

NÍVEL BÁSICO



PROJETO 06

```
(CONTEÚDO DISPONÍVEL) {  
  PRODUCT;  
  HUNT;  
  CLONE;  
  (end);  
}());
```

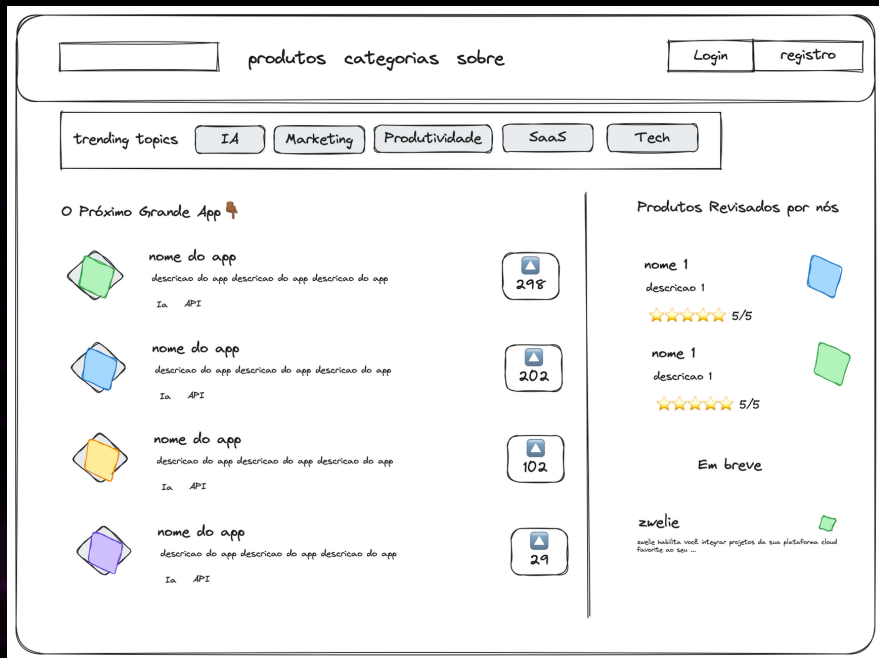
#PORTFÓLIOBOOSTPROGRAM

CONHECIMENTOS REQUIRIDOS:



FULL-STACK

WIREFRAME

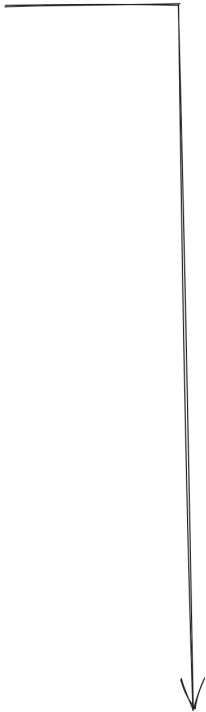


ENTIDADES

Entidades

apenas uma orientação para seus DB models e caso você use TypeScript

```
type User {  
  id: ID!  
  externalID: String!  
  createdAt: Date!  
  updatedAt: Date!  
}  
  
type Product {  
  id: ID!  
  title: String!  
  description: String!  
  url: String!  
  upvotes: Int!  
  createdAt: Date!  
  updatedAt: Date!  
}  
  
type Vote {  
  id: ID!  
  user: User!  
  product: Product!  
  createdAt: Date!  
  updatedAt: Date!  
}
```



Toda a autenticação será feita via clerk,
entretanto vamos salvar os metadados no nosso DB

PRODUCT HUNT

Iremos criar um clone do famoso [product hunt](#) que lista os projetos de startups do momento e permite votação dos usuários, com sistema de upvote



TECH STACK



React



TailwindCSS



NextJS



Vercel (conhecimento básico de serverless)



BaaS(Back-as-a-service)



NodeJS



Prisma



LIBRARIES



ClerkJS



BRIEFING

Uma idéia muito simples que hoje tem um **valuation (valor de mercado) bilionário**. Com um sistema de **UPVOTE** pelos próprios users ele lista os **apps mais visitados e votados seguindo a ordem top-down**.

NÍVEL 1

Vamos somente exibir uma **listagem de projetos** de acordo com a nossa **API** que será um **mock**.

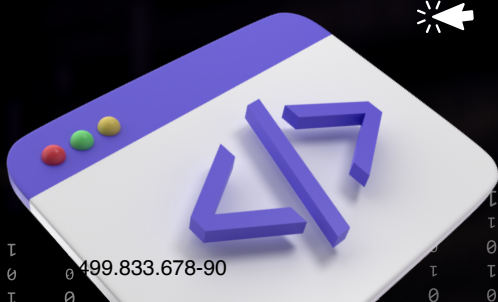
NÍVEL 2

Vamos exibir todos os projetos listados por uma API real que será uma pequena plataforma autenticada na qual o usuário admin irá adicionar os itens.

Esse nível irá requisitar um **refactor**, o que eu recomendo que seja feito em outro **branch**. Por mais que esteja trabalhando sozinho devemos replicar todas as situações do cotidiano de um **dev profissional**.

NÍVEL 3

Vamos criar um filtro de acordo com os **trending topics**: **Inteligência artificial, Produtividade, Marketing, SaaS, Tech**.



REQUISITOS DETALHADOS

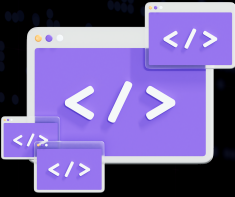
Tecnologias:

- ➔ React
- ➔ TailwindCSS
- ➔ Prisma
- ➔ Railway
- ➔ PostgreSQL
- ➔ ClerkJS
- ➔ Next.js
- ➔ TypeScript (opcional)

Estrutura:

- ➔ O projeto terá um cabeçalho, um rodapé e duas colunas principais.
- ➔ O cabeçalho terá uma barra de navegação com links para as principais páginas do site.
- ➔ O rodapé terá uma área para o footer do site.
- ➔ A primeira coluna principal listará todos os projetos, sem paginação, do maior número de votos para o menor número de votos.





A segunda coluna principal terá duas mini-categorias: "produtos top revisados" e "em breve".

Autenticação:

Os usuários poderão se **inscrever** e fazer **login** no site usando o **ClerkJS**, veja em:

CLERKJS



Dados:

O projeto inicial terá dados **mockados**. No entanto, assim que o projeto entrar no **nível 2**, os dados serão substituídos por **dados reais**.

O **modelo template** para manipular os dados reais estará disponível no **nível 3**.

Filtro:

No **nível 3**, os usuários poderão filtrar os projetos por **trending topics**.

