

Avaliação do Gemini

autor: Henrique Ferreira Moreira

Contexto do Projeto

Nessa apresentação, demonstro o que aprendi sobre o modelo de Inteligência Artificial da Gemini após uma série de testes.


O objetivo dessa série é avaliar e entender o funcionamento dos modelos de IA mais conhecidos da atualidade, definindo para mim e para você como e quando devemos utilizar cada uma delas para máxima eficiência.

O foco dessa pesquisa é a Inteligência Artificial como ferramenta de apoio para programadores, portanto, não espere uma avaliação geral e nem julgue alguma das IAs como ruins apenas por não serem eficazes para essa função. Dito isso, vamos começar...

Série de Testes



Desenvolvi 3 diferentes testes para testar cada modelo, sendo que todos serão submetidos aos mesmos experimentos. São eles:

- **3 problemas simples de programação:** envolve 1 pergunta simples sobre sintaxe, 1 resolução para um erro de lógica básica e 1 que se tratará da explicação de um erro dado um breve contexto.
 - **Guia de Instalação do framework Django:** para esse, a IA deverá explicar com detalhes como instalar o framework em questão. Além disso, ela também deverá criar uma aplicação simples com banco de dados integrado como forma de base para o programador.
 - **Desenvolvimento de um site de vendas:** por fim, a IA deve ser capaz de, a partir de um pedido bem elaborado, desenvolver um site de vendas por completo, tendo 2 chances de fazer isso bem.
- 

