University of Zawia Journal of Engineering Sciences and Technology (UZJEST)
Volume 2, Issue 1, (2024), pp 1-13, ISSN 3006-2764

Modern Front-End Web Architecture Using React.js and Next.js

Hamed Hussen Ben kora1[1]*, Mohamed Sasi Manita[2]

[1] Department of Computer, Faculty of Education, University of Zawia, Zawia, Libya

[2] College of Computer Technology, Zawia, Libya

*Corresponding author email: h.benkora@zu.edu.ly

Received: 20-11-2023 | Accepted: 30-04-2024 | Available online: 15-06-2024 | DOI:10.26629/uzjest.2024.01

ABSTRACT

Before the advent of modern web development technologies, full-stack developers were the norm, handling both front-end and back-end aspects of applications. However, the current landscape has seen a separation of concerns, allowing front-end developers to focus solely on user interface elements like display and animation. This specialization has led to the rise of technologies like the React.js library. While React.js excels in client-side rendering, implementing server-side rendering requires the Next.js framework. This React framework offers a plethora of features, making it ideal for building the front-end of applications for University of Zawia (UZ)'s website and control panel applications. This study delves into the advantages and disadvantages of both React.js and Next.js as core technologies for UZ web application development. It aims to provide a comprehensive overview of the front-end development process and highlight the strengths and weaknesses of these technologies in shaping the user interface of the UZ application.

Keywords: Server-Side Rendering (SSR), Client-Side Rendering (CSR), React.js, Next.js, SEO (Search Engine Optimization)

How to cite this article:
Ben Koral, H.H.; Manita, M.S. Modern front-end web architecture using React.js and Next.js. Univ Zawia J Eng Sci Technol. 2024;2:1-13.

بنية واجهات الويب الحديثة باستخدام React.Js و Next.Js

حامد حسين بن كورة[1]، محمد ساسي مانيطة[2]

[1]قسم الحاسوب، كلية التربية الزاوية، جامعة الزاوية، الزاوية، ليبيا

[2]كلية تقنية الحاسوب، الزاوية، ليبيا

ملخـــــص البحـــــث

قبل ظهور تقنيات تطوير الويب الحديثة، كان مطورو البرامج الكاملة هم الأساس في كامل العمل، حيث كانوا يتعاملون مع الجوانب الأمامية والخلفية للتطبيقات. ومع ذلك، شهد المشهد الحالي فصلاً بين الاهتمامات، مما يسمح لمطوري الواجهة الأمامية بالتركيز فقط على عناصر واجهة المستخدم مثل العرض والرسوم المتحركة. وقد أدى هذا التخصص إلى ظهور تقنيات مثل مكتبة React.js. بينما يتفوق React.js في العرض من جانب العميل، فإن تنفيذ العرض من جانب الخادم

يتطلب إطار عمل Next.js. يوفر إطار عمل React مجموعة كبيرة من الميزات، مما يجعله مثاليًا لبناء الواجهة الأمامية لتطبيقات موقع جامعة الزاوية (UZ) وتطبيقات لوحة التحكم. تتعمق هذه الدراسة في مزايا وعيوب كل من React.js وNext.js كتقنيات أساسية لتطوير تطبيقات الويب UZ. ويهدف إلى تقديم نظرة شاملة عن عملية تطوير الواجهة الأمامية وتسليط الضوء على نقاط القوة والضعف في هذه التقنيات في تشكيل واجهة المستخدم لتطبيق UZ.

الكلمات المفتاحية: العرض من جانب الخادم (SSR)، والعرض من جانب العميل (CSR)، وReact.js، وNext.js، وSEO (تحسين محرك البحث).

1. Introduction

Web applications can now display more than just static data. Customer demand has made modern web applications an integral part of everyday organizations and ecosystems and they have become increasingly popular since their introduction because they provide a secure and efficient way to conduct such communications.

A great online front-end that combines convenience, practicality and functionality can attract consumers. Choosing the right library and front-end framework is one of the factors that affect the improvement of front-end web technologies. In general, JavaScript libraries and frameworks are very important to increase the scalability, maintainability, modularity and interactivity of web pages. So we created the UZ application using the Next.js framework and the React.js library.

Developing the UZ web application was a challenge as this application contributes to higher search engine rankings and is used by all students, members and staff at the university. As the user interface of the UZ system, the front-end web needs to be optimized in terms of user interface, user experience and application performance.

2. Literature Review

2.1 Document Object Model (DOM)

This When a web page is requested in a web browser, the browser displays the document in a tree structure and breaks down the HTML markup into what is called a document object schema [1].

