Part I – Tech Assessment

Sendo o objetivo demonstrar aos clientes um mapa com algum tipo de input de partida e destino, é fulcral que se comece por delinear uma plataforma ou framework que possa providenciar um sistema de mapas, mas também outras tools como criação de percursos, procura de pontos de interesse e geocoding de strings. Posto isto, recomendaria a utilização do Mapbox GL JS para criar o mapa de vetores e as APIs também da Mapbox que contém estas outras funcionalidades referidas anteriormente, mais sobre as mesmas mais a frente no ficheiro.

A framework também contém inúmeros plugins que podem ser adicionados de modo a tornarem possível a utilização de outras frameworks de frontend como React, Angular ou Vue.js. Para fins de organização deste texto escolherei a utilização de React com Mapbox GL JS tornado possível com o plugin react-mapbox-gl (<https://github.com/alex3165/react-mapbox-gl>).

Nota: O Mapbox GL, embora seja open-source tem um limite antes de começar a ser pago, porém continuo a achar uma das melhores hipóteses e alternativas às soluções da Google ou outras libraries e APIs open-source como a OpenRouteService.

Dividindo o projeto em Back e Frontend, a stack tecnológica seria a seguinte:

Frontend – React, Mapbox GL JS

Backend (REST API, DB) – Node.js, Express, MySQL

Começando pela Frontend não é necessário fazer o load do mapa mal o cliente entra em contacto com a web app, portanto de certa forma podemos dividir o software em camadas de acordo com a utilização do cliente. A primeira view seria um prompt de percurso, onde o cliente se depara com um form com dois campos, partida e chegada. Após submeter o formulário, passamos para a segunda view, os dados do formulário são enviados para a API que responderá com até 3 percursos diferentes (limite imposto pela Mapbox Directions API) e uma lista de highlights (pontos de interesse recebidos por outro serviço da Mapbox, Mapbox Tilequery API, selecionando um número fixo de POIs (points of interest ou pontos de interesse), visto que não existe qualquer hierarquia na lista proveniente do request à API). O cliente pode então selecionar um dos percursos que modificará a view existente, deixando apenas o percurso selecionado visível acompanhado de todos os POIs presentes na resposta da API, representados por popups que ao serem clicados contêm o resto da informação recebida da API contida no markup HTML do popup.

Teremos 2 views, uma estática apenas com o formulário inicial, outra dinâmica onde os 3 percursos aparecem primeiro e após seleção de um os outros dois desaparecem e o mapa foca no percurso selecionado.

Em termos de métodos, na primeira view, ou seja, o formulário de partida e chegada, apenas é necessário um método para quando o formulário for submetido. Este método deverá enviar um GET request para a backend de onde receberá a resposta de 3 possíveis percursos e uma lista de alguns pontos de interesse que se possam encontrar em cada percurso (utilizando apenas algumas coordenadas do percurso em vez de todas as recebidas na resposta da call ao Directions API), esta informação irá então ser passada à próxima view mudando o layout da pagina para a view com o mapa, adicionando uma layer ao mapa com as routes recebidas.

Na segunda view, o mapa com os percursos possíveis e lista de highlights aparece e necessitamos de 3 métodos, um de seleção de caminho, outro para adicionar os POIs como layer ao mapa, e um ultimo para criar os popups quando o utilizador clica no ícone de um POI.

Na seleção de caminho deve ser feita uma nova call à API de Tilequery desta vez com todas as coordenadas existentes no percurso (estas coordenadas serão as coordenadas de pontos de viragem ou faseadas ao longo do percurso), estas deverão ser adicionadas ao mapa como uma layer tal como o percurso anteriormente, tendo todas um ícone atribuído.

Por ultimo, ao clicar num ícone de um POI um popup será criado e adicionado ao mapa em cima do ícone, de modo a demonstrar alguma informação sobre o ponto de interesse como o tipo de ponto, ou nome da localização.

Na Backend teríamos duas preocupações, fazer as calls necessárias aos serviços do Mapbox, e possivelmente procurar certa informação contida na base de dados. A única questão seria se a Indie Campers teria dados de POIs pessoais que gostaria de utilizar. Portanto, vou propor a criação da aplicação apenas usando APIs externas mas também o possível modelo de dados se houvessem dados internos de POIs.