

# Trabalho de Pesquisa



**Disciplina:** Projeto e Desenvolvimento de Interfaces WEB

**Autor:** Maurício Correa

## 1. Qual é a sua biblioteca/framework escolhido? Quem produziu e qual é a sua principal referência?

### Angular (com TypeScript)

- **O que é:** Um framework de desenvolvimento de aplicações web (SPA - Single Page Applications) baseado em componentes e escrito em TypeScript.
- **Produtor:** Google (e comunidade open-source).
- **Referência Oficial:** [angular.dev](https://angular.dev) (nova documentação) ou [angular.io](https://angular.io).

### Tailwind CSS

- **O que é:** Um framework CSS *utility-first* (baseado em utilitários) que permite construir designs diretamente no markup sem sair do HTML.
- **Produtor:** Adam Wathan e equipe (Tailwind Labs).
- **Referência Oficial:** [tailwindcss.com](https://tailwindcss.com).

### Lucide Icons

- **O que é:** Uma biblioteca de ícones open-source, leve e consistente (um fork comunitário do extinto Feather Icons).
- **Produtor:** Comunidade Open Source (liderada por Eric Fennis).
- **Referência Oficial:** [lucide.dev](https://lucide.dev).

## 2. Qual é a sua finalidade de uso em IHC-UI-UX?

### **Angular (Interatividade e Feedback):**

- **IHC:** Garante o gerenciamento de estado complexo. Quando o usuário clica em algo, o Angular atualiza a interface instantaneamente (Reatividade), fornecendo feedback imediato, o que é crucial para a heurística de "Visibilidade do Status do Sistema".
- **UX:** Permite a criação de SPAs, onde a página não recarrega inteiramente, proporcionando uma experiência fluida similar a aplicativos nativos.

### **Tailwind CSS (Consistência Visual):**

- **UI:** Mantém o design consistente através de *tokens* pré-definidos (cores, espaçamentos). Isso evita "magic numbers" (valores aleatórios) no CSS.
- **UX:** Melhora a acessibilidade e a hierarquia visual, permitindo ajustes rápidos de layout para diferentes tamanhos de tela (Responsividade) diretamente nas classes.

### **Lucide Icons (Semiótica e Cognição):**

- **IHC:** Ícones atuam como metáforas visuais, reduzindo a carga cognitiva do usuário. O Lucide é conhecido por traços limpos e legíveis.
- **UX:** Ajuda na navegação e no reconhecimento de ações (ex: ícone de lixeira para deletar) em vez de forçar o usuário a ler textos (Reconhecimento em vez de memorização).

---

## **3. Como é o carregamento no código-fonte?**

Em um ambiente moderno de desenvolvimento (build process), o carregamento não é feito apenas com tags `<script>` no HTML, mas sim via importação de módulos e transpilação.

### **Angular (Bootstrapping): TypeScript**

O ponto de entrada é geralmente o arquivo main.ts, que inicializa o módulo raiz ou o componente raiz (em aplicações standalone).

```
// main.ts
import { bootstrapApplication } from '@angular/platform-browser';
import { AppComponent } from './app/app.component';

bootstrapApplication(AppComponent, appConfig)
  .catch((err) => console.error(err));
```

### Tailwind CSS (Diretivas): CSS

O Tailwind é injetado durante o processo de build. No arquivo de estilos global (styles.css), carregam-se as camadas:

```
/* styles.css */
@tailwind base;
@tailwind components;
@tailwind utilities;
```

### Lucide (Importação Modular): TypeScript

Para evitar carregar todos os ícones (peso desnecessário), carrega-se apenas o necessário dentro do componente Angular ou utiliza-se uma biblioteca wrapper como lucide-angular.

```
// No componente Angular
import { LucideAngularModule, Home, User } from 'lucide-angular';
```

## 4. Quais são os principais componentes, tais como: tags, atributos, propriedades, classes, instâncias...?

### Angular

O Angular é baseado em **Classes TypeScript** decoradas com metadados. Ele define *o que* aparece na tela e *como* se comporta.

**Principal Componente:** O `@Component`.

- É o bloco de construção básico.

- Possui um **Seletor** (tag HTML personalizada), um **Template** (HTML) e **Estilos** (CSS).

### Sintaxe de Template (Principais Recursos):

- **Interpolação** `{}{}`: Exibe valores dinâmicos do código na tela.
- **Property Binding** `[]`: Altera atributos HTML dinamicamente (ex: desabilitar um botão).
- **Event Binding** `()`: Captura ações do usuário (ex: cliques).
- **Diretivas Estruturais:** `ngIf` (condicional) e `@for` (loops - nova sintaxe do Angular 17+).

---

## Tailwind CSS

O Tailwind não possui "componentes" no sentido tradicional (como botões prontos). Seus componentes são **Classes Utilitárias** de baixo nível aplicadas diretamente no HTML.

**Principal Elemento:** O atributo `class`.

### Sintaxe de Utilitários (Categorias):

- **Espaçamento:** `p-4` (padding), `m-2` (margin), `gap-2` (espaço entre itens).
- **Flexbox/Grid:** `flex`, `grid`, `items-center`, `justify-between`.
- **Cores e Tipografia:** `bg-blue-500` (fundo), `text-white`, `font-bold`, `rounded-lg` (bordas).

### Modificadores (Prefixos):

- **Estados:** `hover:bg-blue-700` (muda cor ao passar o mouse).
- **Responsividade:** `md:flex` (aplica flexbox apenas em telas médias para cima).

---

## Lucide Icons (A Semiótica Visual)

O Lucide fornece **Componentes SVG** prontos. Em vez de escrever código `<svg>` complexo, você usa tags simples que representam o ícone.

**Principal Componente:** A tag do ícone (ex: `< lucide-icon >` ou a tag direta do ícone dependendo da importação).

### Propriedades (Atributos Configuráveis):

- `name` ou `img`: Define qual ícone será renderizado (ex: 'home', 'user').

- `size` : Tamanho em pixels (ex: `24`).
  - `color` : Cor da linha (aceita hex, rgb ou nomes de cores).
  - `strokeWidth` : Espessura do traço (padrão geralmente é 2).
- 

## 5. Exemplo desenvolvido

Como rodar o projeto:

### Opção 1: Sem instalar dependências:

1. Acesse o StackBlitz (IDE Online): [clicando aqui](#).
2. Aguarde um tempo (até 3 min), o StackBlitz irá compilar o projeto com todas as dependências, deixando tudo pronto pra navegar pelo código e pelo executável.

### Opção 2: Instalando dependências e rodando localmente:

1. Baixe o arquivo .zip [clicando aqui](#).
2. Extraia na sua máquina.
3. Abra a pasta em uma IDE ou no terminal.
4. Digite no terminal: `npm install`
  - a. Certifique-se de ter o Node.js instalado na sua máquina (se não tiver, basta [clicar aqui](#)).
5. Digite no terminal: `npm run start`
6. Abra a porta local informada, geralmente será <http://localhost:4200/>.