Indonesian Journal of Electrical Engineering and Computer Science
Vol. 24, No. 2, November 2021, pp. 978~985
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v24.i2.pp978-985

Modern tools and current trends in web-development

Debani Prashad Mishra[1], Kshirod Kumar Rout[2], Surender Reddy Salkuti[3]
[1,2]Department of Electrical Engineering, International Institute of Information Technology (IIIT) Bhubaneswar, Gothapatna, India
[3]Department of Railroad and Electrical Engineering, Woosong University, Daejeon, Republic of Korea

Article Info

Article history:

Received Jan 7, 2021
Revised Sep 7, 2021
Accepted Sep 16, 2021

Keywords:

Adobe Xd
MongoDB
Nodejs
Reactjs
Redux
SCSS

ABSTRACT

In this paper, a social media platform like LinkedIn and Facebook is made using MongoDB as a database. This paper aims to touch all the modern tools required to make an efficient web app, keeping in mind both the customer satisfaction and the ease for the developers to make their web designs, front-end and back-end. In this application, a user could make an account, add or delete details of their profile, education, and experience fields. The users could post, also comment and even like a post of other users. A monolithic architectural approach is used for simplicity in maintaining the database. Postman application programming interface (API) was used to check the working of the back-end. Git, Github, and Heroku were used to deploy the website. Node package manager (NPM) packages like bcrypt and validator are used to encrypt passwords and to validate a user during login. Media queries are used in cascading style sheets (CSS) to achieve a responsive design. Therefore, the users could view the website through a mobile phone, i-pad and also a personal computer (PC), maintaining the readability and design across all these devices.

Corresponding Author:

Surender Reddy Salkuti
Department of Railroad and Electrical Engineering
Woosong University
Jayang-dong, Dong-gu, Daejeon-34606, Republic of Korea
Email: surender@wsu.ac.kr

1. INTRODUCTION

Since the 1990s, after the introduction of the world wide web (WWW) [1], by Tim Berners Lee, a British scientist. Which was used to communicate, between scientists from all around the world. And at that time, tim had written the three fundamental technologies of a web, which remain as the foundation: hypertext markup language (HTML) [2], uniform resource identifier (URI) and hypertext transfer protocol (HTTP). Followed by many languages such as cascading style sheets (CSS) [3], java, hypertext preprocessor (PHP), JavaScript [4], Python, structured query language (SQL), Angular, and many more were introduced over the years to implement web technology; keeping in mind both web security issues and the ease to programmers in the programming language. Today, for any web application the 3 building block coding languages required are HTML, CSS, and JavaScript. And nowadays, user interface/user experience (UI/UX) design is equally given importance, which involves the extensive use of HTML and CSS.

Being a web developer, the terms monolithic and micro-services architecture [5], [6] must be familiar. What a monolithic application means is that the application has its authentication, posts, profiles, database, and servers all in one place. In the case of a micro-service application, its post section is separately a different application, similarly, the authentication is another application and the profile. An application with monolithic architecture is dependent on all its parts if one of the sections, for example, the authentication

system of that application failed due to some error the entire application would be disrupted. This is not the case with a micro-service architecture. Figure 1 depicts the flow chart of these architectures.

If one of the applications in micro-service fails it wouldn't affect other applications within the application. As in the previous example if the authentication has an error, then all other routes leaving the authenticated routes can be accessed, which is in contrast to a monolithic architecture. Because all these authentication, posts, and profiles services reside in their server. And connectivity is provided to all these services through an event bus, it's a common pipeline of data to all services. For example, if a user is authenticated, the authentication service gives this information to the event bus and from here all other application receives this update, and accordingly the services provides protected routes. The advantage of using a micro-service is the scale-ability and improvements applied to its service rather than the entire application as in the case in the case of a monolithic architecture. A micro-services architecture is achieved through Kubernetes. However, when keeping the time frame in mind, a monolithic architecture could be built faster than a micro-service architecture. And also due to the number of people involved in handling the database in the initial stage would be less, a micro-service application becomes tedious in this case. Figure 2 shows the application of jest.

