International Journal of Scientific Research in Computer Science, Engineering and Information Technology
ISSN : 2456-3307 (www.ijsrcseit.com)
doi : https://doi.org/10.32628/CSEIT23902104

Implementation of MERN : A Stack of Technologies to Design Effective Web Based Freelancing Applications

Karishma Arora, Vaishnavi, Jai Nagpal
Department of Computer Science and Engineering, Delhi Technical Campus, GGSIPU, India

ARTICLEINFO

Article History:

Accepted: 15 April 2023
Published: 06 May 2023

Publication Issue
Volume 9, Issue 3
May-June-2023

Page Number
23-32

ABSTRACT

In today's world with the growth of internet, there is a hike in demand for electronically deliverable services. These services can be provided by freelancers. Freelancers are the personals who work individually and provide services as per the need of client according to their contract for a particular job. Every now and then freelancers need a platform where different businesses can be found and contacted. The idea is to have a software that provides the functionality of a hub where both freelancers and business providers can interact with each other. Our work is known as 'Developer's Hub', which encapsulates the idea of freelancing with MERN stack in an optimized way over the web. MERN stack, as the name suggests is stack of four different technologies which are MongoDB, Express.js, React.js, Node.js. This paper reflects significance of freelancing and the technologies concerned in MERN stack. It includes the implementation of MERN for optimizing freelancing over web. The paper also includes challenges faced by freshers or experienced people in freelancing market with the growth in demand of freelancing world.

Keywords : MongoDB, Express.js, React.js, Node.js

I. INTRODUCTION

As always, every business tries to decrease their overhead and supplement their existing workforce, here is where electronically deliverable services comes into play. Freelancing involves an individual who works with commitment only to particular project and no further involvements. The new generation is more involved into the freelancing as it offers work to freelancers with their own conditions and flexibility of choosing people to work with and along with a choice of projects out of thousands available on different freelancing websites. It allows a vast opportunity of growth and development with strong earning potential for every project chosen. But freelancers face some challenges like they need to find new projects to work on, on frequent intervals. To overcome this, there are multiple websites like Upwork, truelancer.com, guru, freelancing.com which provides both businesses and freelancers multiple options where they can fulfil their requirements. One such website is **Developer's Hub**, where both freelancers and business can find

freelancers and project opportunities respectively. A well established and optimized recommendation system is embedded in the project for business category of users by which they get recommendations of freelancers as per their needs on basis of skills. The main purpose of developer's hub is to help students as current issues faced by students includes difficult to find projects and grow their network. Developer's hub provides them all these opportunities on a single platform, which is free of cost. It permits them to practice from scratch to expertise level. MERN stack is a promising platform that works on back-end as well as front-end. This stack is composed of four marvelous technologies like MongoDB, React.js, Express.js, Node.js. This is proved to be the best suite to the system as MERN is an open-source JavaScript code which can be used to implement complex system in simplest way and gets improvised with time. Front end part of the system is developed using React.js, which is open-source front-end JavaScript framework used to create UI because of it "less code more function" abilities. MongoDB is handling the database of the system as MongoDB is NoSQL database program which are quite useful for working with large sets of distributed data. While Express.js and Node.js is handling the back-end of the website as helps to manage servers and routes.

II. THE ARCHITECTURE of MERN STACK

In MERN Stack the four famous technologies MongoDB, Expess.js, Node.js, and React.js merged are designed to build a robust framework for helping developers to practice with JavaScript components for solving real-life problems and daily development needs. MERN stack is an abbreviation of four different technologies used in the development of dynamic mobile and web applications, where **M** is abbreviated for **MongoDB**, **E** is abbreviated for **Express**, **R** is abbreviated for **React,** and **N** is abbreviated for **Node**. It's one of the most popular tech stacks used by developers nowadays because it makes the development process easier quicker and smoother. MongoDB is referred to as a document database where data is stored in groups of documents known as collection. Express.js is solely used to create a web server and it is inherited from Node itself therefore it contains all the features of the node with some advancement which makes it more flexible and scalable. React is used to create a modern client-side application. Each technology has become the core of web applications in this modern era.

