Journal of Telecommunication, Electronic and Computer Engineering
ISSN: 2180 – 1843 e-ISSN: 2289-8131 Vol. 14 No. 4
jtec.utem.edu.my

React Apps with Server-Side Rendering: Next.js

Harish A Jartarghar[1], Girish Rao Salanke[1], Ashok Kumar A.R[1], Sharvani G.S[1] and Shivakumar Dalali[2]
[1]*Department of Computer Science and Engineering, R.V College of Engineering, Bengaluru, India.*
[2]*Don Bosco Institute of Technology, Bengaluru, India.*
harishaj.cs18@rvce.edu.in

Article Info

Article history:
Received July 19th, 2022
Revised Sep 13th, 2022
Accepted Dec 4th, 2022

Index Terms:
Document Object Model(DOM)
Server Side Rendering (SSR)
User Interface (UI)
Search Engine Optimization (SEO)
Application Program Interface (API)

Abstract

Web applications are developed using a variety of different web frameworks, and developers can pick from a wide range of web frameworks when developing a web application. React.js library provides flexibility for building reusable User Interface (UI) Components. It uses the approach of client side rendering, which loads the HTML content using Javascript. The client side rendering causes the page to load slowly and the client communicates with the server for run-time data only. Next.js Framework solves this problem by using server side rendering. When the browser requests a web page, the server processes the web page by fetching the user's specific data and sending it back to the browser over the internet. Next.js helps the Search engines to crawl the site for better Search Engine Optimization (SEO).

I. INTRODUCTION

The Internet is used by companies all over the world as a low-cost marketing medium as well as a route for contacting with their clients and business partners. Providing a secure and efficient method for this form of communication, the use of web applications has increased since its inception. A web application is a software intended to be used on a web browser. Its usage involves the combination of the server-side scripts to the API request to the server, and the retrieval of information stored in database and client-side scripts to rend the data from the server and to manage the user's interaction within the application.

Next.js is a lightweight React framework used to develop static and server rendered applications. Next.js utilizes a folder directory as the routing method for the web pages. The app is the default page. By using the pages directory, the Next.js provides the page with automatic routing, while the server-side renders fetches and data for each request. [6]

The aim of the paper is to demonstrate the benefits of Next.js Framework which uses server side rendering in optimizing the performance and enhancing the Search Engine Optimization (SEO).

II. RELATED WORK

With the rapid advancement of internet technology over the last decade, customers have been increasingly reliant on e-Business to carry out daily tasks such as shopping, property lending, and tax return. One of the most important factors contributing to this innovative outcome is the appearance of HTML5 technologies, which affects the entire internet development ecosphere. HTML5 technology is a markup language used to create layout and deliver content on global websites [1]. In comparison to previous HTML standards, HTML5 adds and improves several semantic components, such as footer>, aside>, and nav> to clearly describe the web structure and to assist web developers in building their website under distinct structure [2]. HTML5 also introduces new components to gain access application programming interfaces (APIs). For example, the canvas> element allows the website to access the canvas portion of the mobile phone [2]. Web developers can build a website with more complex operations using the sophisticated access characteristics of HTML5.

Although HTML5 introduces many new features, it still has the constraint of low efficiency rendered by any published HTML, which is even worse than FLASH in some cases. Google created the Chrome V8 engine in 2008, which sufficiently tackles the issue that pushes JavaScript to the forefront of HTML5 [3]. Prior to the release of Chrome V8, JavaScript's major job in a website was to interact with Cascading Style Sheets (CSS) to build a better user interface and to assume responsibility for some common script activities, such as form validation. Chrome V8 aims to redefine JavaScript since Chrome V8 JavaScript engine has such an astonishing speed that is more than 56 times quicker than any versions of Internet Explorer (IE) [3]. Traditional web browsers often create JavaScript by parsing byte-code and compile the entire web project to generate the code, which is then executed from a file system [4]. As a result, its

JavaScript execution time is significantly longer than that of the compiled languages, such as Java and C++ [4]. The optimized approach for the V8 engine uses inline caching technology to boost speed without traditional compilation [5].