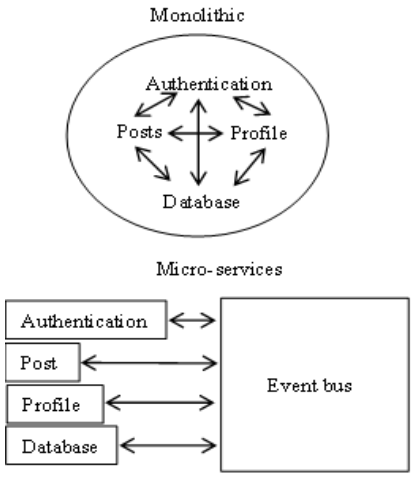

Figure 1. Monolithic and micro-services architecture comparison

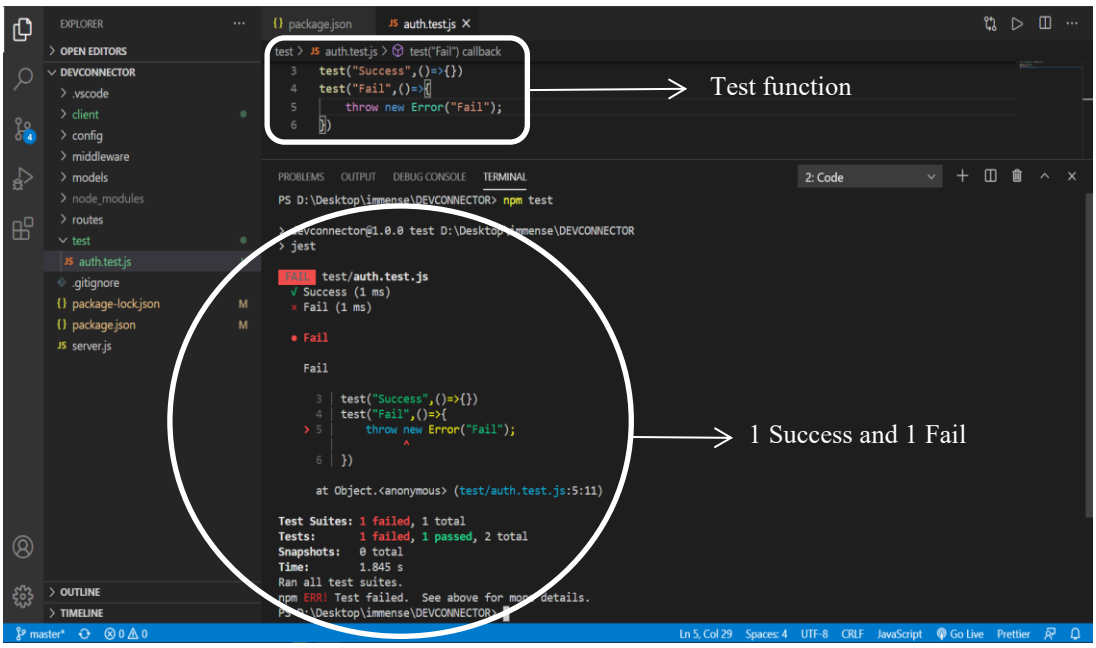

Figure 2. Testing an application using Jest

For an application apart from the selection of the database architecture, it is also important that we test our application's working. Jest and Mocha are the two popular JavaScript testing frameworks. Jest is a comparatively much more user-friendly framework than other frameworks [7], [8]. And jest library is specific to Node applications. The latest version of jest is 26.6. While installing the latest jest, use "npm i jest@26.6.0--save-dev", and we want to save jest as a dependency locally into our machine so that we could test it locally before the production stage of our application. A function test("Success",()=>{}) for pass in jest and a function test("Fail",()=>{throw new Error("Fail");}) to check failed condition in jest. The function test () is globally defined in the jest library, so we could use it anywhere in our application to test the various routes and function operations.

This work is to demonstrate and aware the readers of the present tools that are required to become an efficient web-developer. Efficient meaning that the current requirement in Web-Tech is fulfilled, along with the developer tools that are required for the developers to ease their workflow. Here, a social media platform like LinkedIn and Facebook is made using MongoDB as a database. Adobe Xd is used for UX designing [9]-[12]. Adobe Xd is a user-friendly software where we could make the prototype of our application's UI. For an application, we can't just blindly start coding the CSS and React part without having a plan of what the application must look like. This is when we use Adobe Xd. The UI design was written in SCSS [13]-[15], and then compile to CSS using the syntactically awesome style sheet (SCSS) npm packages. And Firefox provides a CSS environment that is user-friendly as well in building the UI. React is used to render entire applications front-end.

Since the application was still in its development model, a monolithic architectural approach was been used. Moreover, for start-ups a monolithic database is preferable. A micro-service architecture can be used when the business expands. And also testing of an application as the application grows is equally important, it can be achieved through setting up an automated testing application like jest. Postman API was used to check the working of the back-end server and REST API's. Git, Github, and Heroku were used to deploy the website. NPM packages like bcrypt and validator are used to encrypt passwords and to validate a user during login.

