Esquema del flujo completo del código de tartas

```
[Clase abstracta Tarta]
   - Propiedades: sabor, imagen
   - Métodos abstractos: hornear(), ingredientesEspecificos()
   - Método concreto: ingredientes() → combina base + específicos
   - Getter: nombre → devuelve "Tarta de {sabor}"
           |
           v
[Clases concretas]  ---> extienden Tarta con "extends"
   - TartaChocolate
   - TartaFresa
   - TartaQueso
   - TartaLimon
   - TartaZanahoria
   Cada clase:
     - Llama a super(sabor, imagen)
     - Implementa hornear()
     - Define ingredientesEspecificos()
           |
           v
[Array tartas inicial]  ---> comienza vacío
           |
           v
[HTML inicial]
   - <select id="tipo"> → opciones de tarta (Chocolate, Fresa,
Queso, Limón)
   - Botones: "Añadir" y "Hornear todas"
   - <div id="lista"> → contenedor para las tartas
   - <ul id="resultados"> → lista de mensajes de horneado
           |
           v
[Usuario elige tipo de tarta en el <select>]
           |
           v
[Pulsa "Añadir"]
   - JS lee el valor seleccionado
   - llama crearTarta(tipo) → devuelve instancia de la clase
concreta
   - tartas.push(nueva) → añade al array
```

```
   - renderLista() → genera HTML dinámico
           |
           v
[renderLista()]
   - Recorre cada tarta en el array
   - Crea <article class="card"> con imagen, nombre, ingredientes,
botón Hornear
   - Asigna evento a cada botón para llamar hornear()
           |
           v
[Usuario pulsa "Hornear" de una tarta]
   - JS llama tartas[idx].hornear()
   - pushResultado() → añade mensaje a <ul id="resultados">
           |
           v
[Usuario pulsa "Hornear todas"]
   - JS recorre todas las tartas y llama hornear() de cada una
   - pushResultado() → muestra todos los mensajes en la lista
```

Claves:

1. **Clase abstracta → Clases concretas**: polimorfismo y herencia.
2. **Array tartas**: almacena todas las instancias que se crean.
3. **renderLista()**: transforma cada instancia en un `<article>` visible en la web.
4. **Botones Hornear**: llaman al método específico de cada clase, mostrando cómo un mismo método (`hornear()`) se comporta distinto según la clase.
5. **HTML inicial**: permite comprobar el estado inicial y cómo se añaden dinámicamente elementos.