Desafio Angular

Você é responsável por criar uma aplicação de um Planning Poker em Angular!

O Planning Poker é uma ferramenta para estimar tempo e esforço necessário para realizar uma tarefa. Basicamente cada pessoa da equipe de desenvolvimento recebe um conjunto de cartas com diferentes valores em determinada escala, como a de Fibonacci. Para estimar uma tarefa, a equipe conversa sobre a tarefa e cada pessoa da equipe seleciona uma carta, de forma privada, representando a estimativa que ela daria. Depois que todos escolhem, os votos são revelados para toda a equipe. Se todos os votos forem iguais, esta será a estimativa da tarefa. Senão, a equipe conversa sobre os votos discrepantes, e, se necessário, fazem uma nova votação até entrarem em um consenso. Mais informações: <https://www.voitto.com.br/blog/artigo/planning-poker>. Exemplo de aplicação: [https://www.planitpoker.com](https://www.planitpoker.com/).

Requisitos:

A aplicação deve permitir:

* Cadastros:
  + Usuários - Nome, Dono (booleano);
  + História de Usuário - Descrição;
  + Cartas - Valor da carta (numérico);
  + Votos - Id do usuário, Id da carta, Id da história
  + Sessão da Planning – Descrição
* A visualização se todos usuários da sessão de uma planning já votaram em determinada história;
* A visualização de todos os votos de uma determinada história, inclusive a média dos votos;
* Visualização das histórias ativas (que ainda não foram votadas), completadas (que já foram votadas) e do conjunto de todas as histórias (não votadas + votadas);
* Zerar (exclui) os votos de determinada história.

Requisitos não funcionais:

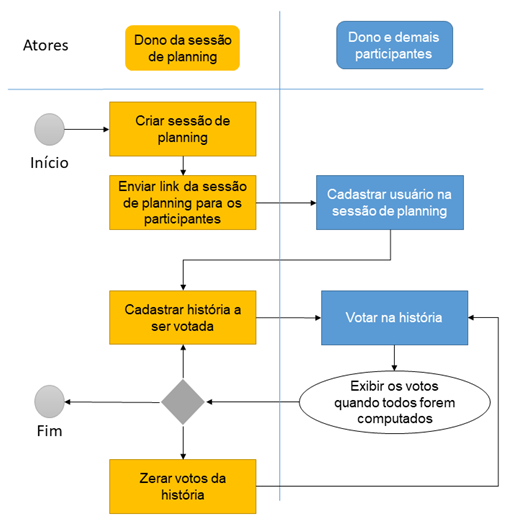
* Utilizar a versão 9 ou superior do Angular;
* Os dados devem ser mocks usando, por exemplo, o Local Storage para armazenamento. Outra opção é utilizar o pacote json-server como backend. Mais informações:
  + [Local Storage;](https://www.youtube.com/watch?v=AphaUpmVguI&ab_channel=LuisTavares)
  + [json-server.](https://www.youtube.com/watch?v=-LF_TOnaH4Y&ab_channel=F%C3%A1bricadeC%C3%B3digo)

Observações importantes:

* Todo o código deverá ser publicado em um repositório público do GitHub;
* Atente-se para ter um código que utilize padrões de desenvolvimento e boas práticas.

Diagrama de Fluxo

Fluxo de funcionamento completo da aplicação (integrando Frontend e Backend):



Etapas de entrega (Releases)

Release 1:

Cadastro de usuários, história de usuário, cartas, sessão da planning e dos votos, ou seja, o primeiro requisito (pode ser entregue apenas em HTML, CSS e JS, não sendo necessário fazer em Angular, embora seja recomendado).

Release 2:

Visualização se todos os usuários já votaram e de todos os votos de uma determinada história de usuário, ou seja, o segundo e terceiro requisito (deve ser entregue em Angular).

Release 3:

Visualização de todas as histórias, incluindo as ativas e completadas, e possibilidade de zerar os votos computados a uma determinada história de usuário, ou seja, o quarto e quinto requisito (deve ser entregue em Angular).