class="box" data-theme="girl">
          <h2>Menina</h2>
          <figure>
            <button class="btn btn-left">
              <img src="assets/icons/arrow-left.svg"
alt="Seta para esquerda" />
            </button>
            <img src="assets/girl_theme.svg" alt="Tema de
menina" />
            <button class="btn btn-right">
              <img src="assets/icons/arrow-right.svg"
alt="Seta para direita" />
            </button>
          </figure>
          <h3>Tema de princesa</h3>
        </article>
      </article>
    </section>
```

Nesse trecho, criamos a seção **Temas**, onde apresentamos as opções de kits de festas disponíveis (menino, neutro e menina).
Utilizamos a tag <section> para delimitar a área e <article> para estruturar cada tema de forma independente, aonde delimitamos cada article em um classname = “box”, aonde esta sendo feito o visual das box de cada article.
Dentro de cada <article>, organizamos o conteúdo com título, imagem ilustrativa e botões de navegação para explorar variações do tema.

Trecho 5:

```
<section id="contact" class="container">
  <h1>Contato</h1>
  <form action="" method="post">
    <!-- Campos do formulário -->
  </form>
</section>
```

Criamos a seção **Contato**, responsável pelo formulário de agendamento.
Usamos <section> para delimitar o bloco principal, <form> para o envio dos dados e <fieldset> para agrupar campos relacionados.

Trecho 6:

```
<fieldset class="full-width">
  <section class="form-field">
    <label for="nome">Nome</label>
    <input type="text" id="nome" name="nome" required />
  </section>
</fieldset>

<fieldset>
  <section class="form-field">
    <label for="telefone">Telefone</label>
    <input type="tel" id="telefone" name="telefone" required />
  </section>
  <section class="form-field">
    <label for="email">Email</label>
    <input type="email" id="email" name="email" required />
  </section>
</fieldset>
```

Primeira parte do formulário coleta **nome, telefone e e-mail**.
Cada campo está dentro de <section class="form-field"> para facilitar o estilo CSS.
Uso de type="tel" permite aplicar máscaras de input no futuro

Trecho 7:

```
<fieldset class="full-width">
  <section class="form-field">
    <label for="tema">Tema da Festa</label>
    <input type="text" id="tema" name="tema" required />
  </section>
</fieldset>

<fieldset class="full-width">
  <section class="form-field">
    <label for="kits">Kits disponíveis</label>
    <select id="kits" name="kits" required>
      <option value="">Selecione um kit</option>
      <option value="kit-compact">Kit Compact</option>
      <option value="kit-profissional">Kit Profissional</option>
      <option value="kit-vip">Kit VIP</option>
      <option value="personalizado">Kit Personalizado</option>
      </optgroup>
    </select>
  </section>
</fieldset>
```

Nessa parte, pedimos o **tema da festa** e o **kit desejado**.
Ira descer um menu suspenso, aonde a pessoa poderá escolher o tipo de kit que ela quer para o tema dito.

Trecho 8:

```
<fieldset>
  <section class="form-field">
    <label for="data">Datas</label>
    <input type="date" id="data" name="data" required />
  </section>
  <section class="form-field">
    <label for="horario">Horários</label>
    <input type="time" id="horario" name="horario" required />
  </section>
</fieldset>
```

Pedimos que você escolha um dia e um horário para a festa.
Nos campos de data e horário, é possível tanto digitar manualmente quanto selecionar pelo calendário e pelo seletor de horários fornecidos pelo navegador.

Trecho 9:

```
<footer id="footer" class="container footer">
      <section class="footer-content">
        <img src="assets/logo.png" alt="Logo Conto de Festas" class="footer-logo" />
        <p>© 2025 Conto de Festas - Todos os direitos reservados.</p>
      </section>
    </footer>

