1) objetivo

entender como quebrar uma página em partes menores chamadas componentes, perceber a hierarquia entre essas partes, identificar quais dados mudam na tela e explicar como isso seria reativo em um framework moderno como React ou Vue

2) escolha da interface

use a página principal index.html
ela contém cabeçalho, seção de início, lista de produtos, seção sobre, seção de contato e rodapé

3) observe a página

O componente que seria atualizado é a **Grid de Produtos**, pois ela é responsável por exibir todos os *Cards de Produto* na tela.
Quando um novo item é adicionado ou um produto é removido, a grid detecta a alteração e atualiza apenas os elementos afetados.
Isso evita o recarregamento completo da página, deixando a experiência do usuário mais fluida e moderna.
Em um framework reativo, como o React ou Vue.js, essa atualização aconteceria automaticamente por meio do gerenciamento de estado, garantindo desempenho e eficiência.

4) liste os componentes que encontrou

preencha uma tabela simples

nome do componente	**função**	**dados exibidos**	**pode ser reutilizado onde**
header	mostrar logo e menu de navegação fixo	imagem do logo e links de navegação	sim em todas as páginas
seção início	destacar a mensagem principal com imagem de fundo	título principal	sim como banner em outras páginas
grid de produtos	agrupar os cards em grade responsiva	coleção de produtos	sim em páginas de listagem
card de produto	exibir um item com imagem título preço e link	imagem nome preço link	sim várias vezes na mesma página

nome do componente	função	dados exibidos	pode ser reutilizado onde
seção sobre	texto institucional e lista com tópicos	parágrafos e lista	estrutura pode ser reaproveitada
seção contato	canais oficiais de comunicação	whatsapp	sim em página de fale conosco
footer	direitos autorais	texto de copyright	sim em todas as páginas

5) crie um diagrama hierárquico

desenhe a relação pai e filho dos componentes

página principal
|__ header
| |__ logo
| |__ menu
|__ seção início
| |__ título
|__ seção de produtos
| |__ grid de produtos
| |__ card de produto 1
| |__ card de produto 2
| |__ card de produto 3
|__ seção sobre
| |__ texto
| |__ lista
|__ seção contato
| |__ canais
|__ footer

6) identifique o estado da interface

pense no que pode mudar e o que precisa ser atualizado quando muda

dado estado	onde aparece	o que muda na interface quando muda
lista de produtos	grid de produtos na página principal	aparecem novos cards ou alguns somem

dado estado	onde aparece	o que muda na interface quando muda
imagem atual do carrossel	páginas individuais de produto	a foto mostrada troca quando o usuário clica nas setas
status sticky do header	topo do site	o estilo do cabeçalho muda ao rolar a página cor e sombra

7) analise a reatividade

A parte mais adequada para ser reconstruída com um framework moderno como React ou Vue.js é a **seção de produtos**.
Essa parte do site é composta por vários *Cards de Produto* que utilizam dados dinâmicos e se repetem na interface.
Com o uso de estados, seria possível atualizar apenas os cards necessários, sem recarregar toda a página, tornando a aplicação mais rápida e eficiente.
Além disso, a hierarquia entre componentes ficaria mais clara e modular, facilitando futuras manutenções, adições de produtos e até integração com bancos de dados.
Isso demonstra como o uso da **hierarquia, do estado e da reatividade** torna o desenvolvimento mais organizado e a interface mais interativa.

8) conclusão

com base na análise de componentes e estados a parte que mais se beneficia de um framework reativo é a seção de produtos. ela é formada por vários cards que recebem dados dinâmicos como imagem nome e preço. ao usar React ou Vue a lista de produtos vira um estado e cada card pode ser renderizado de forma independente. quando um item é adicionado ou alterado apenas a grid e os cards impactados são atualizados. isso melhora desempenho organização da hierarquia e facilita a manutenção do código