2. RESEARCH METHODS

This section contains a brief introduction to all the currently followed trends in web development. The present work is to demonstrate and aware the readers of the present tools that are required to become an efficient web-developer. Efficient meaning that the current requirement in Web-Tech is fulfilled, along with the developer tools that are required for the developers to ease their workflow. Here, the descriptin and the importance of MongoDB/Non-SQL JSON, Mongoose, Node.js and React.js are presented. An explanation as to why these trends are preferred the most are also mentioned in this section.

2.1. MongoDB/Non-SQL JSON type data as a database

For any application today, it must be accessible to its users as quickly as possible. The accessibility of an application is most crucial and also one of the main deciding factors of a customer's satisfaction. And from the business point of view, one requires to scale their application to reach more customers. So, there is a need for a database that could handle multiple users (millions of users). The traditional relational database managed system (RDBMS), which uses structured query language (SQL) has its limitations when it comes to handling huge volumes of data. The time complexity involved in processing data in SQL is significantly high as the number of customers increase.

In 1998 Carlo Strozzi gave an introduction to a non-relational type of data management system which he called "NoSQL". Where the data does not contain a fixed schema, unlike RDBMS, and stores data in JavaScript object notation format (JSON) and not in tables, which makes NoSQL more flexible. Moreover, in such systems rather than the previously used normalized database, these systems use the modern denormalized databases, where the data can be queried at a much faster rate. Currently, the most preferred database management system used extensively is MongoDB [16]-[18]. It uses json format for data storage and is a non-SQL type of database system.

In [16], an analysis of MySQL and MongoDB is made. MySQL uses an RDBMS type database management system and MongoDB is a No-SQL data management system as discussed above. Both MySQL and MongoDB are the top companies in their respective database management services. So, it's apt to compare RDBMS and non-RDBMS technologies using these service providers. A total of three tests were made in [16]. The test was for all operations, insertion, update, and deletion of data. In the 1st test, 500 users were created, 500 individual orders were created and 500 items were created. The worst-case for MongoDB is 0.0474 seconds and for MySQL, it was 0.7954 seconds. In the 2nd test, 1000 users were created, 1000 users patch/put requests to edit data, the worst-case time for MongoDB is 2.3201 seconds and for MySQL, it was 93.0153 seconds. The 3rd and the last test conducted had 7,000 users' creation and 10,000 users update (put/patch request to edit), the worst case in MongoDB is 23.3427 seconds and that of MySQL was a whopping 324.7654 seconds in the worst case. Through the observations, it can be seen clearly that as the number of data increases an RDBMS system takes an exponentially higher amount of time in performing the create, read, update and delete (CRUD) operations.

2.2. Mongoose

A MongoDB collection is queried by an object data mapper (ODM). An ODM is an object associated with a non-SQL database. As the name suggests, an ODM maps a document-related module. And one of the features of an ODM is that it assigns a unique ObjectId to all its data in the MongoDB database. Hence, these objectid's could be used later to fetch all the data stored with respect to that particular Id. For the purpose of querying the MongoDB database, an ODM library called Mongoose [19]-[21] is used in both MongoDB and NodeJs. It manages data relationships, provides schema validations, and is used in MongoDB to translate between objects in code and to represent those objects. While making a mongoose model one must follow certain rules. A Mongoose model function must have a name and a schema that enforces the type of data that can be referred to be accessed and manipulated. The data structure is defined for a model by the Mongoose schema. The model feature provides an interface for interacting with the document information stored at MongoDB. The following example shows a schema for a user model, which has a name, email, and password as its attributes. And all of them are of type string, and the required keyword is called a validator in Mongoose. It's a feature that validates a user based on their name and password.