    <script src="script.js"></script>
  </body>
</html>
```

Esse código cria o **rodapé** de uma página HTML. Ele contém uma **seção** com a **logo** do site e um **texto de direitos autorais**. O `<footer>` tem classes e ID para estilização, e o `<script>` carrega um arquivo JavaScript externo (`script.js`) que adiciona funcionalidades à página.

CSS

Dentro do CSS temos essa estrutura

```
├── styles/
│   ├── index.css      # Chama todos os arquivos de CSS
│   ├── global.css      # Estilos globais e variáveis
│   ├── header.css      # Estilos do cabeçalho
│   ├── about.css       # Estilos da seção "Sobre"
│   ├── themes.css      # Estilos da seção "Temas"
│   ├── contact.css     # Estilos da seção "Contato"
│   ├── buttons.css     # Estilos dos botões
│   ├── footer.css      # Estilos do rodapé
│   ├── section.css     # Estilos gerais das seções
│   ├── utils.css       # Utilitários CSS
```

Index.css

Responsavel por declarar todas as funções de CSS em um código so

```
@import "global.css";
@import "header.css";
@import "about.css";
@import "themes.css";
@import "contact.css";
@import "section.css";
@import "buttons.css";
@import "utils.css";
@import "footer.css";
```

Estamos chamando todas os códigos css para que possam rodar dentro do HTML.
`<link rel="stylesheet" href="styles/index.css" />`

Global.css

Responsavel pela estilização global do site

```
* {
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}

:root {
  --fs-xl: 2rem;
  --fs-lg: 1.25rem;

  --ff-primary: "Inter", sans-serif;
  --ff-secondary: "Berkshire Swash", sans-serif;
  --ff-tertiary: "Josefin Sans", sans-serif;

  --bg-primary: linear-gradient(150deg, #ffffff, #722e43);
  --bg-secondary: #722e43;
  --box-color: rgba(255, 255, 255, 0.3);
  --btn-color: #b098a0;
}
```

Nesse trecho estamos removendo o padding e a margin padrao

 Define tamanhos de fonte **(--fs-xl, --fs-lg).**

 Define fontes **(--ff-primary, --ff-secondary, --ff-tertiary).**

 Define cores e fundos **(--bg-primary, --bg-secondary, --box-color, --btn-color).**

 Facilita alterações rápidas de estilo em toda a página.

```
body {
  background: var(--bg-primary);
  background-repeat: no-repeat;
  background-attachment: fixed;
  height: 100%;
}

.container {
  width: 100%;
  padding: 1rem;
  margin-inline: auto;
}

a {
  color: white;
  text-decoration: none;
```

- **Aplicamos um gradiente de cores no fundo da pagina**
- **Define uma altura da pagina**
- Cria containers **fluídos** com padding interno.
- Centraliza conteúdo horizontalmente.
- • Links são **brancos e em negrito**, sem sublinhado padrão.
- • Ao passar o mouse, o sublinhado aparece, indicando interatividade.

Header.css

Responsável pelo cabeçalho do index.html

```
header {
  position: relative;
  display: flex;
  align-items: center;
  justify-content: space-between;
  width: 100%;
  max-height: 80px;
  background-color: var(--bg-secondary);
  background-size: cover;
  color: white;
  font: var(--fs-lg) / 1 var(--ff-primary);
}

.header-image {
  position: absolute;
  top: 0;
  left: 0;
  margin: 0px;
  padding: 0;
  z-index: 1;
}

.nav-links {
  gap: 3rem;
  margin-right: 5rem;
}

header img {
  width: 70px;
  height: 70px;
}
```

```
header article {
    margin-left: 4rem;
    margin-left: 4rem;
}
```

Estamos estilizando o header. Container flexível que alinha os itens ao centro e os distribui com espaço entre eles. Ele ocupa toda a largura da páginacor de fundo definida pela variável --bg-secondary e texto em fonte média e cor branca.

A imagem de fundo é posicionada no canto superior esquerdo, atrás de outros elementos, sem margens. Os links de navegação têm espaçamento de 3 rem entre si e margem de 5 rem à direita. As imagens no header têm tamanho fixo de 70x70 pixels, e o elemento recebe margem à esquerda de 4 rem, mantendo o layout equilibrado. Esses estilos criam um header organizado e responsivo.

About.css

