Introdução (Backend):

Iremos fazer uma grande atualização e mudanças no nosso código. Por isso, é imprescindível que você saiba Ruby e Rails. Algumas apps nossas são legadas, monolíticas, mesmo no backend você irá precisar lidar com um pouco com frontend (mas fica tranquilo, te ajudamos nisso).
Tempo é fundamental para nós, porém, vamos focar na qualidade do seu trabalho, por isso, leve o tempo que achar necessário.

Desafio:

Crie um CRUD de municipes (Exceto deletar). O munícipe tem status ativo e inativo.
Idealmente, só precisa ser 2 páginas: Listagem de CRUD (com opções para navegar), e o cadastro em si. 2 páginas é apenas uma sugestão, você é livre para montar o UI/UX da forma que achar melhor.
Ter uma entidade relacionada chamada Munícipe. Essa entidade cadastrar cidadãos (pessoas) dentro de um município.

As seguintes regras devem ser seguidas:

1.1 Dados do munícipe: Nome completo, CPF, CNS(cartão nacional de saúde), Email, Data nascimento, Telefone (código do país e ddd), Foto e status.
1.2 Todos os dados do munícipe são obrigatórios;
1.3 CPF, CNS,Email devem ser válidos;
1.4 Tenha atenção a data de nascimento. Valide os casos impossíveis/improváveis de serem válidos;
1.5 Foto do munícipe deve ser tamanhos diferentes para servir vários casos.
Ter uma entidade relacionada chamada Endereço. Essa entidade salva o endereço relacionado ao munícipe. As seguintes regras devem ser seguidas:
2.1 Campos: CEP, Logradouro, complemento, bairro, cidade, UF e código IBGE;
2.2 Todos os dados são obrigatórios, exceto complemento e código IBGE;
2.3 Em termos de MVC, existe apenas a Entidade relacional endereço. O restante é dispensável;

Regras de negócio:

Após criar/atualizar um munícipe, você deve mandar um Email e sms ao mesmo informando sobre o cadastro de suas informações e quando o seu status sofrer alteração;
Filtrar municipes por dados dele e/ou de endereço. É livre a escolha do que deve ser feito.

Dicas:

UI/UX:
É possível otimizar o tempo de cadastro do endereço a partir do UX.
Você deve minimizar o máximo possível a navegação do usuário. Como você faria isso?

Backend:
Pense que essas regras podem ser mudadas com uma frequência alta;

Gostamos de otimização, setups e deploys são sempre automatizados (Docker?)
Não preciso dizer que você precisa testar a maioria dos arquivos, não é mesmo?
Princípios e padrões de projetos são muito bem vindos e essenciais para Seniors;
Reduzir o número de chamadas ao banco de dados é essencial.

Tools:
Ruby, Ruby on Rails e Postgres são obrigatórios;
Elasticsearch/Kafka (opcional, plus);
Utilize ActionView, porém, AssetPipeline/Sprockets ou uma abordagem SPA junto ao rails;

Critérios:

TUDO, absolutamente TUDO, será olhado (sim, teremos carinho pelo seu tempo gasto e olharemos com destreza seu teste);
Esse teste é backend, então, pesará muito como você realizou;
Frontend é opcional para o seu caso, mas, irá contar muito positivo como você fará;
Ver seu teste rodando em modo produção conta MUITO. Fica ao seu critério se faz e em qual plataforma;
Lembre-se que o melhor de um profissional é sua atitude quanto a resolução de problemas.
Interprete este item como desejar ;)
Coloque o código em seu repositório no github e vamos baixar e analisar internamente, após a análise, vamos te dar o feedback.