React JS (Open Source JavaScript Library)

Alok Kumar Srivastava[1], Vaishnavi Laxmi[2], Payal Singh[3], Km Pratima[4], Vibha Kirti[5]
[1]*Assistant Professor, Department of Computer Science & Engineering, Buddha Institute of Technology, GIDA, Gorakhpur*
[2, 3,4]*Department of Information Technology, Buddha Institute of Technology, GIDA, Gorakhpur*

Abstract - **ReactJS, also known as React or React.js, is a very popular open-source JavaScript library for building user interfaces. It is used for handling view layer in single page applications and mobile applications development. It is used to provide speed, simplicity and scalability. Some of its most notable features are JSX, Stateful components, Virtual Document Object Model. JSX which is simply JavaScript extension is optional and not required to use React. However, JSX is a good visual aid when working with UI inside JavaScript because it allows React to show more useful errors and warning messages. Presently it is the most popular front-end JavaScript library. It consolidates the view (V) layer in M-V-C (Model View Controller) design. It is used by Facebook, Instagram, and associations. Respond fundamentally empowers advancement of enormous and complex online applications which can change its information without ensuing page revives. It is very useful for furnish better client encounters and with blasting quick and powerful web applications advancement. It can likewise coordinate with other libraries of JavaScript or structures in MVC, for example, AngularJS. React library can be used in both simple and complex applications. Simple application includes the React library in its script section. React tool chain provides pre-defined structure to bootstrap the application. Also, developers are free to use their own project structure to organize the code. React was conceived to be used in the browser, because of its design but it can also be used in the server with Node.js. Now, the question which arises in front of us is why one should use React. There are so many open-source platforms for making the front-end web application development easier, like Angular. Let us know about the benefits of React over other competitive technologies or frameworks. It is very hard to devote time to learning a new framework with the front-end world-changing on a daily basis especially when that framework could ultimately become a dead end. So, if you're looking for the next best thing but you're feeling a little bit lost in the framework jungle, I suggest checking out React. The most fundamental software or any app development is to select which is the right front end framework or library to go with. The market has a wide variety due to the wide range of problems that developers face every day. If we talk about front end development, then react.js is playing a virtual role and creating new opportunities for developers to build new apps .This paper talks about how react.js is helping in the building of those applications and what advantages it is having in building the front end. There are more than 1,400 developers and over 92,000 sites making use of React.js to build their websites it wouldn't be an overstatement to call React the future of front-end development for web and also for mobile. In this paper, the key features comprising this library were analyzed and its advantages over other frameworks were also analyzed. It is against two-way binding, and also stays away from it and make use of explicit updates instead. React basically allows user to develop large and complex web-based applications which can change its data without subsequent page refreshes. It is used to provide better user experiences and with blazing fast and robust web apps development. ReactJS can also integrate with other JavaScript libraries or frameworks in MVC, such as AngularJS.**

Index Terms - **react, prop types, react dom, react draggable, class names.**

I.INTRODUCTION

React is also known as React.js or React Js. It is a powerful JavaScript library that uses server-side rendering (SSR), and it is an efficient, flexible and declarative JavaScript library for building reusable User Interface components. It is an open source used for creating dynamic and interactive user interfaces for mobile and web applications. React.js is component-based which is used for frontend library responsible only for the view or front layer of the application. It was created by Jordan Walke who had worked as a software engineer at Facebook. Initially it was developed and maintained by Facebook and was later used in its many products like WhatsApp and

Instagram. Facebook developed ReactJS in 2011 in its news feed section, but it was released to the public in the month of May 2013.ReactJS uses virtual DOM based mechanism to fill data in HTML DOM. The ReactJS application is made up of multiple components, each component responsible for outputting a small, reusable piece of HTML code and these components are the heart of all React applications. To create React app, we need to write React components that correspond to various elements. In simple terms, it effectively handles the view layer of mobile and web application. React is only concerned with rendering data to the Document Object Model (DOM). Some of the features of react js are as follows.

1.1JSX(JavaScript Syntax Extension):

JSX is a combination of HTML and JavaScript. It's an XML or HTML like syntax used by ReactJS. JSX makes codes easy and understandable. It extends the ES6 so that HTML like text can co-exist with JavaScript react code. It is easy to learn if you know HTML and JavaScript.

```
 const n="Home";
const ele = <h1>Welcome to {n}</h1>;
```

1.2Components:

Applications of ReactJS are made up of multiple components, and each component has its own logic and controls. So the component logic which is written in JavaScript makes it easy and run faster and can be reusable.