The Document Object Model (“DOM”) defines the logical structure of an HTML document and essentially serves as an interface to web pages. Programming languages such as JavaScript allow you to access the DOM to manipulate and make your website interactive. When the page structure changes, the web page's DOM is also updated. You can think of the DOM structure as a tree diagram that represents all the elements in an HTML document that is loaded by a browser. Relationships between HTML tags can be represented in a dendrogram. An example of this is shown in Figure 1. This is a representation of the DOM structure of the code in the same diagram.

Front-end frameworks and libraries are becoming an important part of modern web development stacks. Like React js, React.js is a front-end library that is slowly becoming the preferred framework for modern web development in the JavaScript community.

2.1.1 React JS Library -is a declarative, efficient, and flexible open source JavaScript library carefully developed by Facebook to simplify the complex process of creating interactive user

interfaces. Think of a UI built with React as a collection of components responsible for outputting reusable HTML snippets [2].

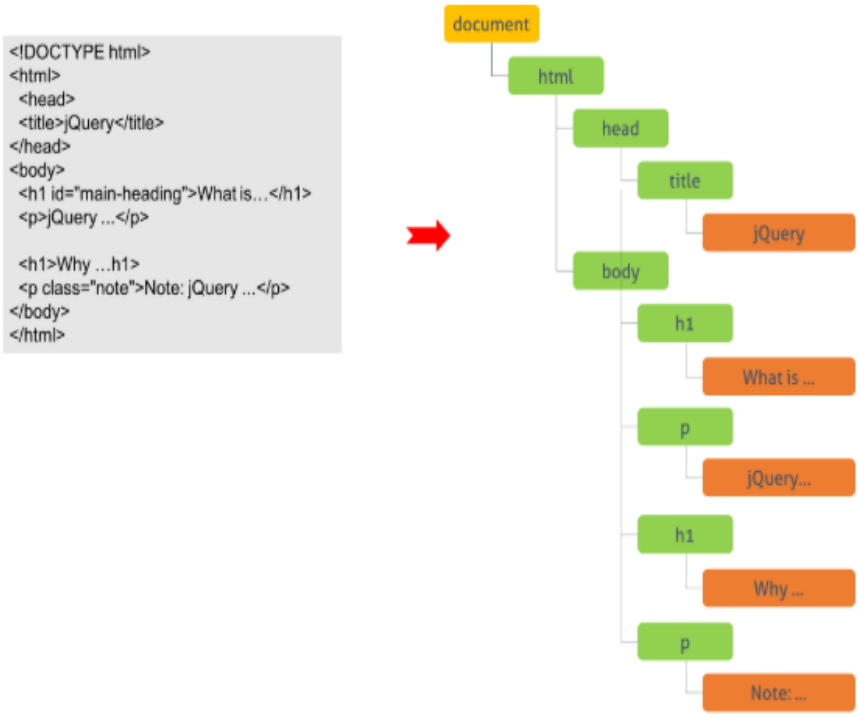

Figure 1 Syntax of HTML and DOM tree of the syntax

2.1.2 React Component - One of the core building blocks of React. In other words, all applications developed with React are made up of parts called components. Components make it easy to create user interfaces. You can see that your UI is divided into separate parts (called components), edit them individually, and then merge them all into a parent component that represents final UI. In Figure 2 below, Zawiya University's home page is divided into separate components

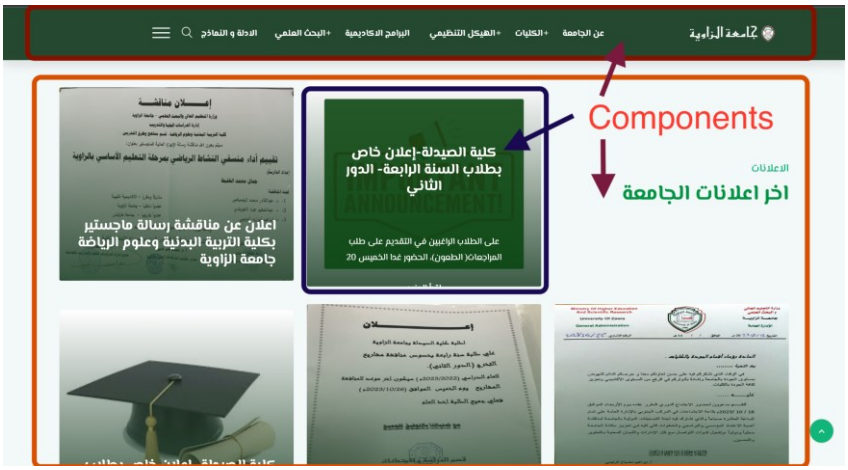

Figure 2. Zawiya University's home page components

React components essentially return JSX code that specifies what to render on the screen. There are two main types of components in React.

