# Seeking Alpha React Test

Instructions:

This task should be implemented client side only, using Javascript(ES6), HTML, CSS and [React Framework](https://reactjs.org/). You can load React from CDN or use any of build systems you prefer. You are free to use any React state container if you need one, What we’re looking for in your code is readability and easy maintenance. We want to see code that reveals its intent to the reader and follows best practices. To accomplish this you can use any paradigm you want.

You also have to make sure the code really works and use any tool or technique you need to accomplish this.

The Problem:

1. You should start with a two-dimensional grid of **50x50** square *cells* and each of these cells are either *alive* or *dead*. You can grid as a **<div> or <td> with white background** showing dead cells (we use 0 for example) and living cells using **<div> or <td> with black background** (we use 1). The initial configuration of living cells in this grid is arbitrary and should be random for every page refresh. Here’s a smaller 5x5 example:

00000

00000

01110

00000

00000

2. This grid is subject to changes on what is called a *tick*. When a grid “ticks”, these are the rules to determine the next state of the grid:

* Any live cell with fewer than two live neighbours dies (underpopulation).
* Any live cell with two or three live neighbours lives on to the next generation.
* Any live cell with more than three live neighbours dies (overcrowding).
* Any dead cell with exactly three live neighbours becomes a live cell (reproduction).

Try to look at the first state of this grid and apply the rules above. The result will be the second grid, as shown below:

00000 00000

00000 00100

01110 - tick -> 00100

00000 00100

00000 00000

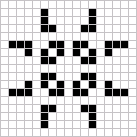
**That’s all!** Just to help you, here are images of some grids ticking every ~0.4s (white is dead and black alive). You can use them to check if your code is correct:











What we expect to receive:

1. Link to Github with fully functional development (not modified and not obfuscated) code.
2. If needed, please include a brief description of how to run the game.
3. Use Flow.js or typescript
4. Write several unit-tests (not snapshots)

Good luck!