**React Assignment: 1**

1. Install NodeJs, Visual Studio Code
2. Create a Welcome component to greet the user using a JavaScript function, with a property called as name. Apply some CSS styling. Use the component in App component.

**1 import** React, { Component } **from** "react";

**2 export** **default** *class* Welcome *extends* Component {

**3** state **=** { name: "Emmanuel Joseph" };

**4**  render() {

**5**    *return* (

**6**      <**div**>

**7**      <**h3**>Welcome {*this***.**state**.**name} </**h3**>

**8**      </**div**>

**9**    );

**10**  }

**11** }

1. Create a component to add two numbers.
2. **import** React, { Component } **from** "react";
3. **import** "./Add.css";
4. **export** **default** *class* Add *extends* Component {
5. constructor(props) {
6. super(props);
7. *this***.**state **=** {
8. num1: null,
9. num2: null,
10. sum: null,
11. };
12. }
13. inputA **=** (event) *=>* {
14. *this***.**setState({
15. num1: event**.**target**.**value,
16. });
17. };
18. inputB **=** (event) *=>* {
19. *this***.**setState({
20. num2: event**.**target**.**value,
21. });
22. };
23. add **=** (event) *=>* {
24. *this***.**setState({
25. sum: Number(*this***.**state**.**num1) **+** Number(*this***.**state**.**num2),
26. });
27. };
28. clearBtn **=** () *=>* {
29. *this***.**setState({
30. sum: "",
31. num1: "",
32. num2: "",
33. });
34. };
35. render() {
36. *return* (
37. <**div**>
38. <**div** *className***=**"container">
39. <**form**>
40. <**h4** *className***=**"title">Input 2 numbers</**h4**>
41. <**input** *className***=**"inputField" *onChange***=**{*this***.**inputA}></**input**>
42. <**input** *className***=**"inputField" *onChange***=**{*this***.**inputB}></**input**>
43. <**br** />
44. <**button** *type***=**"button" *onClick***=**{*this***.**add}>
45. Calculate
46. </**button**>
47. <**button** *onClick***=**{*this***.**clearBtn}>Clear</**button**>
48. <**div** *className***=**"results">Sum:&**nbsp**;{*this***.**state**.**sum}</**div**>
49. </**form**>
50. </**div**>
51. <**div**></**div**>
52. </**div**>
53. );
54. }
55. }
56. Modify the card component to add a props age and perform the validation to check it’s a number.
57. **import** React, { Component } **from** "react";
58. **export** **default** *class* Card *extends* Component {
59. state **=** { value: null, result: null };
60. getInput **=** (event) *=>* {
61. *this***.**setState({ value: event**.**target**.**value });
62. };
63. checkInput **=** () *=>* {
64. **if** (isNaN(*this***.**state**.**value)) {
65. *this***.**setState({ result: "The input is not a number." });
66. } **else** {
67. *this***.**setState({ result: "The input is a number." });
68. }
69. console**.**log(isNaN(*this***.**state**.**value));
70. };
71. clearBtn **=** () *=>* {
72. *this***.**setState({
73. value: "",
74. result: "",
75. });
76. };
77. render() {
78. *return* (
79. <**div**>
80. <**form**>
81. <**h3**>This will check if the input is a number or not.</**h3**>
82. <**input** *onChange***=**{*this***.**getInput} />
83. <**button** *type***=**"button" *onClick***=**{*this***.**checkInput}>
84. Check Result
85. </**button**>
86. <**button** *onClick***=**{*this***.**clearBtn}>Reset</**button**>
87. <**p**>Result:&**nbsp**;{*this***.**state**.**result}</**p**>
88. </**form**>
89. </**div**>
90. );
91. }
92. }
93. What is a Virtual DOM in ReactJS.

Virtual DOM is a representation of the real DOM. In which a virtual DOM is constantly being kept in sync with the “real” DOM. This process is called reconciliation. Reconciliation is a feature in ReactJS wherein whatever changes you apply to your project, it is made sure that it will constantly be updated.

The “DOM" stands for Document Object Model. It is a programming API used for HTML and XML documents.