1.3One-way Data Binding:

The name itself says that it is a one-direction flow. The flow of data in react is only in one direction i.e. it is transferred from top to bottom i.e. from parent components to child components. If the flow of data is in another direction, then it requires an additional feature. It is because their components are supposed to be immutable and the data within them cannot be changed. Flux is nothing but the pattern that helps to keep your data unidirectional. This keeps everything modular and fast.

1.4Virtual DOM:

DOM stands for Document Object Model. A virtual DOM object is nothing but the representation of the original DOM object. It works like a one-way data binding. Whenever modification is done in the web application, the whole virtual DOM is updated first and finds the difference between real DOM and Virtual DOM. Once it finds the difference between real DOM and virtual DOM, then DOM updates only the part that has changed recently and everything remains the same. This makes the application faster, and there is no wastage of memory.

1.5Extensions:

React is extended with Flux, Redux, React Native, etc. which helps us to create good-looking UI. React not only supports mobile app development but also provides server-side rendering. There are many extensions of React that we can use to create full-fledged UI applications.

II.LITERATURE SURVEY

It seems react.js solve a problem we became aware of a few years ago. Back then, we had realized that the trend for web development was going toward highly interactive DOM manipulation that usually resulted in markup on the client that was very different from what was sent from the server. Being a man of extremes, we decided to experiment with building UIs that were 100% made in JavaScript and were inserted into the DOM manually (with jQuery) as the user interaction required. Our idea was that we might be able to build very complex UI components (like blog layouts, or submission forms) and pack them into JavaScript libraries. Then we could 'build' a whole site just by including a couple of JavaScript libraries and putting an empty <body> tag on the home page. The results of our 100% JavaScript experiment were not positive for the following reasons:

Incrementally building the mark-up for deeply nested UI components requires that you consider and track the dependency graph of your UI components.

You have to take care not to introduce circular dependencies in whatever rendering method you use.

Modularity becomes very difficult to maintain as the size of the app scales.

The above problems are where we believe react.js fits in as a good solution.

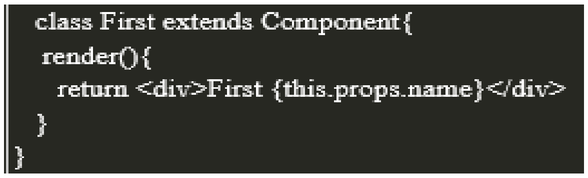
III.REACT LIFECYCLE METHODS

If we analyze then we can see that every component in React goes through a lifecycle of events.

- Mounting – Birth of your component
- Update – Growth of your component
- Unmount – Death of your component

Now let's understand more about how they work.
Common React Lifecycle Methods
render()
This method is the most used lifecycle method in react. We will see it in all React classes. This is because it is the only required method within a class component in React.
It handles the rendering of our component to the User Interface. It happens during the mounting and updating of our component.

Let's understand by an example of a simple render() in React.

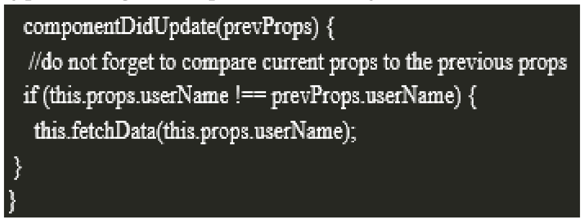


```
class First extends Component{
 render(){
   return <div>First {this.props.name}</div>
 }
}
```

As we can see in the example above, the render() method returns JSX (JavaScript extension) that is displayed in the UI. It can also return a null if there is nothing to render for that component.
A render() method has to be pure with no side-effects. React requires that our render() is pure. Pure functions are those functions that do not have any side-effects and they will always return the same output when the same inputs are passed. This means that we cannot setState() within a render().
We cannot modify the component state within the render().
If we do need to modify state that would have to happen in the other lifecycle methods, therefore keeping render() pure.
Furthermore, keeping your render() simple and clean without state updates makes our app maintainable.

componentDidMount()
Now our component has been mounted and ready, that's when the next React lifecycle method componentDidMount() comes in play.
This method is called when the component is mounted and ready. It's a good place to initiate API calls, if we need to load data from a remote endpoint.
Unlike the render() method, componentDidMount() also allows the use of setState(). Calling the setState() here will update state and cause another rendering but it will happen before the browser updates the UI. This is to ensure that the user will not see any UI updates with the double rendering.