A. MongoDB

MongoDB is the most prominent NoSQL database which provides an alternative to a relational database system where all the data is stored in tables in form of rows and columns. In this data is stored in documents and a group of documents is known as collections. In MongoDB, data is stored in form of BSON format which is the extension of JSON format due to this feature its performance level increases making it more scalable and reliable. It is more widely used than SQL databases because of its strong query processes, easy accessibility, customizable schema, and security.

B. Express

Express.js is one of the most valuable NodeJS packages that can help us in creating server-side web applications. It helps developers to write lesser lines of code thus saving time and reducing the complexity of code. It allows developers to execute customized, reliable, and more scalable servers to connect with the front end. It also helps to set up middleware and develop API routes for the transmission of data from the front end to the back end.

C. React

React.js is a popular and openly available front-end JavaScript library for creating Web applications. The fact that React is component-based is a major advantage. Since they are independent of each other, we can easily reuse them on different pages or projects. This saves us both time and money. It allows the creation of creative interfaces using its built-in components. It uses JSX to create ad use components

with HTML and thus allows us to directly write HTML react applications. With JSX we can write expressions inside curly braces and use its synchronous behaviour to call functions in a sequential pattern.

D. Node

Node.js is an openly available server environment that can run on any platform such as Linux, Windows, and Mac. It provides a run time environment to run JavaScript outside the browser. It is used to create server-side applications because of its asynchronous nature and event-driven process. It also has its package manager popularly known as npm. It can be installed by using the simple command "npm install". It helps users to directly add useful packages to their applications by writing a single line of code. It is free and mostly used by large organizations to run real-time server applications.

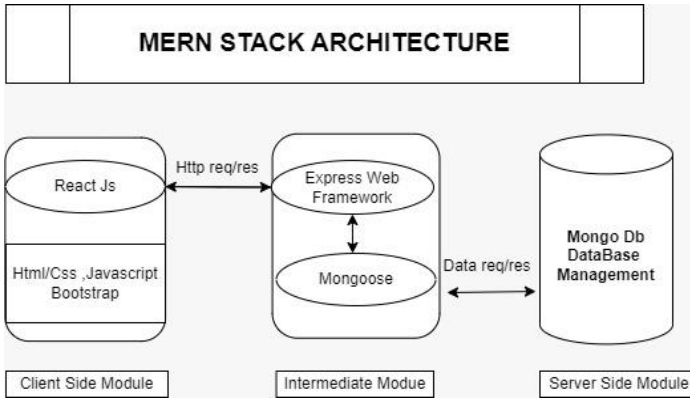

Figure 1. Architecture

The **Model-View-Controller (MVC)** is coming into existence in 1979 by computer scientist Trygve Mikkjel Heyerdahl Reenskaug. He defined an architectural pattern that divides the complexity of web applications into three logical components: the **model**, the **view**, and the **controller.** Laravel, Ruby on Rails, and Angular are one of the few popular web frameworks which follows MVC architectural pattern.

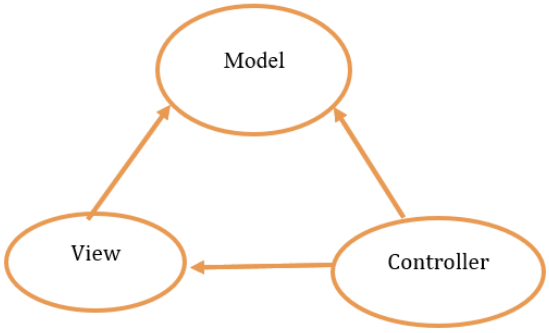

Figure 2. MVC Model

MERN Stack breakdown of MVC pattern:

1) **Model**: Mongoose serves as a 'Model' of the MVC pattern at the server-side module. It handles all the data-related logic that the user works with. Mongoose stores all data components of the application that needs to have a function.

2) **View**: React serves as a 'View' of the MVC pattern at the client-side module. The module will be written in JavaScript, HTML, and CSS, using ReactJS as the framework. This works as an interface between the user and the features of the application.

3) **Controller**: Express and Node serves as the 'Controller' of the MVC pattern. Express and node handle all the functional programming aspects and acts as an interface between Models and Views. It sends the React module to the client's device and receives HTTP requests from the client and responds according to them.