2.3. Node.js

Node.js (node) is a server-side JavaScript environment [22]-[24]. It uses Google's open-source V8 engine as its default JavaScript engine which executes the JavaScript code. Both V8 and node are mostly written in C/C++, concentrating on productivity and low memory usage. But, the purpose of a V8 is to support JavaScript in the browser (Google Chrome browser), and that of node is to run a server process. Node.js is an asynchronous I/O eventing model. As JavaScript is an asynchronous language, so does node. A single thread is used to handle multiple requests. What this means is that all the requests are accessed, and the longest request is executed last. Behind the scene, the longest request is running in the background while the other requests are executed and have been served at the same time, using the async and await keywords. This is in contrast to a synchronous architecture where a request is served one at a time, so until an ongoing request is not yet served all other incoming requests are temporarily put to a halt. And this explains why MongoDB is much faster than MySQL. An RDBMS system is synchronous, which makes it slower. And this becomes more noticeable as the number of user increase. In a business making your customers wait is undesirable. So in order to overcome this issue, we must increase the number of threads, so that more users could access a request. For more threads to be available more hardware is required. This is the major disadvantage of synchronous architecture. In case the number of users increased in an exponential manner, we require much more hardware to avail the threads. This is not economical. This is where an asynchronous architecture like Node.js along with MongoDB becomes a big advantage. In [24] a performance analysis is made on Node and compared with an internet information services (IIS) server. According to the results they achieved, Node is most preferable for an I/O intensive application.

2.4. React.js

The two popular JavaScript frameworks used for front-end development at present are React [25], [26] and Angular [27]. Angular is maintained by the Google community and React is developed by Facebook. Both are open source frameworks and these help the developers in making single-page applications. YouTube, PayPal, Walmart, and G-mail use Angular. And React is used in Facebook, Instagram, and Whatsapp. In [28], [29], a comparison of React with Angular is made. Angular uses real document object model (DOM) whereas React uses virtual DOM. A virtual DOM is lightweight and faster than a real DOM since, in a real DOM, a small change in the status of the DOM would cause a re-rendering of the entire DOM in Angular, this is not the case in React. Think about the condition where the DOM has a large number of data to be rendered, in this case, a small change in DOM causing an entire re-rendering of the DOM is undesirable. A virtual DOM on the contrary only re-renders that certain block/component that has been updated, which makes React much faster than Angular in DOM rendering applications. One more advantage in React is that, unlike Angular which uses a Two-way data binding, React uses a one-way data-binding using the store where all the data to be rendered to the front-end is created and there is an action and event dispatcher, details of which would be explained in the next section. One way data flow/binding gives a better understanding and control over the data.

3. RESULTS AND DISCUSSION

In this section, the step-by-step approach used to build this project is explained. Throughout this work, visual studio code was used as the code editor. It's user-friendly, simple to use and it comes with many extensions like emmet, prettier, bracket pair colorizer, auto rename tag, live server, and many more extensions which help in detecting bracket pairs, detecting an error, and also to deploy a local server.

3.1. Creating a Mongoose schema

The various mongoose schema used were user schema, profile schema and post schema. User schema had the name, email, password, and date of creation as attributes. Profile schema had attributes: company, website, location, skills, experience and education. So that each user could document their job status, which company they're working in, their skills, their experience, and education. Post schema had attributes: text, name, likes, comments, and date. Likes and comments were also used to populate users to keep a record of all the users who liked or commented on a given post. Now in the MongoDB cloud database the three folders, one of the user, second of the profile, and the third of posts would be made. Figure 3 shows the three collection folders been made in the MongoDB cloud, one for users, the second for profiles, and the third for posts.

