## 1. Prompts prontos para Lovable

Use cada prompt separadamente para que a IA desenvolva as etapas do projeto:

Prompt 1 — Estrutura inicial do projeto

Desenvolva a estrutura de um site para a "TV Calango", inspirado nos modelos da Brasil Paralelo, Revista Oeste e Jovem Pan.

O site deve ter:

Página inicial com destaque para programação ao vivo (12h às 00h).

Quando não houver programação ao vivo, exibir notícias recentes.

Seção de notícias estilo blog, categorizadas.

Seção de vídeos gravados e reprises.

Integração com streaming ao vivo (YouTube, Vimeo ou servidor próprio).

Layout moderno, responsivo e leve.

Desenvolva o projeto usando React.js no front-end, Node.js no back-end, Supabase como banco de dados, e Prisma ORM.

Prompt 2 — Estilo e design

Aplique um estilo inspirado nos sites:

https://revistaoeste.com/

https://www.brasilparalelo.com.br/

https://jovempan.com.br/

Priorize cores neutras e tipografia clara, com destaques em amarelo/dourado ou vermelho.

O design deve transmitir credibilidade jornalística e modernidade.

Prompt 3 — Funcionalidades técnicas

Crie a lógica para:

Gerenciamento de notícias (CRUD completo).

Gerenciamento de vídeos (links do YouTube ou uploads).

Sistema de agenda para definir horários ao vivo e exibir automaticamente na página inicial.

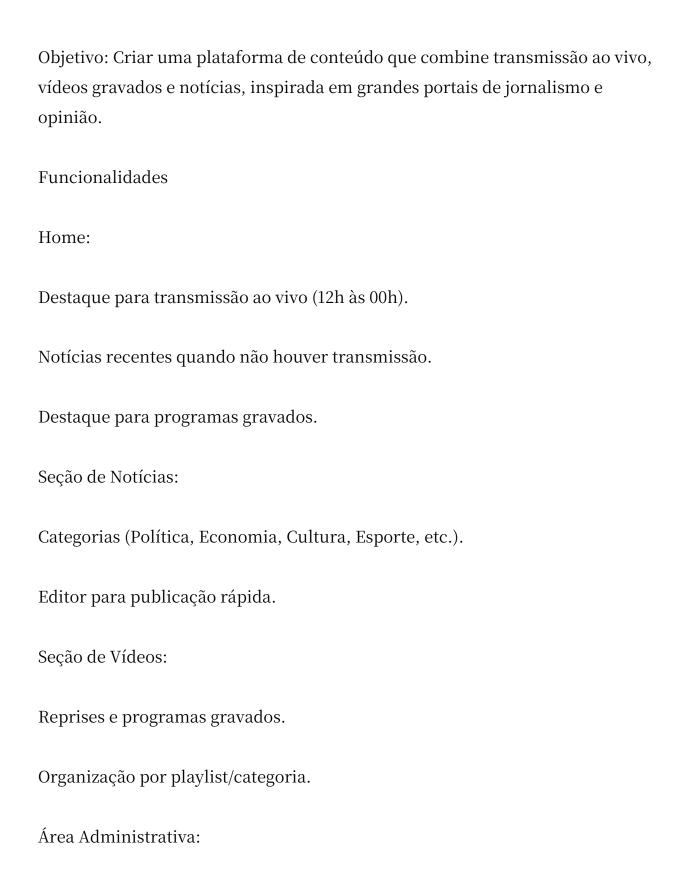
Área administrativa restrita para upload de notícias e vídeos.

Integração com redes sociais para compartilhamento automático.

2. PRD (Product Requirements Document)

Nome do Projeto: TV Calango

Responsável Web: Gustavo Possidonio



Login restrito.
CRUD de notícias e vídeos.
Definição da agenda ao vivo.
Integração:
Redes sociais.
YouTube/Vimeo para streaming.
3. Flowchart (Descrição textual)
[Visitante]
$\downarrow$
[Home]
├── [Ao Vivo] → Player streaming
├── [Notícias] → Lista + Categorias → Página da notícia
└── [Programas Gravados] → Lista vídeos → Player
[Admin]
↓ Login
↓ Dashboard
—— Gerenciar Notícias
├── Gerenciar Vídeos
└── Gerenciar Agenda Ao Vivo

```
4. Schema Prisma
model User {
id
      Int
            @id @default(autoincrement())
        String
name
        String @unique
email
password String
       String // admin, editor
role
createdAt DateTime @default(now())
model News {
            @id @default(autoincrement())
id
      Int
title
      String
content String
category String
imageUrl String?
authorId Int
createdAt DateTime @default(now())
updatedAt DateTime @updatedAt
               @relation(fields: [authorId], references: [id])
author User
model Video {
            @id @default(autoincrement())
id
      Int
title
      String
description String?
videoUrl String
category String
```