**# Checklist for NC Games Front End**

​

**## README - write your own and make sure that it:**

​

- [ ] has a link to the deployed version

- [ ] provides general info about your app

- [ ] includes links to your back end repo

- [ ] specifies the minimum version of Node required to run locally (check your Node version, `node --version` and use the major version that you are on)

- [ ] has clear instructions on how to run your project locally (`git clone <repo-url>, cd ...`)

​

**## UX**

​

- [x] Basic styling added

- [x] Responsive design

- [x] Items aligned

- [ ] Content legible (not too wide, obstructed, etc)

- Could limit the width a little on your page content - have a look at reddit as an example of this! The `max-width` CSS attribute may help.

- [ ] Refreshing doesn’t cause an issue on sub-pages

- [ ] No errors in the console

- A few references to react lists needing unique `key` props, and some other issues (specifically regarding trying to render a single review)

- [ ] Votes / Posts / Deletions happen instantly *\_OR\_* give user indication of loading

​

**## Functionality**

​

**### Login**

​

- [x] Some indication of who is logged in (this can be hardcoded)

​

**### Reviews**

​

- [x] Serves all reviews / top reviews

- [ ] Can vote on reviews

- I can vote on them but it seems to happen twice per click! When I come back to the page it has only gone up by 1, meaning the backend is only being hit once (which is what we want!)

- [ ] Can vote a maximum of once in either direction per page load

- [ ] Votes are persistent when page is refreshed

- [x] Reviews by category pages load only relevant reviews (especially when navigating from one category page to another)

- [ ] Can sort reviews by date created / comment\_count / votes

- Can sort happily by some of these - just not quite all!

​

**### Individual Review / Comments**

​

- [x] Individual reviews are served with comments

- [ ] Can vote on comments

- Same issue regarding voting on an article

- [ ] Can vote a maximum of once in either direction per page load

- [x] Votes are persistent when page is refreshed

- They are persistent with the single BE call (rather than the FE showing two upvotes)

- [x] Can post new comments, which are persistent

​

**### Additional functionality:**

​

- [x] Can only delete comments of logged in user

- [x] Deleted comments don’t re-appear on re-render/refresh

- [ ] sort comments by date created / votes

- [ ] navigate over pages of reviews (if implemented in back-end)

- [ ] navigate over pages of comments (if implemented in back-end)

- [ ] filter / display reviews by specific user

- [ ] post new review

- The form is there for this, but I look to be getting a 404 from the BE

- [ ] delete logged in user's reviews

​

**## Error Handling**

​

- [ ] Bad url

- [ ] Bad category slug in url

- [ ] Bad review\_id in url

- [ ] Post comment: (No text in comment body / Can you post without logging in?)

​

**## Code**

​

- [ ] Well named components

- [ ] Components reused where possible (`Reviews` / `Voter`...)

- The `CommentVote` and `ReviewVote` are fairly similar... perhaps they could be merged somehow?

- [x] Minimal state - don't hold derivable data in state

- [x] Set state correctly, using previous state where possible

- [ ] Handle asynchronicity clearly (i.e. isLoading pattern)

- Adding some loading state wherever you have to wait for an API call to come back will go a long way to improving the user experience!

- [x] Functions are DRY (`handleChange` for controlled components / api calls)

- [x] Use object destructuring where possible

- [ ] Tidy? If not: ESLint / Prettier

- [x] `node\_modules` git ignored

- [ ] No `console.log`s / comments

- [ ] remove unnecessary files (e.g. App.test.js)

​

**## General pointers**

​

**### App**

​

- When I refresh the page, I get logged out. I can see that you're using `localStorage` to keep track of the user, which is great! But it seems as though when you get back the user from `localStorage`, you're missing a call to `setLoggedInUser`. Additionally, it is struggling on initial render because in `ReviewVote` (and a couple of other places) it is trying to read `loggedInUser.username` but `loggedInUser` is `null` until it has read from `localStorage`

- Not sure if there is any need for both a `user` and `loggedInUser` in state

- This is another point for an `isLoading` pattern - it can help to prevent things that shouldn't be trying to happen before the page has properly loaded

​

**### Code**

​

- In `Categories`, no need for a react fragment `<></>` if you are only returning a single child element (as you are inside the `.map`)

- **\*\*NOTE\*\***: There is an infinite loop that is causing repeated re-renders, and also causing multiple requests to your BE every second - it's very easy to do, but in `ReviewPage`, you have put the `comments` in the `useEffect` dependency array, which means that `setComments` will trigger the effect to run again, which will trigger another `setComments` and so on and so forth (same for the `singleReview`!)

- Changing this will also fix the voting issue!

- Looks like the `Home` component is fairly empty, save for a call to `getUsers()`. Perhaps it could be merged with the `UserLogin` component?

- Make sure to get rid of any lingering code that is no longer used - like the logic around fetching categories in `Nav.jsx` that don't end up being rendered

- In `PostReview`, it looks like you're hardcoding the list of available categories for adding a review. Perhaps this can be made dynamic by using the `getCategories` api function?

​

**## MAKE SURE ALL TESTS ARE STILL PASSING IN BACK END**

​

**## Once everything else is complete, here are some extra challenges:**

​

- [ ] Use `aXe` extension to check for a11y issues

- [ ] Make sure any pure functions are extracted and tested with `Jest`

- [ ] Use Context API for sharing logged in user amongst components

- [ ] Create a user page where you can change their profile information if they are "logged in as the right user". This will require having an additional PATCH endpoint on your backend

- [ ] Create a view for all the reviews a user has liked. This will require additional functionality on your backend

- [ ] Make use of [web sockets](https://en.wikipedia.org/wiki/WebSocket) to allow your page to automatically update with a little notification if there have been any recent posts. [socket.io](https://socket.io/) is quite a good one to use and has some good getting started guides. This will require additional functionality on your backend for recent reviews e.g. last 10 minutes