Simple Browser Based Game

Frontend Technical Assignment

Hello, player. The objectives are not simple and will require several days (hopefully not full) of your time to complete. It is however intended for developers that have their toolsets and framework in hand and do not need to implement/learn from scratch.

Objective One - The Game

Create a small browser based game using Angular and the Golang server provided. The game should highlight your skills as a Frontend dev. We are not just looking for clean framework and service code (although essential), but also a flair for frontend implementation and design. We want the app to sing - We are a consumer facing games company so this is essential. A backend Golang server is provided for you to interface with.

Objective Two - The Test

Create at least one integration test that we can use to integrate into our CICD pipeline. The objective here is to show us that you understand and can implement a decent structure for providing our QA a scaffold to build and maintain integration tests on the app. Unit and component tests will also be awarded points, but are not the primary objective of this task.

Objective Three - Feedback

Please provide feedback to justify your implementation; highlight where you feel you shined; explain where you fell short; highlight where we fell short (bugs/implementation); This is our first front-end test, so any feedback at all is really appreciated.

Game Mechanics

When you choose to join the game you will need to provide a name (display name) and pick two numbers between 1 and 10 inclusive. It is these details that shall be used to calculate the players score each round and display them on a leaderboard.

When at least 2 players have joined the game it will start to countdown from ten to game start.

A game lasts for a maximum of 10 rounds with roughly 1 second in between each round. For each round the server generates a random number (between 1 and 10 inclusive), and calculates the score for each player.

The winner is the player with the highest score at the end of 30 rounds or any player that scores exactly 21 points at the end of round.

In case of a tie the player with the higher upper bound wins, followed by the lower bound and finally alphabetical order. For frontend development you can assume that the server will always return a single winner to the game, and not an array.

Interface

The main interface for the game should be a leaderboard with the highest scoring player on top and lowest scoring at the bottom. The leaderboard should update in the browser without the players or observers needing to interact with the game.

When a game ends the winner should be showcased and new players can join before the next game starts. The time between games is set to ten seconds.

If a player attempts to join in the middle of the game they have to wait until the end of the current game and then are automatically joined for the next game. Until that point, they should remain as observers to the current game.

Constraints & Requirements

- Frontend implementation should be done using AngularJS
- No authentication is required
- Games do not persist between server restarts, so you only need to worry about the current game
- The game should operate correctly on the latest version of Chrome
- A self contained and concise readme file should be included that details on how to:
    - build, deploy and run the game (without fail). We suggest building into the current docker-compose to ensure both server and client launch
    - build ,deploy and run an integration test
- Games that do not run are not marked or graded, sorry. Best to start here!

SBBG Server

A zip file should be provided with this document that contains the golang server. The server contains its own README.md file that describes its setup and the API. You need this server in order to complete the assignment.