

# 10 React Practice Projects

*"... but what do I build?"*

React seems worth learning, and the basics make some sense, but you need more practice.

But most projects are either too **small** or too **big**.

You want *some* direction, but you want less hand-holding than a step-by-step tutorial (this is supposed to be practice, after all). And no more To Do apps.

In this guide you'll find 10 projects to **level up** your React skills.



# Before you start...

Make sure you have the tools installed:

node with npm (or yarn)

```
npm install -g create-react-app
```

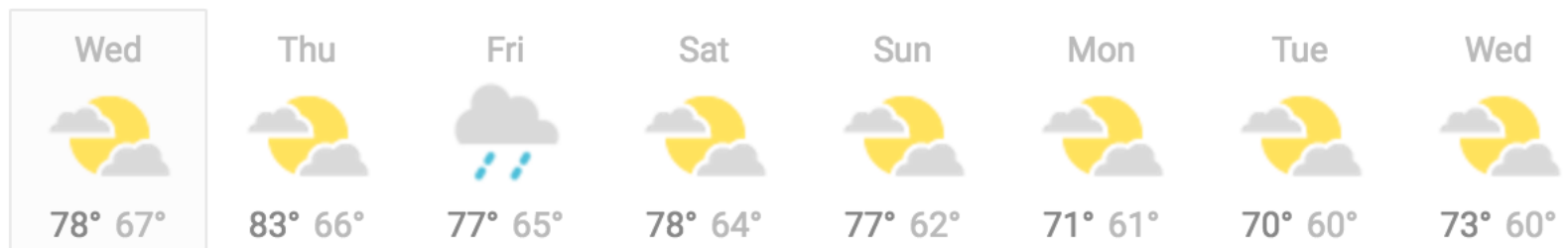
You should already understand some of the basics:

- JSX, conditionals
- How to make a component
- How to pass props to a component
- What state is, how to update it, how to pass it down as props
- How to pass callback functions down as props and use them to update state in parent components

If you have *no idea what I just said*, I strongly recommend running through the official React tutorial first 😊 You can find it here:

<https://facebook.github.io/react/tutorial/tutorial.html>

# #1 Weather



Display a weather forecast, where each day shows the high and low temperatures, and an image for sunny/rainy/cloudy/snowy.

Use fake, hard-coded data until you've got everything rendering correctly.

## Bonus

- Click on a day to see its hourly forecast (maintain the current view in the top-level App state)
- Use React Router to map `/` to the full forecast and `/name-of-day` to that day's hourly forecast.
- Sign up for a free API key from Open Weather Map and fetch a real 5-day forecast.

# #2

## Calculator

0			
clear			÷
7	8	9	-
4	5	6	+
1	2	3	=

Clicking the numbers or the operations should perform the corresponding action.

