Objetivo: Avaliar domínio de stack, boas práticas, raciocínio lógico e estruturação

Para backend: Crie uma API que:

- Permita cadastrar clientes de uma loja de brinquedos
- Liste os clientes com opções de filtros (ex: por nome ou e-mail)
- Permita deletar um cliente
- Permita editar informações de um cliente
- Requer autenticação para acessar as rotas
- Adicione testes automatizados
- Crie uma tabela adicional chamada sales para armazenar vendas por cliente
- Crie uma rota de estatísticas que retorne o total de vendas por dia
- Crie outra rota que retorne:
 - O cliente com o maior volume de vendas
 - O cliente com a maior média de valor por venda
 - O cliente com o maior número de dias únicos com vendas registradas (frequência de compra)

Para front-end: Crie uma aplicação que:

- Permita adicionar clientes com nome, e-mail e data de nascimento
- Liste os campos conforme achar pertinente
- Adicione autenticação simples
- Consuma a API de estatísticas para:
 - o Exibir um gráfico com o total de vendas por dia
 - Destacar visualmente:
 - O cliente com maior volume de vendas
 - O cliente com maior média de valor por venda
 - O cliente com maior frequência de compras
- Adicione um campo visual que indique, para cada cliente, a primeira letra do alfabeto que ainda não apareceu no nome completo do cliente. Se todas as letras de a-z estiverem presentes, exibir '-'.
- Ao consumir a API de listagem de clientes, considere que o endpoint pode retornar uma estrutura desorganizada ou com dados redundantes (ex: múltiplos campos aninhados, duplicações, ou propriedades desnecessárias). Faça o tratamento e normalização desses dados no front-end antes de renderizá-los na interface. Abaixo está o JSON que será retornado pela API de listagem de clientes. Esse é o formato exato com o qual o candidato deverá trabalhar:

```
JavaScript
{
   "data": {
```

```
"clientes": [
    {
      "info": {
        "nomeCompleto": "Ana Beatriz",
        "detalhes": {
          "email": "ana.b@example.com",
          "nascimento": "1992-05-01"
       }
      },
      "estatisticas": {
        "vendas": [
          { "data": "2024-01-01", "valor": 150 },
         { "data": "2024-01-02", "valor": 50 }
    },
    {
      "info": {
        "nomeCompleto": "Carlos Eduardo",
        "detalhes": {
          "email": "cadu@example.com",
          "nascimento": "1987-08-15"
        }
      },
      "duplicado": {
        "nomeCompleto": "Carlos Eduardo"
      },
      "estatisticas": {
       "vendas": []
 1
"meta": {
 "registroTotal": 2,
 "pagina": 1
```

```
},
"redundante": {
    "status": "ok"
}
```

• O candidato deve extrair corretamente os dados relevantes e ignorar as informações desnecessárias ou duplicadas.

Requisitos:

- API com autenticação rest
- Banco de dados obrigatório no backend
- Front pode consumir API real ou mockada via ferramenta
- Testes obrigatórios no backend