- Functional Components: Functional components are simply JS functions. React allows you to create functional components by writing JS functions. These functions may or may not take data as parameters. The following example shows a valid functional component in React [3].

```
const App = () =>
 <p>
    Hello World
 </p>
```

- Class components: Class components are slightly more complex than function components. Functional components are unaware of other components in the program, whereas class components can operate on each other. We can pass data from one class component to another class component. We can create class-based components in React using JS ES6 classes [4]. The following example shows a valid class-based component in React:

```
class App extends React.Component {
  render() {
     return  <p>
           Hello World
         </p>
  }
}
```

2.2 Virtual Document Object Model (VDOM)

React also uses a virtual DOM to help build web applications faster. Rather than updating all components again as in traditional web applications, the virtual DOM compares the previous state of components and updates only the elements that have changed in the real DOM. In Figure 3 below is an illustration of the working of VDOM.

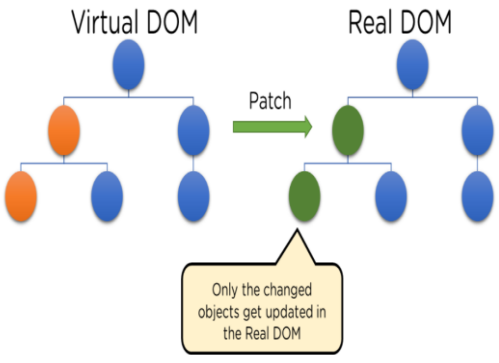

Figure 3 VDom

Redux - Redux is a very popular open source library that makes state management in applications easier. It is an open source, cross-platform library inspired by Facebook's Flux architecture. It removes unnecessary complexity present in Flux architecture. Redux is typically used with React using React-Redux, but it can also be used with other UI frameworks such as Angular, Vue.js, and Vanilla Javascript.

The main goal of Redux is to introduce predictability without sacrificing async speed when it comes to managing and updating application state. This approach limits how updates can be made, creating a more stable and testable application. One consequence of these limitations is that developers find Redux simple and effortless to use. If a problem occurs, the ability to time travel between previous application states provides accurate and efficient debugging [5].

Whatever components require state from a larger application, they can inherit it from a container or store created by Redux, without the need to pass props from one component to another.

Redux consists of three basic components and provides store, reducer and action as shown in Figure 4. This storage acts as a repository for application state and stores information. Triggered by a change in application state, performs an action to communicate the change to the Redux store. Alternatively, reducers act as pure functions, taking the current state of the application, performing operations and rendering the new state.

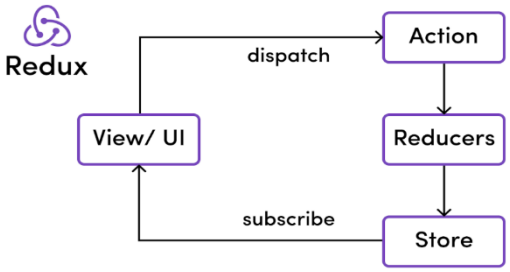

Figure 4 Redux's three basic components

2.3 Client-Side Rendering (CSR)

React.js, Angular, and vue.js use client-side rendering that runs in the browser. Client-side rendering refers to using JavaScript to render content in the browser. Therefore, on first load you receive only a minimal HTML document with a JavaScript file [2] Instead of receiving all content directly from the HTML document, the rest of the website is rendered using the browser. Although the initial page load is often a bit slow in client-side rendering, all subsequent pages load very quickly. This technology mainly communicates with the server to obtain runtime data. Additionally, the entire user interface does not have to be reloaded every time the server is called. By re-rendering specific DOM elements, the client-side framework is able to update the UI with updated data. Figure 5 illustrates the idea of CSR.

2.4 Server-Side Rendering (SSR)

Next.js uses server-side rendering, The server sends the HTML response to the browser over the Internet [6]. The browser then creates 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 as shown in Figure 6, and the entire process takes just a few milliseconds.