```
#about {
  display: flex;
  justify-content: center;
  margin-top: 2rem;
}

#about article {
  display: grid;
  margin-top: 3rem;
  background-color: var(--box-color);
  border-radius: 2.5rem;

  gap: 2rem;
  align-items: center;
  justify-items: center;

  max-width: 90rem;
  padding: 5rem 10rem;
  border: solid 2px var(--bg-secondary);
}

p {
  max-width: 50rem;
  font: var(--fs-lg) / 1.5 var(--ff-tertiary);
  font-weight: 300;
}
```

- O container principal usa Flexbox para centralizar o conteúdo e tem margem superior de 2 rem.
- O elemento <article> em about é um grid com margem superior de 3 rem, fundo semitransparente, bordas arredondadas, e borda sólida. Ele tem espaçamento interno de 5 rem vertical e 10 rem horizontal, com espaçamento entre itens de 2 rem.
- O parágrafo (p) tem largura máxima de 50 rem, fonte terciária, tamanho médio, espaçamento entre linhas de 1,5, e peso leve (font-weight: 300).

Themes.css

Responsavel pela parte de Temas no index.html

```
#themes {
  margin-top: 5rem;
}

#themes img {
  width: 18rem;
  height: 18rem;
}

.box {
  background-color: var(--box-color);
  border-radius: 2.5rem;
  border: solid 2px var(--bg-secondary);
}

.boxs {
  margin-top: 3rem;
  display: flex;
  flex-wrap: wrap;
  justify-content: center;
  gap: 1rem;
}

.boxs h2,
.boxs h3 {
  font-size: var(--fs-lg);
  font-family: var(--ff-secondary);
  color: black;
  text-align: center;
}

.box h2 {
  margin-top: 1rem;
}

.box h3 {
  margin-top: 1rem;
  margin-bottom: 3rem;
}
```

Sabendo que isso seria tudo para a parte de Temas no index.html

- Adiciona **espaçamento superior** de 5 rem, afastando a seção de elementos acima.
- Define tamanho fixo para **imagens da seção**, garantindo uniformidade visual.
- Cada **caixa** tem fundo semitransparente (`var(--box-color)`), **bordas arredondadas** e **borda sólida** com a cor secundária do site.
- Adiciona **margem superior** de 3 rem.

- Usa **Flexbox** para organizar as caixas, permitindo **quebra de linha automática** (`flex-wrap: wrap`) e **centralizando-as**.
- Mantém **espaçamento de 1 rem** entre as caixas.
- • Títulos dentro das caixas (`h2` e `h3`) usam **fonte secundária**, cor preta e estão centralizados.
- • O tamanho da fonte é médio (`var(--fs-lg)`).
- Ajusta **margens internas** para criar espaçamento consistente entre título, subtítulo e conteúdo dentro de cada caixa.

```
figure {
  position: relative;
  display: inline-block;
  padding: 1.5rem;
}

figure img {
  display: block;
}

figure button {
  position: absolute;
  top: 50%;
  transform: translateY(-50%);
  z-index: 10;
}

.btn-left {
  left: 0.5rem;
}

.btn-right {
  right: 0.5rem;
}
```

- Cada `<figure>` é **inline-block**, com **posição relativa**, permitindo posicionar elementos internos de forma absoluta (como botões).
- Adiciona **padding interno** de 1,5 rem.
- Garante que a **imagem dentro do figure** ocupe todo o bloco e não tenha espaços extras.
- • Botões dentro da figura são **posicionados verticalmente no centro** da imagem.
- • `z-index: 10` garante que fiquem **acima das imagens**.
- Define a posição horizontal dos botões **à esquerda ou à direita** da figura, com pequeno espaçamento das bordas.

Contact.css

Responsavel pela parte de Contatos no index.html

```
#contact {
  display: flex;
  justify-content: center;
  margin-top: 3rem;
  margin-bottom: 5rem;
}
```

- Centraliza horizontalmente o conteúdo da seção usando **Flexbox**.
- Adiciona **margem superior** e **inferior** para espaçamento em relação às outras seções.

