Exercício
Samplemed

Passando aqui para te parabenizar! Você passou na nossa etapa de entrevistas, e agora estamos no **TESTE TÉCNICO**.

Essa etapa é importante para avaliarmos sua experiência e organização durante o desenvolvimento;
1) A ideia não é ver se você vai acertar no bit. Queremos ver como se organiza e implementação da solução (boas práticas, etc).
2) Caso você tenha qualquer tipo de dúvida, pode entrar em contato conosco.

>>> **SEGUE O TESTE**

Contexto

Faça um MVP (Produto mínimo viável) de um blog utilizando Python, Django e Django Rest Framework, contendo minimamente a autenticação, cadastro e listagem de artigos do blog. Se possível utilize um modelo que desacople o frontend do backend através de chamadas API com autenticação JWT. Se preocupe mais com o código interno das APIs (organização, consistência, etc) do que com as telas. Não há necessidade de criar telas se preferir, pode mostrar exemplos de chamadas de APIs com o Postman por exemplo. Estamos mais preocupados com a estrutura do Backend.

Imagine que um artigo do blog tenha palavras-chave, e essas palavras-chave precisam ser criadas em outra tabela. Na API para cadastrar um post você precisa enviar no JSON de cadastro as palavras-chave que serão cadastradas em conjunto com o post do artigo. Queremos entender como vai fazer a amarração dos viewsets e serializadores. Na próxima página, **anexo A** está um exemplo de envio do JSON na api de cadastro de artigos

Arquitetura
Crie um diagrama com a estrutura das tabelas necessárias, mesmo que não use todas no código, além da tabela de posts desenhe outras tabelas necessárias para o funcionamento correto do sistema de um blog (tabela de usuários, de palavras chave, comentários, etc). Pode colocar no repositório um arquivo com o schema, um DBML (https://www.dbml.org/), ou mesmo uma imagem (procure colocar este artefato junto com o projeto - git, etc). Você não precisa utilizar todas essas tabelas no MVP, só queremos entender sua organização e conhecimento na arquitetura do banco de dados.

Considerações
- **Organização**: Agrupe os módulos em contextos de forma que essa associação seja algo que faça lógica no mundo real, se possível utilize todos os nomes de funções e commits em inglês.
- **Repositório**: O histórico de commits deve ser claro e representarem uma mudança unitária do projeto.

Detalhe em um documento a parte os motivos de determinadas tomadas de decisão e da codificação:

- **Resiliência**: O que fazer para mitigar possíveis erros e controlar os possíveis erros recebidos da API?
- **Performance**: Quais boas práticas são aplicadas em banco de dados e no código para garantir performance?
- **Segurança**: Como garantir segurança para as APIs do sistema?
- **Simultaneidade**: Como trabalhar com simultaneidade se milhares de requisições forem solicitadas simultaneamente?

Compartilhe o código do exercício em um repositório git e nos envie.

>>> **FIM DO TESTE**

Muito sucesso para você e até breve!
Um abraço,

Anexo A: Exemplo de JSON para cadastrar um artigo

```
{
    "title": "Nome do post",
    "subtitle": "Subtítulo do post",
    "type": 0,
    "content": "Conteúdo do post",
    "status": 1,
    "keyword_set": [
            {
                "name": "aviação"
            },
            {
                "name": "tripulação"
            }
        ]
}
```