Portfolio sites:

This site: <https://github.com/GNolanDev/PortfolioSite>

Timerthing – [www.timerthing.com](http://www.timerthing.com)

Sailor’s Mate -

Github pages:

(see Euroballs)

<https://gnolandev.github.io/teacosy/> / <https://github.com/GNolanDev/teacosy> = Following a design brief given image assets – “Tea Cosy” shop website

<https://gnolandev.github.io/excursion/> - <https://github.com/GNolanDev/excursion> = another design challenge to follow a brief, given some assets to work with. Once complete, found it didn’t look good on smaller screens, so added some responsive css. Learning point – embedded videos don’t autoplay unless muted.

Netlify:

<https://euroballs.netlify.app/> & <https://gnolandev.github.io/MixedMessages/> / <https://github.com/GNolanDev/MixedMessages> - fairly simple javascript app – randomises real comments to make up commentator nonsense for Euro 2020.

<https://gnolandevjammming.netlify.app/> / <https://github.com/GNolanDev/jammming> - A project set by Codecademy as part of the full stack path. Uses Create-React-App as a base, html templates given. My job was to create all the components and connect them up, then connect to the Spotify API to fetch songs & create new playlists on the user's account. This was a good overview of how React single page apps are arranged, a chance to see how API documentation works, an opportunity to use build tools, and an introduction to a simple way of deploying via 'surge'.

<https://gnolandev-reddit-project.netlify.app> / <https://github.com/GNolanDev/reddit-project> – practice using 3rd party api (reddit) to fetch data, practice with react-redux & testing react-redux.

Technologies: ECMAScript 6, React, Redux, CSS, React Testing Library & MSW for api mocking.

To create a single page application which:

as a user will show me a list of subreddits as collected from the Reddit API

as a user will allow me to choose a subreddit & show me a list of posts from that subreddit

as a user will allow me to input a search term and display only those posts with the search term in them

as a user will show me the number of comments on each post and allow me to expand the comments section

<https://gnolandevquiz-app.netlify.app> / <https://github.com/GNolanDev/quiz-app> – create flashcards for simple quizzes, practice with react. A project set by Codecademy as part of the full stack path. Uses Create-React-App as a base, html templates and React Router basic framework given. My job was to create the slices and get all the components talking to each other via actions/reducers/selectors etc.

This was a good overview of how React single page apps make use of <BrowserRouter> and Slices to organise the Redux Store, another chance to consider Separation of Concerns, and another opportunity to use build tools.

The original readme for the project mentions normalisation - I believe there is a little more to be done here. The given structure has "topic" objects holding a list of associated quizzes, but also has each "quiz" holding the id of an associated topic. This seems like duplication of data sources that could potentially contradict each other. An improvement would be to only hold the data regarding which quizzes are associated with which topics in a join table, and look this up whenever necessary.

<https://gnolandevappointmentplanner.netlify.app> / <https://github.com/GNolanDev/appointmentplanner> – very simple contacts & appointments page, created as practice with using React <BrowserRouter> in a single page app. Uses Create-React-App as a base, html templates and React Router basic framework given. My job was to create all the components and connect them up, maintaining controlled inputs keeping "App" stateful & children stateless.

Non-hosted:

In-console game “find my hat” - <https://github.com/GNolanDev/FindHatProject> - To create a simple keyboard event driven game - navigate around a randomly initialised 'maze' to get to your "hat" without falling in any "holes".

Codecademy shared files / <https://github.com/GNolanDev/codecademy_sharing> :

Valid credit card number checker – returns the company name of any invalid cards in the array.

Mysterious organism – creates an array of randomly generated ‘organisms’ with some random dna bases which match certain ‘survival’ criteria, & compare similarity function for 2 organisms.