III. CHALLENGES for MERN STACK

Web development technologies go through exemplary changes with time. HTML, CSS, and JavaScript have become the cornerstone of web development but no longer fulfil the demands of modern world websites like having better user experience, lesser loading time, and responsiveness to different size devices. Even the MERN stack with the above features is not enough to meet the inevitable expectations of users and developers. Challenges faced by developers while using MERN Stack are:

A. MongoDB

1) Query: MongoDB is a non-relational database system, it doesn't support join functionality. Join functionality is the ability to query between different collections. MongoDB must do separate queries when there are conditions that can be solved by multiple collections. The developers are still working to develop a function that allows them to do so.

2) Mobile Problems: MongoDB doesn't provide complete support when it comes to mobile. As a result, developers have to write code separately to synchronize data on mobile devices with data on servers. This problem leads to the requirement of constant good connectivity in order to meet application needs.

3) Scalability: MongoDB limits the ability to use the application by many concurrent users on a single node. Since in MongoDB data is stored at only in memory, the loading time is more as compared to relational databases.

4) Locking Mechanism: The transaction that is being modified gets locked and other sessions which are modifying the transactions are aborted and they are required to retry the transaction. This results in the requests in non-deterministic orders.

B. React

- React.js components are difficult to understand and implement in complex web applications.
- React virtual DOM is not much precise.
- Complicated asynchronous programming is required to interact with the server.
- HTML templates are neither complete nor powerful.
- Long list of react.js can lead to poor performance on low-specs laptop and mobile devices.

C. Node

1) Rules: Node.js doesn't provide any standard rules or guidelines describing how the code should be written. That's why beginners find it difficult to understand and learn since each application requires its own unique approach to writing the code.

2) Poor documentation: Documentation of any software are roots for a developer to learn a new skill. Poor documentation leads to more development time and difficulty in relating the components of the software. Since the open-source community offers a different method for common practices, it is difficult for users to implement them as per their needs.

3) Scalability: As Node.js uses a single-threaded process, scaling is more complicated. Applications that require more CPU consumption would require the code to be more modular.

IV.DEVELOPER'S HUB: A MERN STACK WEB APPLICATION

Developer's Hub is a full-stack web application developed using MongoDB, Express.js, React.js, and Node.js. The objective of the application is to develop an environment that can help freelancers whether he or she is an experienced individuals or a student by providing them with hundreds of projects on a single platform with just a few tap aways. The main aim of the project was to provide students an opportunity to start their careers in the freelancing world without any mandatory need for a degree or any other certificates. By evaluating similar applications and articles it can be easily found that when a freelancer struggles to find a job, they must conquer multiple turbulence to reach out to a single project owner. Such turbulence is bidding processes, with irrelevant tests and costly courses to be completed before entering some communities. This application will showcase suitable freelancers for every job posted based on the skill requirements of business owners. To develop this recommendation system, the application consists of a content-based recommendation system that works on a real-time basis. The intended user of the applications is freelancers, Businesses, or any recruiters and an admin who can manage data at MongoDB databases. Following are the user stories explaining how the application works for Freelancers and Businesses.

To have all the privileges of the application any freelancer who has skills can register themselves on the web app and create their profile. The application allows users to view or save necessary details like contact numbers, email ids, social links, etc. It also allows freelancers to mark their interests in any project posted by businesses. Once a user creates their profile, it can be seen in the freelancer's list by other freelancers and businesses. Businesses can also see the freelancer's profile when they mark interest in the jobs posted by them. An updating feature is also implemented in the app to add new experiences and education details. To communicate with the job posters their email has been provided on their profile.

Businesses can also register themselves and create their profile. For businesses, the application allows entering details like contact numbers, email ids, company links, etc. They also have access to create a job posting and mention the required skills and salary range they wanted it to be. After creating these job posts they can see a list of interested freelancers, recommended freelancers, and a list of all freelancers at any time. If required, they can contact to the freelancer with the email id provided for them.

V. IMPLEMENTATION

As with every other MERN Stack application Developer's hub is also divided into sections i.e., client-side and server-side.

A. Server-side Implementation

The server side of the application is implemented by using built-in modules or customized modules of Node.js.Node.js has more than one million packages to install various frameworks or development tools used to develop the project. To install npm packages, execute "npm install" command in the command line interface. Such dependencies installed on the developer's hub application are jsonwebtokens, react-icons, mongoose etc. Another part of server-side implementation is database management which is executed with MongoDB. That's why server-side implementation is further divided into sections which are database management, Routes, middleware, Models, and server setup and configuration.

