DOCUMENTAÇÃO

FILOSOFIA DE DOCUMENTAÇÃO

Entenda quando e onde documentar, seguimos padrões elevados de documentação e isso garante boas tomadas de decisão no futuro, saiba o escopo da sua documentação.

Se precisa documentar um processo ou decisão, documente no **Notion**. Se existe uma dependência no seu projeto ou método específico para executá-lo, use o **README**. Mas, se um trecho do seu código não é fácil de ser entendido, utilize comentários.

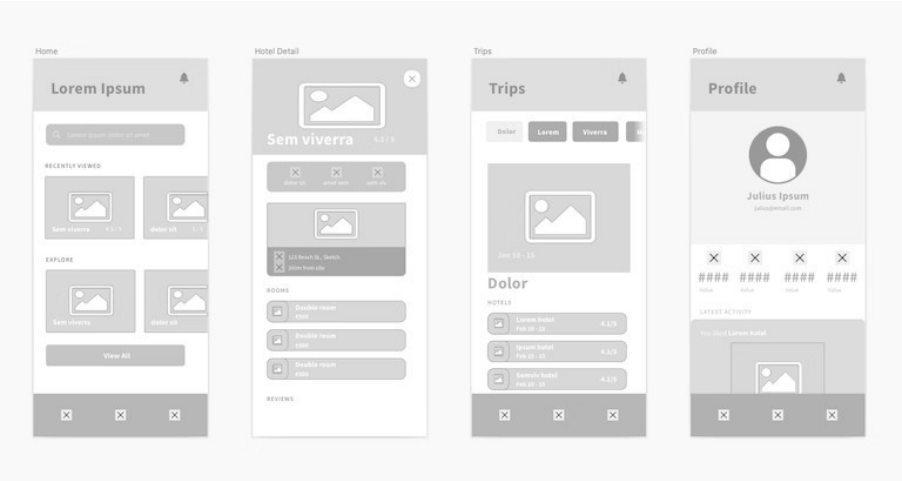
Entenda que existem exceções, porém geralmente se você precisa comentar muito seu código, é porque ele precisa ser refatorado. Utilize nomes de variaveis e funções bem, para que o código fique auto explicativo.

COMO DOCUMENTA UM PROCESSO

A documentação é uma tarefa manual que deve ser executada com atenção aos detalhes e de forma organizada para garantir a precisão e a clareza das informações registradas. Na figura abaixo, temos uma explicação de como funciona todo esse processo:

1 – Planejar o escopo do processo inicial

Desenvolver o escopo das informações iniciais e criar uma breve descrição com base nas metas, no cronograma e na prioridade do projeto. Também planejar quais membros vão trabalhar neste projeto e criar um wireframe do que será realizado. Abaixo está um exemplo de wireframe:



a - O que é Wireframe: É um protótipo da página de um site ou aplicativo, em outras palavras, nada mais é que um mockup de como a tela dele ser.

2 – Definir os limites do processo:

Após obter as informações iniciais do processo, definimos os limites. Descrevemos onde o processo se encaixa em cada membro da equipe, onde o processo começa e onde termina, e quem é afetado por ele. Definindo esses limites, podemos ajudar a estabelecer diretrizes de tarefas mais claras para a implementação do novo processo.

3 – Determinar as entradas e saídas do processo:

É analisado as entradas do processo, os recursos necessários para concluí-lo e as saídas, além dos resultados esperados ao final do processo. Determinamos as saídas analisando os objetivos de projeto iniciais e selecionando indicadores específicos e determináveis. Ao determinar as entradas e saídas, será possível dividir cada uma dessas metas em etapas menores no futuro.

4 – Definir os participantes e delegar as tarefas

Nesta parte, definimos os membros que farão parte do projeto, definindo as tarefas que cada um deverá realizar e entregar no final de cada sprint.

5 – Criar um Jira para documentar o processo

Nesta parte, definimos as sprints de cada semana do processo. Nelas determinamos as histórias e definimos o tema principal de cada atividade que será feita e delegada por cada membro da equipe. Dentro das histórias, podem conter tarefas, que são pequenas atividades que, quando finalizadas, são submetidas a análises para verificar se estão de acordo com os critérios de aceitação e, caso estejam, concluímos a história em si.

a -O que é o Jira: É uma ferramenta que permite realizar o monitoramento de tarefas e acompanhamento de projetos, garantindo o gerenciamento das atividades.

6 - Criar teste unitário:

Conforme o andamento das sprints, devemos realizar testes devido o critério de aceitação da sprint da semana. Ao realizar os testes, identificamos os locais em que os problemas surgem ou em que há potencial de risco, caso seja identificado, deve ser corrigido os erros. Está é uma oportunidade para aperfeiçoar o processo.

7 – Testar o processo:

Durante todo o processo de desenvolvimento, realizamos testes do processo para garantir o seu devido funcionamento. Nesta etapa, concluimos os testes finais e analisamos se o projeto está no seu devido critério de aceitação para passar para etapa de entrega.

8 – Resultado final e entrega

Após realizado os testes e correções, podemos finalizar o processo e entregar o projeto para o nosso cliente. Tendo em vista todos os processos acima, temos a certeza que estamos entregando um projeto bem documentado, funcional e com excelência.

COMO DOCUMENTA UM CÓDIGO

A documentação tem como principal finalidade aprimorar a capacidade de manutenção do produto. Com uma boa documentação, fica mais simples e fluído a transferência de conhecimento, pois tendo um código bem documentado, qualquer desenvolvedor consegue entender com facilidade como ele funciona. Algumas das boas práticas da documentação seriam manter o código simples e conciso, manter o código sempre atualizado, documentar quaisquer alterações no código e utilizar linguagem simples e de forma adequada.