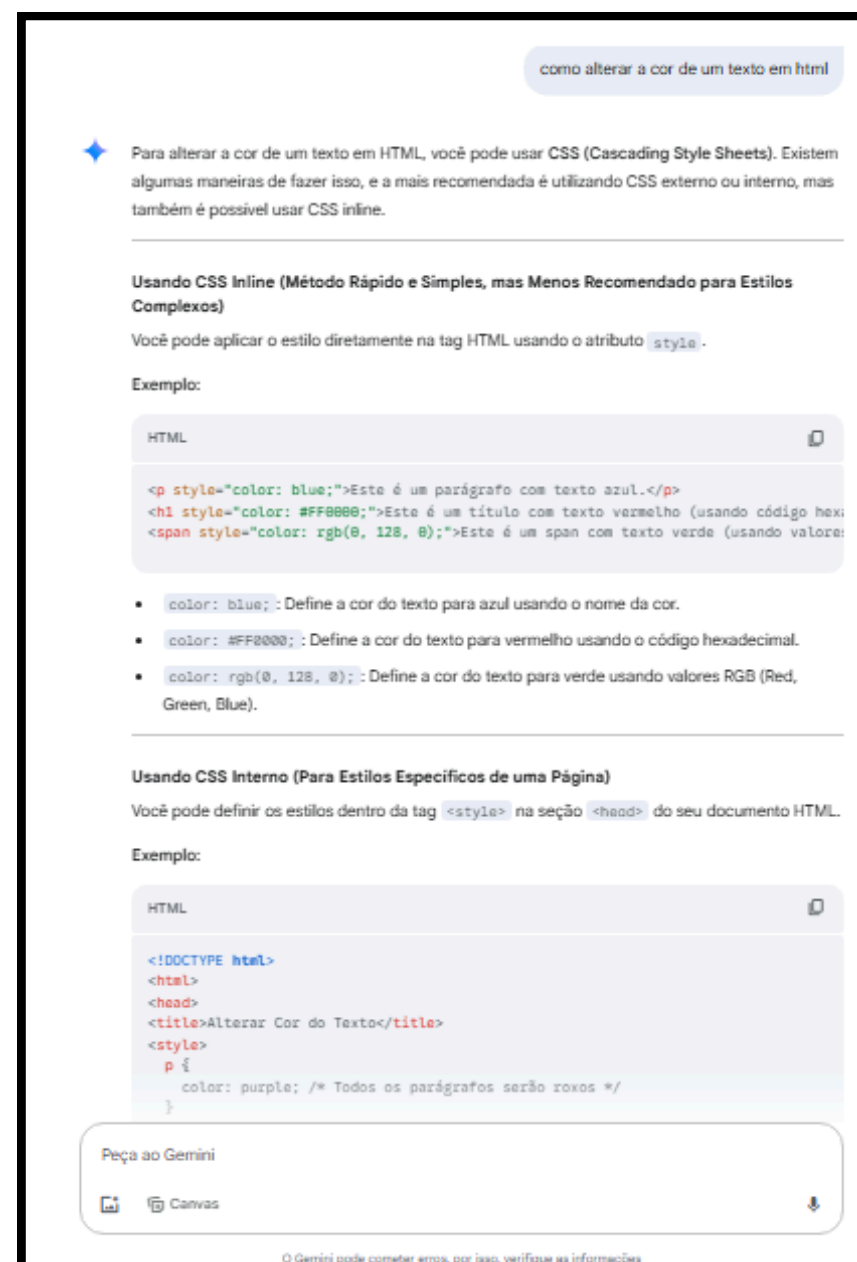
Perguntas Básicas

Para esse primeiro teste, fiz perguntas básicas sobre sintaxe, lógica de programação e explicação de erros. O objetivo aqui é ver o quanto direta e precisa a IA consegue ser quanto a esses problemas.

Acredito que o resultado ideal aqui seria uma resposta rápida e breve, mas que também explique o código com detalhes ao final, o que é mais útil para quem sabe bem o que está fazendo e só se perdeu em um pequeno detalhe, mas também ajuda quem procura entender o porquê da resposta.

Explicado isso, vamos a primeira pergunta...

Resultados do Gemini



Primeira Pergunta: “*como alterar a cor de um texto em html*”

A abordagem do Gemini foi bastante abrangente, mostrando todas as formas possíveis (e consideradas como boas práticas) de se alterar a cor de um texto em html.

Obviamente, isso não foi de todo bom, já que com uma resposta tão longa, a eficiência e a praticidade de apenas obter a resposta imediatamente acaba sendo perdida, mas a vantagem de trazer uma resposta completa dá para um usuário que realmente não entende do assunto a oportunidade de aprender, o que também deve ser valorizado.

Resultados do Gemini

Segunda Pergunta:

Estou tentando fazer com que, quando executo a função imprimirTabuleiro(), o tabuleiro principal seja imprimido no console, mas ele aparece de forma quebrada, como conserto isso?

```
tabuleiro_principal = [ [torre(False, "♖ "), cavalo(False, "♘ "), bispo(False, "♗ ") [...]]
def imprimirTabuleiro(jogador):
    impressao = ""
    tabuleiro_suporte = copy.deepcopy(tabuleiro_principal)
    numeros_tabuleiro_suporte = numeros_tabuleiro
    print("_____")
    linha_atual = 0
    impressao = impressao + f"{numeros_tabuleiro_suporte[linha_atual]}" linha_atual+=1
    for linha in range(len(tabuleiro_suporte)):
        for coluna in range(8):
            impressao = impressao + " |" + tabuleiro_suporte[linha][coluna].aparencia
            if (coluna+1) % 16 == 0 and linha+1 != 8:
                impressao = impressao + " |\n"
                impressao = impressao + f"{numeros_tabuleiro_suporte[linha_atual]}"
                linha_atual+=1
            elif (coluna+1) % 16 == 0:
                impressao = impressao + " |" print(impressao)
    print("-----")
    print(" a b c d e f g h")
```

Para ser honesto, não gostei muito da resposta do Gemini para este caso. Apesar de sim, explicar meu problema logo no primeiro paragrafo, reescreveu meu código de uma forma muito esquisita por assumir que os objetos que utilizei não existiam.

Claro, é apenas uma pequena crítica, principalmente porque eu poderia utilizar apenas a parte que me interessa da resposta e ignorar o resto, mas me causa um grande incômodo por me fazer sentir que preciso sempre enviar o código completo ou dar um grande contexto para receber respostas completas e corretas, o que não considero necessário para esse caso.

OBS: os [...] no tabuleiro principal representam uma abreviação para que a pergunta coubesse aqui, a IA recebeu a tabela 8x8 por completo.

