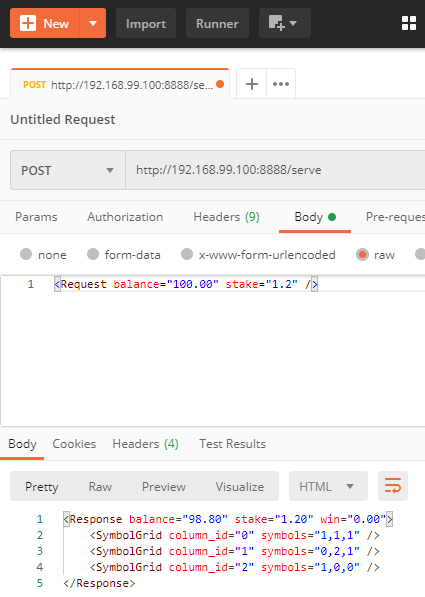
**SLOT MACHINE APP**

**Technologies:** ES6, HTML, CSS, Webpack, Babel, Docker API, Visual Studio Code, GitHub, PostMan, PixiJs, TweenJs

Download the app from github link.

[https://github.com/derik09/SlotApp](https://github.com/derik09/SlotMachineApp)

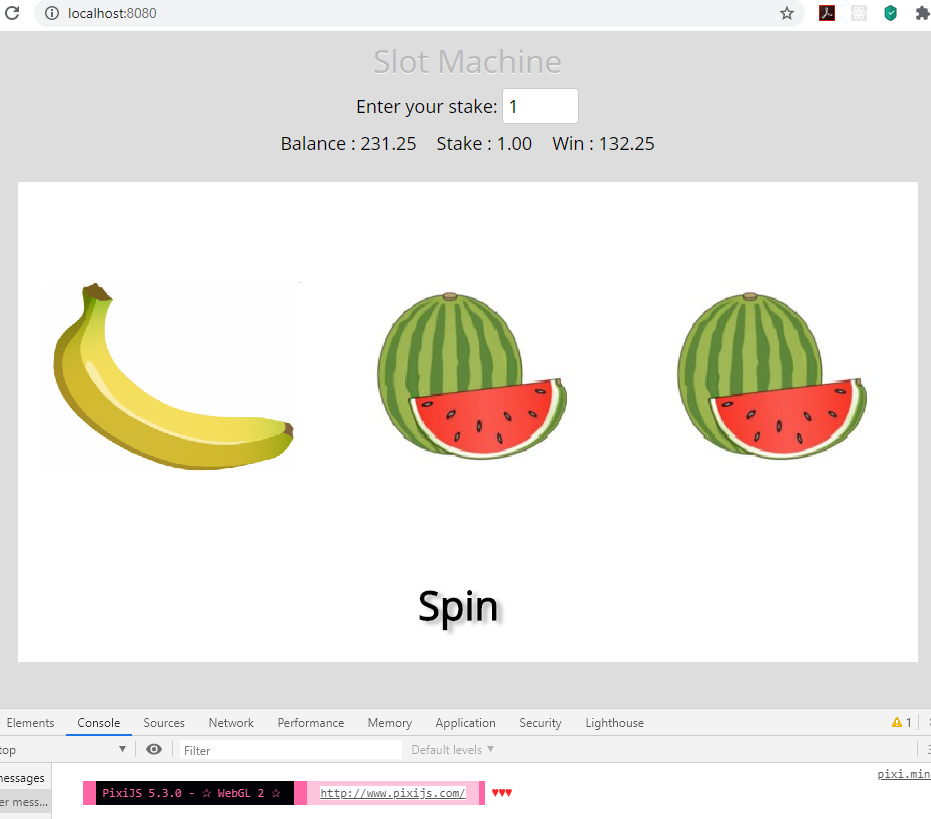
Docker API needs to be running as <http://localhost:8888/serve> or the port you are using.



Used the docker API , provided by you. On each spin the first element of each symbol grid is sliced and mapped with the fruits. For example, in above figure 1, 0, 1 is taken from each of the first three inner elements and displayed as banana, apple, banana.

