Implementation of E-Commerce Responsive Website

[1]R.Ramprashath, [2]R.Gomathi, [3]Vivek Biswas, [4]S.Madhuvarshine, [5]K.Tamil Selvan
[1,2]Assistant Professor, [3,4,5] Student,
[1,2,3,4,5]Department of MCA,
[1,2,3,4,5]Karpagam College of Engineering, Coimbatore, Tamil Nadu.

Abstract : This work aims to provide a seamless shopping site using ReactJS, Bootstrap 4, SASS (SCSS) and HTML 5 for frontend. ReactJS is preferred here, as it provides the concept of Virtual DOM. By using the Virtual DOM concept, the webpage can be updated without having to update or refresh the entire webpage. Also, manipulation of physical DOM involves a tedious process which affects the site performance. This proposed work aims to eliminate this problem by using the diffing method where the initially rendered Virtual DOM is compared with the previous state when a change occurs and only the changed part is updated. The diffing process is backed by the component method. In ReactJS, every element can be derived from a component. This makes the diffing process easier. The backend used in this project uses MongoDB which is an unstructured database that stores data in the form of JSON data unlike the traditional table structured data. The interface that acts between the frontend and the backend is Springboot.

IndexTerms **– ReactJS, Virtual DOM, JSON, MongoDB, Bootstrap, Product Decsription Page , Spring Boot, Java Script.**

I. INTRODUCTION

The proposed work aims to provide an attractive shopping website for the client. It uses the latest and the best the current web technology trend could possibly provide. The website is developed using ReactJs, SCSS (SASS CSS), HTML, JavaScript, JSON, NodeJS and Bootstrap. Some of the components on the website are also developed using Skava Studio. The Backend is handled by a separate team which can be fetched through API calls in JSON format and these can be customized by passing headers or query parameters during request. The backend is designed using MongoDB and Springboot (Java using Maven build tool).

II. EXISTING SYSTEM

The existing system is designed using various technology like plain HTML or JS or CSS. Mostly the rendering of the website happens at the client side thus reducing the performance of the site on client side. The client, on loading the site, will begin downloading all the scripts that render the page, this could at times cause freezing of the browser or website. Most web technology modifies the physical DOM element causing a single change to re-render the entire webpage. The most common databases used are MySQL, Oracle, MariaDB etc which follows the RDBMS storage method which increases the memory usage. These type of storage methods also increase the time required for storage and retrieval of data.

III. DRAWBACKS OF EXISTING SYSTEM

- Performance while loading contents on a web page is poor.
- The over usage of either Server-Side Rendering or Client-Side Rendering could cause exhaustion of resources on both client and server end.
- Modifications to physical DOM elements reduce the performance of a website.
- Creation of individual web page is required, thus complicating site maintenance.
- Maintaining data in the form of relations take up more space.
- Usage of traditional RDBMS techniques are becoming slower due to the time taken to query and process the query every time a request is made.
- May lead to complexities while handling very large application with huge amounts of web documents.

IV. PROPOSED SYSTEM

- The proposed system utilizes the features of Node.js, ReactJs, React-Bootstrap, HTML5, SCSS, ES5 or 6 and JSON.
- ReactJs uses the concept of virtual DOM.
- Virtual DOM creates multiple copies of a page to identify the changes and update the current page accordingly. This process is known as diff-ing.
- React-Bootstrap is a package used for styling components in the webpage.
- JS is powerful that it will soon be able to interact with the backend directly rather than just making API calls.
- Using MongoDB improves the performance drastically by reducing the query processing time and reducing the storage consumption.

4.1Scope

- The scope of this project is to design a single page application using ReactJS.
- Utilize the full potential of Virtual DOM concept to increase performance and dynamicity of the webpages.
- Provide better backend support with the help of MongoDB and Springboot to efficiently store or retrieve data from the Database server.
- Obtain maximum client satisfaction by providing the best quality end product.
- Reduce load on client and server side using appropriate rendering.

4.2 System Requirement Specification

4.2.1 Functional Requirement

- A UI containing details of all available products, promotions and product description.
- Feature to enable auto subscription to products from the Product Description Page (PDP).
- Login and Signup page for users, also to provide them with appropriate recommendations (Targeted Recommendations).
- Secured payment method and should abide to the laws and regulations of the United States.

4.2.1 Non Functional Requirement

- Support for translation to Chinese and Russian language.
- Compliant to all the Government policies stated by the geographical boundary.