Resultados do Gemini

Terceira Pergunta:

“estou tentando identificar se uma resposta para uma questão é true ou false, mas obtenho erro na hora de imprimir a escolha da pessoa: `TypeError: unsupported operand type(s) for +: 'bool' and 'str'`”

Para essa última pergunta senti que o Gemini foi bastante preciso, explicando o que causa esse tipo de erro logo no primeiro parágrafo enquanto explicava com mais detalhes e exemplos no restante da resposta, o que eu diria que foi o ideal para esta situação.



Guia de Instalação

Para esse segundo teste, enviei para cada uma das IAs a série de mensagens a seguir:

“Descreva detalhadamente para mim todo o processo de instalação do framework de desenvolvimento web django, me explicando o que devo fazer desde meu projeto vazio até a página de "servidor funcionando" do django”

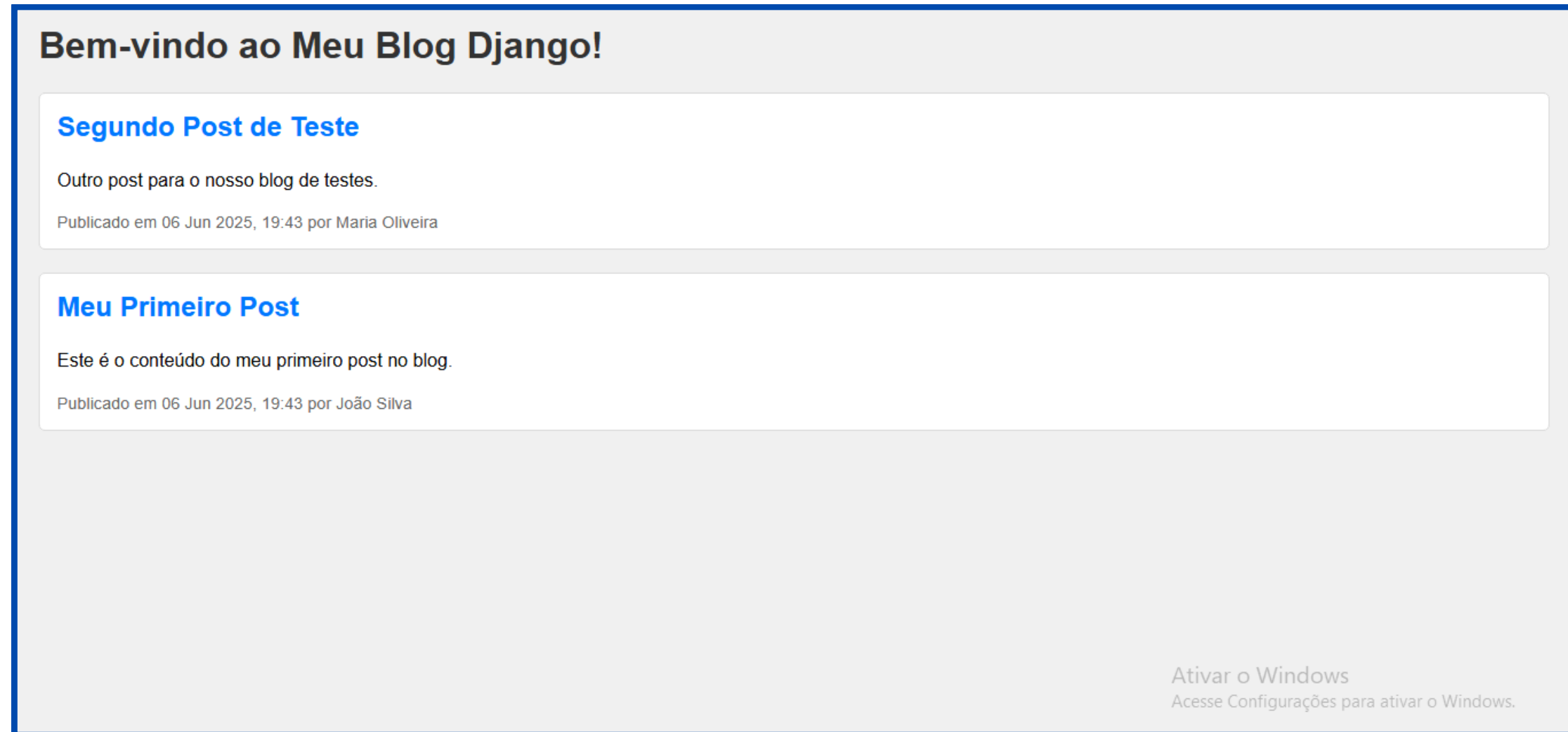
“agora crie e utilize um modelo de banco de dados nesse mesmo projeto, explique para mim de forma detalhada”

Acredito que qualquer profissional humano seja capaz de compreender o que estou pedindo, por isso, contabilizei cada mensagem extra como uma pequena falha interpretativa.

