# Assignment 4 – Conditionals & Lists

Hands on Task

1. Conditionally show/hide the Student component.

**import React from "react";**

**class Student extends React.Component{**

**render(){**

**return (**

**<div>**

**<hr/>**

**{this.props.children}**

**<p> Name : {this.props.name} </p>**

**<p > Age : {this.props.age} </p>**

**</div>**

**)**

**};**

**}**

**export default Student;**

**import React from "react";**

**import Student from "./Student";**

**class App extends React.Component {**

**state = {**

**students:[**

**{name:"John", age :10},**

**{name: "Mary", age: 11}**

**],**

**showHide: true**

**};**

**showHideStudents = ()=>{**

**this.setState({showHide:!this.state.showHide})**

**}**

**render(){**

**let studentJsx = null;**

**if(this.state.showHide){**

**studentJsx = this.state.students.map((student,index)=>{**

**return(**

**<Student**

**key= {index}**

**name = {student.name}**

**age = {student.age}**

**index={index}**

**increaseAge= {this.increaseIndividualAge}**

**// delete = {this.delete}**

**/>**

**)**

**});**

**}**

**return (**

**<div className = "App">**

**<div>{studentJsx} </div>**

**<button onClick = {this.showHideStudents} > Show-Hide </button>**

**</div> );**

**}**

**}**

**export default App;**

2.Bind a method in constructor (Refer the **Binding a method** slide)

* 1. Avoid binding a method using public class fields syntax
  2. Avoid binding a method using arrow function in the callback

**import React from "react";**

**class Student extends React.Component{**

**constructor(props) {**

**super(props);**

**this.function1 = this.function1.bind(this);**

**}**

**function1() {**

**console.log('Click happened');**

**}**

**render() {**

**return <button onClick={this.function1}>Click Me</button>;**

**}**

1. Implement deleteStudent by array index. Make sure you are maintaining the state immutability.

**deleteStudent = (index) => {**

**const students = this.state.students;**

**let resultStudents = this.state.students.filter((student)=>{**

**return student !==students[index]**

**})**

**this.setState({students:resultStudents})**

**}**

**render(){**

**let studentJsx = null;**

**if(this.state.showHide){**

**studentJsx = this.state.students.map((student,index)=>{**

**return(**

**<Student**

**key= {index}**

**name = {student.name}**

**age = {student.age}**

**index={index}**

**increaseAge= {this.increaseIndividualAge}**

**delete = {()=>{this.deleteStudent(student.index)}}**

**/>**

**)**

**});**

**}**

**return (**

**<div className = "App">**

**<div>{studentJsx} </div>**

**</div> );**

**}**

**}**

## **Research**

1. Read the lecture thoroughly. In your opinion, what are the 3 main points? Please share.
2. Do a research regarding immutable in software engineering and immutable in react. Share your findings.
3. Do a research regarding reconciliation in react. Share your findings.