After the V8 engine is released, JavaScript have comparable running performance to Java or C++. As a result, the V8 JavaScript engine allows web projects to run at the same pace as the traditional desktop software. Because of the excellence of the V8 JavaScript engine, different JavaScript platforms based on the V8 engine has appeared, ushering in a new era in Internet development history. Node.js, a development platform that combines with the V8 JavaScript engine, was introduced in 2009 [6]. Node.js broadens the developers' understanding that JavaScript can do more than just running a simple script on a webpage; it can also be used to write an event-driven server-side application with ease [7]. Despite the fact that Node.js was released nine years ago, several new JavaScript frameworks have appeared and have an impact on the Internet development. The paper will next go over the key front-end frameworks, like Next.js and React.js libraries in the following section.

III. WEB FRAMEWORKS AND LIBRARY

A. Document Object Model (DOM)

When a web page is requested in the web browser, the browser renders the document into a tree, dividing the HTML tags called a Document Object Model [3].

```
<HTML>
    <head>
        <meta charset="utf-8" />
        <title>Next.js Benefit</title >
    </head>
<body>
    <h1>Next.js</h1>
    <p>Pros</p>
    <u l>
        < l i >Optimized</ l i >
        < l i >SEO Friendly</ l i >
        < l i > Flexible</ l i >
    </u l>
</body>
</HTML>
```

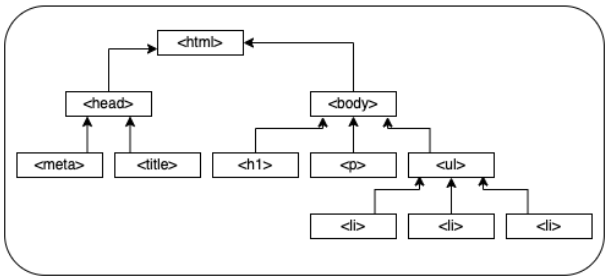

Figure 1. Syntax of HTML and DOM tree of the syntax

A web page's DOM will also update if the structure of the page is modified. The DOM structure can be seen as a tree chart that represents all the elements in an HTML document that has been loaded by the browser [3]. The relationships between the HTML tags can be seen in the tree chart. An example of this can be seen in Figure 1, which is a representation of the DOM structure of the code in the same figure.

B. React Library

React is a declarative, efficient and flexible JavaScript library meant for interactive User interfaces (UIs) and building reusable UI components. It is component-based, which means it is made up of smaller, independent parts called components; thus, making the code easier to handle and more predictable. [1]

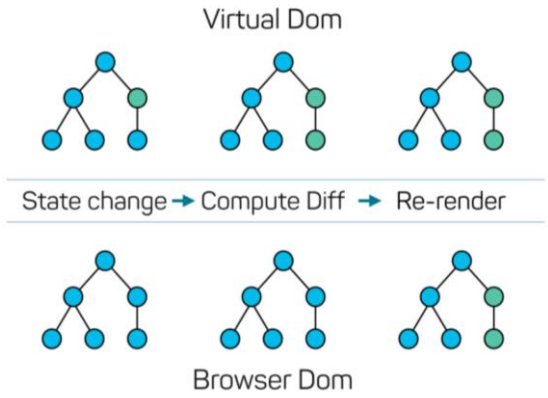

Figure 2. Re-rendering of the actual DOM [20]

As the DOM manipulation is a heavy task and affects the web page performance, the virtual DOM comes into action. The changes are first made in the virtual DOM and the difference is computed between the actual and the virtual DOM. Further, the nodes which have changed will be re-rendered. This way computing simple JS Objects in memory is much faster than manipulating DOM objects for every transformation.

C. Virtual DOM

React.js updates the Virtual DOM rather than the real DOM directly. W3.org defines DOM as "the logical structure of documents as well as how documents are accessed and updated." Updating a DOM is as quick as updating any JavaScript object. Figure 3 illustrates how the browser renders a web page.

