Teste Front-end - Parte 2 – Frameworks

1 - Com quais frameworks para criação de SPAs você já trabalhou? Qual foi o último que você usou e em qual tipo de projeto?

Do final do ano passado para cá estou estudando VueJS (uma derivação do React) e AngularJS. Já tive muita prática com JQuery e Ajax e alguma coisa de SPA eu implementava sem auxílio de framework algum, programava no JS puro com auxílio de PHP. Nos últimos anos estive um pouco afastado do frontend pois estava estudando Jogos Digitais e porém agora que eu estou voltando ao Frontend pois o mercado de jogos aqui no Brasil ainda é muito pequeno. Vi forte evolução dos Frameworks Javascript e estou procurando seguir esta evolução. Trabalho com Javascript faz 20 anos e não estou tendo dificuldade alguma nestas tecnologias, aliás para mim é bem claro seu uso e para que servem. Não tenho projetos utilizando frameworks modernos, apenas material de estudo.

2 - Sobre a utilização de SPA e do framework citado no projeto acima, como você vê que SPA e o framework utilizado agregaram para o projeto? De maneira geral, em sua opinião, quais as vantagens de uma abordagem SPA e quando é melhor evitar?

A utilização de SPA é fundamental para melhorarmos a fluidez da navegação evitando a troca de páginas reaproveitando componentes que já foram carregados e diminuindo o número de requisições em um servidor, de modo geral um servidor permite à uma página um número pequeno de requisições de pacotes por um mesmo número IP. Mesmo que o tamanho do pacote seja pequeno o servidor limita suas requisições tornando a fluidez da navegação um problema. Em uma SPA, teríamos o site como uma aplicação aonde todos os componentes necessários para a visualização da página seriam pré-carregados e funcionariam de forma dinâmica fazendo consultas ao servidor a partir da interação do usuário com estes componentes. Imagens, elementos html, estilos e código não precisariam ser carregados a cada troca de página apenas o conteúdo é requisitado e alterado via DOM. De certa forma isto realmente causa grande economia porém em sistemas mais complexos, com muitos módulos, carregar uma SAP é um desperdício de uso dos recursos do cliente. Daí seria melhor, independentemente do framework utilizado, carregar a página sob demanda, lazy loading, disponibilizando partes da aplicação conforme ela é requisitada pelo usuário. Exemplo: Em uma aplicação de streaming de vídeo, uma função “playerVídeo()” poderia ser implementada apenas após o usuário de fato tenha iniciado a exibição de um vídeo.

3 - Caso no item 1 não tenha citado ReactJS: Já trabalhou com ReactJS? Como era o projeto e como o ReactJS agregou?

Minhas experiências com páginas reativas eu tenho implementado em VueJs em páginas CRUD ou formulários. A grande vantagem de páginas reativas é que a interação do usuário seja por mouse ou por teclado é armazenada e pode ser utilizada em tempo real em uma página apenas. Páginas reativas também reagem a eventos disparados automaticamente ou por interação do usuário e estes eventos podem ser manipulados para melhorar a interação do usuário com a página.

4 - Já trabalhou com SSR? Qual o problema você visava solucionar com SSR? Qual framework você utilizou para fazer? Como era o último projeto que utilizou com SSR?

Não mas conheço o problema à ser resolvido, o maior deles é a questão do SEO pois o estado aplicação exibida no browser possui um conteúdo que é construído dinamicamente durante a interação do usuário impossibilitando os robôs de busca indexarem este conteúdo de forma eficaz, não é possível indexar o estado desta página. Atualmente o GoogleBot consegue, pagando né, mas na busca natural, aquela por relevância, isto não é possível. Além disto, de brinde, ganhamos performance pois o conteúdo é entregue ao cliente pré-renderizado.

5 - Já trabalhou com as ferramentas ESLint e Prettier? O que você pensa sobre estas ferramentas? Em quais projetos você acredita ser importante usar e em quais não?

São formatadores de código. A utilização deles é importante pois mantém o código limpo e legível, facilita a manutenção e o teste. Geralmente utilizo formatadores de código existentes nos próprios SDK´s, o Angular CLI faz um Lint através do ng lint. O ideal é utilizá-los em todos o projeto para garantir a integridade dos dados que são negociados entre as camadas do programa.

6 - Há indícios de que uma parte da aplicação ou um componente específico esteja executando renderizações excessivas ou usando mais processamento do que deveria... Quais os procedimentos que você adotaria para diagnosticar e averiguar a situação?

Eu começaria pelos logs e eventos disparados no console do browser, caso não fosse possível descobrir eu faria testes unitários nos últimos serviços atualizados ou criados.

7 - Qual foi a última tecnologia, tópico ou conceito que você aprendeu relacionado a front-end? Quais os problemas que você visa solucionar ou solucionou com este aprendizado?

AngularJS e VueJS. O problema inicialmente foi me ambientar com o Typescript e o ECMA 6, a parte lógica não é problema mas toda mudança de sintaxe dá um pouco de trabalho, já possuía experiência com MVC mas o entendimento de como podemos implementar os frameworks JS nestes projetos e como estes manipulam o DOM foram, não digo problemas, mas passos naturais do aprendizado. Agora voltando à tecnologias mais antigas os principais problemas que tento resolver são problemas de performance e desempenho no lado do cliente, no servidor o ambiente é controlado, mas no cliente não. Geralmente trabalhamos em computadores “parrudos” e os resultados vistos por nós nem sempre é o mesmo que conseguimos em hardware mais modestos. Quando trabalhei com frontend em Actionscript isto acontecia, quando trabalhei com players de vídeo isto acontecia, jogos em Html5 e Javascript isto também acontecia. Um bom código, refatorando iterações, reformulando recursos, garantem uma melhoria contínua do código seja qual for a linguagem.