

Common issues/questions

- Where do I start?
- How do I handle reload?
- My elements don't exist!
- How do I handle this 401 error?
- I hate having to rebuild/restart

Where Do I Start?

- Start by writing a basic render method
 - For the first part you will need to render
 - Not everything yet
 - Shows what state variables you need
 - Everything is based on render()!
- Then add event listeners and service calls
 - For the elements you have rendered
- Write a little bit at a time
 - Confirm it works
 - Only then do you add more

How do I handle reload?

If you reload the page

- ...and assumed they have to login
- ...jarring experience

Instead, on initial load

- Service call to check for session
 - If session, load word then render
 - If no session, render

Do AFTER writing initial render() and login

- And after testing that those work

My Elements don't exist!

`render()` replaces your elements

- Adding/removing eventListeners is complex
- Not removing eventListeners is memory leak

Instead use **event delegation**

- Event delegation is relying on **event propagation**
- Attach event listeners to ancestor element
 - That is NOT replaced by `render()`
- Confirm if `event.target` is desired source of event

How do I handle a 401 error on page load?

- This is not an error you need to report to user
 - 401 on check for session on load is expected
 - Just render while state says not logged in
 - Showing login form IS the error handling
- Other errors should be reported to user
 - So user can fix
 - Add current error to state
 - Let render() report it

Restarting is annoying!

During development you can automate restarts:

- Install/use nodemon to handle server restarts
 - Not for js-service-calls, but is part of project2

```
nodemon server.js
```

- webpack can run in "watch" mode to auto-rerun

```
npx webpack --watch
```

- These should NOT be your `start` or `build` scripts!
 - But ARE options for differently named scripts

Lessons for Project 2

- Services won't always return perfect data
 - Services not specific to this app/this version
 - May need to call multiple services
 - Translate returned data into what app needs
- Client State !== Server State
 - Only one user, for example
 - Different goals
- Usually render HTML, don't show/hide with CSS
 - Show/hide gets more complex as app grows
 - Make it always able to render()
 - Decoupled from momentary logic