

**Tech Slot**

**Chess Quality Assurance Session**

**We would like to invite you to demonstrate your skills in quality assurance by creating a test suite and writing some e2e and unit tests for a small chess application**

You will be provided with an application which:

* ~~Allows two people to play a chess game on a chess board.~~
* ~~White and black should be allowed to make moves in turns, none of them can make 2 or more consecutive moves~~
* ~~Validate Only one move per player is allowed~~
* ~~The moves should be transferred to the back-end and saved in the database – - Assert/Validate the Backend Moves~~
* ~~Only~~ **~~valid~~** ~~moves should be stored in the database.~~
* ~~Assert Valid Moves and Assert/Should not have invalid moves in DB~~

Valid are all moves with own piece that satisfy the below  
Piece is allowed to move:

* on the board – Assert Piece Movement with in the board
* ~~to a free square – Assert movement to any position (except it’s own)~~
* ~~to a square where opponent’s piece is placed – Assert movement in opposite place~~

Piece is **not** allowed to move

* ~~to a square where own piece is placed~~
* ~~to a square where a king is placed~~

The application is written in

* Vue.js with typescript and ‘vue-property-decorator’ for the front-end part
* Node.js, typescript + express for the server side
* MongoDB + mongoose for storing the data

**The Task**

1. **Create a test suite describing the test cases that should be created to ensure the quality of the application – Done**
2. **Write any unit tests that you think are necessary using any preferred by you testing framework among mocha, jest or jasmine - Done**
3. **Write any end-to-end tests that you think are necessary using selenium web driver + a preferred language of yours –**
   * + For the configuration, a file
     + /src/config.ts file is offered for the frontend, for the backend the file backend/.env

The frontend config contains the uris to the REST API of the backend as well as the uri to the WebSocket Server

export const CONFIG\_API = 'http://localhost:3000/api/lastsession'

export const CONFIG\_WS = 'ws://127.0.0.1:3030'

```

The backend config contains the corresponding configuration as well as the connection uri to MongoDB

LISTEN\_PORT=3000

SOCKET\_PORT=3030

LOG\_LEVEL=2

MONGO\_URI=mongodb://127.0.0.1:27017

**Browser 1:** Player 1 : Launch the chess app – Validates the pre conditions - Make it Reusable **Script1**

* + Validate the Chess24/ class .logo - Validated
  + Validate Start New Game/ tagname button
  + Validate Chess board/ Class .chessboard

Player 1: Click Start new Game

* + Assert Wait for opponent to join.
  + Assert Quit game
  + Click Quit Game
  + Assert DOM should not contain Wait for opponent to join
  + Assert DOM should not contain Quit game
  + Assert DOM should Contain Start New Game
  + Assert that no Pieces are movable

**Browser 2:** Player 2: Launch the chess app – Validates the pre conditions - Make it Reusable **Script2**

* + Validate the Chess24/ class .logo
  + Validate Join a Game/ tagname button
  + Validate Chess board/ Class .chessboard

Player 2: Click Join a Game

* + Assert - It is your opponent's turn. You play the black pieces. (1)
  + Assert Quit Game
  + Assert that no pieces are movable in Board

**Browser 1:** Player 1

* + Assert - It's is your turn. You play the white pieces.
  + Assert - Black keys are not movable
  + Action-Move White key /Soldier before Queen 2 steps forward

- Assert the instance stored in DB backend

* + Assert - It is your opponent's turn. You play the White pieces. (2)
  + Action- try to move the moved key back

- Assert user cannot move the key back.

**Browser 2:** Player 2

* + Assert - It's is your turn. You play the Black pieces.
  + Assert - White keys are not movable
  + Action - Move any Black key/Soldier before king 2 steps forward

- Assert the instance is not stored in DB backend

* + Assert - It is your opponent's turn. You play the Black pieces. (2)
  + Action- try to move the moved key back

- Assert user cannot move the key back.

**Browser 1:** Player 1

* + Assert - It's is your turn. You play the white pieces.
  + Assert - Black keys are not movable
  + Action-Move same White key /Soldier and **cut Black Soldier**

- Assert the instance stored in DB backend and **Black soldier is not available in Board**

* + Assert - It is your opponent's turn. You play the White pieces. (2)
  + Action- try to move the moved key back

- Assert user cannot move the key back.

**Browser 2:** Player 2

* + Assert - It's is your turn. You play the Black pieces.
  + Assert - White keys are not movable
  + Action - Move any Black key/Soldier and put in black king’s square

- Assert the instance is not stored in DB backend

- Assert Soldier not allowed to make the move in king square

* + Assert - It's is your turn. You play the Black pieces.
  + Assert - White keys are not movable
  + Assert user cannot move the key back.
  + Action Click Quit Game
  + Assert Start New Game

**Good luck!**