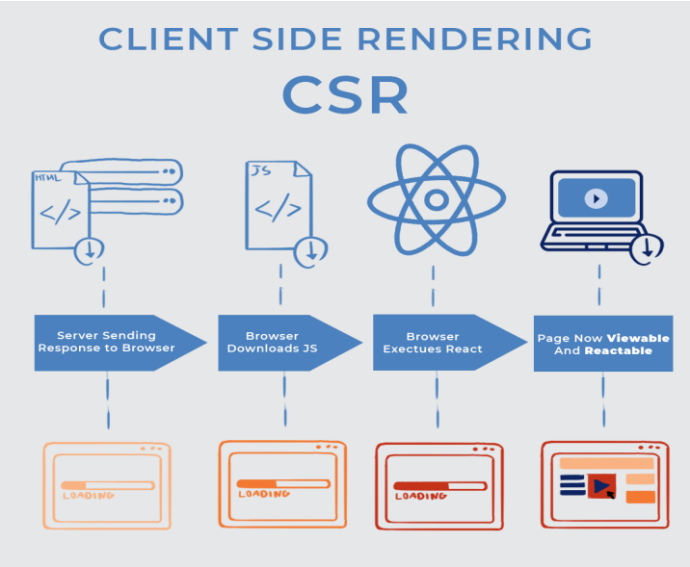

Figure 5 Client-Side Rendering

Webservice -The development of application software that uses web services is growing rapidly. Many technology companies use web services in their business. Web services are described as a method of exchanging or communicating information between devices over a network. The web server accesses the database to request the data required by the client and returns response data in JSON/XML format. SOAP (Simple Object Access Protocol) and REST (Representational State Transfer) are the two protocols most commonly used to exchange messages. REST API is a simple and easy-to-maintain API based on REST architecture.[7]

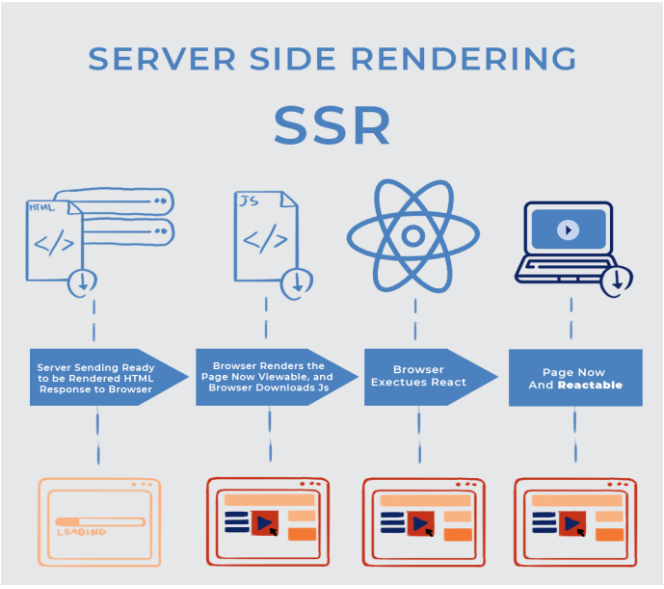

Figure 6 Server- side Rendering

API **-** Is a set of definitions and protocols for building and integrating application software. It is sometimes called a contract between an information provider and an information user, specifying what the consumer requests (the call) and what the producer requests (the response). For example, a weather

service's API design might require the user to provide a zip code and require the manufacturer to respond with a two-part response, the first part indicating high temperatures and the second part indicating low temperatures [8]. In other words, if you want to interact with a computer or system to retrieve information or perform a function, an API helps you tell the system what you want so that it can understand and fulfill the request.

Think of an API as an intermediary between users or clients and the resources or web services they want to retrieve. It's also a way for companies to share resources and information while maintaining security, control, authentication and determining who has access to what.

REST **–** REST is a set of architectural constraints, not a protocol or standard. API developers can implement their REST in different ways. When a client request is made via a RESTful API, a representation of the resource status is passed to the requestor or endpoint [9]. This information or representation can be provided over HTTP in any of the following formats: JSON (JavaScript Object Notation), HTML, XLT, Python, PHP, or plain text. Despite its name, JSON is the most commonly used file format because it is language independent and readable by humans and machines.

Other things to remember: Headers and parameters are also important for HTTP methods in RESTful API HTTP requests because they contain important identifiers for metadata, authorization, Uniform Resource Identifiers (URIs), caches, cookies, etc. Request more content. There are request headers and response headers, each containing its own HTTP connection information and status code.