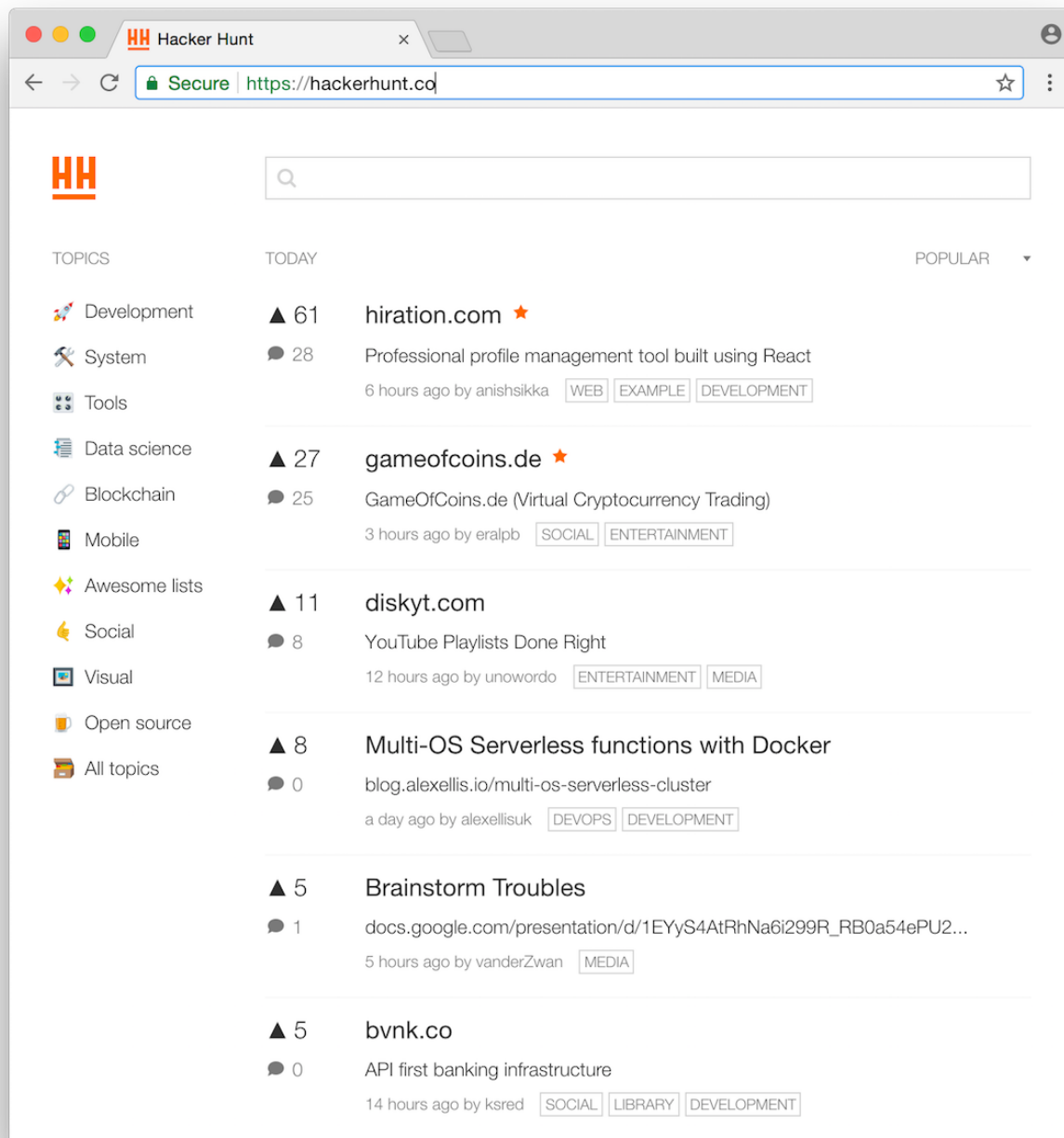
Think about what to store in state. Just the numbers on the display? When you type a new number, does it replace the display, or append?

### Bonus

Respond to keyboard input without using an `<input>` control.