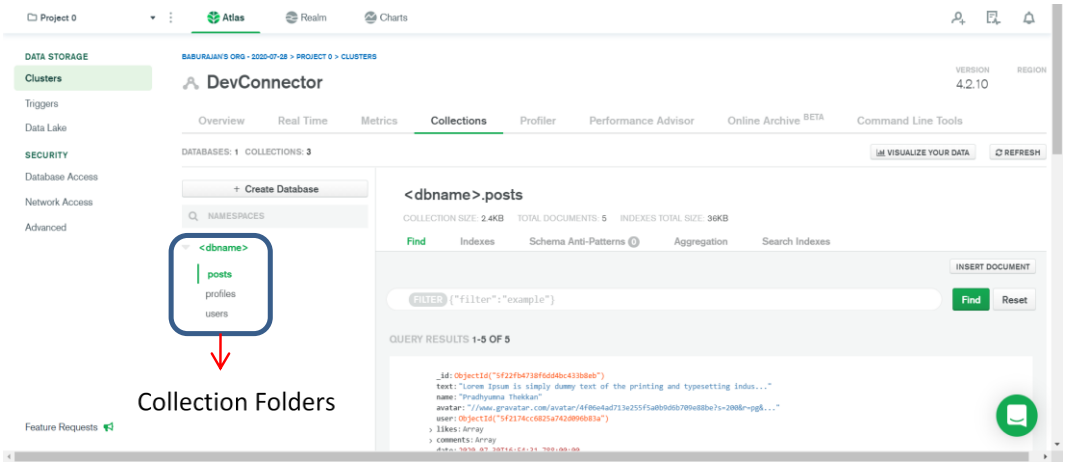

Figure 3. MongoDB collection

3.2. Rest API

Three routes for creation (post request), one for user, second for profiles, and the third for posts, were created. One route is created to authenticate the user so that a user could enter their email and password and then they could give a post request to create and delete a post or profile. The profile and post routes also had a get request to read the data of other users posts and profiles. And JSON web tokens (JWT) were used to authenticate the users while logging in. Only authenticated users could access the routes. Postman API was used to check the working of REST API. When a user is logged in, a JWT is created for that user.

Now using this JWT they could access all the protected routes. But the JWT has an expiring time which is set to 3600 seconds or 1 hour, after which the current generated JWT becomes invalid. After the expiring time of 3600 seconds, the user needs to login once again in order to create a new valid JWT to access the routes. Figure 4 depicts the user is logged in, a new JWT is created and the routes are accessed successfully with this new JWT.

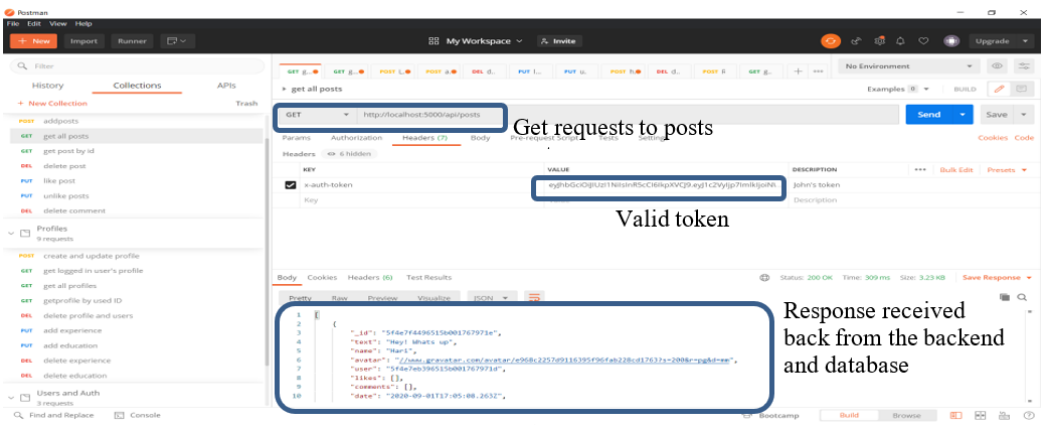

Figure 4. Valid JWT and response from the backend and database

3.3. Front-end

A front-end of a Web-app is the web design and the client-side rendering of web content. This includes the CSS which is the UI part and reactjs which involves rendering the content. So, the goal here is to render the data at nodemon server (backend-server) to the node server or client-server so that the users could view posts, comments and profiles in real-time. To achieve that both the servers are to be activated simultaneously. A NPM package concurrently was deployed, so that we don't have to run nodemon and client-server separately instead with a single command both of them would be activated. This section will be dealing with tools that are required for front-end development. Now to the backend while requesting axios, we make a proxy: "proxy": http://localhost:PORT. So that we could make an axios.get/post/delete method directly to our back-end routes. Axios is a client-side HTTP service within React and Angular, it is used to connect the front-end requests to the back-end.

3.3.1. UI/UX design

For any Web application, its interface and design play an important role in business, because it directly reflects a customer's usage. More the customers are satisfied, it helps in growing your business, you get much more customers. Keeping this in mind, it's hard to straight away start designing the UI without having a feel of how our application would look like to our customers. This is where a need for UX arises. The most popular tool for UX/UI designing in Adobe XD. It's user-friendly with many features and is available for both macOS and Windows. The part that any web designer would like about this software is that it gives a feel of the application we intend to make, a functional application, and design. So a developer could make a rough sketch of their entire application along with the design and its functionaries like a pop-up window (a modal in CSS), the link between pages in the web application, animation effects, fill color property, and many more. One more major advantage to a UI designer is that the names of the features used in Adobe Xd are the same in coding CSS properties.