Contudo, documentação gasta recursos, como qualquer atividade corporativa. Por isso, documentar em excesso acaba se tornando um gasto desnecessário, tanto em termos de tempo e esforço pessoal quanto de espaço para manutenção do código. Porém, documentar de menos ou documentar mal, também é um desperdício de recursos.

BOAS PRATICAS DE DOCUMENTAÇÃO

Endentação: Como boas práticas, uma boa endentação é essencial para o entendimento do código. Um código de forma desorganizada e desalinhada, acaba dificultando a sua compreensão. Entretanto, um código bem organizado e endentado, facilita rapidamente o seu entendimento.

Comentários: O princípio de um bom comentário no código se mantém em ser breve, relevante e contextualizado. Escrever apenas o necessário e sendo direto, sempre

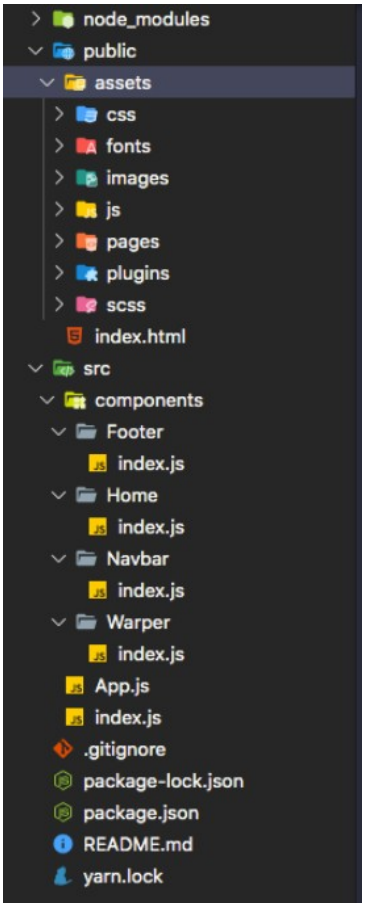
é o melhor caminho. Comentários em exceção podem acabar poluindo o código e atrapalhando o seu entendimento. Já por outro lado, documentar pouco, acaba se tornando irrelevante.

Nomes de variáveis, classes, métodos: Utilizando as boas práticas, colocando nomes adequados e padronizados, utilizando preferencialmente nomenclaturas em inglês, tornamos a leitura do programa muito mais simples e objetiva. Note alguns exemplos de como nomear:

Variáveis: Criar sempre nomes curtos (name, firstName) e significativos, em que bater o olho no nome, já identificamos para que aquela variável está sendo utilizada. Essas regras também são aplicáveis para os Métodos.

Classes e Interfaces: Começamos a nomeação com a primeira letra sendo maiúscula (Employees, Customers, Products). Para palavras compostas, a primeira letra de cada palavra deve ser maiúscula (HomePage, MainSector). Essas regras também são aplicadas para Interfaces.

Organização dos arquivos: Separar os códigos por pastas, é uma excelente prática para manter a organização do seu projeto. Com os arquivos bem organizados, conseguimos identificar com facilidade a sua funcionalidade. Um exemplo básico seria: dentro da pasta **src**, conter outras pastas como **html, css, img, components** e dentro dessas pastas, os arquivos específicos. Entretanto, se todos os arquivos estivessem na mesma pasta e misturados, acabaria dificultando o encontro dos arquivos. No exemplo abaixo, temos uma demonstração de como separar os códigos por pastas:



COMO DOCUMENTAR UM PROJETO DE FRONT END

Na documentação do front-end, temos alguns recursos que nos facilitam documentar todo o projeto. Sendo eles:

Escreva um README: Criar um arquivo README.md no diretório principal do projeto, onde você podemos fornecer uma descrição geral do projeto, instruções sobre como executá-lo localmente, como contribuir, e quaisquer outras informações importantes que outros desenvolvedores precisam saber.

Este exemplo, também pode ser utilizado para **criar um README da API.** Em um arquivo README do front-end, é importante ressaltar os seguintes tópicos:

- Título
- Uma breve descrição sobre o projetos
- Pré-requisitos de como instalá-lo
- Passo a passo de como fazer a instalação
- Mencionar os testes
- Linguagens utilizadas
- Versão do projeto
- Autores
- Licença
- Expressões de gratidão, contando como foi finalizar o projeto e agradecimentos.

Para complemento, vide o link GitHub

Documente seus componentes: Criar uma documentação para cada componente criado, incluindo informações sobre o propósito do componente, quais propriedades ele recebe, quais eventos ele emite e exemplos de uso.

Utilize comentários: Adicione comentários ao código-fonte do projeto para explicar o que cada seção faz. Fazendo isso, podemos ajudar outros desenvolvedores a entender sobre o que o código se trata e também, a facilitar a manutenção do código no futuro.

Crie diagramas: Utilizar diagramas de arquitetura e fluxo de dados para representar a estrutura do seu projeto. Isso pode ajudar outros desenvolvedores a entender como as diferentes partes do seu projeto se conectam.

Use ferramentas de documentação: Existem várias ferramentas de documentação disponíveis para Front-end, como o **JSDoc**, **Storybook**, **Style guide**, que pode ajudá-lo a gerar documentação automaticamente a partir do seu código-fonte.

COMO DOCUMENTAR UM PROJETO DE API COM SWAGGER

O **Swagger** é uma ferramenta de código aberto que permite documentar APIs de maneira fácil e padronizada. Ele permite criar uma especificação de API em formato YAML ou JSON, que descreve a estrutura da API, seus endpoints, parâmetros, respostas e outros detalhes.