3. METHODOLOGY

Our goal in this research is to develop Modern Front-end Web Architecture Using React.Js and Next.Js. To build such a system the Proposed Methodology will follow:

3.1. Project Setup and Planning:

- Defining Project Requirements: The functionality, features, and user experience goals of the Zawia University website are clearly defined.
- Choose a Folder Structure: A scalable folder structure is defined to organize React components, styles, data fetch logic, and selected Next.js files (pages, API paths).
- Define State Management: The state management library (Redux) is used to manage the global application state.

3.2. Building Components with React:

- Component Hierarchy: The UI is divided into reusable React components.
- Component Functionality: Implement logic and data handling within each component using React Hooks (useState, useEffect, etc.).
- Styling: Standard CSS modules, JS libraries used for encapsulated and maintainable styles.

3.3. Leveraging Next.js Features:

- Routing: Utilizing Next.js built-in routing for defining page structures and navigation. Next.js supports features like dynamic routing and nested layouts for complex UIs.

- Data Fetching: Leveraging Next.js Data Fetching methods like getStaticProps (for pre-rendering at build time) or getServerSideProps (for server-side rendering) to optimize performance and SEO.
- API Routes: Creating serverless API routes using Next.js to handle data fetching from external APIs or databases. This allows separation of concerns and keeps your UI logic clean.

3.4. Additional Considerations:

- Code Splitting: Next.js automatically code-splits your application, improving initial load times.
- Static Site Generation (SSG): Because zawia university website consider content-heavy website, we used SSG to pre-render content at build time, resulting in fast loading and improved SEO.
- Server-Side Rendering (SSR): SSR offers a better user experience as content is rendered on the server for the initial page load.

3.5. Testing and Deployment:

- Unit Testing: Testing React components to ensure their functionality and prevent regressions.
- End-to-End Testing: Using Postman tool for end-to-end testing to simulate user interactions and test overall application behaviour.
- Deployment: Deploying Next.js application (Zawia University) to a hosting platform.

4. Results and Discussion

This stage aims analyze the data flow in the React component structure of the UZ application. after connecting the API URL to the application, the data provided by the backend can be displayed in the application view.

4.1 Advantage of REACT.JS

Based on the literature review and UZ application case analysis, the advantages of the React.js library are explained as follows:

- ***JavaScript Extension*** - JSX means you can write JavaScript code within HTML code. Additionally, JSX makes code easier to understand and debug.

Creating dynamic web applications that exclusively use HTML strings is difficult because it requires complex coding.uses JSX (JavaScript Extensions) solves this problem and simplifies it .It requires less coding and provides more functionality.

- ***Reusable Components*** - The UZ web application consists of multiple components, each with its own logic and controls. These components are responsible for outputting a small piece of reusable HTML code that reused wherever needed. Reusable code helps make UZ applications

easier and maintain. By nesting these components with other components, complex UZ applications built using simple building blocks.

- ***ReactJS uses a virtual DOM***-based mechanism to populate data into the HTML DOM. Virtual DOM runs fast because it only changes individual DOM elements instead of reloading the entire DOM every time.

4.2 Disadvantages of REACT.JS

Based on the literature review and UZ application case study, the shortcomings of the React.js library are explained as follows:

- ***Using third-party libraries for page Routing*** - To create an application with multiple page routes additional libraries such as react-router-dom needed.as shown in figure7

