**Star Ratings Component**

### Document Properties

| Item | Details |
| --- | --- |
| Document | Technical Specification Document |
| Application | Anant Nerkar |
| Author | Anant Nerkar |
| Creation Date | 20/9/2020 |
| Last Updated | 24/9/2020 |
| Status | Completed |

Contents

[Document Properties 1](#_Toc51418511)

[1. Component Overview 3](#_Toc51418512)

[1.1 Objective 3](#_Toc51418513)

[1.2 Use Cases 3](#_Toc51418514)

[1.3 Out of Scope 3](#_Toc51418516)

[2. Technical Specifications 3](#_Toc51418517)

[2.1 NPM details 3](#_Toc51418518)

[2.2 NPM repo 3](#_Toc51418519)

[3.Unit Testing 3](#_Toc51418520)

[4. Assumptions 3](#_Toc51418521)

[5. Concerns and Issues 3](#_Toc51418522)

**1. Component Overview**

The star component is used to show users the quality of hotels for them to make well-informed decisions. The initial purpose was of informing travelers on basic facilities that can be expected, but now the objectives of hotel ratings have expanded to focus on the hotel experience.



This is how user can see the hotel ratings based on the user experience.

**1.1 Objective**

The objective of this project is to create a project having a strategy where all your multiple projects are stored in the single repository instead of handling individual repositories.

Here, in this repository we have created multiple projects, or we can say web components that can be used by installing its package from the registry.

**1.2 Use Cases**

The star rating is used to show users the quality of service to be expected from a hotel.

Following are the different scenarios where a star rating is used:

1. Component is used to show the class and quality of the selected hotel.
2. Component is used to show the overall experience of the users.

**1.3 Out of Scope**

Scope of filtering the hotels along with checkboxes to help the users filter hotels of their desired star-ratings.

**2. Technical Specifications**

If you plan to publish your package, the *most* important things in your package.json are the name and version fields as they will be required. The name and version together form an identifier that is assumed to be completely unique. Changes to the package should come along with changes to the version. If you don’t plan to publish your package, the name and version fields are optional. The name is what your thing is called.

Some rules for package name:

* The name must be less than or equal to 214 characters. This includes the scope for scoped packages.
* The name can’t start with a dot or an underscore.
* New packages must not have uppercase letters in the name.
* The name ends up being part of a URL, an argument on the command line, and a folder name. Therefore, the name can’t contain any non-URL-safe characters.

About ratings component, we have created a component using attribute named as “rating-value”. User can use this component inside another component by installing its package using **“npm install @my-first-project/ratings”** and then importing its package inside the component using **“import ‘@my-first-project/ratings’** **”**

By default, this component will render 5 stars with default color and user has to provide the rating-value to use this component like this way:

<orxe-ratings rating-value="4"></orxe-ratings>

Based on the rating-value it will render the 4 stars with color filled and 1 star with default color.

Ratings component CSS variables are created in ratings-map.scss file and in that we have used the CSS variables names are used from the global css file from below path:

[**https://components.qa.cnxloyalty.com/theme/css/imports.css**](https://components.qa.cnxloyalty.com/theme/css/imports.css)

**2.1 NPM details**

Following packages are installed for project:

“lit-element” – “^2.2.1”

“lit-html” – “^1.1.2”

Following are the command for creating and running the component:

* Installing orxe/cli - npm install -g @orxe3/cli
* Create a new component workspace - orxe new component-workspace my-first-project
* Create a component – orxe g c star-ratings
* Serving the component – orxe serve
  1. **NPM repo**