```
 #contact form {
   background-color: var(--box-color);
   border-radius: 2.5rem;
   border: solid 2px var(--bg-secondary);
   padding: 3rem;
   width: 100%;
   max-width: 50rem;
   display: flex;
   flex-direction: column;
   gap: 1.5rem;
 }
```

- Fundo semitransparente (`var(--box-color)`) com **bordas arredondadas** e **borda sólida**.
- Padding interno de 3 rem e largura máxima de 50 rem.
- Layout **flexível em coluna** com espaçamento de 1,5 rem entre os campos.

```
fieldset {
  display: flex;
  gap: 1rem;
  border: none;
  padding: 0;
  margin: 0;
}

fieldset.full-width {
  flex-direction: column;
}

section.form-field {
  display: flex;
  flex-direction: column;
  flex: 1;
}
```

- fieldset organiza campos lado a lado (display: flex) com espaçamento de 1 rem.
- full-width permite que campos ocupem toda a largura em coluna.
- section.form-field organiza label e input/select/textarea em coluna.

```
section.form-field label {
  font-family: var(--ff-secondary);
  font-size: var(--fs-lg);
  color: black;
  margin-bottom: 0.5rem;
  font-weight: 400;
}
```

- Labels utilizam **fonte secundária**, tamanho médio, cor preta e peso normal, com pequena margem inferior.

```
section.form-field input,
section.form-field select,
section.form-field textarea {
  padding: 0.8rem 1rem;
  border: none;
  border-radius: 1.5rem;
  background-color: rgba(255, 255, 255, 0.8);
  font-family: var(--ff-tertiary);
  font-size: 1rem;
  color: #333;
  outline: none;
  transition: all 0.3s ease;
}
section.form-field input:focus,
section.form-field select:focus,
section.form-field textarea:focus {
  background-color: rgba(255, 255, 255, 0.95);
  box-shadow: 0 0 0 2px var(--bg-secondary);
}

section.form-field input::placeholder,
section.form-field textarea::placeholder {
  color: #666;
  font-style: italic;
}
```

- Campos têm padding interno, bordas arredondadas e fundo claro semitransparente.
- Mudança de estilo ao focar (:focus) com sombra e fundo mais claro.
- Placeholder estilizado em cor cinza e itálico.

```
section.form-field select {
  cursor: pointer;
  appearance: none;
  background-image: url("data:image/svg+xml;charset=UTF-8,%3csvg
xmlns='http://www.w3.org/2000/svg' viewBox='0 0 24 24' fill='none'
stroke='currentColor' stroke-width='2' stroke-linecap='round' stroke-
linejoin='round'%3e%3cpolyline points='6,9 12,15
18,9'%3e%3c/polyline%3e%3c/svg%3e");
  background-repeat: no-repeat;
  background-position: right 1rem center;
  background-size: 1rem;
  padding-right: 3rem;
}
```

- Cursor indica **interatividade**, remove estilo padrão e adiciona **seta customizada** no lado direito

```
section.form-field textarea {
  resize: vertical;
  min-height: 4rem;
  border-radius: 1rem;
}
```

- Permite **redimensionamento vertical**, define altura mínima e bordas arredondadas.

```
.submit-btn {
  background-color: var(--btn-color);
  color: white;
  border: none;
  border-radius: 1.5rem;
  padding: 1rem 2rem;
  font-family: var(--ff-secondary);
  font-size: var(--fs-lg);
  cursor: pointer;
  transition: all 0.3s ease;
  align-self: flex-start;
  margin-top: 1rem;
}

.submit-btn:hover {
  background-color: var(--bg-secondary);
  transform: translateY(-2px);
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);
}

.submit-btn:active {
  transform: translateY(0);
}
```

- Botão com **cor de fundo**, bordas arredondadas, fonte personalizada e **cursor pointer**.
- Efeitos de **hover** e **click** para animação de movimento e sombra, deixando a interação mais intuitiva.