4.3 End-User System Reuirements

4.3.1 Hardware

- Requirements are dependent on the web browser used by the end user.

4.3.2 Software

- Google Chrome (73.x or above)
- Firefox (66.x or above)
- Internet Explorer 11
- Safari (8.x or above)
- Microsoft Edge (38.x or above)

V. MODULE DESCRIPTION

5.1 Home Page

- Default or landing page for all users.
- Displays all ongoing offers or special sales.
- Lists top products based on user purchase and feedback.

5.2 Sign-Up Page

- Creates a new user account when all the mandatory fields are filled and are valid.
- The new user is asked to enter valid Email ID, password, phone number and full name to register for an account. If valid the user account will be created, else the user will be asked to check the inputs again and correct them.

5.3 Sign-In Form

- Performs necessary operations for the user to login into the site safely.
- Provide the option to reset password for the user.
- Allow a new user to create an account.

5.4 Product Listing Page

- Lists the available products and possible filters.
- Filters such as price, product type and ounce are available for narrowing the choices

5.5 Checkout Page

- A checkout page where user needs to fill out necessary details for successful delivery of the product.
- This page requires the full name, address, payment method and card details of the customer.

VI. PSEUDOCODE

6.1 Signin:

GET EmailID

GET Password
IF (EmailID == EnteredEmailID and and Password == EnteredPassword) THEN
Login Successful
ELSE
Login Failed.
ENDIF

6.2 Signup:

GET EmailID
GET FirstName
GET LastName
GET Password
GET ConfirmationPassword
GET PhoneNumber
IF (Username != ‘’ and and Password != ‘’ and and ConfirmationPassword != ‘’ PhoneNumber != ‘’ and and FirstName!=‘’ and and LastName != ‘’ and and Password == ConfirmationPassword) THEN
Proceed Signup Process
ELSE
Correct or provide valid data.
ENDIF

6.3 Cart:

GET Quantity
IF (Quantity > 0 and and Quantity < StockAvailability) THEN
IF (Quantity < 6) THEN
Navigate to checkout page.
ELSE
Message: Only 5 items per user is allowed.
ENDIF
ELSE
Product out of stock.
ENDIF

