Available online at : http://www.advancedscientificjournal.com
http://www.krishmapublication.com
IJMASRI, Vol. 3, issue 1, pp. 916 - 921, Jan. -2023
https://doi.org/10.53633/ijmasri

INTERNATIONAL JOURNAL OF MULTIDISCIPLINARY ADVANCED SCIENTIFIC RESEARCH AND INNOVATION (IJMASRI)

ISSN: 2582-9130 **IBI IMPACT FACTOR 1.5** **DOI: 10.53633/IJMASRI**

RESEARCH ARTICLE

A COMPREHENSIVE REVIEW OF REACT TECHNOLOGY: LIMITATIONS AND PERFORMANCE IMPROVEMENT

Priya Bihani[1] and Ms. Tejna Khosla[2]

Department of Information Technology, Maharaja Agrasen Institute of Technology, Delhi, India.

Abstract

ReactJs is the most sought-out front-end JavaScript library. The user interface is an essential aspect of web apps as it dictates how a user interacts and sees the front end of your website. It can create reusable components, and if properly implement it creates much faster interfaces then possible with old technologies such as templating languages like EJS, handlebars etc. It is very useful in creating dynamic User Interface but it also has its disadvantages, which we are also going to discuss in this research paper. It also has many functional benefits. React has many advantages for building robust, scalable, and cost-efficient web app development projects. With React.js, you can increase your overall productivity, shorten development time, reduce web development and maintenance costs, and produce more standardized outputs. However, just like any other web app framework, there are disadvantages and limitations to utilizing React.js. It has a very steep learning curve, lots of libraries to be installed in order for ReactJs to properly function, state management, large chunk size, and poor SEO are some of the problems faced by ReactJs.

Keywords: ReactJS, ReactJS Limitations, ReactJS Advantages

Introduction

ReactJs is developed by Facebook to build responsive user Interfaces. It's used in web versions of FB and FB-owned products — Instagram and WhatsApp. Since 2013, when Facebook open-sourced the library, millions of developers and thousands of tech companies have taken to it, and for a reason. The critical advantage of React-based projects lies in their capability to change data elements — like news feeds or geolocations — without reloading the entire page. This makes apps much faster, efficient than templating languages.

S is a JavaScript library for building user interfaces. It was initially released in 2013 by Facebook and has since grown in popularity, becoming one of the most widely-used libraries for building web applications. React allows developers to

build reusable UI components and manage the state of those components efficiently.

One of the key features of React is its use of a virtual DOM. The virtual DOM is a lightweight in-memory representation of the actual DOM, and React uses it to keep track of changes to the UI. When a component's state changes, React updates the virtual DOM and then uses a process called "reconciliation" to determine the minimal set of updates necessary to update the actual DOM. This can result in better performance, especially for complex and dynamic web applications.

React also makes use of a powerful concept called "components". A component is a piece of UI that can be defined using JavaScript code. React components are reusable and can be nested inside other components to create complex UI. Components also have a lifecycle, which allows developers to control when a component is created, updated, and destroyed. This provides a powerful way to manage the state of a component and the data that it uses.

Another important concept in React is "props". Props are a way to pass data from a parent component to a child component. This allows for a more modular and reusable codebase, as well as a more flexible way to manage the state of a component.

React also has a large ecosystem of tools and libraries available to assist with building web applications. Some popular libraries include React Router for client-side routing, Redux for managing application state, and Next.js for server-side rendering.

React is also highly customizable. Some popular libraries like Material-UI, Ant Design, or Semantic UI allows developers to use pre-built components that follow the guidelines of well-known design systems, so they can focus on building their application logic.

In conclusion, ReactJS is a widely-used JavaScript library for building user interfaces that allows developers to create reusable UI components, manage the state of those components efficiently, and build complex and dynamic web applications. Its virtual DOM, component-based architecture, and powerful features such as props and lifecycle methods make it a popular choice among web developers. Additionally, with a large ecosystem of tools and libraries available, React can be highly customizable to fit the specific needs of any web development project.

Literature Survey

State Management in Redux: We have presented a study and discussed issues related to success factors for cross-platform web application development. Our work includes an in-depth assessment of react redux platform app development frameworks with industry anchored research goals. At the end of this survey we can clearly see the advantages of using react redux for certain circumstances. Management of states and props is also in the most efficient way in terms of data rate and response time. The survey together with academic literature and discussions with the case company formed the basis for the requirements and the technical implementations. Moreover, this provides insights for practitioners through mapping survey findings to common issues identified in react redux framework app development. We confirmed user experience, technical implementation, app performance, and testability to be the most common issues related to react redux framework. There are still many problems to tackle, such as the human side particularly in the form of end-user experience.

