**State:**

* Open Demos>React>state>state-demo in VS Code
* In the terminal, ensure you are in the state-demo directory and run ‘npm start’
* Show the students that the page loads 2 empty ‘post’ components
* Open src>App.js and show students that each post is its own component, one class based, and one functional
* Open PostClass.js and walk students through the file
* In the constructor add a state object that includes a name, profile image (using the womanProfileImg variable), and some text
* Use the variables in state to fill out the img src, h3 text, and p text
* Show students that the information provided in the state is now visible in the post on the browser
* Show students that the onClick event handler calls the empty nextPost method and binds the word this to the current instance of the class calling nextPost
* Add a function body to nextPost that changes the state variables for the profile image (to the parrotProfileImg variable) and the text
* Demonstrate that when the Next Post button is clicked the image and text change, but the name stays the same because we did not re-set it
* Open PostFunction.js and add separate state variables for name, profile image, and text
* Use the state variables to fill out the img src, h3 text, and p text
* Show students that the information provided in the state is now visible in the post on the browser
* Show students that the onClick event handler calls the empty nextPost method
* Add a function body to nextPost that changes the state variables for the profile image (to the parrotProfileImg variable) and the text
* Demonstrate that when the Next Post button is clicked the image and text change, but the name stays the same because we did not re-set it
* Rewrite the state variables into a single user object similar to that used in class variables
* Demonstrate that the post that renders and functions the same

**Prop Drilling:**

* Open Demos>React>prop-drilling in VS-Code
* In the terminal, ensure you’re in the prop-drilling directory and run ‘npm start’
* Explain to the students that the divs are all nested within each other like Russian nesting dolls
* Open src>App.js in VS Code and explain the state variables defined there and that they are being passed to DollOne as props
* Open DollOne and explain that the num prop is being used to display the doll’s number, then that the num prop + 1 and tiny prop are being passed to Doll Two
  + Repeat for dolls two and three
* When you get to tiny doll explain that the tiny prop is finally being used after being passed through all the props
* Remove tiny={props.tiny} from DollTwo line 8
* Demonstrate to students that the app still loads properly but now TinyDoll is empty
  + This is because the tiny prop must be passed through all of the components to make it to the inner most component
* Undo your change to DollTwo
* Demonstrate props destructuring in DollOne
  + Change DollOne(props) to DollOne({ num, tiny })
  + Remove all uses of “props.”
  + Show that the app renders the exact same