A design section is where we could make the web designs. In this section, we could use the repeat grid feature which can be used to repeat a grid cell throughout the web page. There is also a feature called components in the design section which is used to symbolize. A component is a feature that can be used in multiple pages, like a sidebar and a navbar, which are the repeating elements throughout pages. What the component feature does is that, once we select a particular section like the sidebar or navbar of a page and declare it as a component, then we could use that over many pages. So, we need to only design once and declare it as a component, after that we could use this design through multiple pages. Other than the design section, Adobe Xd provides a prototype section. Here we could link between pages and provide the functionality to the modals (pop-window).

Also in syntactically awesome style sheets (SASS), we could use the block element and the identifier notations easily, which is the trend at present to organize elements of a block item. Figure 5 shows the block element notation. Where gallery is a block and 'item' is a block element. The figure also shows the SCSS file being compiled to CSS. SCSS uses the SASS features but the curly braces, unlike SASS where we require indentation and the block element notation becomes much easier to use.

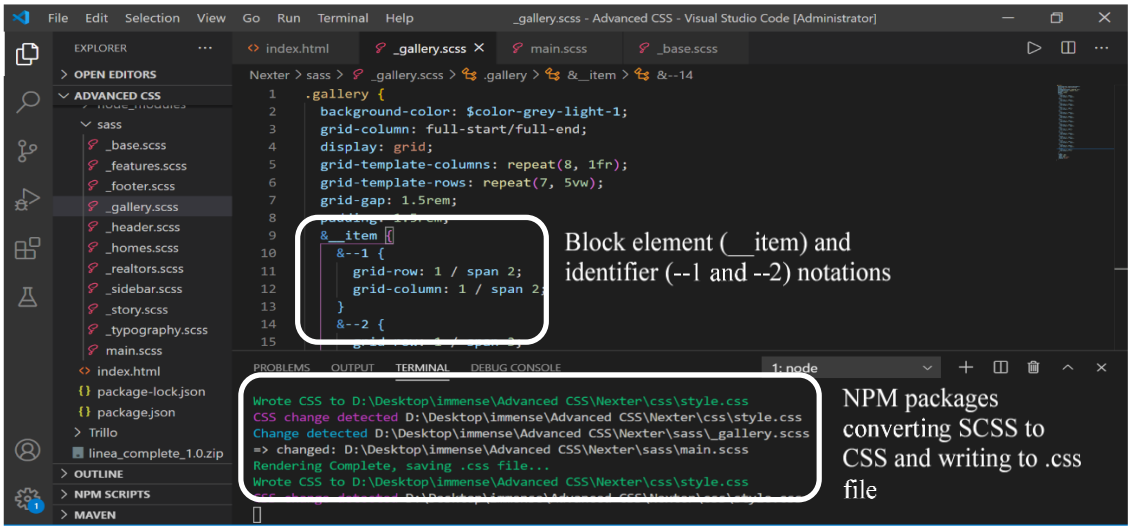

Figure 5. SCSS Code and NPM package

After the complete plan out of a UX design, and the prototype of how the UI of our application needs to look is made, the next step is to code the CSS. Traditional CSS coding poses many difficulties to designers, due to its inflexibility. Instead of using CSS the trend now is SASS. SASS is a preprocessor scripting language that is compiled into CSS. So, ultimately the code written in SASS is converted to a CSS file. SASS provides flexibility to CSS, we could declare multiple variables for colors once and use them throughout the design. And if we later wanted to change the color we could re-assign that variable with a new color. And one more major advantage of SASS is that we can write the code in different files and import all the files using the @import key, so we could maintain different components like the sidebar component in a different file and navbar in a different file and then also reuse these components for a different project too.

3.3.2. Redux

Redux is an open-source JavaScript library that handles the status (or commonly called a state) of an application. It is widely used to create the user interface with libraries like React or Angular. With Redux one could connect all their front-end components. A state in Redux is a large term (also known as the state tree), but it generally refers to the single state value in the Redux API that is controlled by the store and returned by the getState() function. It reflects the entire state of a Redux programme, what this means is that any change in any of the components of our application would result from a change in the entire application's rendering. Middleware in Redux offers a medium to dispatched actions before it reaches the reducer. An action is a payload that carries information from the back-end of our application and is send to the store so that the state could access the particular action. In the example above where the user gets authenticated, it is the passing of information from the back-end to the front-end, because our application's back-end is connected to our MongoDB database and in turn, the front-end renders the information at the back-end. So, this part of carrying the information from the back-end to the store is carried out by the actions. And the store is where the root reducers, initial state, and middleware are created. Chrome has an extension for Redux through which we could see and analyze our states.