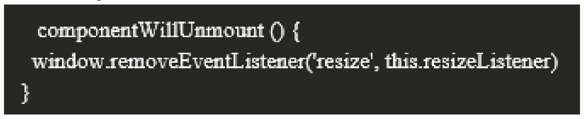
componentDidUpdate()
This lifecycle method is called when the updating happens. If we talk about the common use case for this method then it is updating the DOM in response to prop or state changes. In this lifecycle, we can call setState() but we have to keep in mind that we shall need to wrap it in a condition to check for state or prop changes from previous state. If there is incorrect usage of setState() then it can lead to an infinite loop.
Let's understand by an example below that shows a typical usage example of this lifecycle method.



```
componentDidUpdate(prevProps) {
  //do not forget to compare current props to the previous props
 if (this.props.userName !== prevProps.userName) {
   this.fetchData(this.props.userName);
 }
}
```

We can see in the above example that we are comparing the current props to the previous props. This is for checking if there has been a change in props from what it currently is. In this case, there would not be a need to make the API call if the props did not change.

componentWillUnmount()
This method is invoked just before the component is unmounted and destroyed. If there are any cleanups actions that we would need to do, this would be the right spot.
We have to keep in mind that we cannot modify the component state in componentWillUnmount lifecycle.
This is the component which will never be re-rendered and because of that we cannot call setState() during this lifecycle method.



```
  componentWillUnmount () {
 window.removeEventListener('resize', this.resizeListener)
 }
```

Common cleanup activities performed in this method include, clearing timers, cancelling api calls, or clearing any caches in storage.

Uncommon React Lifecycle Methods

Now we have a good idea of all the commonly used React lifecycle methods. Besides that, there are some other lifecycle methods that React offers which are sparingly used or not used at all.

shouldComponentUpdate()
This lifecycle can be handy sometimes when we don't want React to render our state or prop changes.
Anytime setState() is called, the component re-renders by default. The shouldComponentUpdate() method is used to let React know if a component is not affected by the state and prop changes.
We have to keep in mind that this lifecycle method should be sparingly used, and it exists only for certain performance optimizations. We cannot update component state in this lifecycle.

```
 shouldComponentUpdate(nextProps, nextState)
{
  return this.props.title !== nextProps.title ||
   this.state.input !== nextState.input
}
```

We can see in the example above, this lifecycle should always return a boolean value to the question, "Should I re-render my component?"

static getDerivedStateFromProps()
It's one of the newer lifecycle methods which is introduced very recently by the React team.
We can say that this will be a safer alternative to the previous lifecycle method componentWillReceiveProps().
It is invoked just before calling the render() method.
This is a static function that does not have access to "this". getDerivedStateFromProps() returns an object to update state in response to prop changes. It can return a null if there is no change to state.
It also exists only for rare use cases where the state depends on changes in props in a component.

```
 static getDerivedStateFromProps(props, state)
{
   if (props.currentRow !== state.lastRow)
  {
   return {
    isScrollingDown: props.currentRow > state.lastRow,
    lastRow: props.currentRow,
   };
  }
 // Return null to indicate no change to state.
 return null;
}
```

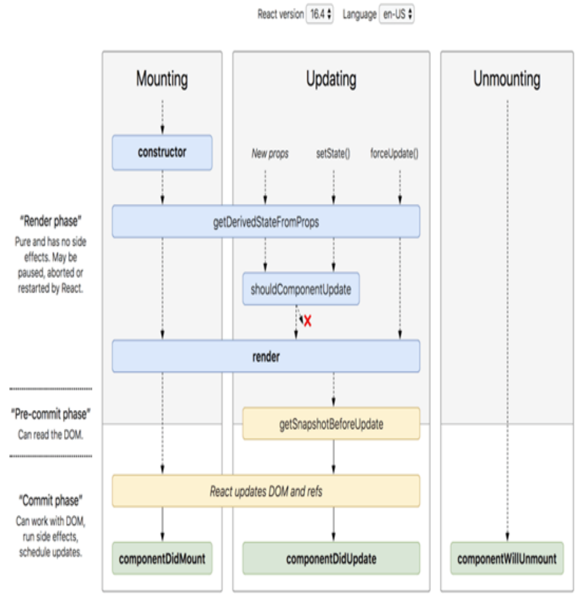
We have to keep in mind that this lifecycle method is fired on every render.
An example use case where this method may come in handy would be a <Transition> component that compares its previous and next children to decide which ones to animate in and out.