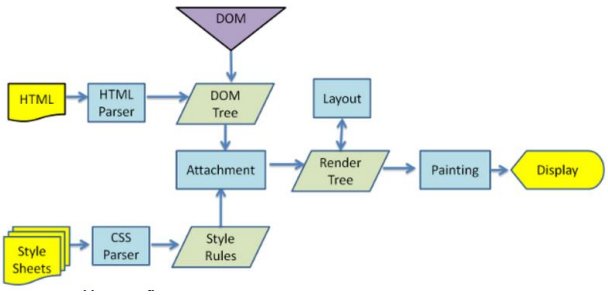

Figure 3. Rendering of DOM in browser [21]

All browsers have rendered engines such as V8 engine used in the google chrome, These Javascript engines are responsible for parsing the HTML page. Javascript engines also parse the CSS to each DOM element and create a render tree. This process is called attachment.

Virtual DOM is an in-memory representation of real DOM. It is a minimal JavaScript object that reflects the Real DOM. When the setState method is used, React.js creates the entire Virtual DOM from scratch. The speed at which a whole tree can be created means that it has no impact on performance.

At any one time, React.js maintains two virtual DOMs, one with the updated state and the other with the previous state.

D. Redux: State Manager

Many businesses have gone forward to deploy effective and faster UI frameworks as a result of global technological improvement. Any UI framework, including React, Angular, Vue, and plain old JS, can use the Redux UI framework library. Redux and React are widely used together, despite the fact that they are independent of one another and that Redux is compatible with all frameworks.

When utilizing Redux with another UI framework, we will typically utilize a "UI binding" library or a linking library to connect Redux to our UI framework, rather than directly dealing with the store from your UI code. React Redux is the original React UI binding library. As a result, the use of a binding layer is reduced and direct connectivity is provided.

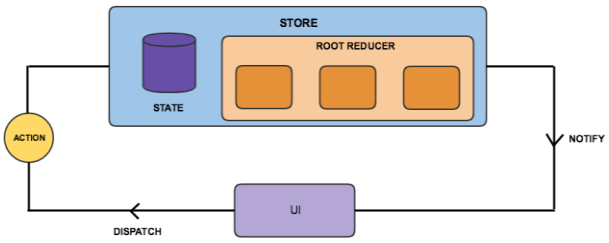

Figure 4. Redux Architecture

Advantages of using Redux:

- The official React Redux binding is called Redux. It enables your React components to send actions to the main data store to modify data and receive data from a Redux store. Developers may easily manage various application states due to the ability to access global data.
- Any change to the state interrupts the child components, consequently hurting performance. The Redux library, on the other hand, centralizes the application's state management. This enables the developer to employ important development tools such as undo/redo, state persistence, and much more.
- Tracking the application's state during the debugging process is quite challenging in React. Redux, on the other hand, offers a fantastic developer experience that allows for "time-travel debugging" and even sends comprehensive error reports to a server.
- React has a rich UI, making data flow challenging when multiple components were used to share the same data. However, Redux is adaptable across all UI levels and has a big ecosystem of add-ons to meet your needs.
- Since the components in React are strongly tied with the root component, it is quite difficult to reuse them. Redux eliminates this complexity and enables global accessibility, which aids with the development of apps that can be tested and run in a variety of situations (client, server, and native).

IV. ANALYSIS METHODOLOGY

A. Client Side Rendering (CSR)

React.js, angular and vue.js uses client side rendering, which runs on the browser. The client-side rendering refers to the use of JavaScript to render content in the browser. As a result, just a minimal HTML document with a JavaScript file in initial loading is received. Rather than receiving all the content directly from the HTML document, the remainder of the site is then rendered using the browser. Although the initial page load is typically a little slow with client-side rendering, subsequently all the pages load are fast. This technique mainly communicates with the server to obtain run-time data. Additionally, following each call to the server, the complete user interface does not need to be reloaded. By re-rendering specific DOM element, the client-side framework is able to refresh the user interface with the updated data..

