# *FRAMEWORKS FRONT END*

## Proposta de Resolução

**Autoria:** Anderson da Silva Marcolino

**Leitura crítica:** Paulo Henrique Santini

# Proposta de Resolução

Para criarmos a nossa demanda, criando três equipes para cada um dos *frameworks front end*, recomenda-se iniciar pela criação de um quadro que apresenta as especificidades de cada uma destas tecnologias. É o que foi construído no Quadro 2.

**Quadro 2 – Especificidades dos *frameworks front end.***

| ***Framework*** | **Especificidades técnicas** | **Observações** |
|---|---|---|
| Vue.js. | Padronização para desenvolvimento, fórum para dúvidas, tamanho intermediário da biblioteca, renderização de projetos é rápida, linguagem JavaScript. | Menos conhecido, pode dificultar o processo de aprendizagem, além de ter uma documentação predominantemente em inglês. |
| Angular. | Não segue um padrão específico, não possui fórum oficial da comunidade, maior tamanho da biblioteca, renderização de projetos é mais lenta, linguagem TypeScript e JavaScript, melhor divisão dos arquivos do projeto. | Documentação vasta, mas a quantidade de versões e mesmo a existência do Angular 1 podem gerar confusão no momento de se estudar. |
| React. | Padronização para desenvolvimento, fórum para dúvidas, menor tamanho da biblioteca, renderização de projetos é rápida, linguagem JavaScript, JSX. | Documentação ampla e boa documentação em português, porém, o *framework* não possui boa divisão dos arquivos de seus componentes, o que dificulta seu uso. |

Fonte: elaborado pelo autor.

Tendo como base o Quadro 2, podemos criar conjuntos um conjunto inicial de recursos humanos para cada uma das tecnologias:

a) Vue.js: desenvolvedor (identificador 1), design (identificador 3) e analista de qualidade (identificador 4).
b) Angular: design (identificador 3) e analista de qualidade (identificador 4), e desenvolvedor/ design (identificador 5).
c) React: desenvolvedor (identificador 1), design (identificador 3), analista de qualidade (identificador 4) e desenvolvedor/design (identificador 5).

Com base nesta seleção, podemos descrever a justificativa para cada seleção e permitir um refinamento de cada equipe. Para facilitar a visualização, um quadro pode ser criado, como o Quadro 3.

**Quadro 3 – Membros por *frameworks front end.***

| Membro | Vue.js | Angular | React | Justificativa |
|---|---|---|---|---|
| Desenvolvedor (1). | Sim. | Não. | Sim. | Considerando que o Angular apresenta uma divisão maior dos arquivos e a curva de aprendizagem pode ser maior. Além de que React apresenta conceitos diferentes de separação de responsabilidades, por meio do JSX e que o desenvolvedor 1 não ter conhecimento em JavaScript e TypeScript, justifica-se sua alocação ao Vue.js e React, pois ambos os *frameworks* são em JavaScript, o que pode dinamizar o processo de aprendizagem para o mesmo. |
| Desenvolvedor (2). | Não. | Não. | Sim. | A decisão principal da alocação do desenvolvedor 2 é a sua dificuldade em aprender novas tecnologias. Como Angular possui várias versões e o Vue.js possui sua documentação em grande parte em inglês e mais restrita, indica-se a alocação da equipe de React. Adicionalmente, os conhecimentos do mesmo o tornará apto a trabalhar com o *framework* mais rapidamente. |
| Design (3). | Sim. | Sim. | Sim. | Como o design é peça-chave na criação de interfaces e todos os *frameworks* são de *front end*, alocou-se o mesmo para as três opções. Um ponto importante a se destacar é que não foi considerada a carga horária de cada recurso. Neste caso, recomenda-se a alocação do mesmo em dois *frameworks* apenas, e o desenvolvedor/ design (5) na tecnologia em que o design (3) não tivesse sido alocado. |
| Analista de qualidade (4). | Sim. | Sim. | Sim. | Considerando que o analista de qualidade é o responsável por realizar testes de caixa preta, o que é pertinente para o contexto de projetos *front end*, este foi alocado em todas as equipes. |
| Desenvolvedor/ design (5). | Não. | Sim. | Sim. | Pensando estrategicamente o desenvolvedor/ design (5) foi alocado em dois projetos, com o objetivo de integrá-lo à equipe e permitir que o mesmo possa então, reduzir suas dificuldades |

| | | | | |
|---|---|---|---|---|
| | | | | de trabalho em equipe. Adicionalmente, o mesmo possui pouco conhecimento em JavaScript, visto que sabe utilizar apenas bibliotecas desenvolvidas em tal linguagem, como o Ajax e JQuery, logo, poderá se familiarizar com TypeScript no Angular e o JavaScript e JSX no React. Finalmente, por ser também design, sua participação poderá ser útil para suprir necessidades específicas na ausência do design (3). |

Fonte: elaborado pelo autor.

Com base nesta divisão e justificativas, é importante destacar algumas especificidades dos *frameworks* que não interferiram na seleção dos recursos:

- A renderização de projetos e tamanho das bibliotecas dos *frameworks* não deverão influenciar na escolha de recursos, mas na decisão de projetos de cada solução a ser desenvolvida. Para isso, uma análise junto ao cliente solicitante deverá ser realizada para definir qual tecnologia escolher.
- Fórum de comunidade é um fator importante, mas não decisivo, visto que soluções como o *Stack Overflow*, site onde desenvolvedores tiram dúvidas sobre problemas no desenvolvimento, são ferramentas que concentram vários questionamentos e respostas à dúvidas.

Assim, um dos fatores mais relevantes foram os recursos que possibilitam uma aprendizagem mais rápida, somada ao conhecimento prévio dos desenvolvedores. Desse modo, temos uma divisão realista considerando nossos recursos e os *frameworks.*

**Bons estudos!**