Run :> npm start

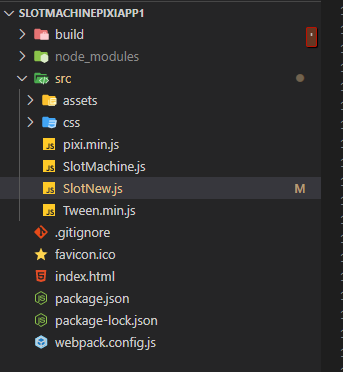
App would be built on local and could be browsed at link localhost:8080/ or whichever link appears in your node cli. I have used Pixi.js for the canvas and Tween.js for the tweening. Pixi.js is displayed in the console.



Stake amount can be changed within the enter your stake textbox. Balance is hardcoded at the moment at 100. Could modify it if required.

## **Review Criteria**

* **Modular/Code organization.**

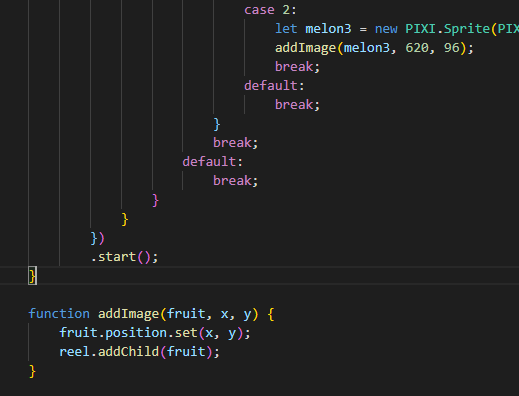


All Js files and css files are under the src folder. Have used babel and webpack to bundle all js files into es5 compatible file - build/app.bundle.js.



The Slotmachine.js is set as the entry point.

* **Generic where appropriate.**



Combined common functionality into addImage method

* **Clarity/Self documenting.**

Added comments wherever necessary.

* **Indenting.**

Developed app in Visual Studio Code. Shortcut Key => Ctrl + K + D used for indenting code.

* **Use of camel case.**

Used within the code.

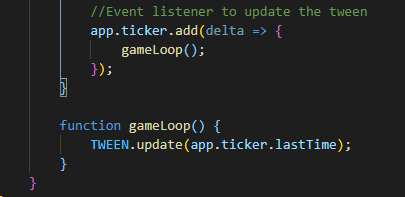
* **Demonstrate understanding of JS scope.**

Used global and local variables.

* **Optimised code.**

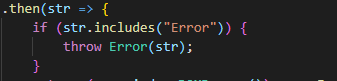
Webpack and Babel used to bundle all javascript files into backward compatible files (from es6 to es5). Since javascript is minified into app.bundle.js (es5) , javascript would work in all browsers

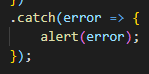
* **Use of callbacks / timers.**

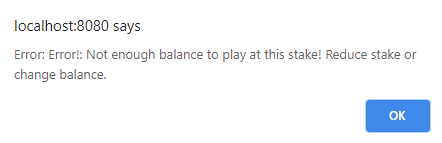


* **Error Handling.**

Error handling is maintained as below. Error from Docker web api is taken care of as well.



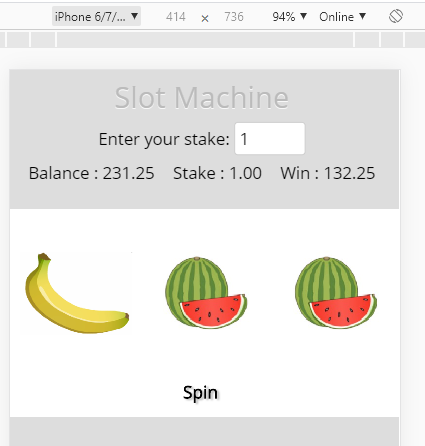


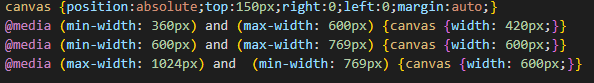


Please note, in IE browser when I make a call to the web api localhost:8888/serve the api returns with below error. I have enabled/disabled CORS but the issue still exists within the api response. Not quite sure how to fix this.



* **Responsive design handling for small screen**





**THANK YOU**