Fullstack Engineer Test

Lobby Wars

We are in the era of "lawsuits", everyone wants to go to court with their lawyer Saul and try to get a lot of dollars as if they were raining over Manhattan

The laws have changed much lately and governments have been digitized. That's when **Signaturit** comes into play.

The city council through the use of [Signaturit.com] (https://www.signaturit.com/en) maintains a registry of legal signatures of each party involved in the contracts that are made.

During a trial the justice only verifies the signatures of the parties involved in the contract to decide who wins. For that, they assign points to the different firms depending on their signers roles.

For example, if the plaintiff has a contract that is signed by a **notary** he gets 2 points, if the defendant has in the contract the signature of only a **validator** he gets only 1 point, so the plaintiff party wins the trial.

We want you to automate this process, given a contract with your 2 parties involved and their signatures and indicate which one wins the test

Roles

- K - King - 5 points
- N - Notary - 2 points
- V - Validator - 1 point

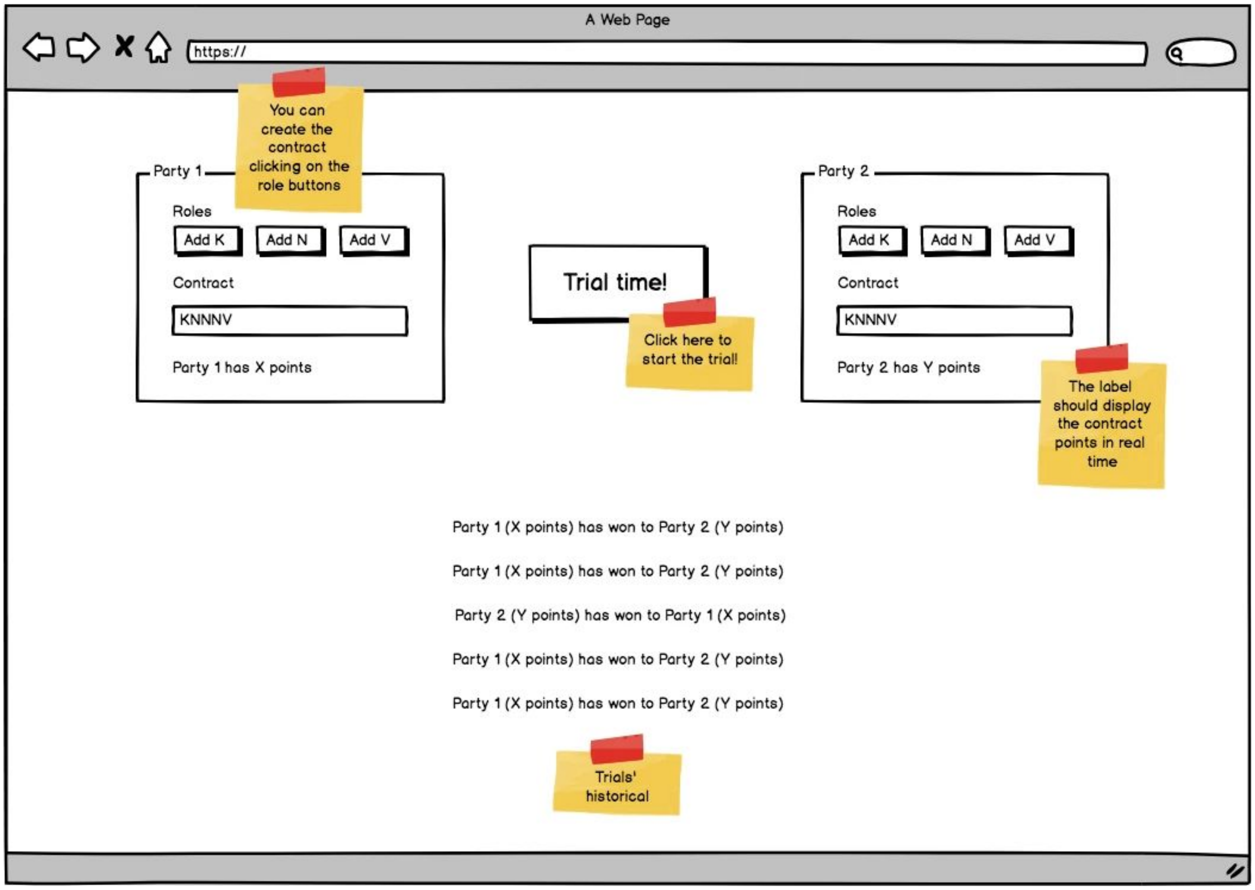
Keep in mind that when a King signs, the signatures of the validators on his part have no value.

First Phase

Make a REST API endpoint that accepts two contracts in the **KN** vs **NNV** format and returns the winner of the trial.

Second stage

We want you to create a web app connected to the API that you have created on the first phase. It should allow you to create contracts and display the results of the trials. You will find some ideas on the following wireframe:



You can create the contract clicking on the role buttons

Click here to start the trial!

The label should display the contract points in real time

Party 1 (X points) has won to Party 2 (Y points)

Party 1 (X points) has won to Party 2 (Y points)

Party 2 (Y points) has won to Party 1 (X points)

Party 1 (X points) has won to Party 2 (Y points)

Party 1 (X points) has won to Party 2 (Y points)

Trials' historical

We will take into account the use of:

- Preprocessors
- ES6/7 or TypeScript
- Mobile-first (offline first, it’s a plus)
- Do not use CSS frameworks that offers ready made components like Bootstrap, Bulma, Material, Foundation, etc (albeit you’re allowed to use frameworks like Susy, Compass)
- Apply best practises and quality code.