Documentação do Projeto Angular - Golden Raspberry Awards

**Introdução**

Este documento fornece uma visão detalhada da estrutura do projeto Angular Golden Raspberry Awards. Ele inclui informações sobre os principais arquivos e pastas, como realizar manutenção e possíveis expansões futuras. Este guia é destinado a desenvolvedores e equipes responsáveis pela evolução do software.

**Estrutura do Projeto**

Abaixo está a descrição das principais pastas e arquivos do projeto, com detalhes sobre suas funções e orientações para manutenção.

**Como testar/rodar o Programa no desktop ou computador?**  
Para testar e rodar o programa no seu computador, você precisa de algumas ferramentas essenciais configuradas no ambiente de desenvolvimento. Abaixo está uma lista detalhada do que deve estar instalado:

Baixe o programa descompacte e acesse o diretório.

**1. Node.js e npm**

* **Por quê?** O Angular utiliza o Node.js para gerenciar dependências e compilar o código.
* **Instalar:**
  + [Download Node.js](https://nodejs.org)
  + Durante a instalação, o npm (Node Package Manager) será incluído automaticamente.
* **Verificar instalação:**

bash

Copiar código

node -v

npm -v

Isso deve retornar as versões do Node.js e npm instalados.

**2. Angular CLI**

* **Por quê?** Ferramenta oficial para criar, construir, rodar e testar aplicações Angular.
* **Instalar Angular CLI globalmente:**

bash

Copiar código

npm install -g @angular/cli

* **Verificar instalação:**

bash

Copiar código

ng version

**3. Editor de Código**

* **Sugestão:** [Visual Studio Code (VS Code)](https://code.visualstudio.com/)
* **Extensões recomendadas:**
  + Angular Language Service (para autocompletar e ajuda com Angular).
  + Prettier (para formatação de código).
  + ESLint (para garantir boas práticas de codificação).

**4. Dependências do Projeto**

Após clonar ou baixar o projeto, você precisa instalar as dependências:

* **Comando para instalar dependências:**
  + Navegue até a pasta do projeto no terminal.

bash

Copiar código

cd <nome\_da\_pasta>

npm install

Isso instalará todas as bibliotecas e pacotes definidos no package.json.

**5. Executar o Programa**

Depois de configurar as dependências, você pode rodar o servidor de desenvolvimento Angular.

* **Comando para rodar o servidor:**

bash

Copiar código

ng serve

* **Acessar no navegador:**
  + Abra o navegador e acesse: http://localhost:4200

**6. API Externa**

* O programa utiliza uma API externa para buscar dados dos filmes. Certifique-se de que a URL da API configurada no código esteja acessível:

bash

Copiar código

https://challenge.outsera.tech/api/movies

* + Verifique no arquivo movies.service.ts a URL base para garantir que esteja correta.

**7. Browser Moderno**

* Certifique-se de que está usando um navegador moderno e atualizado, como:
  + Google Chrome
  + Microsoft Edge
  + Mozilla Firefox

**8. Git (Opcional, para Clonar o Projeto)**

* **Por quê?** Necessário se você estiver clonando o projeto de um repositório.
* **Instalar:** [Git](https://git-scm.com/)
* **Comando para clonar o projeto:**

bash

Copiar código

git clone <URL\_DO\_REPOSITORIO>

Se você seguir todos os passos, deverá conseguir rodar e testar o programa sem problemas. Caso algo não funcione ou tenha dúvidas, é só avisar!

**Para testar no GitHub:**

Para rodar o projeto no GitHub (ou seja, hospedá-lo e disponibilizá-lo para acesso público via GitHub Pages), você pode seguir os passos abaixo. Este processo é para hospedar projetos Angular no GitHub Pages:

**Passo 1: Configurar Repositório no GitHub**

1. **Criar um Repositório no GitHub:**
   * Acesse [GitHub](https://github.com) e crie um novo repositório.
   * Exemplo:
     + Nome do repositório: meu-projeto-angular.
   * Não é necessário adicionar nenhum arquivo inicial, como README.md, caso você já tenha o projeto localmente.
2. **Conectar o projeto Angular ao GitHub:**
   * Navegue até a pasta do projeto no terminal e execute os comandos:

bash

Copiar código

git init

git remote add origin https://github.com/SEU\_USUARIO/meu-projeto-angular.git

git branch -M main

git add .

git commit -m "Primeiro commit"

git push -u origin main

**Passo 2: Instalar a Dependência para GitHub Pages**

1. **Instalar o pacote angular-cli-ghpages:**
   * Este pacote ajuda a publicar aplicações Angular no GitHub Pages.
   * Execute o seguinte comando no terminal:

bash

Copiar código

npm install -g angular-cli-ghpages

1. **Construir o projeto Angular para produção:**
   * Antes de publicar, você precisa gerar os arquivos estáticos do projeto:

bash

Copiar código

ng build --output-path=dist --base-href "https://SEU\_USUARIO.github.io/NOME\_DO\_REPOSITORIO/"

* + - Substitua SEU\_USUARIO pelo seu nome de usuário do GitHub.
    - Substitua NOME\_DO\_REPOSITORIO pelo nome do seu repositório.