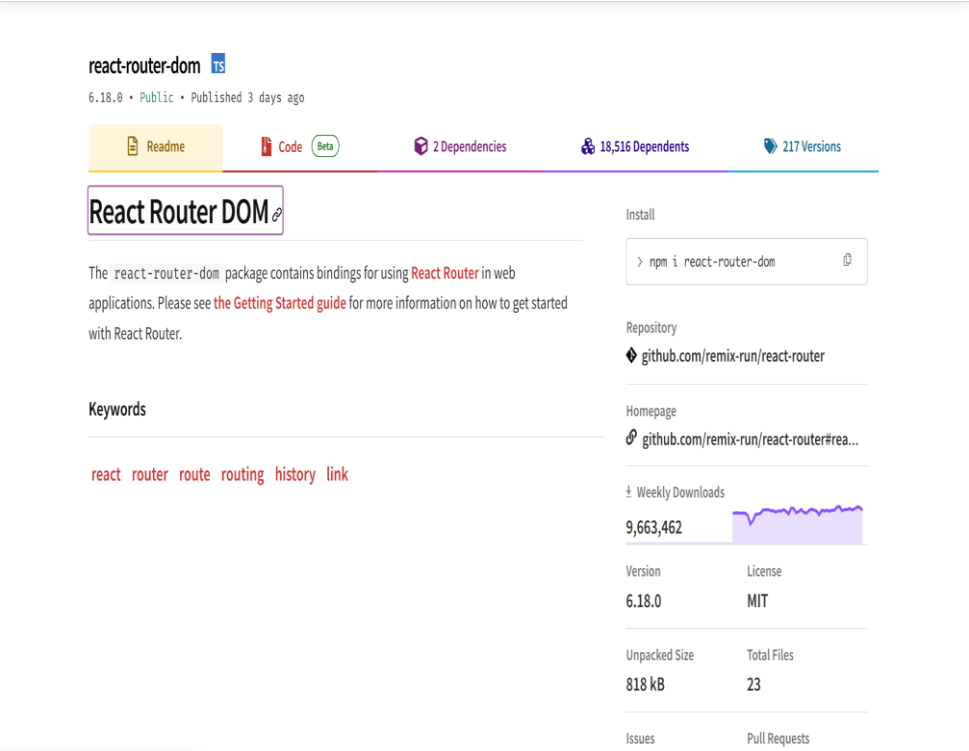

Figure 7 React-router-Dom Package

- ***Not Good for SEO -*** Although React makes it easier to create beautiful dynamic content, it is still a single-page application. This means your entire web application runs on a single page. It's great for dynamic applications, but not so great for SEO. It's difficult for meta tags to describe every relevant aspect of your application. Secondly, single page applications do not require additional routing libraries. In order to establish expertise and authority on the Internet, your application must have a content-rich website with valuable articles, semantic keywords, and properly tagged meta data. In our case when applied Google chrome SEO tool in react page the result in figure 8,9 shows the loss of meta data in react.