Resultados do Gemini

O Gemini conseguiu gerar o que pedi, mas apenas depois de:

- 1 pergunta extra por não ter entendido que precisava criar uma página visual do projeto.



Mesmo sem precisar, a página até que ficou bastante apresentável.

Resultados do Gemini

no diretório atual, evitando uma pasta aninhada extra. Se você omitir o `.`, uma pasta `meu_site_django` será criada dentro da pasta `meu_site_django` que já existe.

Após executar este comando, a estrutura de diretórios do seu projeto será algo assim:

```
meu_projeto_django_env/  
├── meu_site_django/  
│   ├── manage.py  
│   └── meu_site_django/  
│       ├── __init__.py  
│       ├── asgi.py  
│       ├── settings.py  
│       ├── urls.py  
│       └── wsgi.py  
└── db.sqlite3 (será criado após a primeira migração)
```

Passo 4: Entendendo a Estrutura do Projeto

Vamos dar uma olhada rápida no que cada arquivo significa:

- `manage.py`: Um utilitário de linha de comando que permite interagir com seu projeto Django de várias maneiras (executar o servidor de desenvolvimento, fazer migrações de banco de dados, etc.).
- `meu_site_django/` (o diretório interno): Este é o pacote Python do seu projeto.
 - `__init__.py`: Um arquivo vazio que indica que este diretório deve ser tratado como um pacote Python.
 - `asgi.py`: Uma entrada para servidores compatíveis com ASGI para servir seu projeto. Usado para aplicativos assíncronos.
 - `settings.py`: As configurações/configurações do seu projeto Django. Aqui você define coisas como configurações de banco de dados, aplicativos instalados, fusos horários, etc.
 - `urls.py`: As declarações de URL para o seu projeto. É aqui que você mapeia URLs para as views do seu aplicativo.
 - `wsgi.py`: Uma entrada para servidores compatíveis com WSGI para servir seu projeto. Usado para aplicativos síncronos.

Passo 5: Executando as Migrações Iniciais

O Django vem com alguns aplicativos internos (como autenticação, administração, etc.) que precisam de tabelas no banco de dados para funcionar corretamente.

1. Certifique-se de que você está no diretório que contém `manage.py`:

Peça ao Gemini

Canvas

O Gemini pode cometer erros, por isso, verifique as informações

Sobre a instalação, o Gemini foi bastante preciso, deixou todos os processos bastante claros e ainda não cometeu erros quanto ao programa de base pedido. Além disso, ao final de quase todas as linhas de código que deviam ser escritas no terminal, ou então acompanhado dos diversos arquivos que surgem após determinado comando, havia uma explicação detalhada do que tinha acabado de ser feito.

Não tenho muito o que criticar, para ser honesto, senão pelas explicações mais estruturais que vi pela documentação do django, diria que ter perguntado diretamente ao Gemini como instalar a biblioteca teria sido mais rápido e prático.

Site de Vendas

Para o terceiro e último teste, envio para a Inteligência Artificial um grande texto explicando diversos detalhes sobre o que quero em minha aplicação, como você poderá ver no slide seguinte. O objetivo de tantos detalhes é deixar perfeitamente claro o que quero, assim, erros podem ser diretamente interpretados como falta de compreensão básica ou dificuldades em escrever um código tão complexo.

O objetivo desse último teste é ver até onde vale a pena utilizar o Gemini como forma de desenvolver a estrutura e design de um site, já que é um processo muitas vezes repetitivo e que ninguém gosta de fazer. Vale mencionar que apenas farei perguntas adicionais se o programa realmente não estiver funcionando, fora isso, quaisquer bugs funcionais ou visuais serão mencionados como falhas da IA. Dito tudo isso, vamos para os resultados...