```
.kit-detalhes {
  display: none;
  margin-top: 1rem;
  padding: 1rem 1.5rem;
  background: rgba(255, 255, 255, 0.8);
  border-radius: 1rem;
  border: 2px solid var(--bg-secondary);
}

.kit-detalhes h4 {
  margin-bottom: 0.5rem;
  font-family: var(--ff-secondary);
  color: #333;
}

.kit-detalhes p,
.kit-detalhes li {
  font-family: var(--ff-tertiary);
  font-size: 1rem;
  color: #333;
  line-height: 1.5;
}

.kit-detalhes ul {
  padding-left: 1.5rem;
  list-style-type: disc;
}
```

- Inicialmente ocultos (`display: none`) e aparecem apenas quando um kit é selecionado.
- Estilo de **caixa com padding e borda**, títulos e texto com fontes diferentes para hierarquia visual.
- Listas com **marcadores e espaçamento** apropriado.

```
.modal-overlay {
  position: fixed;
  top: 0;
  left: 0;
  width: 100%;
  height: 100%;
  background-color: rgba(0, 0, 0, 0.8);
  backdrop-filter: blur(5px);
  z-index: 9998;
  opacity: 0;
  transition: opacity 0.3s ease;
}

.modal-overlay.show {
  opacity: 1;
}
```

- Fundo escuro semitransparente cobrindo toda a tela.
- show controla a visibilidade com transição suave de opacidade.

```
.modal-3d {
  position: fixed;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%) perspective(1000px) rotateX(-90deg);
  z-index: 9999;
  opacity: 0;
  transition: all 0.4s cubic-bezier(0.68, -0.55, 0.265, 1.55);
}

.modal-3d.show {
  transform: translate(-50%, -50%) perspective(1000px) rotateX(0deg);
  opacity: 1;
}
```

- Modal centralizado, com efeito 3D e rotação no eixo X para animação de entrada.
- show ativa a transição para a posição final e torna visível.

```
.modal-content {
  background: linear-gradient(
    135deg,
    rgba(255, 192, 203, 0.95),
    rgba(221, 160, 221, 0.95)
  );
  border-radius: 2rem;
  border: solid 3px var(--bg-secondary);
  padding: 3rem;
  max-width: 500px;
  width: 90vw;
  text-align: center;
  box-shadow: 0 20px 40px rgba(0, 0, 0, 0.3),
    0 0 0 1px rgba(255, 255, 255, 0.2) inset;
  transform-style: preserve-3d;
  position: relative;
}

.modal-content::before {
  content: "";
  position: absolute;
  top: -10px;
  left: -10px;
  right: -10px;
  bottom: -10px;
  background: linear-gradient(
    45deg,
    transparent,
    rgba(255, 255, 255, 0.1),
    transparent
  );
```

- Conteúdo do modal com gradiente de fundo, bordas arredondadas e sombras para efeito 3D.
- Pseudo-elemento ::before cria um reflexo sutil ao redor do modal.

```
.success-icon {
  width: 80px;
  height: 80px;
  background: linear-gradient(135deg, #4caf50, #45a049);
  border-radius: 50%;
  display: flex;
  align-items: center;
  justify-content: center;
  font-size: 3rem;
  color: white;
  margin: 0 auto 2rem;
  box-shadow: 0 10px 20px rgba(76, 175, 80, 0.3);
  animation: bounceIn 0.6s ease-out 0.2s both;
}
```

- Ícone de sucesso circular, com gradiente verde, centralizado e com **animação de entrada (`bounceIn`).**

```
.modal-content h2 {
  font-family: var(--ff-secondary);
  font-size: 2.2rem;
  color: #333;
  margin-bottom: 1.5rem;
  font-weight: 600;
  text-shadow: 0 2px 4px rgba(0, 0, 0, 0.1);
}

.modal-content p {
  font-family: var(--ff-secondary);
  font-size: 1.1rem;
  color: #333;
  line-height: 1.6;
  margin-bottom: 2rem;
}
```

- Títulos e textos usam fontes secundárias, cores escuras e espaçamento apropriado.
- Textos com sombra leve para destaque visual.