# #3

# Hacker Hunt



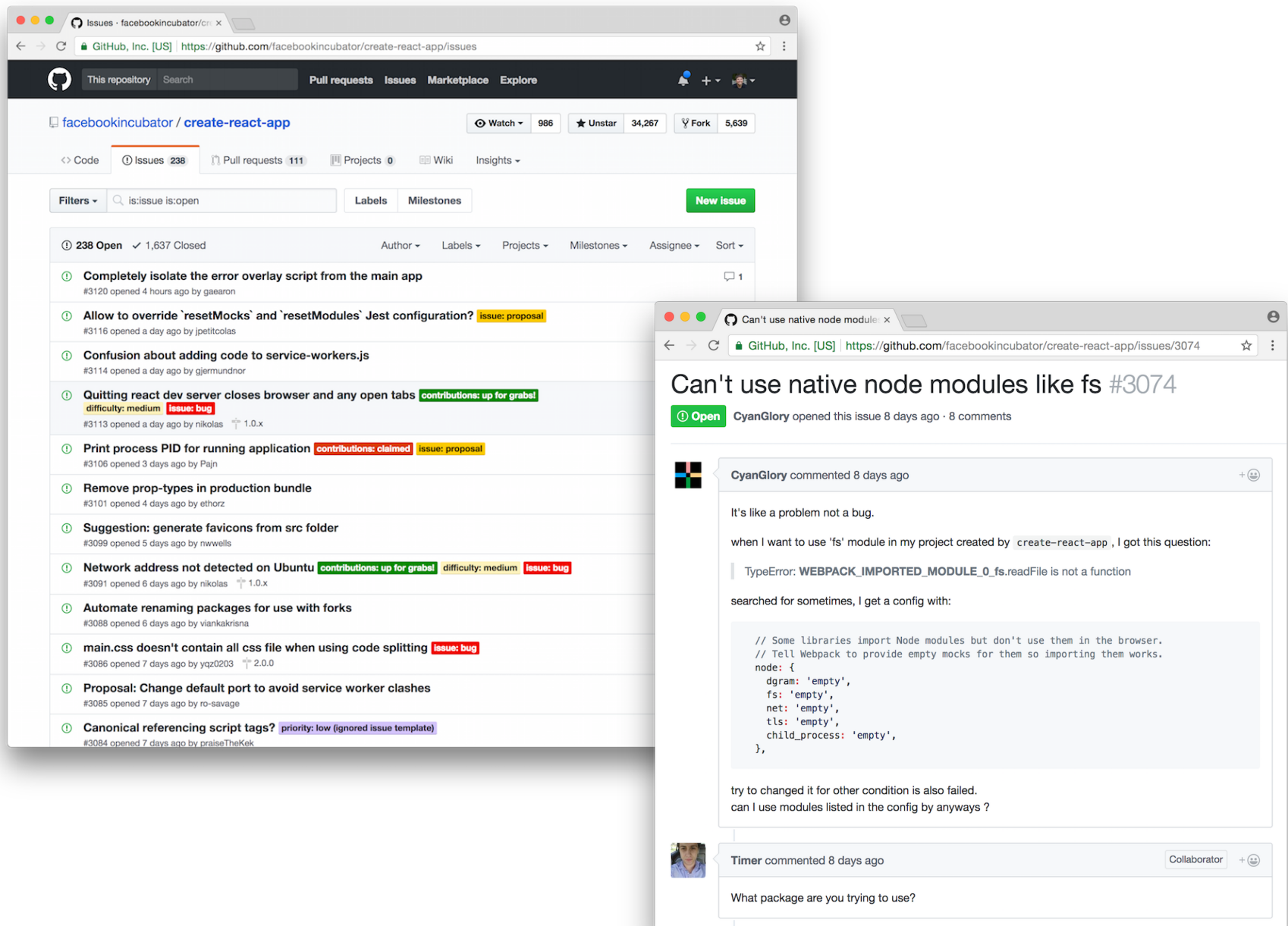
Make a clone of Hacker Hunt.

Use static data until you get the app rendering correctly.

## Bonus

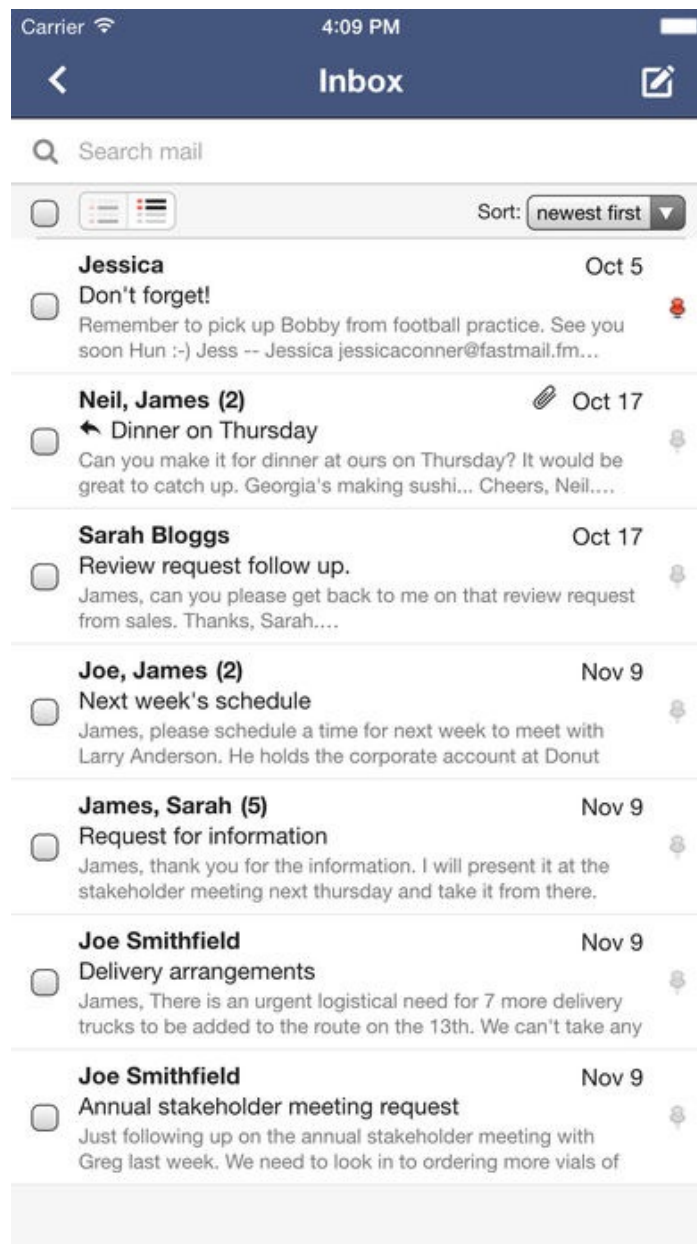
- Use their API to fetch real data: <https://hackerhunt.co/api/daily/0>
- Implement pagination

