FrangoAlert

A Verdade Nu e Crua: Combate à Desinformação

A Missão

O Cenário: Vivemos na era da desinformação. Fake news se espalham 6x mais rápido que fatos reais.

A Proposta: O FrangoAlert nasce como um portal ágil e visual para verificar informações.

> *"Usar tecnologia para separar o joio do trigo (ou o milho do cascalho)."*

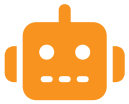
Funcionalidades Principais

Sistema de Usuários

Controle de acesso seguro com Login, Cadastro (com "Easter Egg" para Admin) e perfis personalizados.

Engajamento

Interatividade em tempo real. Usuários podem curtir e salvar notícias favoritas para ler depois.



IA Gemini

Integração nativa com Google Gemini para verificar a veracidade das notícias instantaneamente.

Poder da IA (Google Gemini 2.5)

O sistema não apenas exibe notícias, ele as **entende**.

Verificação de Fatos

Um botão em cada card consulta a IA para analisar o texto e dar um veredito: Confiável ou Fake.

Redator Automático

No painel administrativo, criamos um agente que gera rascunhos completos (título, texto e classificação) baseado apenas em um tema.

```
# Exemplo de Resposta da API Gemini
{
  "result": "Análise IA",
  "text": "O conteúdo sobre 'Terra Plana' é
cientificamente falso. Imagens de satélite
comprovam a curvatura da Terra.",
  "confidence": 0.99
}
```

Código e Banco de Dados (CRUD)

Tecnologias

Utilizamos **Python (Flask)** com **SQLAlchemy** (ORM) para gerenciar o banco de dados SQLite.

Estrutura de Dados

- ✓ Tabela **Users** (Autenticação)
- ✓ Tabela **Posts** (Conteúdo)
- ✓ Tabelas Pivô **Likes/Saves** (Relação N:N)

```python
# Modelagem de Relacionamento (Muitos-para-Muitos)

class User(db.Model):
  id = db.Column(db.Integer, primary_key=True)
  username = db.Column(db.String(100))

  # Relação N:N para Likes
  liked_posts = db.relationship(
    'Post',
    secondary=post_likes,
    backref='liked_by'
  )

# Rota de Like (Update)
def like_post(post_id):
  post = Post.query.get(post_id)
  if post in current_user.liked_posts:
    current_user.liked_posts.remove(post)
  else:
    current_user.liked_posts.append(post)
  db.session.commit()
```

Arquitetura Técnica

Backend: Python com Flask (Roteamento, Lógica de Negócio e API).

Banco de Dados: SQLite com SQLAlchemy (Relacionamento N:N para Likes/Saves).

Frontend: HTML5, CSS3 (Variáveis CSS) e JavaScript (AJAX para interações sem refresh).

AI Service: Google Generative AI (SDK Oficial) conectando ao Gemini.

Perguntas & Respostas

FrangoAlert | 2025