Mensagem enviada para a IA



“Objetivo geral: Site de vendas onlines feito com o framework django

Funcionalidades principais: Clientes podem acessa-lo para adicionar produtos ao seu carrinho de compras. Administradores podem acessa-lo para criar, atualizar, observar e remover produtos já cadastrados. Adicione o que mais achar necessário.

Modelo de dados: cada produto deve possuir categorias (um mesmo produto pode possuir várias categorias), identificador único de 5 algarismos, nome, quantidade, preço, descrição, prioridade, data de cadastro e descrição curta. Para cada categoria, uma subcategoria deve poder ser criada, fazendo parte da busca caso sua categoria principal seja mencionada, mas servindo como filtragem extra. Os clientes também devem ser armazenados de forma semelhante, possuindo nome, identificador, carrinho, email, gênero, data de nascimento e data de cadastro. Adicione o que mais achar necessário.

Estrutura de páginas: página inicial (deve apresentar a empresa, promoções, produtos, novidades, etc...), info do produto (mostra informações do produto quando clicamos nele), lista de produtos (mostra todos os produtos da loja em ordem de prioridade e categoria, pode receber uma ou mais categorias, nome, data de cadastro ou margem de preço como possibilidade de filtragem), carrinho (painel por onde o usuario pode verificar os produtos adicionados e valor total, assim como remover ou alterar a quantidade de qualquer produto). Adicione o que mais achar necessário.

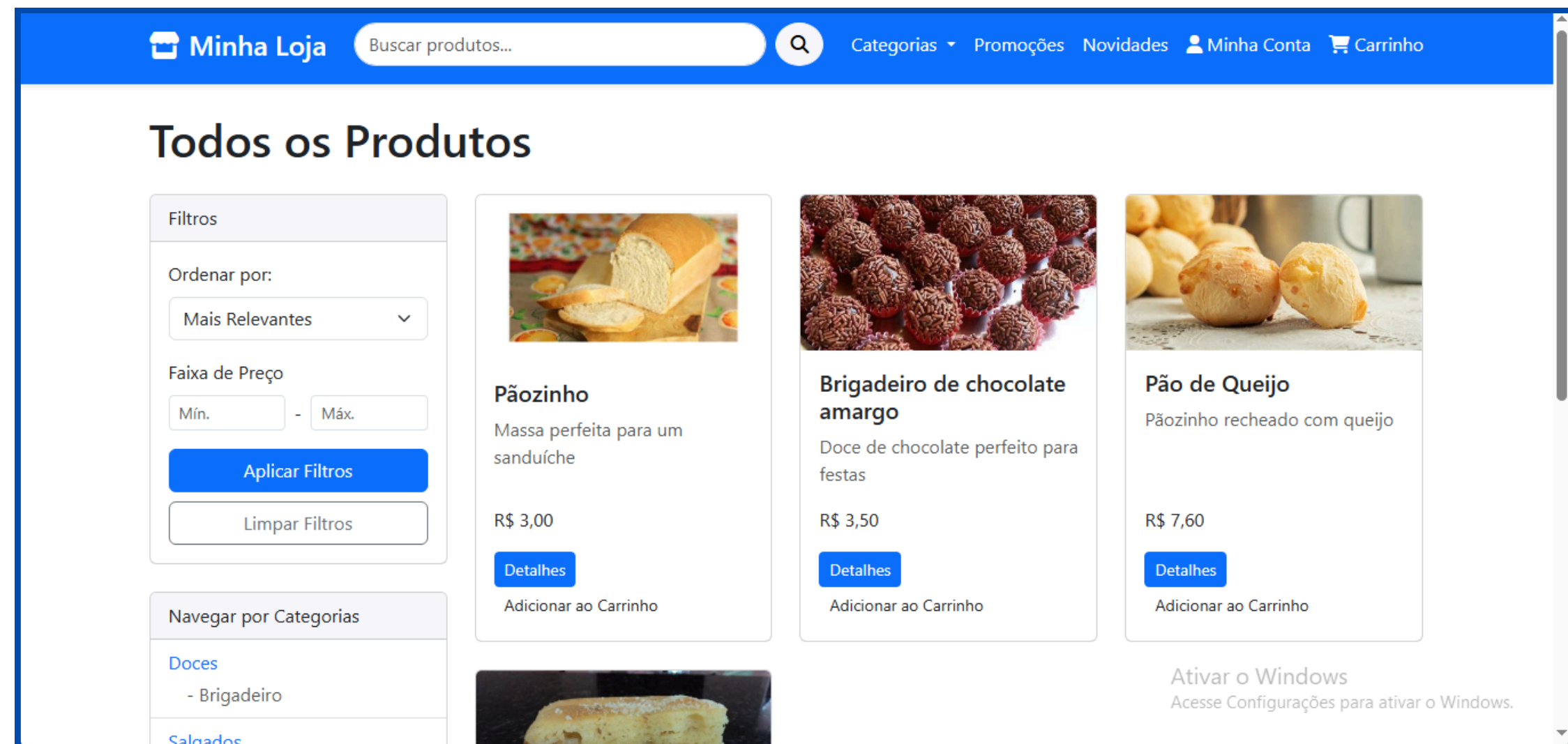
Estilo e design: Escolha cores com o objetivo de transmitir conforto ao usuário. Quanto ao layout, siga a convenção da maioria dos sites de venda (carrinho no canto superior direito, logo da empresa no canto superior esquerdo, barra de pesquisa e nome da empresa também no cabeçalho, etc...). Adicione o que mais achar necessário.