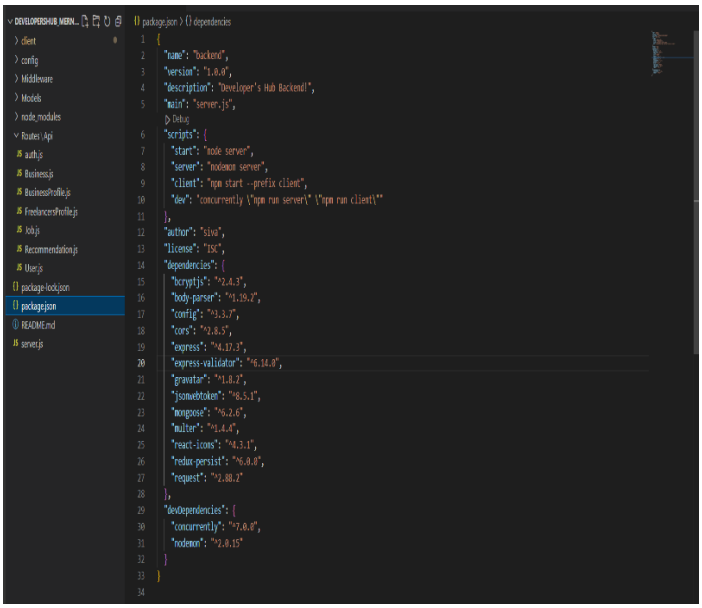

Figure 3. Package.json

1) Database Management: MongoDB utilizes data by storing them in form of clusters. Clusters handle large volumes of data and the number of requests. It makes monitoring and automation easy through data reduction and load balancing. To set up the cluster of this project create a cluster by setting up the cloud provider and region. MongoDB also provides backup and size options for data at very reasonable prices. After creating the cluster, connect the database and application by pasting a link into the config folder of the project. This connection allows live updates and modification of data done by the client directly to the database. Atlas admin can also monitor or set conditions if required to the database. In this application, default conditions are perfectly fine with the requirements.

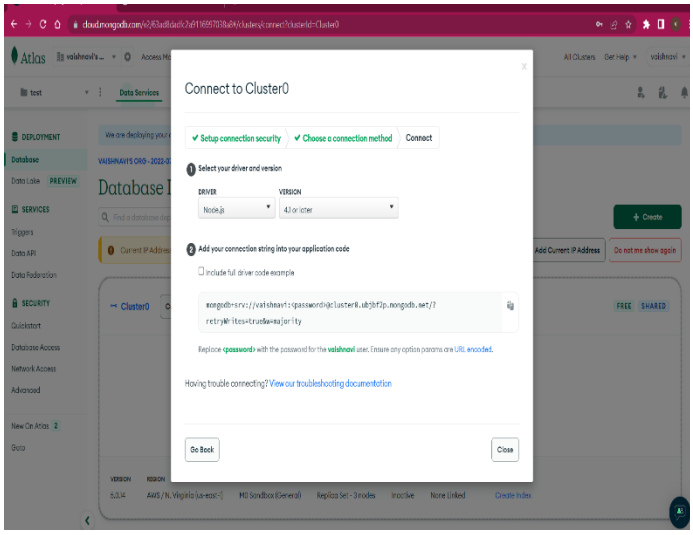

Figure 4. Connect cluster

2) Server Setup: Start connecting the server with express.js, cors, multer, and body-parser with require() in the server.js file to include its external node modules. To connect the server on visiting to a particular route on any browser app.use() is implemented with two parameters that will download the customized react components on accessing them.