VII. SCREENSHOTS

8.1 Home Page

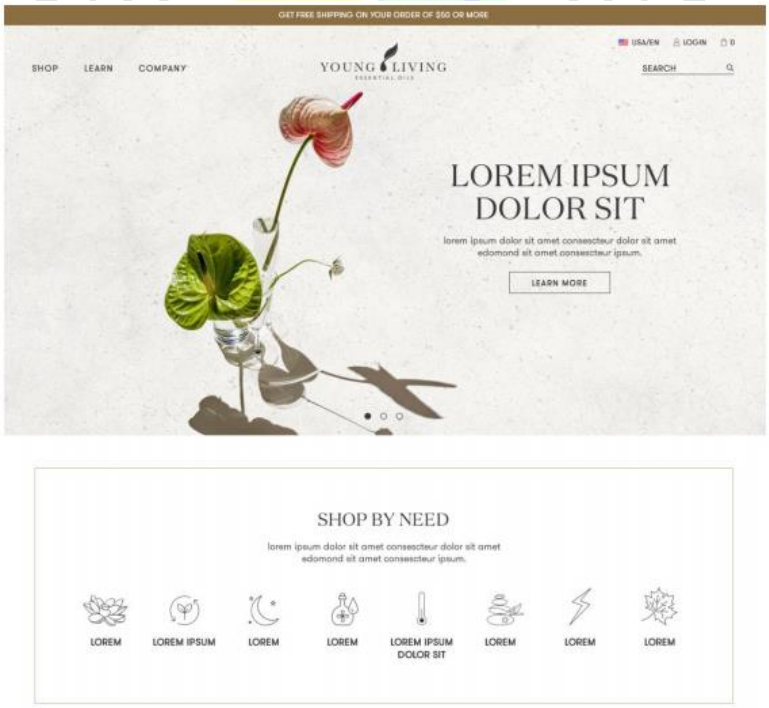

Figure 8.1Home Page

8.2 Sign-Up Page

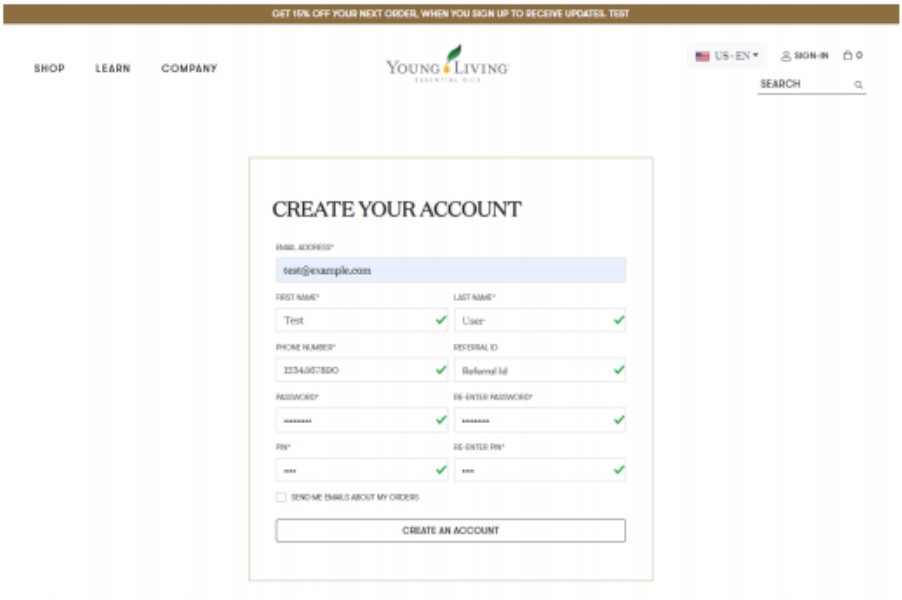

Figure 8.2 Sign-Up Page

8.3 Sign-In Page

8.3.1 E-Mail Address

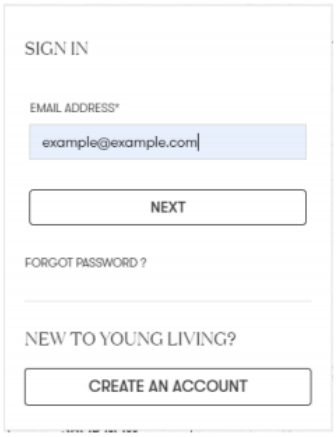

Figure 8.3.1 E-Mail Address

8.3.2 Password

Figure 8.3.2 Password

8.4 Product List Page

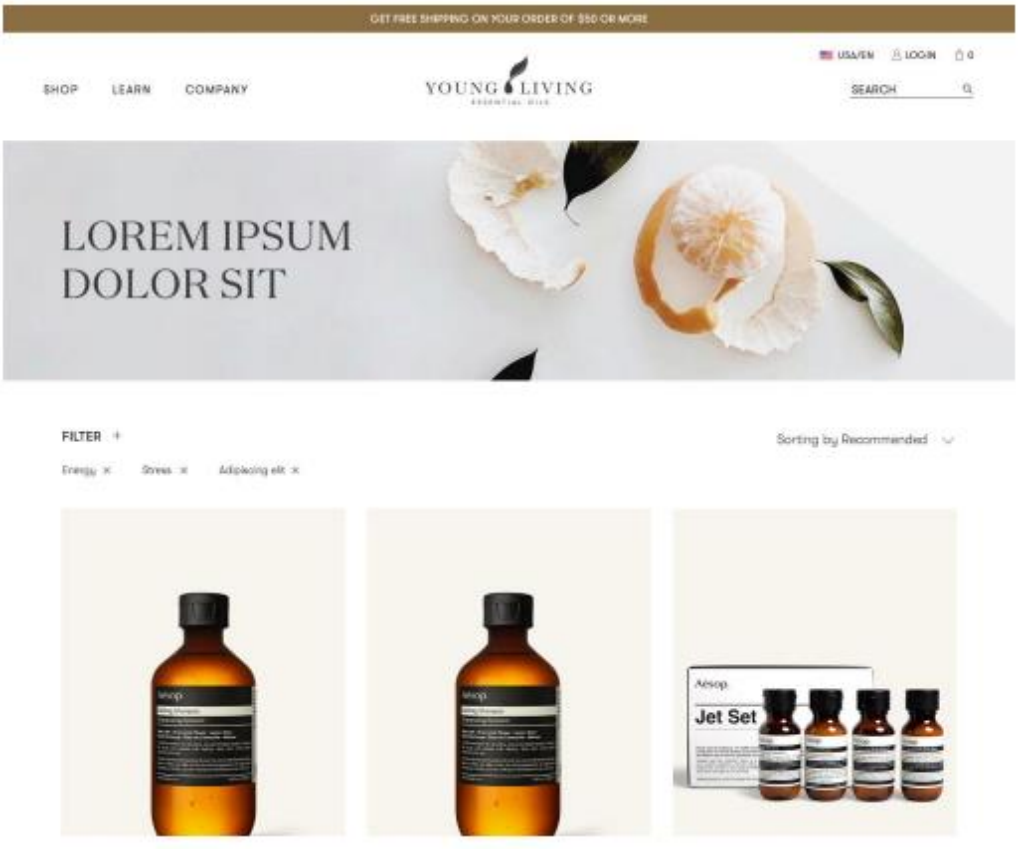

Figure 8.4 Product List Page

8.5 Checkout Page

8.5.1 Shipping Address

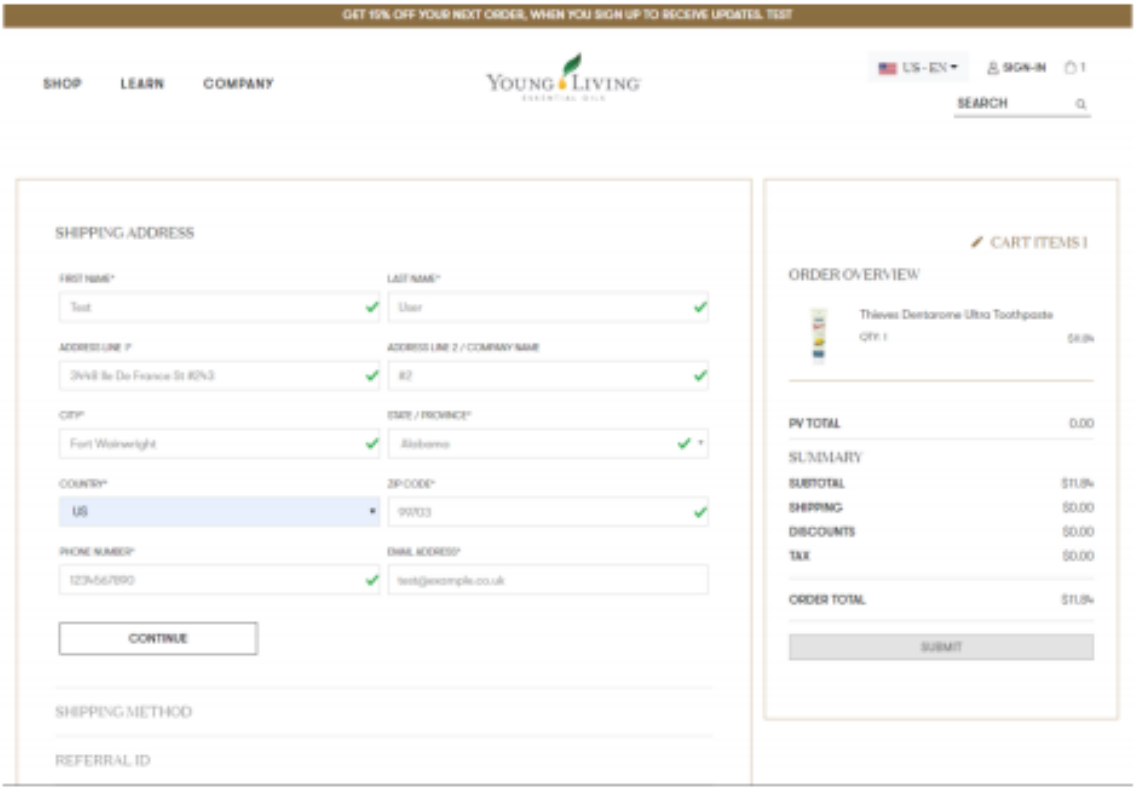

Figure 8.5.1 Shipping Address

8.5.2 Shipping Method

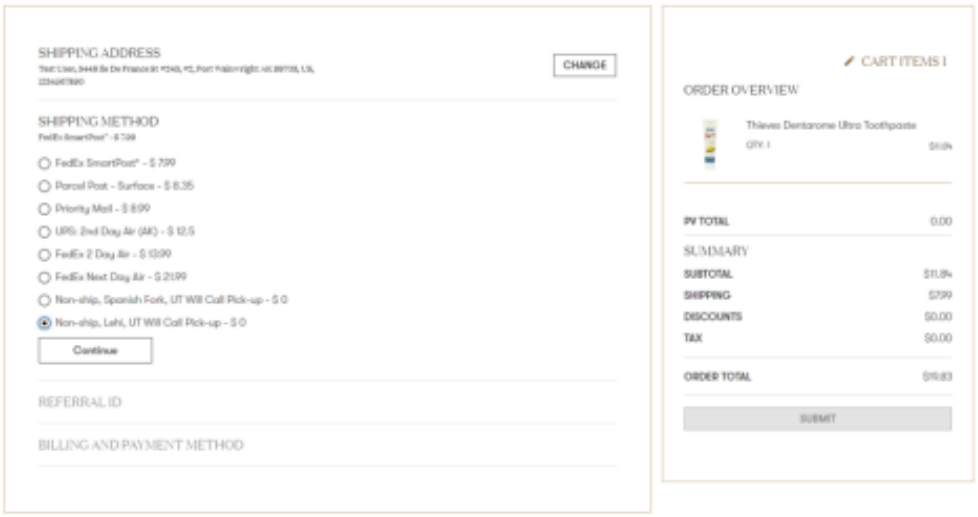

Figure 8.5.2 Shipping Method

8.5.2 Payment