**https://github.com/anantnerkar/star-ratings-component**

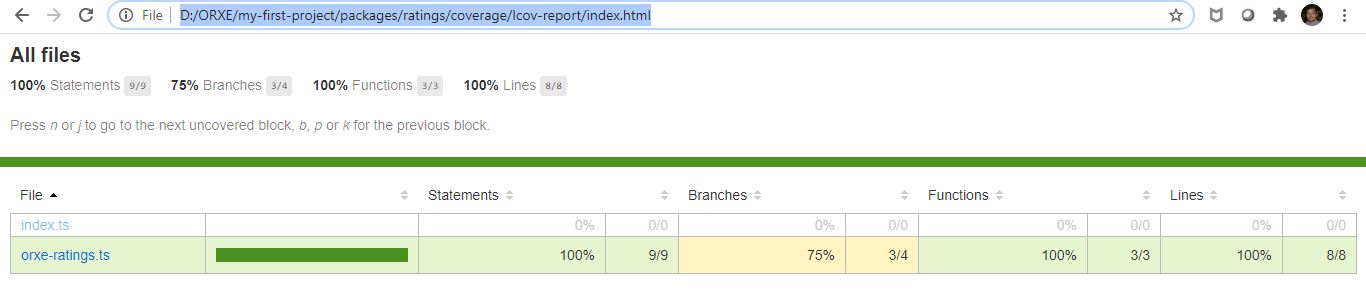
**3.Unit Testing**

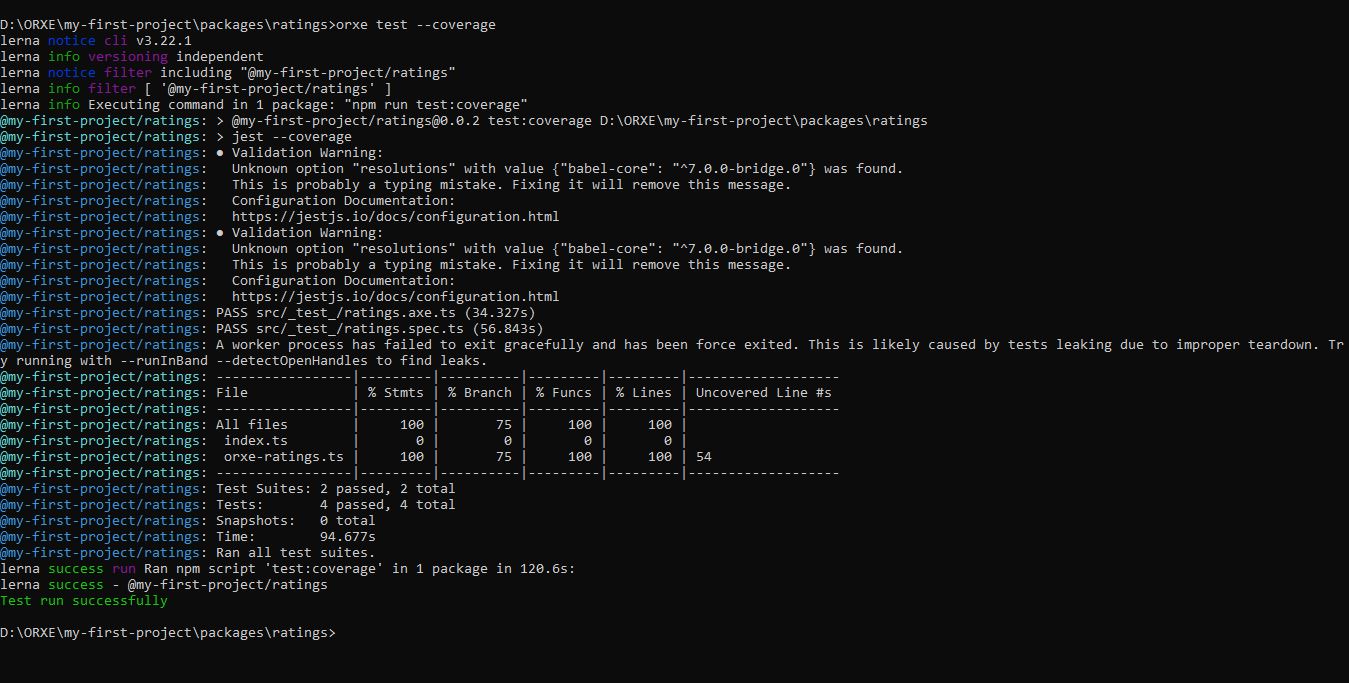
The test cases written for orxe-ratings component it consists of two test suites. One is default ratings.axe file and another ratings.spec file. All function and line coverage are done with spec file.

All test cases of orxe-ratings component are written in ratings.axe.ts and ratings.spec.ts file. Test cases are available below.

1. Should check default attribute.
2. Should function render is call.

The unit test cases coverage report is available below.





**4. Assumptions**

This component is developed based on the design guidelines provided by Tavisca on the confluence page.

https://tavisca.atlassian.net/wiki/spaces/ORXE3/pages/780665440/Star+Rating+0.3

**5. Concerns and Issues**

1. Issue with loading icon from the orxe repository. I tried to use <orxe-icon icon=”ic-star-filled”></orxe-icon> it does not load the icon when I open it. So, instead I used the icon as “ic-star” for now. And highlight it based on the ratings value.

2. Also, as per the design guidelines, I tried to fit all the 5 stars inside 88px of container size including 2px of margin between each star. However, when I set this the last star goes on the next line and to fix that, I have increased the size of the container.