Desafio Back-End

Construir uma SPA de Lista de Tarefas (To-Do List) usando uma stack de alta performance, segurança e minimamente escalável!

Dicas:

- Trabalhe sempre pensando em **a)** reutilização de código, **b)** adoção de padrões de projeto, **c)** serviços REST com o Back-End e Front-End totalmente separados, **d)** escalabilidade e **e)** níveis mínimos de segurança.
- Quanto **maior** o grau de senioridade você aplicar, **mais** será cobrado em função destes pontos acima!
- Não é obrigado a seguir à risca as sugestões ou telas propostas, inove!
- **Acima de tudo: divirta-se! Dê à solução a sua cara!**

Funcionalidades Gerais

As Single Page Applications serão desenvolvidas usando Vue.js e Webpack! Para tanto é necessário fazer o scaffolding com o **vue-cli** (https://github.com/vuejs/vue-cli) para gerar o esqueleto do projeto para o nosso front-end do projeto. Para template sugerimos o **webpack** padrão!

E vai ser super legal fazê-lo! Sua SPA vai ser alimentada com as APIs que você montar no seu Back-End, que pode ser um serviço tipo json-server!

Todo servidor possui funcionalidades de servir páginas estáticas, neste caso o webpack vai gerar um compilado de todas as dependências preparado para o navegador e todos os estilos, você pode simplesmente indicar uma pasta sendo um diretório de arquivos estáticos!

Esteja livre para utilizar qualquer biblioteca de componentes visuais como Bootstrap, Foundation, Semantic, Materialize Ui, etc! Para trabalhar com datas no javascript recomendamos o Moment.js e posicionamento de tarefas o Flexbox para css!

SPA de listagem de tarefas

- Criação de parâmetro de renderização do dia, semana, mês para o sistema (o padrão será a semana atual);
- Listar tarefas com filtro por realizadas ou não realizadas;
- Forma de visualizar tarefas por dia, semana ou mês;
- Botão de cadastro de uma nova tarefa;
- Modal (ou solução personalizada) para criação de tarefa com possibilidade de adicionar ou remover tags;
- Botão de remoção da tarefa;
- Botão de edição da tarefa;
- Modal (ou solução personalizada) para edição de tarefa com possibilidade de adicionar e remover tags;

Sugestão de atributos de um objeto **Tarefa**:

- Descrição;
- Data & hora que a tarefa acontecerá;
- Tempo de duração em minutos, desde que seja um natural múltiplo de 5;
- Lembrete de **X** minutos antes, tal que x é um natural múltiplo de 5;
- Data & hora de criação da tarefa;
- Data & hora de criação da última edição;
- Data & hora de remoção;

Nível Júnior

Tente implementar o maior número de itens possíveis descritas no setor de funcionalidades!

Expectativa e Desafios

- Implemente - sempre que possível - conceitos mínimos de padrões de projetos;

Nível Pleno

Gosta de desafios? Vamos dar uma turbinada nos conceitos básicos da nossa To-Do-List! Que tal inserir algumas tags para pesquisa? A partir de agora a solução via API se torna obrigatória e não somente um diferencial!

SPA de listagem de tarefas

- Campo de busca de tags;

SPA de listagem de tags

- Lista tags cadastradas;
- Modal (ou solução personalizada) para criação de uma tag;
- Botão de remoção de uma tag;
- Botão de edição de uma tag;
- Modal (ou solução personalizada) para edição de uma tag;

Sugestão de atributos de um objeto **Tag**:

- ID único;
- Descrição;
- Data & hora de criação;
- Data & hora da última edição;
- Data & hora de remoção;

SPA de busca de tarefas

- Listagem de resultado da busca de tarefas;
- Busca por tags ;

- Busca por comparação de string com a descrição da tarefa;

Notificações

- Disparar uma notificação usando a api do Chrome para lembrete de evento, usando o campo que define os minutos do lembrete;

Expectativa e Desafios

- Implemente - sempre que possível - conceitos mínimos de padrões de projetos;
- Montar uma solução com persistência e separação de serviços. Nossa sugestão é usar Docker com um docker-compose.yml no repositório.
- Scripts de inicialização de banco de dados com/sem seed (caso o MVC não ofereça). Não é obrigatório que seja relacional;
- Buscas podem ser feitas de inúmeras maneiras: sistemas de arquivos indexados

---

Nível Sênior

Para os monstros! Que tal além de uma busca de tags e notificações, adicionarmos também um dashboard e camadas se segurança e escalabilidade?

Leve em consideração as adições do Nível Pleno!

Validação de API

- Toda requisição via API deverá enviar um JWT validado por um middleware. Você pode forçar um login de um usuário na aplicação ou fornecer usuário e senha no **readme.md.**

Dashboard

- Gráficos de resoluções de tarefas;
- Gráfico de tags com maior número de tarefas resolvidas;

Escalabilidade

- Sua arquitetura tem que implementar um sistema com várias máquinas servido ao mesmo IP com distribuição de requests para as várias instâncias e a sessão dos usuários deve persistir independentemente da instância daquela request. Já configurou Load Balancer antes? Sessão não pode ficar só na máquina, tem vários outros drivers para sessão além de cockies, temp files, etc!
- Sua aplicação tem que rodar na porta 80 para garantir fácil usabilidade! Sugestão de configurar um proxy reverso!

- Sabemos que banco de dados relacionais têm problemas com escalabilidade. Caso ele caia a aplicação muito provavelmente não resistirá! Já configurou uma Read Replica?

Expectativa e Desafios

- Implemente - sempre que possível - conceitos mínimos de padrões de projetos;
- Montar uma solução com persistência e separação de serviços. Nossa sugestão é usar Docker com um docker-compose.yml no repositório.
- Scripts de inicialização de banco de dados com/sem seed (caso o MVC não ofereça). Não é obrigatório que seja relacional;

Envio da Solução

Passos

1. Caso ainda não tenha uma conta no Github, crie! Pra ontem! Vai!
2. Crie um repositório para o seu projeto com o nome **desafio-vitta!** Sim, a padronização do nome nos ajuda a verificar se houve cópia!
3. Responda o e-mail do desafio com o link para o repositório!
4. Qualquer dúvida enviar para filipe.forattini@vitta.me com o assunto “Desafio VITTA | Dúvida | Seu Nome Aqui”;

Equipe de Engenharia da Vitta!