// pseudo code

// MVP

// user selects a body part from the drop down menu.

// when App.js mounts, go grab a list of exercises, names of body parts and a GIF that demonstrates the exercise from the API.

  // Use AXIOS to make this request.

  // Want to make this API call when the component mounts, then we're done.

  // Need to make our API call in USE EFFECT with an empty DEPENDANCY ARRAY.

  // After retrieving the data, we need to organise it to the needs of our app. In this case, displaying a list of example exercises with an image depending on what body part the user chooses from.

  // When the user selects another body part, it will trigger a re-render of our component, so that our JSX can update to display new data.

  // To do this, we must use STATE to trigger a re-render of the page.

  // have a FORM on the page which allows the user to select what body part they want exercises for.

  // Put this form in its own COMPONENT - meaning we will need to get the user's selection UP from the form component back to App.js.

  // We can pass a function down as a PROP from App.js to the form component. Then when we call the function in the form component, we can pass it the user's selection as an argument, which will be received in App.js.

  // Once the user’s choice of body part is received in App.js we will use this info to FILTER the original list of exercises/images that fit's the user's choice.

  // After filtering, we need to put the information somewhere in order for it to be rendered on to the page, thus we must put it into STATE. This will trigger a re-rendering of the page and display new JSX on to the page.

  // Display the JSX as a narrowed down dataset i.e., a list of exercises that specifically targets a particular body part.

  // Need to MAP it to the page, which we will write in it's own component - this is going to be passed down as a PROP to this component.

// stretch goals

// include some user input on what equipment the user has available.