Autenticação e segurança: O site exige login para acessar ou utilizar o carrinho, mas não para navegar pelo restante do site

Considerando as informações acima, desenvolva por completo a aplicação WEB descrita. Tente não se limitar a algum limite de caracteres, e se necessário para isso, me peça para digitar "S" para sim no final de uma etapa do desenvolvimento, assim você continua o desenvolvimento considerando mesmo contexto”

Resultados do Gemini

Após uma série de “S” e respostas do Gemini, não precisei de nenhuma pergunta adicional por conta de bugs diretos, porém, a falta de clareza se fez presente enquanto definia os diretórios da aplicação, além do que, o Gemini pediu algumas vezes por comandos que exigiam coisas que só foram descritas e explicadas depois, ou seja, deveriam ter sido pedidas ao final de tudo, e não naquele momento.



Resultados do Gemini



Modelo de Dados:

O site quase não apresentou problemas quanto a modelagem de dados, na verdade, poderia dizer que o mal funcionamento da parte do perfil dos clientes está mais relacionada às funcionalidades do site, então sim, o Gemini foi perfeito em relação aos objetos e ao banco de dados do site.

Admito que não é um processo difícil nem para humanos, mas ainda assim é um ponto positivo para o Gemini.

Resultados do Gemini

Estrutura de Páginas:

O site apresentou um problema quanto a estrutura das paginas, sendo ele:

- ausência de possibilidade de filtragem por categoria (as categorias aparecem, mas nada muda quando acessamos qualquer uma delas) e por nome (até funciona bem com 2 ou mais letras, mas ele parece identificar letras usadas na descrição longa).