```
.close-modal-btn {
  background: linear-gradient(135deg, var(--btn-color), var(--bg-secondary));
  color: white;
  border: none;
  border-radius: 2rem;
  padding: 1rem 2.5rem;
  font-family: var(--ff-secondary);
  font-size: 1.1rem;
  font-weight: 600;
  cursor: pointer;
  transition: all 0.3s ease;
  box-shadow: 0 8px 16px rgba(0, 0, 0, 0.2);
  transform: translateZ(20px);
}

.close-modal-btn:hover {
  transform: translateZ(30px) translateY(-2px);
  box-shadow: 0 12px 24px rgba(0, 0, 0, 0.3);
}

.close-modal-btn:active {
  transform: translateZ(20px) translateY(0);
}
```

- Botão com **gradiente de cores**, bordas arredondadas, sombra e efeitos de hover/click para interação 3D.

```
@keyframes bounceIn {
  0% {
    transform: scale(0.3) rotateY(180deg);
    opacity: 0;
  }
  50% {
    transform: scale(1.05) rotateY(90deg);
  }
  70% {
    transform: scale(0.9) rotateY(45deg);
  }
  100% {
    transform: scale(1) rotateY(0deg);
    opacity: 1;
  }
}
```

- Animação que faz elementos entrar com efeito de crescimento e rotação, ideal para ícones ou modais.

Button.css

Responsável pelos botões no index.html

```
.btn {
  background-color: var(--btn-color);
  border-radius: 50%;
  width: 2.5rem;
  height: 2.5rem;
  border: none;
  cursor: pointer;
  font-size: 1.8rem;
  font-weight: 300;
  color: white;
  display: flex;
  align-items: center;
  justify-content: center;
}

.btn img {
  width: 2.5rem !important;
  height: 2.5rem !important;
}
```

Funcao global para os botos do index.html

- Vai definir a cor de fundo do botão
- Tamanho do botão
- Mostrar que o botão é clicável ao passar o mouse em cima.
- Centraliza o texto do botão no centro do bloco
- Para imagens dentro de um botão, ele ira manter a imagem do mesmo tamanho do botão

Footer.css

Responsável pelo rodapé do index.html

```
footer.footer {
  background-color: var(--bg-secondary);
  color: white;
  padding: 2rem 1rem;
  text-align: center;
}
```

- Define a cor de fundo do rodapé
- Centraliza o texto

```
.footer-content {
  display: flex;
  align-items: center; /* logo e texto no mesmo eixo */
  justify-content: center;
  gap: 1rem; /* espaço entre logo e texto */
  margin-bottom: 1rem;
}
```

- Usa flexbox para colocar a logo e o texto lado a lado.
- Garante que ambos fiquem centralizados no meio da tela.
- Cria um espaço entre eles (gap) para não ficarem grudados.
- Deixa uma margem embaixo para separar dos outros elementos.

```
.footer-logo {
  width: 100px; /* 🔷 aumentei o tamanho */
  height: 100px;
}
```

Define o tamanho da logo que aparece no rodapé.

- Força a logo a ter 100px x 100px, para que fique grande e visível.

```
.footer-content p {
  font-family: var(--ff-secondary);
  font-size: 1.2rem;
}
```

- Troca a fonte para a secundária (mais decorativa).
- Aumenta um pouco o tamanho da letra (1.2rem).