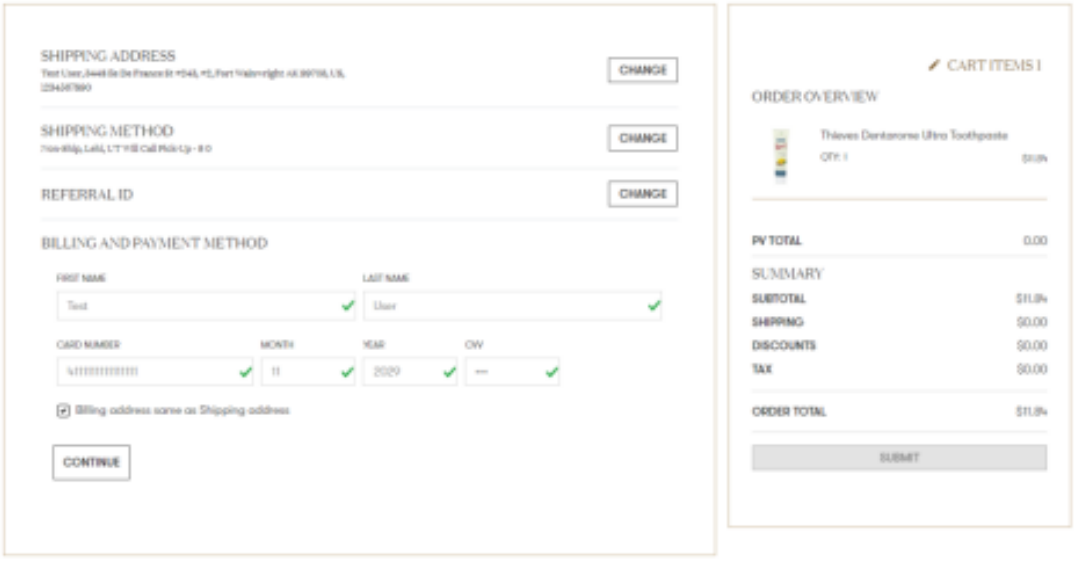

Figure 8.5.2 Payment

VIII. CONCLUSION

The software is built using ReactJS, which is an advanced framework of JavaScript. It helps to develop a single page application by utilizing the Virtual DOM. Since ReactJS supports server-side rendering and client-side rendering, it is easy to manage load on both server-side and the client-side. ReactJS also uses the Loadable-Component package to break the page specific JavaScript files into smaller chunks to allow the user to load pages quickly by only downloading the necessary JavaScript and Style Sheet files for a particular page. As ReactJS is booming at present, it is receiving more attention and better community support along with frequent updates.

REFERENCES

[1] Anthony Accomazzo, Nate Murray & Ari Lerner "Fullstack React: The Complete Guide to ReactJS and Friends" - 2017.
[2] Kirill Konshin "Server-side rendering done right" - 2018.
[3] Sagar Ganatra "Routing in React applications made easy" -2018.
[4] Jacob Lett "A Beginner's Guide to Building Responsive Layouts with Bootstrap 4" – 2018.
[5] Jacob Lett "Bootstrap 4 and 3 Cheat Sheets Collection" - 2018.
[6] www.reactjs.org/docs/getting-started.html
[7] www.github.com/jamiebuilds/react-loadable
[8] www.jaredpalmer.com/formik/docs/api/formik