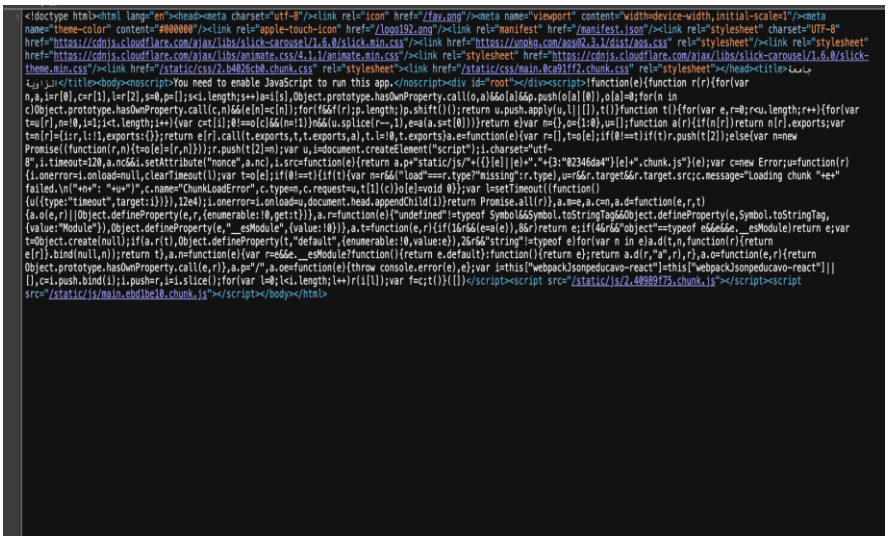

Figure8 View source react page

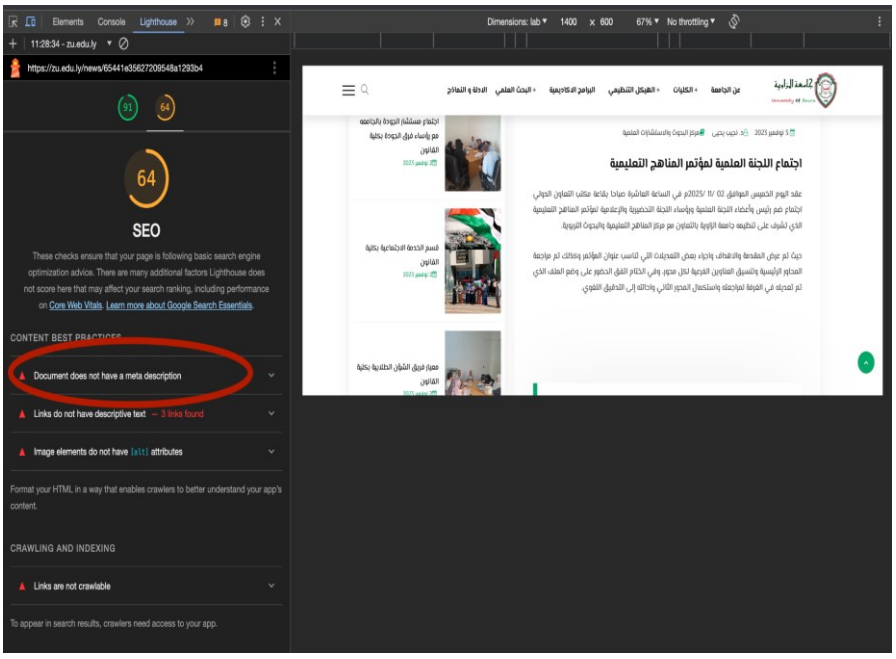

Figure 9 Google chrome SEO tool result

4.3 Advantage of NEXT.JS

Based on the literature review and UZ application case analysis, the advantages of the React.js library are explained as follows:

Simple routing mechanism - The Next.js framework already supports a simple routing mechanism in the pages folder, so no third-party libraries are required for routing. Create a folder or file in the Pages folder and name it what you want your route to be called as shown in figure 10. Then the URL address bar automatically points to the file name in the page folder. To redirect between pages, you can use the Next.js link component and pass the appropriate URL to redirect to.

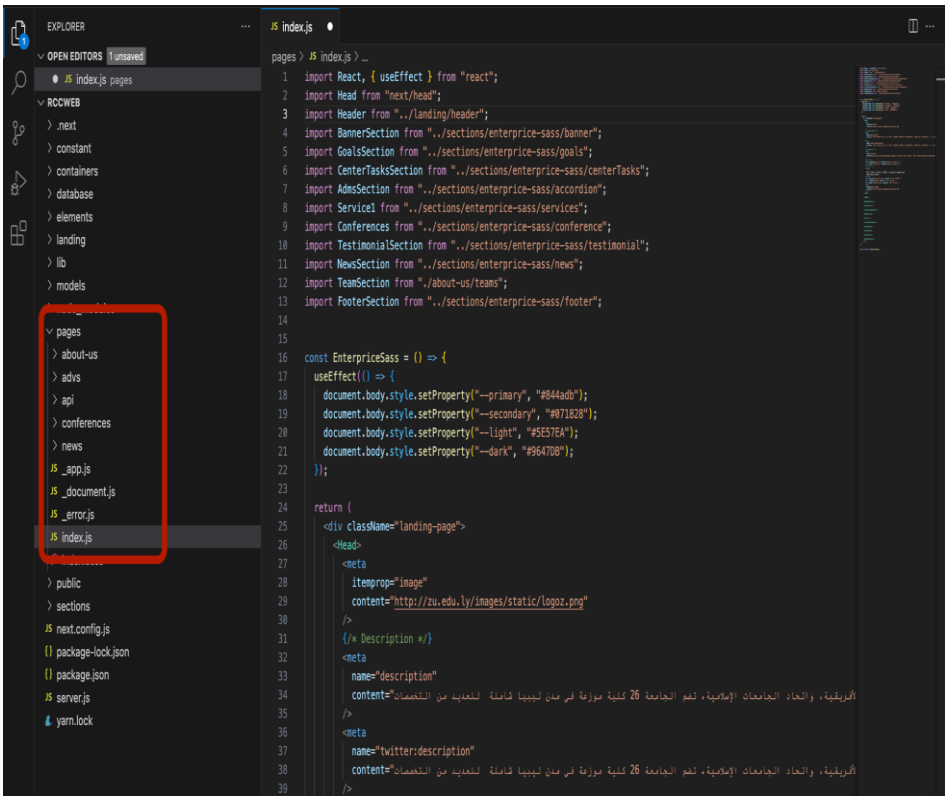

Figure10 Routing mechanism for Next.js

Various Data Fetching Mechanism - According to the official documentation, unlike simple React.js applications that only support data retrieval using CSR, the Next.js framework supports data retrieval mechanisms through CSR, SSR, SSG, and ISR. So that the data retrieval mechanism can be customized according to the application needs.

Good for SEO – Since Next.js uses SSR, SSR converts React code to HTML on the first request. In comparison, a typical SPA site delivers a heavy Javascript payload to the browser. Most of the HTML code displayed by browsers is generated by Javascript. This difference in loading strategy has a significant impact on search rankings.