```
.footer-links {
  display: flex;
  justify-content: center;
  gap: 2rem;
  margin-top: 0.5rem;
}

.footer-links a {
  color: white;
  font-family: var(--ff-secondary);
  font-size: 1.1rem;
  text-decoration: none;
  transition: 0.3s;
}

.footer-links a:hover {
  text-decoration: underline;
  color: #ffd6e0;
}
```

- Coloca os links lado a lado (com flexbox).
- Centraliza o grupo de links no meio da página.
- Dá um espaço entre cada link.

- Remove o sublinhado padrão dos links.
- Deixa uma transição suave para quando passar o mouse

Section.css

Responsável por todos os <h1> na section

```
section h1 {
  display: flex;
  align-items: center;
  justify-content: center;

  font: var(--fs-xl) / 1 var(--ff-secondary);
}
```

- Títulos fiquem centralizados
- Ganham estilo de fonte maior e mais bonita

Utils.css

```
.flex {
  display: flex;
}
```

- Transforma o elemento em um container flexível

JavaScript

```
document.querySelectorAll(".box").forEach((box) => {
  const themes = {
    boy: [
      { img: "assets/boy_theme.svg", title: "Tema de urso" },
      { img: "assets/boy_theme2.png", title: "Tema de carros" },
      { img: "assets/boy_theme3.png", title: "Tema de dinossauro" },
    ],
    neutral: [
      { img: "assets/neutral_theme.svg", title: "Tema de urso" },
      { img: "assets/neutral_theme2.png", title: "Tema de natureza" },
      { img: "assets/neutral_theme3.png", title: "Tema de animais" },
    ],
    girl: [
      { img: "assets/girl_theme.svg", title: "Tema de princesa" },
      { img: "assets/girl_theme2.png", title: "Tema de flores" },
      { img: "assets/girl_theme3.png", title: "Tema de unicórnio" },
    ],
  };

  const themeType = box.dataset.theme;
  const themeData = themes[themeType];
  let currentIndex = 0;

  const imgElement = box.querySelector("figure img:not(.btn img)");
  const titleElement = box.querySelector("h3");
```

```
box.querySelector(".btn-left").addEventListener("click", () => {
   currentIndex = (currentIndex - 1 + themeData.length) % themeData.length;
   updateTheme();
  });

  box.querySelector(".btn-right").addEventListener("click", () => {
   currentIndex = (currentIndex + 1) % themeData.length;
   updateTheme();
  });

  function updateTheme() {
   imgElement.src = themeData[currentIndex].img;
   titleElement.textContent = themeData[currentIndex].title;
  }
```

- Cria uma sequencia de temas para cada `.box`.
- Os temas (menino, menina, neutro) têm imagens e títulos armazenados em arrays.
- O dataset.theme da div indica se a caixa é "boy", "girl" ou "neutral".
- As setas (`.btn-left` e `.btn-right`) mudam o índice atual, trocando imagem e título.
- A função `updateTheme()` atualiza o conteúdo exibido.

```
document.addEventListener("DOMContentLoaded", function () {
  const form = document.querySelector("form");

  form.addEventListener("submit", function (e) {
   e.preventDefault(); // Previne o envio padrão do formulário
```

- Espera o carregamento completo da página.
- Captura o `<form>` e, no evento de submit, bloqueia o envio normal (`e.preventDefault()`), para que a página não recarregue.

```
const overlay = document.createElement("div");
   overlay.className = "modal-overlay";

   // Cria o modal 3D
   const modal = document.createElement("div");
   modal.className = "modal-3d";
   modal.innerHTML = `
     <div class="modal-content">
       <div class="success-icon">✓</div>
       <h2>Agendamento confirmado com sucesso!</h2>
       <p>Em breve, nossa equipe entrará em contato para conhecer mais
detalhes sobre sua festa. Agradecemos por escolher nossos serviços e estamos à
disposição para tornar seu evento ainda mais especial.</p>
       <button class="close-modal-btn">Fechar</button>
     </div>
   `;

   // Adiciona o overlay e modal ao body
   document.body.appendChild(overlay);
   document.body.appendChild(modal);

   // Adiciona classes para animação
   setTimeout(() => {
     overlay.classList.add("show");
     modal.classList.add("show");
   }, 10);
```

- Cria dinamicamente o fundo escuro (overlay) e o modal 3D.
- O modal exibe a mensagem de sucesso + botão de fechar.
- Insere os dois no body.
- Usa setTimeout para dar um efeito de animação (classes .show ativam CSS).