# #4 Github Issues



- Start with just the list by itself. Ignore the star rating, search/filter controls, etc.
- Fetch data from Github's API
- Add pagination controls
- Use React Router to allow navigating directly to a page
- Click an issue to go to its detail page
- Render the issue's text + comments with react-markdown

# #5 Email



Make a clone of your favorite email client.

Start by rendering the list of emails, and use static data.

Once that's working, enable clicking on emails to view their contents. For crazy extra bonus points, fetch the actual mail over IMAP. [Here's a library for that.](#)



# #6 Airbnb

A great way to practice is to copy existing apps that you're familiar with. This is known as copy work.






Make a clone of Airbnb's home page. The screenshot below has some potential components already outlined.

**Airbnb** Book unique homes and experience a city like a local.

Where Anywhere	When Anytime	Guests 1 guest ▾	Search
-------------------	-----------------	---------------------	--------

- FOR YOU
- HOMES
- EXPERIENCES
- PLACES

### Experiences See all >

 <p><b>\$19</b> Enjoy unique views at an artist home ★★★★★ 4 reviews</p>	 <p><b>\$35</b> Make a flamenco bag with a designer</p>	 <p><b>\$35</b> Hatha Yoga &amp; local tea in a greenspace Art Gallery</p>	 <p><b>\$31</b> Street art lessons in Ipanema from a local artist</p>	 <p><b>\$57</b> Learn how to cook perfect crepes with a chef</p>
---	--	--	--	---



# #7 Calendar

## September 2017

Su	Mo	Tu	We	Th	Fr	Sa
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

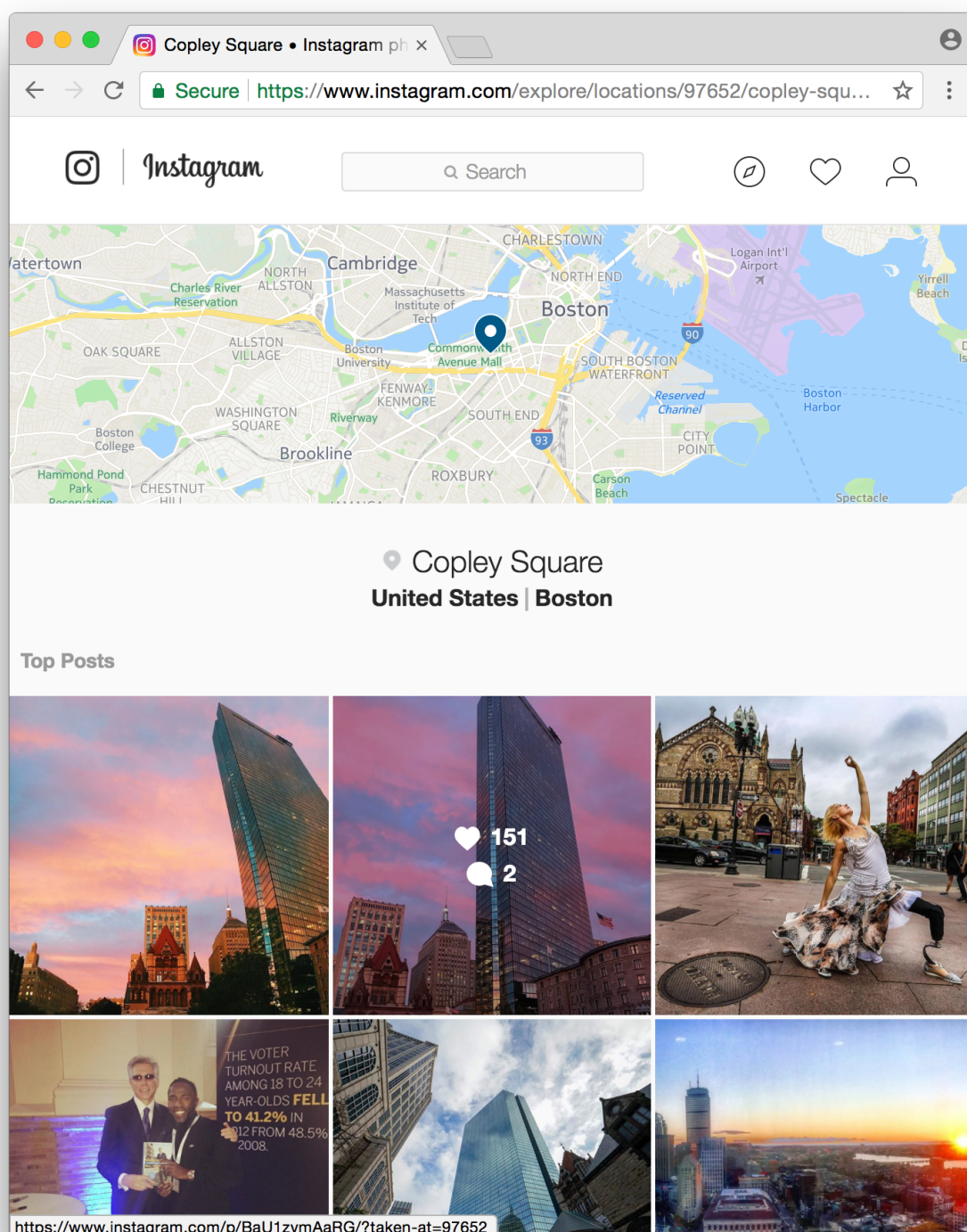
- Start by making a Calendar component that displays a given month/year
- Add interactivity: selecting dates, and date ranges
- Add controls to move forward/backward between months — do these become part of the Calendar component, or are they in a parent? Where should the state be maintained for maximum reusability?