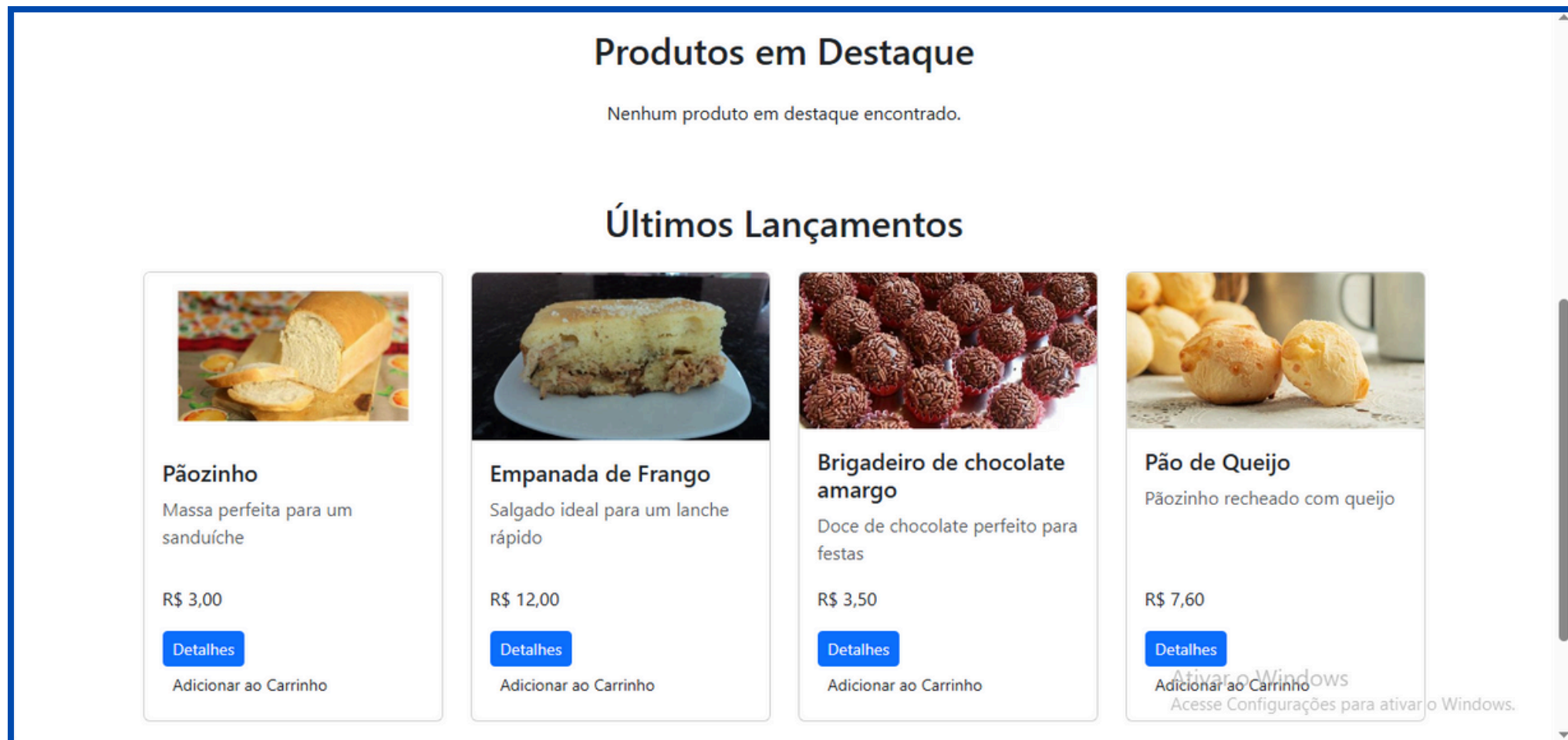
Me preocupa que o Gemini tenha sido capaz de criar os campos para as categorias e nome do jeito correto, mas falhado em dar a funcionalidade aos mesmos. Apesar disso, a filtragem dos demais itens foram implementados com excelência.

Resultados do Gemini

Estilo e Design:

Apesar dos bugs já mencionados, não tenho nada a criticar quanto ao design do site. Apesar de ser bem genérico, a combinação de cores e o equilíbrio entre efeitos visuais e performance dão um aspecto profissional impressionante.

Outro ponto a se ressaltar é a adição de um sistema que te permite adicionar imagens para os produtos, o que realmente me surpreendeu na hora.



Conclusão

Analizando os resultados, pude perceber que o Gemini não se dá muito bem em situações de pouco contexto, procurando sempre dar respostas completas, mesmo quando não há necessidade. Contudo, quando ele recebe o esperado contexto em forma de pedidos precisos e bem estruturados, a resposta beira o impecável.

Outra qualidade marcante está na facilidade de compreender o que acontece no programa, já que muitas das linhas possuem comentários que explicam exatamente o que é feito em cada momento. Além disso, na própria resposta do Gemini, diversos pontos mais complexos são explicados cuidadosamente. O Gemini me pareceu uma ótima escolha de IA para perguntas complexas, seja instalar um framework ou criar um projeto complexo do zero, mas apenas se você gastar um tempo para dar a ele o devido contexto.