1. **Publicar no GitHub Pages:**
   * Após construir os arquivos estáticos, publique no GitHub Pages com:

bash

Copiar código

npx angular-cli-ghpages --dir=dist

**Passo 3: Configurar GitHub Pages no Repositório**

1. **Acesse o repositório no GitHub:**
   * Vá para as configurações do repositório (Settings).
2. **Configurar GitHub Pages:**
   * No menu lateral, clique em **Pages**.
   * Na seção "Source", selecione o branch gh-pages e salve.
3. **Acessar a aplicação:**
   * O GitHub Pages gerará um link, geralmente no formato:

plaintext

Copiar código

https://SEU\_USUARIO.github.io/NOME\_DO\_REPOSITORIO/

**Passo 4: Atualizar o Projeto**

* Sempre que você fizer mudanças no projeto, deve reconstruí-lo e publicá-lo novamente:

bash

Copiar código

ng build --output-path=dist --base-href "https://SEU\_USUARIO.github.io/NOME\_DO\_REPOSITORIO/"

npx angular-cli-ghpages --dir=dist

**Restrições do GitHub Pages**

* GitHub Pages é voltado para sites estáticos.
* APIs que o projeto consome devem estar acessíveis publicamente, pois o servidor do GitHub Pages não suporta backends dinâmicos.

Se você seguir esses passos, seu projeto Angular será acessível no GitHub Pages. Caso tenha dúvidas ou encontre problemas, é só avisar!

**Pastas e Arquivos**

*node\_modules*

Esta pasta contém todas as dependências instaladas pelo npm necessárias para o funcionamento do projeto. Ela é gerada automaticamente e não deve ser alterada manualmente. Caso seja necessário atualizar as dependências, use o comando 'npm install' no terminal.

*src*

A pasta 'src' contém o código-fonte principal do projeto Angular. Ela é subdividida em várias pastas importantes:  
- app: Contém os componentes e serviços do projeto.  
- assets: Contém arquivos estáticos, como imagens e ícones.  
- environments: Configurações para diferentes ambientes (desenvolvimento, produção).

*angular.json*

Este arquivo de configuração define as propriedades do projeto Angular, incluindo caminhos de arquivos, estilos, scripts e configurações de build. Alterações comuns incluem:  
- Adicionar novos estilos ou scripts.  
- Configurar diferentes ambientes de build.  
Exemplo de manutenção: Para adicionar uma biblioteca CSS, adicione seu caminho à seção 'styles'.

*package.json*

Este arquivo lista todas as dependências do projeto, além de scripts que podem ser executados, como 'start' e 'build'. É essencial para gerenciar pacotes e versões. Para instalar uma nova dependência, use 'npm install <nome-do-pacote>'.

*Detalhamento dos Componentes*

Os componentes são a base do projeto Angular. Cada componente contém três arquivos principais:  
- .ts: Lógica do componente.  
- .html: Estrutura da interface.  
- .css: Estilo do componente.

*dashboard.component.ts*

Este arquivo contém a lógica do dashboard principal. Ele inclui chamadas para serviços, manipulação de dados e métodos para filtrar e exibir informações. Um exemplo de manutenção seria ajustar a lógica de filtragem no método 'filterMovies'.

*dashboard.component.html*

Contém a estrutura da interface do dashboard. Para ajustar o layout ou adicionar novos elementos, edite este arquivo. Certifique-se de manter a consistência visual.

*movies.service.ts*

Este serviço é responsável por realizar chamadas à API e retornar os dados necessários para os componentes. Exemplo de manutenção: Para adicionar um novo endpoint, crie um método correspondente neste serviço.

*Boas Práticas e Comentários*

Todos os arquivos devem seguir as boas práticas de codificação e estar devidamente comentados. Isso facilita a manutenção e garante que outros desenvolvedores possam entender o código. Exemplos de boas práticas incluem:  
- Uso de nomes de variáveis claros e descritivos.  
- Documentação de métodos com JSDoc.  
- Organização lógica de funções e classes.

*Expansão do Projeto*

Para expandir o projeto, siga estas diretrizes:  
- Adicionar novos componentes: Use o comando 'ng generate component <nome>' para criar componentes consistentes.  
- Implementar novas funcionalidades: Centralize a lógica nos serviços e mantenha os componentes limpos.  
- Atualizar dependências: Sempre teste o projeto após atualizar pacotes para evitar incompatibilidades.

*Conclusão*

Esta documentação fornece uma base sólida para entender e manter o projeto Angular Golden Raspberry Awards. Seguindo as boas práticas descritas e utilizando os exemplos fornecidos, você estará apto a realizar alterações e melhorias com eficiência.