Now an action contains a type and payload. The type specifies the action to be performed (it's a label to the action) like for example to get the profile of a user, the type specified to that action is "GET_PROFILE". The payload contains the information from the back-end of a particular action queried. In this example of the "GET_PROFILE" type action, a get request through axios is made to the back-end and the response data from the back-end is send as the action's payload. Now the action's payload is passed to the Redux's store, from here the payload is then passed to the state and then the entire application's front-end receives the changes applied. Now if the login is failed, the action gets updated to type "LOGIN_FAIL" and is authenticated attribute is set to false, indicating the user entered invalid credentials. A notification of "Invalid credential" is sent from the back-end to the front-end through the action type of "LOGIN_FAIL". Now this notification is a division ( ) which would remain in the login page. To remove this notification action of type "REMOVE_ALERT" is set after 5 seconds using the setTimeout method in JavaScript. What this action does is that it would set the display block to none (display: none) after 5 seconds.

4. CONCLUSION

Many Web technologies are being made to ease the connection between a retailer and customers across the globe. The most booming tech at present in Web tech is the MonogoDB database, Nodejs, and Reactjs. Monogodb being a non-relational database management system (non-RDBS), is the preferred database at present due to its capability to handle huge volumes of data which a traditional RDBMS system like MySQL fails. Monogodb uses JavaScript object notation (JSON) data format to store data into its database, in JSON format data is stored in a key-value pair. So, unlike MySQL where data is to be stored in a table, JSON data is much easier to handle. In case of making a schema to reference between data in MongoDB database like that of MySQL (relating between tables data), Mongoose is used to make a schema and provides the connection between the back-end of the application with the MongoDB database. And the back-end of an application is build using Nodejs. Reactjs is used for the client-side rendering of the website (as a front-end). In the case of React, it uses a virtual DOM which only re-renders the particular component that is been updated, rather than re-rendering the entire content of the DOM as in the case of Angular.

ACKNOWLEDGEMENTS

This research work was funded by "Woosong University's Academic Research Funding-2021".

REFERENCES

[1] K. S. Alaoui, J. Foshi, and Y. Zouine, "Radio over fiber system based on a hybrid link for next generation of optical fiber communication," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 9, no. 4, pp. 2571-2577, Aug. 2019, doi: 10.11591/ijece.v9i4.pp2571-2577.

[2] G. Stuart, "HTML5 and the Canvas Element," *Introducing JavaScript Game Development*, pp. 3-16, 2017, doi: 10.1007/978-1-4842-3252-1.

[3] J. Attardi, "Introduction to CSS," *Modern CSS*, 2020, pp. 1-15, doi: 10.1007/978-1-4842-6294-8.

[4] A. Guha, C. Saftoiu, and S. Krishnamurthi, "The Essence of JavaScript," *European Conference on Object-Oriented Programming–Object-Oriented Programming*, 2010, pp. 126-150, doi: 10.1007/978-3-642-14107-2_7.

[5] K. T. Song and S. H. Park, "Design of Master-Slave-Slave Replication Model to Balance Master Overhead for Key-value Database," *The Journal of Korean Institute of Information Technology*, vol. 15, no. 2, pp. 7-14, 2017, doi: 10.14801/JKIIT.2017.15.2.7.

[6] M. Waseem, P. Liang, M. Shahin, A. D. Salle, and G. Márquez, "Design, monitoring, and testing of microservices systems: The practitioners' perspective," *Journal of Systems and Software*, vol. 182, 2021, doi: 10.1016/j.jss.2021.111061.

[7] M. Thakkar, Building React Apps with Server-Side Rendering, New York, USA: Apress, 2020, doi: 10.1007/978-1-4842-5869-9.

[8] G. Sayfan, "Testing React.js Applications with Jest-A Complete Introduction to Fast, Easy Testing," *SpringerLink*, 2019. Available. [Online]. https://link.springer.com/video/10.1007%2F978-1-4842-3980-3

[9] Available. [Online]. https://www.coursehero.com/file/69776207/BTM495-AA-Outline-Fall-2020-V032-RA-with-Linkspdf/

[10] Y. Qi, "The role of mobile web platforms in the development of critical, strategic and lateral thinking skills of students in distance physical education courses," *Thinking Skills and Creativity*, vol. 42, 2021, doi: 10.1016/j.tsc.2021.100935.

[11] A. Stadnicki, F. F. Pietroń, and P. Burek, "Towards a Modern Ontology Development Environment," *Procedia Computer Science*, vol. 176, pp. 753-762, 2020, doi: 10.1016/j.procs.2020.09.070.