React Router: A common library for routing in React is called React Router. It permits switching between views of different React Application components, permits changing the browser URL, and keeps the UI in sync with the URL. To learn how the React Router functions, let's build a straightforward React application.

State in ReactJS: The process of handling the data that React components require in order to render themselves is known as react state management. Usually, this information is kept in the component's

state object. The component will re-render itself when the state object changes. In essence, a React app's state management component is React.

Problem Statement

Traditional multi-page applications (MPAs) have several issues that can make them difficult to maintain and less performant than modern single-page applications (SPAs). Some of the issues with traditional MPAs include:

Slow performance: Traditional MPAs require a full page reload with each navigation, which can lead to slow and unresponsive user experiences.

Limited capabilities: MPAs are typically built with server-side technologies, which can limit their capabilities and make it difficult to implement certain types of functionality, such as real-time updates.

Complexity: MPAs often have complex routing and navigation structures, making them difficult to understand and maintain.

Limited reusability: Traditional MPAs typically have a lot of duplicated code and limited reusability of components, making them hard to maintain.

Harder to test: Because of the complex nature of MPAs, it is harder to test the application and fix bugs.

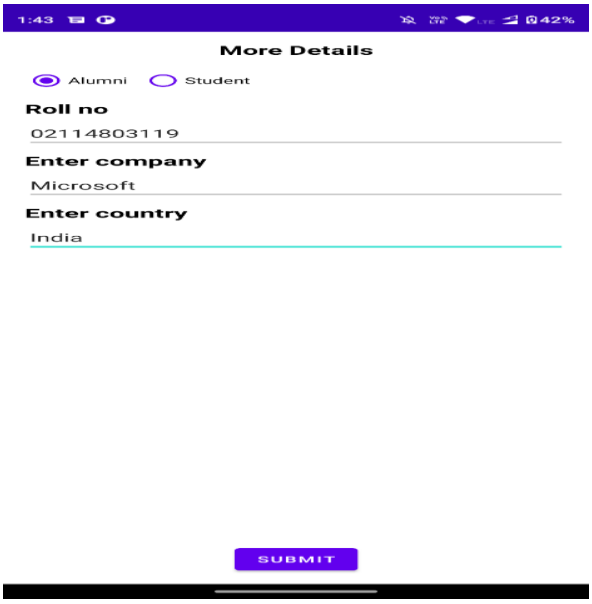
Limited SEO: MPAs can have limited SEO capabilities, and it's harder to make them discoverable by search engines.

Single-page applications (SPAs) built with JavaScript libraries and frameworks, such as React, Angular, or Vue, can overcome many of these issues and provide a more efficient and performant user experience.

Single Page applications and ReactJS

Single Page applications and ReactJS

Single page applications (SPAs) are web applications that load a single HTML page and dynamically update that page as the user interacts with the application. Unlike traditional multi-page applications (MPAs), SPAs do not require a full page reload with each navigation, making them faster and more responsive SPAs use JavaScript to dynamically update the content of the page, typically by re-rendering components in response to user actions. This allows for a more seamless user experience, as the page does not need to refresh with each navigation. The JavaScript framework or library that is mostly used to build SPA is React, Angular, Vue.js etc. These frameworks provide a structure for building reusable components and managing the state of the application. SPAs are often used for complex web applications, such as online marketplaces, social media platforms, and web-based productivity tools, as they can provide a more efficient and performant user experience than traditional MPAs.



4.1. Why React JS and not other libraries ?

There are several reasons why one might choose to use React over other JavaScript frameworks for building single-page applications (SPAs):

Reusable Components: React's component-based structure allows for the building of reusable components, making it easier to build and maintain large and complex web applications. Virtual DOM: React uses a virtual DOM, which improves the performance of web applications by limiting the number of changes that need to be made to the actual DOM. Popularity: React is one of the most popular JavaScript libraries and has a large and active community of developers. This means that there is a wealth of resources and tutorials available, as well as a large ecosystem of third-party libraries.

Flexibility: React is a view library and does not force a particular way of building an application, it can be integrated with other libraries and frameworks for routing, state management, and more.

Learning Curve: React's approach to building web applications is relatively simple and easy to learn, which can make it a good choice for developers who are new to building SPAs.

Performance: React's ability to handle large datasets and updates smoothly, it can be a great option for high-performance applications.

It's worth noting that different frameworks and libraries may be better suited for different types of projects, and ultimately the choice will depend on the specific requirements of the project and the preferences of the development team.

JSX & Syntax

JSX is a syntax extension for JavaScript that allows you to write HTML-like elements in your JavaScript code. It is used in React to describe the structure of a user interface using a syntax that is similar to HTML.