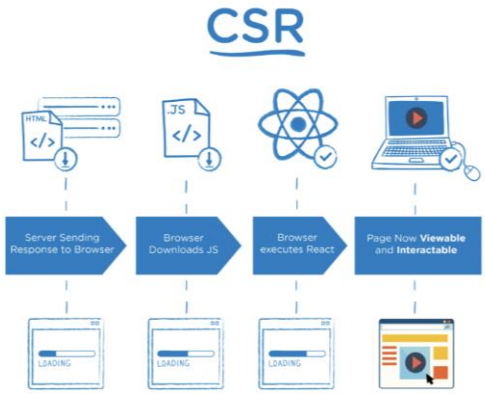

Figure 5. Client Side Rendering [23]

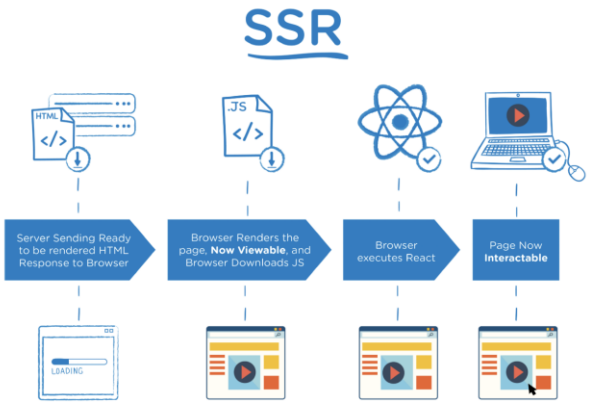

Figure 6. Server- side Rendering [22]

B. Server Side Rendering (SSR)

Next.js uses server side rendering, that is the server sends the HTML response and the browser over the internet. The browser then forms the content and renders the page. The entire process of requesting data from the database, creates an HTML page and sends it to the browser, which happens in mere milliseconds.

C. SSR vs CSR

The primary distinction between SSR and CSR is that for the SSR, the server sends a ready-to-render HTML web page to the browser in response, but for the CSR, the browser just receives a sparse document with a link to the JavaScript. This

means that instead of having to wait for all of the JavaScript to download and run, the browser will immediately begin rendering the HTML from the server. Reaction will need to be downloaded in both situations, and it will go through the same steps of creating a virtual DOM and adding events to make the page interactive. However, with the SSR, the user can begin seeing the page while all of process is taking place. For the page to be accessible, the CSR requires all of the aforementioned events to take place before the virtual DOM is sent to the browser's DOM.

D. Performance Comparison

The analysis shown in Figure 7 is done on the structuring of the web application to both the SSR and the CSR. These are the chrome network screen tabs of the rendered pages. Web application using SSR approach renders more quickly and loads with CSR resulting in a blank, white page.

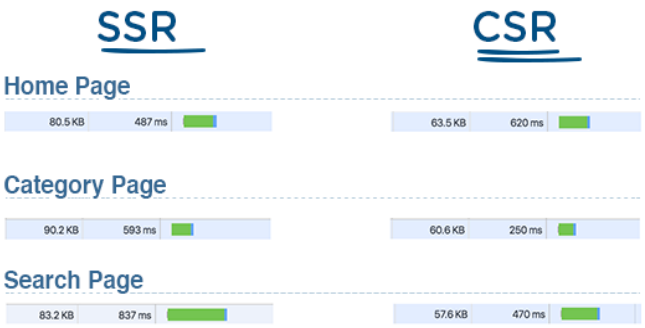

Figure 7. Network Tabs of SSR vs CSR

V. NEXT.JS: SSR FRAMEWORK

This section will highlights the advantages and disadvantages of using Next.js:

A. Advantages of Next.js

1) Automatic code splitting

The technique of separating the application's bundle into smaller parts required by each entry point is known as code-splitting. The goal is to reduce the initial load time of the application by just loading the code needed to run that page.

2) Lazy Loading

Next.js supports dynamic imports, some components are imported during the runtime or later part time whenever necessary. Deferred loading helps to improve the initial loading performance by decreasing the amount of JavaScript necessary to render the page. Components or libraries are only imported and included in the JavaScript bundle when they're used.

3) Built-in CSS

With Next.js, enables importing CSS styles from a JavaScript file to be used online for faster rendering.

4) Better Image Optimization

Graphics are scaled and supplied using the best, most recent formats, such as WebP (while being open to future formats), and images are engineered to adapt to smaller viewports.

5) Search Engine Optimization (SEO)

Titles and keywords for each page are simple to develop for individuals wishing to improve their SEO. They can use the featured Head components to add them to each page.