getSnapshotBeforeUpdate()
It is another new lifecycle method introduced in React recently. Here also we can say that this will be a safer alternative to the previous lifecycle method componentWillUpdate().

```
 getSnapshotBeforeUpdate(prevProps, prevState)
{
  // ...
 }
```

This method is called right before the DOM is updated. The value which is returned from getSnapshotBeforeUpdate() is passed on to componentDidUpdate().
We have to keep in mind that this method should also be used rarely or not used at all.
Resizing the window during an async rendering is a good use-case of when the getSnapshotBeforeUpdate() can be utilized.

React Component Lifecycle Diagram
We can see the diagram below which is from the official React documentation showcasing the different React lifecycle methods and when they are invoked.



IV.CONCLUSION

Finally, we can say that React JS has come at a good time and helping the developers to build highly engaging web apps and user interfaces within quick time. It allows the developer to break down the components and enables them to create a single page application with less coding.

It is also known to be SEO friendly. We can develop large scale apps with frequently changing data. It is due to these major advantages that React JS has gained much spotlight.

As we know React is a JavaScript-based front-end development framework that offers immense benefits to developers and business owners – small wonder then, that this framework shot to popularity rapidly, and is the hot favorite today. Learning "ReactJS" is the need of the hour and completely makes sense as it is providing much-needed ease to developers in building highly engaging web applications and user interfaces in a very less time, where they create large-scale apps with frequently changing data. React's also benefits of being robust, advanced, responsive, non-risky and user-friendly far exceed its disadvantages, and because developers and organizations understand React's relevance in the market, so they are promoting its learning and deployment whole heartedly. With this we hope that now you know why React JS is great frontend framework.

V.FUTURE SCOPE

ReactJS is very easy to learn, and it is very popular than some other JavaScript frameworks. Many businesses are shifting, or we can say adopting React library because of the simplicity it provides and ease of use. The best advantage of React is Ease of Learning as compared to other popular front-end frameworks like Angular and Vue, we can say that React JS is not going anywhere because according to Stack Overflow, it is the number 1 web framework used by software developers across the world. Therefore, react is far ahead in competition compared to its rivals like jQuery or Vue. The trend of React JS is not only visible in the US, but even in developing countries like India. A recent report shows the open positions of React JS programmers have increased by 184% post-COVID. Therefore, we can say React will dominate many years in the future on a global scale.

REFERENCE

[1] Fu Kaifang "Design and implementation of an instant messaging architecture for mobile collaborative learning" Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on, vol.3, no., pp.287-290, 8-9 Aug. 2009.

[2] Cherry, S.M "Talk is cheap; text is cheaper [mobile messaging]", Spectrum, IEEE, vol.39, no.5, pp.55, May 2002

[3] Butler, M "Android: Changing the Mobile Landscape", IEEE, vol.10, no.1, pp.4-7, Jan.-March 2011.

[4] Vipul Kaushik, Kamal Gupta, Deepali Gupta(2018). React Native Application Development,10.1007/978-981-10-6620-7_59.

[5] Tim A. Majchrzak, Comprehensive Analysis of Innovative Cross-Platform App Development Frameworks Available FTP: https://www. valentinog.com/blog/redux/

[6] M. Miškuf and I. Zolotová, "Architecting React Applications with Flux," 2015 IEEE 13th International Symposium on Applied Machine Intelligence and Informatics (SAMI), Herl'any, 2015, pp. 193-197.

[7] Talaoui Y., Kohtamäki M., Rabetino R. (2017) Business Intelligence—Capturing an Elusive Concept. In: Kohtamäki M. (eds) Real-time Strategy and Business Intelligence. Palgrave Macmillan, Cham.

[8] Basques, K. (2020) Performance analysis reference https://developers.google.com/web /tools/chrome-devtools/evaluate-performance/ reference(Accessed on 10.12.2020)