[12] D. Schwarz, "*Jump start Adobe XD. SitePoint*," 2017. Available. [Online]. https://www.sitepoint.com/premium/books/jump-start-adobe-xd

[13] D. Mazinanian and N. Tsantalis, "An Empirical Study on the Use of CSS Preprocessors," *IEEE 23rd International Conference on Software Analysis, Evolution, and Reengineering*, 2016, doi: 10.1109/SANER.2016.18.

[14] A. Libby, "Introducing Dart Sass", 2019, pp. 1-33. Available. [Online]. https://dokumen.pub/introducing-dart-sass-a-practical-introduction-to-the-replacement-for-sass-built-on-dart-1st-ed-978-1-4842-4371-8-978-1-4842-4372-5.html

[15] D. Cederholm, "*SASS pour Les web designers*," 2015. Available. [Online]. https://www.eyrolles.com/Informatique/Livre/sass-pour-les-web-designers-9782212141474/

[16] C. Gyorödi and R. Gyorödi, "A Comparative Study between the Capabilities of MySQl Vs. MongoDB as a Back-End for an Online Platform," *International Journal of Advanced Computer Science and Applications*, vol. 7, no. 11, pp. 73-78, 2016, doi: 10.14569/IJACSA.2016.071111.

[17] T. E. Somefun, C. O. A. Awosope, and C. Sika, "Development of a research project repository," *TELKOMNIKA (Telecommunication, Computing, Electronics and Control)*, vol. 18, no. 1, pp. 15-165, Feb. 2020, doi: 10.12928/telkomnika.v18i1.10452.

[18] Available. [Online]. http://img105.job1001.com/upload/adminnew/2015-04-07/1428393304-PTFEKAZ.pdf

[19] S. B. Uzayr, N. Cloud, and T. Ambler, "Mongoose. In JavaScript frameworks for modern web development: The essential frameworks, libraries, and tools to learn right now," *Apress*, 2019, doi: 10.1007/978-1-4842-4995-6.

[20] A. H. Chillón, D. S. Ruiz, J. G. Molina, and S. F. Morales, "A Model-Driven Approach to Generate Schemas for Object-Document Mappers," *IEEE Access*, vol. 7, pp. 59126-59142, 2019, doi: 10.1109/ACCESS.2019.2915201.

[21] S. Holmes, "Mongoose for application development," *Packt Publishing*, 2013. ISBN-13: 978-1782168195.

[22] S. Tilkov and S. Vinoski, "Node.js: Using JavaScript to Build High-Performance Network Programs," *IEEE Internet Computing*, vol. 14, no. 6, pp. 80-83, 2010, doi: 10.1109/MIC.2010.145.

[23] A. Mardan, "Practical Node.js: Building real-world scalable web apps," *Apress*, 2018, doi: 10.1007/978-1-4302-6596-2.

[24] O. H. Jader, S. R. M. Zeebaree, and R. R. Zebari, "A State of art survey for web server performance measurement and load balancing mechanisms," *International Journal of Scientific and Technology Research*, vol. 8, no. 12, pp. 535-543, Dec. 2019.

[25] L. P. Chitra and R. Satapathy, "Performance comparison and evaluation of Node.js and traditional web server (IIS)," *International Conference on Algorithms, Methodology, Models and Applications in Emerging Technologies (ICAMMAET)*, 2017, pp. 1-4, doi: 10.1109/ICAMMAET.2017.8186633.

[26] N. McClay, "*MEAN cookbook: The meanest set of MEAN stack solutions around*," Packt Publishing, 2017.

[27] A. Bhawiyuga, S. A. Kharisma, and B. J. Santoso, "Cloud-based middleware for supporting batch and stream access over smart healthcare wearable device," *Bulletin of Electrical Engineering and Informatics (BEEI)*, vol. 9, no. 5, pp. 1990-1997, 2020, doi: 10.11591/eei.v9i5.1978.

[28] Y. Chika and O. K. Esther, "Financial stock application using websocket in real time application," *International Journal of Informatics and Communication Technology (IJICT)*, vol. 8, no. 3, pp. 139-151, 2019, doi: 10.11591/ijict.v8i3.pp139-151.

[29] P. Gupta, S. Banerjee, D. P. Mishra and S. R. Salkuti, "Connection status report generator," *Indonesian Journal of Electrical Engineering and Computer Science (IJEECS)*, vol. 22, no. 2, pp. 1069-1077, Sept. 2021, doi: 10.11591/ijeecs.v22.i2.pp1069-1077.