B. Disadvantages of using Next.js:

1) Cost of flexibility

Because Next.js does not have many pre-built front pages, one must build the entire front-end layer from the ground up.

2) Lack of built-in state manager

As a result, if one requires a state manager, it will also require Redux, MobX, or something similar.

3) Low on plug-ins

Next.js cannot utilize as many easy-to-adapt plugins in comparison to the Gatsby.js.

VI. SEO OPTIMIZATION WITH NEXT.JS

As Next.js uses the SSR, the SSR converts the React code to HTML on the initial request. In contrast, a typical SPA site transmits a large Javascript payload to the browser. The vast majority of the HTML shown by the browser will be generated by the Javascript. This distinction in the load strategies has a significant impact on search rankings.

SEO bots crawl the web, examining the HTML of each URL they come across. If it reaches a non-SSR SPA page, it will deliver a large glob of Javascript instead of HTML. SEO bots do not comprehend Javascript, and they do not always wait for it to convert to HTML. This means that the bots who determine what appears on the first page of search results are unable to read the page. If they could not read the page, Google will not serve it up as the "best" option answer to a user's search. This is where Next.js can help.

By encapsulating the SPA page in NextJS, any queries to the site's URLs will always result in a bot-friendly HTML page. It can have the best of both worlds. It may create SPA sites while still reaping the SEO benefits of traditional website architecture.

VII. DISCUSSION AND FUTURE WORK

The above discussion highlights the advantages of Next.js and its performance optimization. Next.js has some drawbacks, and one of the drawbacks is that the SSR makes some features to be more complicated and costly. The cost of flexibility is more as Next.js does not provide built-in front pages and it has to create the frontend layer from ground up. Less plugins are adaptable to the SSR approach and it could not be integrated with Next.js. Managing the state is a challenging task in SSR applications as compared to CSR. Research and open contributions are being made to overcome the above drawbacks for optimizing the SSR.

VIII. CONCLUSION

React apps with server side rendering can be built using Next.js frameworks, which optimizes the application performance. Next.js also enhances the SEO for the web applications and makes crawling easily accessible. The two most significant building components supporting the overall digital experience are theNext.js and the React. They have the ability to accelerate the web through applications that improve performance, reduce development costs, and have larger production rates.

REFERENCES

[1] Lawson, B. and Sharp, R, “Introducing HTML5”, 5th ed, pp. 1-13.

[2] BRIGHT, P., 2014. HTML5 specification finalized, squabbling over specs continues. From https://arstechnica.com/information-technology/2014/10/HTML5-specification-finalized-squabbling-over-who-writes-the-specs-continues/

[3] M. Kovatsch, M. Lanter and S. Duquennoy, "Actinium: a Restful runtime container for scriptable Internet of Things applications" in 3rd IEEE International Conference on the Internet of Things, Wuxi, 2012, pp. 135-142.

[4] K. Lei, Y. Ma and Z. Tan, "Performance Comparison and Evaluation of Web Development Technologies in PHP, Python, and Node.js," in IEEE 17th International Conference on Computational Science and Engineering (CSE), Chengdu, China, 2014, pp. 661-668.

[5] Eric Molin. "Comparison of Single-Page Application Frameworks". PhD thesis. KTH Royal Institute of Technology School of Computer Science and Communication, 2016, pp. 12-105

[6] Dasari Hermitha Curie, Jaison Joyce, Yadav Jyoti, et al. "Analysis on Web Frameworks", in Journal of Physics: Conference Series, pp. 10-15.

[7] Stefanov Stoyan, "React: Up and Running: Building web Applications" in first Edition, 2016. pp. 1-200

[8] Horton Adam, Vice Ryan, "Mastering React", February 23, 2016, pp. 12-19

[9] Stein Johannes, "ReactJS Cookbook", December 6, 2017, pp. 150-170

[10] Masiello Eric, "Mastering React Native", January 11, 2017.

[11] Pratik Sharad Maratkar , Pratibha Adkar "React JS - An Emerging Frontend JavaScript Library" Iconic Research And Engineering Journals Volume 4 Issue 12 2021, pp. 99-102