```
function closeModal() {
     overlay.classList.remove("show");
     modal.classList.remove("show");

     setTimeout(() => {
       document.body.removeChild(overlay);
       document.body.removeChild(modal);
     }, 300);
   }

   // Event listeners para fechar
   modal
     .querySelector(".close-modal-btn")
     .addEventListener("click", closeModal);
   overlay.addEventListener("click", closeModal);
 });
```

- Define uma função closeModal() que remove as classes de animação e, após 300ms, deleta os elementos da tela.
- Fecha o modal tanto ao clicar no botão "Fechar" quanto clicando no overlay escuro.

```
const kitsSelect = document.getElementById("kits");
 const detalhesDiv = document.getElementById("detalhes-kit");

 kitsSelect.addEventListener("change", function () {
   let conteudo = "";

   if (this.value === "kit-compact") {
     conteudo = `
       <h4>Kit Compacto inclui:</h4>
       <ul>
         <li>3 mesas cilíndricas</li>
         <li>Painel circular com tecido</li>
         <li>Bolo personalizado</li>
         <li>4 bandejas decorativas</li>
         <li>3 bonecos temáticos</li>
         <li>2 garrafas decorativas</li>
         <li>1 kit de flor</li>
       </ul>
     `;
   } else if (this.value === "personalizado") {
     conteudo = `
       <h4>Kit Personalizado:</h4>
       <p>Montamos conforme sua necessidade. Entre em contato para definir os
itens.</p>
     `;
}
```

```
} else if (this.value === "kit-profissional") {
     conteudo = `
       <h4>Kit Profissional inclui:</h4>
       <ul>
         <li>3 mesas cilíndricas com tecido</li>
         <li>Painel circular com tecido</li>
         <li> Painel retangular com tecido</li>
         <li> Bolo personalizado</li>
         <li> 4 bandejas decorativas</li>
         <li> 3 bonecos temáticos</li>
         <li> 2 garrafas decorativas</li>
         <li> 1 kit de flor</li>
         <li> 1 apoio de chão</li>
       </ul>
     `;
   } else if (this.value === "kit-vip") {
     conteudo = `
       <h4>Kit VIP inclui:</h4>
       <ul>
         <li>3 mesas cilíndricas com tecido</li>
         <li>Painel circular com tecido</li>
         <li> 2 Paineis Romanos</li>
         <li> Bolo personalizado</li>
         <li> 4 bandejas decorativas</li>
         <li> 3 bonecos temáticos</li>
         <li> 2 garrafas decorativas</li>
         <li> 1 kit de flor</li>
         <li> 1 apoio de chão</li>
       </ul>
     `;
   }

   if (conteudo) {
     detalhesDiv.innerHTML = conteudo;
     detalhesDiv.style.display = "block";
   } else {
     detalhesDiv.innerHTML = "";
     detalhesDiv.style.display = "none";
   }
  });
});
```

- Captura o select dos kits e a div onde os detalhes aparecem.
- Sempre que o usuário mudar a opção (change), o script insere os detalhes correspondentes em HTML.
- Se não houver valor selecionado, esconde a div.

Como foi feito o código

1. Projeto inteiro foi feito em um editor de código (Visual Studio Code)
2. Foi instalado uma extensão chamada Live Server, possibilitando abrir o index.html
3. Separamos em pastas:
    a. Pasta assets para as imagens no site
    b. Pasta styles aonde esta armazenada toda estrutura de CSS
4. Arquivo index.html sendo o arquivo principal para rodar o código
5. README.md para mostrar mais sobre o código

Rodar o Código

1. Va para o editor
2. Abra o Conto de festas
3. De cd src
4. Digite no terminal
    - open index.html