

## NÍVEL BÁSICO



PROJETO 10

(CONTEÚDO DISPONÍVEL) {  
GALERIA;  
DE;  
VIDEO;  
(end);  
})();

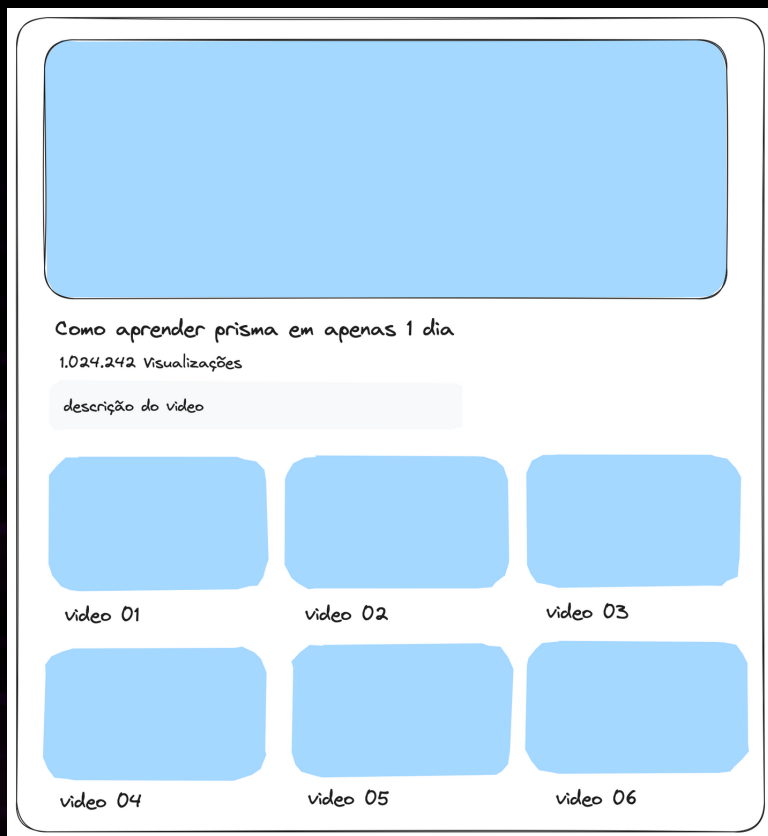
#PORTFÓLIOBOOSTPROGRAM

CONHECIMENTOS REQUIRIDOS:



FULL-STACK

# WIREFRAME



# GALERIA DE VIDEO

Crie uma galeria de tutoriais em vídeo extraídos de uma lista de reprodução React do YouTube.

## TECH STACK

- ➔ React
- ➔ NodeJS



## LIBRARIES

- ➔ React-youtube
- ➔ YouTube API



## BRIEFING

As listas de reprodução do YouTube são uma ótima maneira de **coletar vídeos** com um tema maior, sejam seus próprios vídeos ou de outra pessoa.

Ele pode ajudar as pessoas a se **concentrarem** em um tópico específico, como um conjunto específico de tutoriais.



## NÍVEL 1

O melhor do YouTube é que eles facilitam a **incorporação do player**. Isso nos permite adicionar facilmente uma lista de reprodução a qualquer página.

Crie uma página que incorpore uma lista de reprodução usando o player do YouTube.

## NÍVEL 2

Ao usar o player do YouTube, você não tem muito controle sobre a aparência. A boa notícia é que eles têm uma **API** que nos permite obter **informações** sobre uma lista de reprodução para que possamos usá-la em nosso App.

Busque a lista de reprodução do YouTube usando a **API do YouTube** e exiba os resultados em uma **página com links para cada vídeo**.

## NÍVEL 3

O link para o YouTube ajuda a permitir que as pessoas assistam facilmente a um vídeo, mas isso as afasta do seu site. Como podemos **permitir que as pessoas assistam aos vídeos** e **evitar que naveguem para longe**?

Adicione uma **incorporação de reprodução automática de um vídeo** à página quando alguém selecionar um vídeo.



# REQUISITOS DETALHADOS

## Back-end:

- ➔ Obtenha uma lista de vídeos de uma lista de reprodução do YouTube. O código de **back-end** precisará usar a **API do YouTube** para buscar uma lista de vídeos de uma lista de reprodução do YouTube.
- ➔ A **API do YouTube** fornece um método chamado **listPlaylistItems** que pode ser usado para buscar uma lista de vídeos de uma lista de reprodução.
- ➔ O método **listPlaylistItems** usa alguns parâmetros, **incluindo o ID da lista de reprodução** e o **número de vídeos** a serem buscados. Armazene os dados em um banco de dados.
- ➔ O código de **back-end** precisará armazenar os dados em um **banco de dados**. O banco de dados pode ser usado para armazenar **títulos de vídeo, descrições, miniaturas e outros metadados**.
- ➔ Crie uma **API REST** para expor os dados ao **front-end**. A **API REST** precisará fornecer métodos para buscar uma lista de vídeos, buscar um vídeo específico e incorporar um vídeo.

## Front-end:

- ➔ **Busque dados da API REST**. O código **front-end** precisará buscar dados da **API REST**. A **API REST** pode ser usada para **buscar uma lista de vídeos, buscar um vídeo específico e incorporar um vídeo**.

- ➡ Crie uma galeria de tutoriais em vídeo. O código do front-end precisará criar uma galeria de tutoriais em vídeo. A galeria pode ser criada usando um componente React. O componente React precisará buscar dados da API REST e exibir os vídeos em uma lista.
- ➡ Incorpore vídeos ao clicar. O código do front-end precisará incorporar vídeos ao clicar. Os vídeos podem ser incorporados usando a API do YouTube. A API do YouTube fornece um método chamado embed que pode ser usado para incorporar um vídeo em uma página da web. Maiores infos em:

## DEVELOPERS



### Vale lembrar que:

- ➡ O back-end deve ser responsável por buscar dados, armazenar dados e fornecer uma API REST.
- ➡ Caso você deseje desenvolver somente a parte back-end ou Front-end, tudo bem. Este app permite isso, na parte do front ao invés de salvar no banco de dados. Caso deseje ter um case full-stack, desenvolva ambas as partes.
- ➡ O front-end deve ser responsável por exibir dados, interagir com o usuário e fornecer uma interface de usuário.
- ➡ As duas partes devem ser dissociadas para que possam ser desenvolvidas de forma independente.

### O que isso quer dizer?

- ➡ Separe a lógica em folders que tenham nomes bem semânticos e distintos de acordo com a sua responsabilidade. Exemplo: Web para Front e API para back.