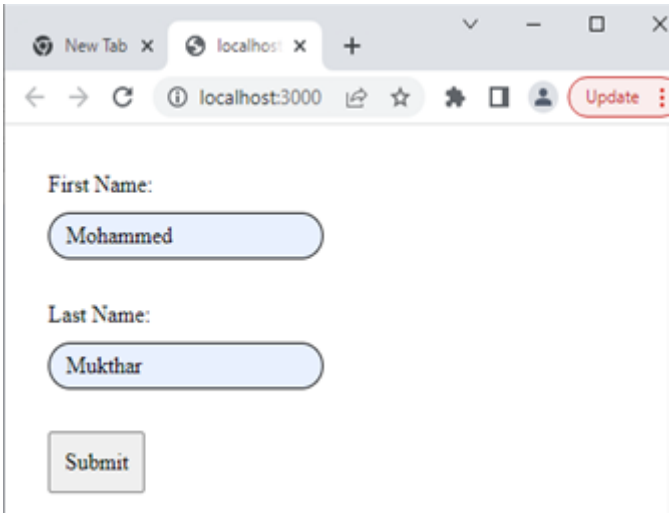
Hands-On - #8

Q1. Use the LAB-p4q1 folder.

Copy the files as stated below:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'App.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

Run the application, the web-browser should look something like the one shown below:



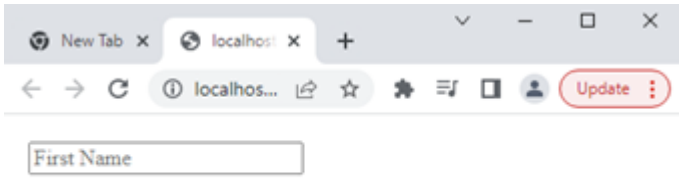
Enter the form data, open the console window and observe the output.

Q2. Use the LAB-p4q2 folder

Copy the files as stated below:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'Form.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

Run the application, the web-browser should look something like the one shown below:



[a] For the 'onChange' event, assoicate the 'handleChange' function

Inside the handleChange function, console log 'Changed!'

Check the same in the console window

[b] The 'handleChange' function can use the 'event' object.

Console log the event object and record your observation.

The 'event' object 'target' will specify the HTML element

The 'event' object 'target.value' will get the value from the text box.

As the text is entered, the console log will be displaying for every character entered.

[c] Your challenge is to update the firstName state on every keystroke

[d] Your next challenge is to track the applicants lastName as well

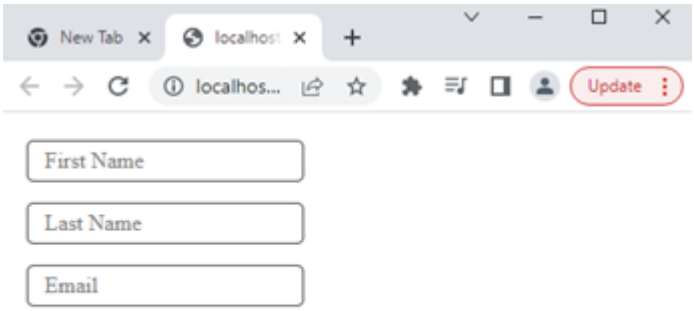
Q3. Use the LAB-p4q3 folder

Copy the files as stated below:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'Form.js' files in the 'src' folder

NOTE: Replace the file(s) in the destination.

Run the application, the web-browser should look something like the one shown below:



[a] Can you imagine if the form has 20+ widgets. Should we have an individual state for each of the widget/control and a separate handler for each of the widget/control

Not an ideal solution.

Your task is to improve the source code.

[b] Thus instead of using individual states for different widgets use the object and process the object with a single handler.

HINT:
The 'event.target.name' will help in identifying the control
The 'event.target.value' will get the value

Run the application and check the console of every change in the controls.

[c] Your next challenge is to 'email' field/state to the form and process.

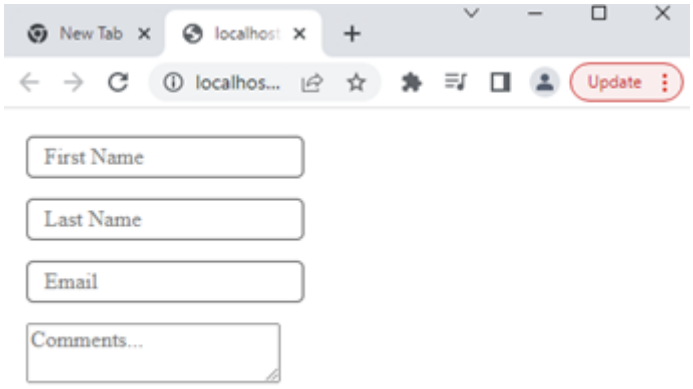
Run the application and check the console for every change in the controls.

Q4. Use the LAB-p4q4 folder

Copy the files as stated below:

- 'index.html' & 'style.css' files in 'public' folder
- 'index.js' and 'Form.js' files in the 'src' folder
  NOTE: Replace the file(s) in the destination.