The advantage of using JSX in React is that it makes it easy to understand the structure of a user interface and how different components interact with each other. Because JSX uses a syntax that is similar to HTML, developers who are already familiar with HTML can quickly pick up how to use JSX in React.

JSX also allows developers to write reusable components in a clean and readable way, which makes it easy to understand the overall structure of the application.

Another advantage of JSX is that it can improve the performance of React applications. When a component's props or state change, react uses the virtual DOM to determine which parts of the actual DOM need to be updated. Because JSX allows React to understand the structure of the component, it can more efficiently update the DOM and minimize the number of changes that need to be made.

JSX also allows for the easy integration of logic and expressions inside the components, making the code more expressive and less repetitive.

Overall, JSX is an important part of React that makes it easy to build and maintain user interfaces, improves performance and allows for more expressive and less repetitive code.

Routing in ReactJS

React, routing is the process of mapping URLs to different components that are rendered in the browser. The most commonly used library for routing in React is React Router.

React Router provides a set of components that allow you to declare the routes for your application. The most basic component is the <Route> component, which maps a specific URL path to a component that should be rendered when that path is accessed. For example, the following code maps the path "/about" to the About component:

```
<Route path="/about" component={About} />
```

Fig. 1 Router component

The <Switch> component allows you to group multiple <Route> components together and only render the first one that matches the current URL. For

example, the following code will only render the Home component if the path is "/" and the About component if the path is "/about":

```
<Switch>
  <Route exact path="/" component={Home} />
  <Route path="/about" component={About} />
</Switch>
```

Fig. 2 Switch in React Router

React Router also provides a <Link> component that allows you to create links between different routes. For example, the following code creates a link to the "/about" route:

Additionally, React Router also provides a <Router> component that is used to configure the browser's history and handle the actual navigation. The <Router> component can be configured to use different types of history, such as browser history or a in-memory history.

Finally, React Router also provides a useParams hook, useLocation hook, useHistory hook which can be used to extract dynamic parts of the URL, the current location and navigate the user respectively.

Overall, React Router provides a powerful and flexible set of tools for handling routing in React applications, making it easy to map URLs to different components and navigate between them.

State and state management

In React, state refers to the data or variables that determine a component's behavior and render the view. It's an object that holds information that can change over time, such as a form input value, a toggle button status, or a list of items.Each React component has its own state, and the component can update its state by calling the setState method. When the state of a component changes, React will re-render the component, updating the view to reflect the new state.

For example, consider a simple counter component that has a state variable count that stores the current count value, and two buttons that increment or decrement the count:

```
class Counter extends React.Component {
  constructor(props) {
    super(props);
    this.state = { count: 0 };
  }

  handleIncrement = () => {
    this.setState({ count: this.state.count + 1 });
  }

  handleDecrement = () => {
    this.setState({ count: this.state.count - 1 });
  }

  render() {
    return (
      <div>
        <h1>{this.state.count}</h1>
        <button onClick={this.handleIncrement}>Increment</button>
        <button onClick={this.handleDecrement}>Decrement</button>
      </div>
    );
  }
}
```

Fig. 3 Example of component with state variable

Performance enhancement in ReactJS

React is one of the popular web frameworks that has gained importance over other frameworks such as Angular, Vue, etc. This is because of its implementation of Virtual DOM; whose primary objective is to enhance the overall performance of the application. However, there are certain things that one has to keep in mind before designing the applications. Failing to anticipate the problems that may occur component hierarchy will lead to performance degradation. Some of the commonly faced problems are component re-rendering, application lag due to background computations being run, lag due to processing large data sets in a single stretch, etc. This paper will describe some of the practical ways of overcoming such problems within the application, thus enhancing the performance of the ReactJS App in a production environment. The paper will also describe a time-efficient search algorithm that can be used for searching objects in a large data set.

1. Conclusion and Future Scope

These days, practically everyone uses React, one of the most widely utilised JavaScript libraries. It also has a large contributor base and is one of the most starred repositories. One of the most effective and agile ways to develop an application is by using the modern React API in conjunction with JSX. It comes with everything new users need to get going and offers just the right amount of customizability. As this article has discussed, ReactJS has a few downsides that can discourage an organisation from implementing it. However, there is usually a workaround for every problem. Due to the strong community support, developers have produced remedies for every problem.

The SEO problems are eliminated with a framework called "NextJS," which incorporates server side rendering right out of the box. Redux, HookState, and Beaver are a few solutions that can help you manage state. Additionally, "ContextAPI" is a built-in feature of React that enables native state management. Create React Application, another feature of React, allows users to launch the application without worrying about the compilation building, or serving processes.

Reference

1. https://reactjs.org/docs/getting-started.html
2. https://redux.js.org/
3. https://blog.hubspot.com/website/react-js
4. https://mobx.js.org/

*****