# #8 Instagram

- Clone a single page from Instagram - use static data at first
- Download the images and serve them locally, use a screenshot of the map instead of a real map, etc.

## Bonus


- Use the API to fetch real data  
<https://www.instagram.com/developer/>
- Implement the “Load More” button, and infinite scrolling





# #9 Comment System


- Render a nested list of comments (start with static data)
- Add the ability to add comments at the top level
- Add the ability to Reply to a comment
- Make the upvote/downvote buttons functional
- How do you handle deep nesting (e.g. > 3 or 4 levels?)




 **Vinod** · 6 months ago  
Any tips on server side rendering using react-create-app? Specifically, if I want to send the index.html from my express app.  
6 ^ | v · Reply · Share ›

 **Eli** · 6 months ago  
I got this working locally, thanks. Now how do i go about deploying both apps to a server? Is it better if i host the react client on one server and the api on another? or can i have the express app server the react app? and if so how?  
4 ^ | v · Reply · Share ›

 **Joshua Terrill** → Eli · 6 months ago — | v  
It's entirely up to you if you want to keep them on the same server or separate. I personally would learn towards keeping them on the same server just for convenience sake. Usually I would put all of my API calls at a url scheme of '/api/endpoint-name' and then return all other URL calls to an index.html file that has your react app on it. A step-by-step on how to do this can be found all over online, here is the first one I found that looks like it explains it pretty well:  
<https://medium.com/@patrici...>  
3 ^ | v · Reply · Share ›

 **Matt Lockyer** · 6 months ago  
A lot of people will stumble with multiple terminals or think they need to use Linux screen.  
I suggest using concurrently (npm concurrently) and adding an "express" script to package.json that runs both express and react-scripts using the PORT environment variable for express like you suggested. :)  
Bonus is you can run build and serve the build from your express server using express.static('build'). Add compression and you're close to a real production environment! Use chrome tools and throttle network to see how users will really experience your app!  
2 ^ | v · Reply · Share ›

 **tamj0rd2** → Matt Lockyer · 5 months ago  
Would you be able to give more details about express.static('build')? Where would I need to add that line?  
Edit: Nevermind, I found it in the docs :) <https://expressjs.com/en/st...>  
^ | v · Reply · Share ›

# #10

# Chat App

- Make a clone of something like Apple's Messages app
- Assume 2 participants
- After it's working with static data, add the ability to send messages

## Bonus

- Create a back end server to handle the messages
- Connect the React app to the back end and send real messages back and forth
- Add the ability for more than 2 participants in a single chat