Run the application, the web-browser should look something like the one shown below:

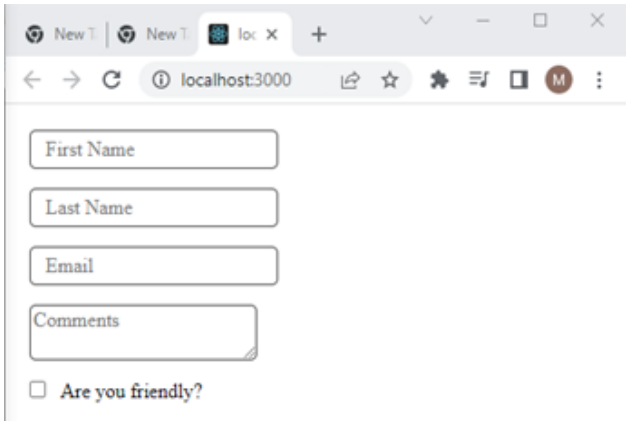


[a] Your challenge is to set the inputs to the state

HINT: Set the 'value' attribute to state-object's data

[b] Your Challenge: Add a **textarea** for "comments" to the form
    Make sure to update state when it changes.

[c] You form should now also have a checkbox with the label 'Are your friendly?' and defaulted to checked.

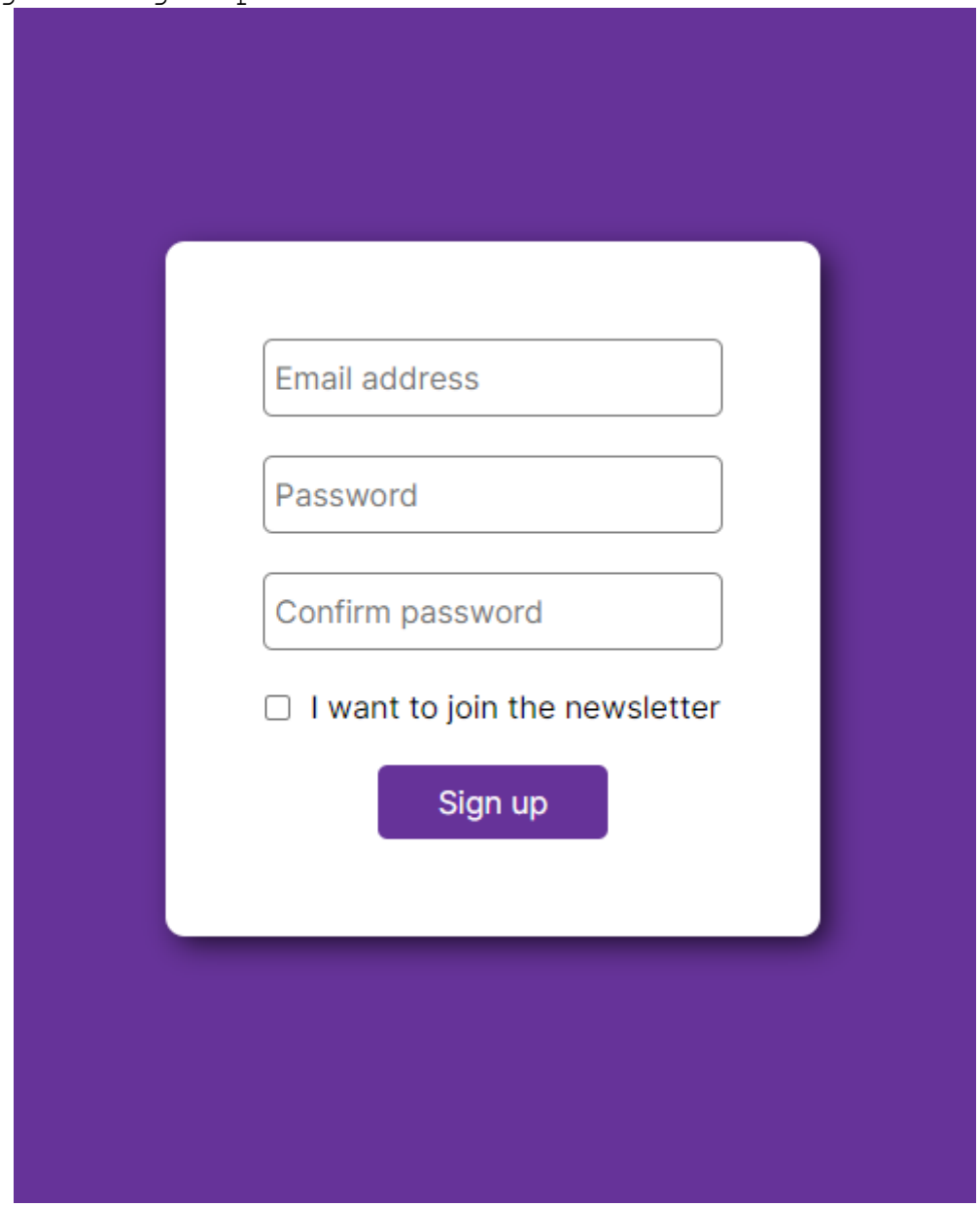


NOTE: In React, checkbox is specified by 'input' element with 'type' as 'checkbox'

Q5. The Sign-up Form

Your challenge is!

Design a Sign-up Form which looks as shown below:



Connect the form to local state

1. Create a state object to store the 4 values we need to save.

2. Create a single handleChange function that can manage the state of all the inputs and set it up correctly

3. When the user clicks "Sign up", check if the password & confirmation match each other.

   If so, log "Successfully signed up" to the console.

   If not, log "passwords do not match" to the console.

4. Also when submitting the form, if the person Checked the "newsletter" checkbox, log "Thanks for signing up for our newsletter!" to the console.