Day IV

Focus (Day IV)

- Be able to use styled-components to add functional styles to your React Components
- Be able to explain why state driven views and components are useful patterns for scalability, composability, and reuse and how React enables those patterns

Daily Setup (Day III)

- By now you have been working on laying out your application, adding functionality and even building out a login system that only shows your app if a user is logged in.
- Today we're going to be adding `styled-components` to our application. For more on the motivation behind this approach watch this styled-components.

Tasks (Day III)

- Try and replace all of the styles you have previously written. Change everything to be a Styled Component. (Except for where you need to reference your icon classNames.)
- Re-factor `SearchBar` to use `styled-components`.
- Create a `Header` styled-component.header that wraps your entire SearchBar.
    - We recommend working left to right so begin changing out your styles on the instagram logo.
    - Create a `LogoHeader` styled-component.div
    - Create a `LogoImage` styled-component.img that fills in the background of your
- Re-factor your `PostContainer` component use only styled components
    - Start with the `UserThumbnail` and `Username`
    - Note that the `Username` styles could most certainly be used by your `CommentSection` component as well.
    - To achieve this, you'll simply just build out your reusable `styled-component` inside of a `Styles/Reusables` directory. This is where you could store all of your reusable styled components and export them out of their respective files to then import them for reuse.
        - Notice that the `Username` on top of the Post is a bit bolder than the `Username`'s found in the comment section. If there is a way to make this reusable styled component accept a prop that will distinguish it from being now is the time to figure that out.
- Now move onto the `CommmentSection` and get rid of any CSS that you don't need.
- By now you're a pro at using `Styled-Components` and we hope you armed with the ammo necessary to choose whether or not you like using it as opposed to vanilla CSS or even a Pre-Processor. We don't want you to believe that there is never a time and place for both native CSS/Pre-processing to exist. But at least now you have a variety of weapons you can choose from to get the job done.
- Now is the time to take a step back and look at what you have accomplished this week. Start from the Day 1 README file and see what the tasks were there? Think about how we asked you to approach this problem. Think about the different components you have built? Did you separate out your Comments array from the rest of the application state? If so, what types of advantages did you gain by doing so? React is a very powerful tool and the next steps in learning about the React Ecosystem will all revolve around other libraries that we will plug into our application for extending it's use.
- Read this article about 'State Driven Views' and come up with a short paragraph as to why you feel that `state-driven` views are important to us as software developers today? What are state-driven applications? Why are they powerful? How does React enables those patterns?

Stretch Problems (Day III)

- Add the functionality to select a single Post. If a user clicks on a post, only show that post.
  - You will need to research into `React Router V. 4` (this is what we're learning about next week) to get this to work in a fluid fashion.
  - Be sure to declare routes for your home `/` page (`<App />` or `<LoginPage />`);
  - Declare a Route for `/single-post` and mount the single post that the user clicks on.