SEO bots crawl the web and check the HTML of every URL they encounter. When it reaches a non-SSR SPA page, it serves a lot of Javascript instead of HTML. SEO robots don't understand Javascript and don't always wait for it to be converted to HTML. This means that the robots that determine what appears on the first page of search results cannot read that page. If they can't read the page, Google won't show it as the "best" answer option for the user's search. This is where Next.js can help. In our case when applied Google chrome SEO tool in Next.js page the result in figure 11,12 shows the fetching of meta data in Next.js.

By encapsulating SPA pages in NextJS, all queries to the website URL always generate robot-friendly HTML pages. It offers the best of both worlds. It can create SPA websites while getting the SEO benefits of traditional website architecture.

Figure 11View source Next.js page

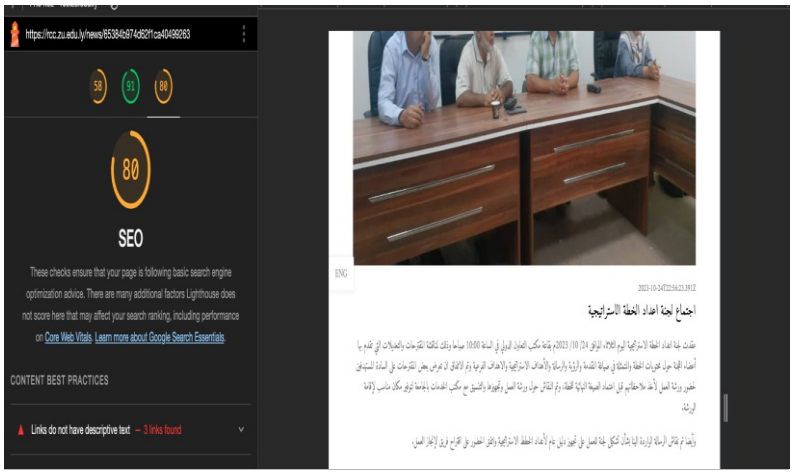

Figure 12 Google chrome SEO tool result

4.4 Disadvantages of NEXT.JS

Based on the literature review and UZ application case analysis, the shortcomings of the Next.js framework are as follows:

- High server load Since Next.js framework uses SSR (Server-Side Rendering) method in UZ application, the server load increases due to frequent requests to the server.
- You must wait for the website to become interactive Pre-rendering using SSR (Server-Side Rendering) can cause the user to enter the page and then the HTML file has been rendered and served by the server. However, until the browser successfully runs React, the user can only see the page view and cannot interact with the page.
- Full page reloads by using SSR (Server-Side Rendering), when the user changes the page, the entire page reloads because the user has to request an HTML file that has been rendered on the server.

5. Conclusion

The study discusses the use of React.js and Next.js in developing front-end applications for the University of Zawia's website. React.js simplifies creating interactive user interfaces, while Next.js offers features like server-side rendering for better SEO. React components and JSX enhance code readability and reusability. However, the results show that React.js may require additional libraries for routing and can be challenging for SEO. Next.js simplifies routing, supports various data fetching mechanisms, and improves SEO through server-side rendering. Yet, it may lead to high server loads and delays in interactivity.

REFERENCES

[1] "JavaScript HTML DOM." Accessed: Nov. 14, 2023. [Online]. Available: https://www.w3schools.com/js/js_htmldom.asp
[2] H. A. Jartarghar and G. R. Salanke, "React Apps with Server-Side Rendering: Next.js," vol. 14, no. 4, Art. no. 4.
[3] "React component guide: Class vs functional," Educative. Accessed: Nov. 05, 2023. [Online]. Available: https://www.educative.io/blog/react-component-class-vs-functional
[4] "ReactJS Class Components," GeeksforGeeks. Accessed: Nov. 05, 2023. [Online]. Available: https://www.geeksforgeeks.org/reactjs-class-components/
[5] D. Bugl, Learning Redux. Packt Publishing Ltd, 2017.
[6] "Client-Side Vs. Server-Side Rendering." Accessed: Nov. 16, 2023. [Online]. Available: https://www.searchenginejournal.com/client-side-vs-server-side/482574/#close
[7] M. F. S. Lazuardy and D. Anggraini, "Modern Front End Web Architectures with React.Js and Next.Js," vol. 7, no. 1, Art. no. 1.
[8] "What is an API?" Accessed: Nov. 03, 2023. [Online]. Available: https://www.redhat.com/en/topics/api/what-are-application-programming-interfaces
[9] Chathumofficial, "REST API & Koa JS," Medium. Accessed: Nov. 02, 2023. [Online]. Available: https://medium.com/@chathum2000official/rest-api-koa-js-1e84affc5040