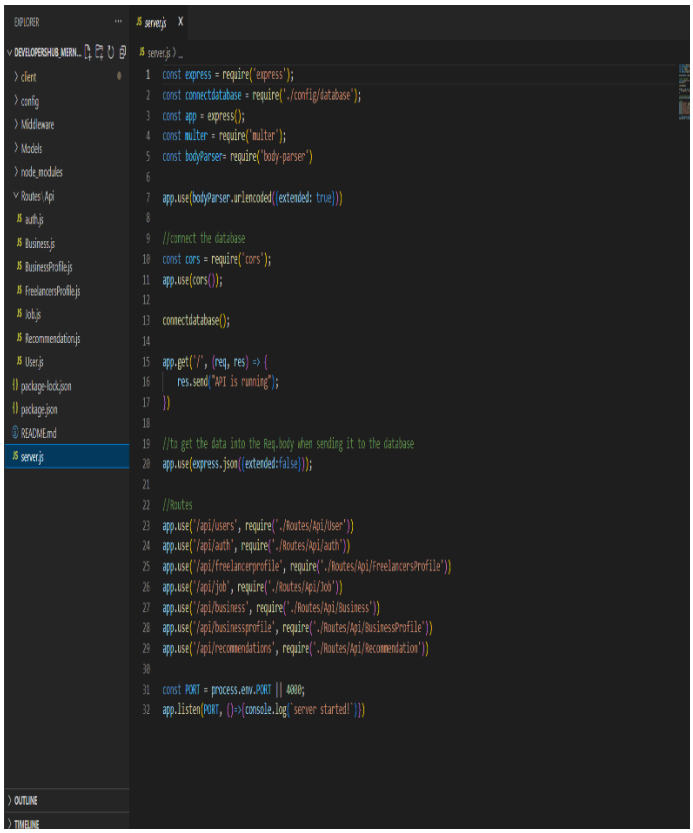

Figure 5. Server.js

3) Models: Once the database and server are set, to define the structure of the application start creating schemas. The mongoose models which define the application's instances, and their properties are Business.js,BusinessProfile.js, FreelancerProfile.js, Jobs.js, User.js. Every property which has the required method as "true" is mandatory to carry out operations while if it is "false" it is optional.

Business.js and user.js models are customized to register in the application by businessman and freelancer respectively while BusinessProfile.js and FreelancerProfile.js is customized to create profiles. In job.js, "mongoose.Schema.Types.ObjectID" is implementing a very fine feature of the application. It is connecting the jobs with the interested freelancer defined under the user.js model.

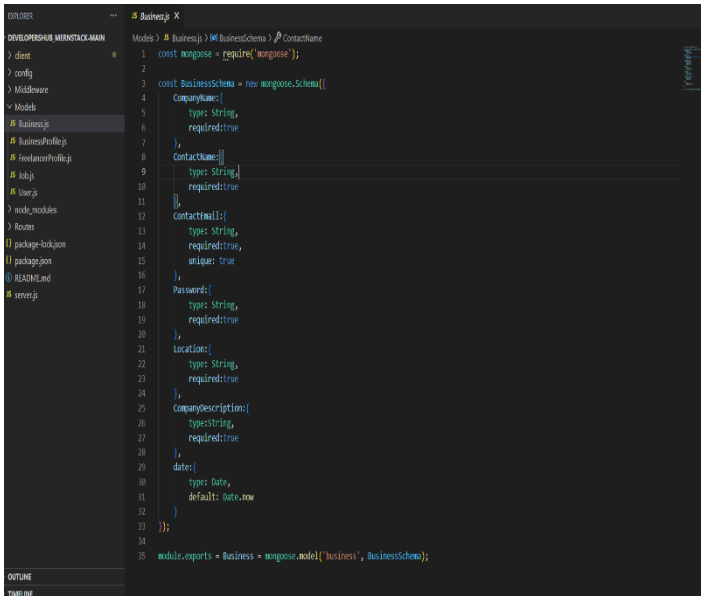

Figure 6. Business.js

The password property defined in these models is not stored in form of plain text in the MongoDB database. It is first hashed using bcrypt which is a very popular npm module. gensalt() function in bcrypt is used to implement the hashing of the password. This hashed password is further encrypted by the hash() function and send to the mongodb.

If the password is needed in any sign in or sign-up activity of the application, an authentication process is carried out in the web application. For security purposes Json web token authentication is the standard

security level of the application. Whenever a user register on the website a token is generated using jwt.sign() and verified each time anyone log in to the website using jwt.verify().

4) Middleware: Middleware in this application is the authentication files that contain objects and functions which is executed during the request and response cycle of the application. Req is known as the request object, Res is known as the response object and next is used for the next middleware function. Middleware works with the JWT authentication system to secure the web application. JWT can be installed using the command "npm install jsonwebtoken" on the command line interface. Whenever a user enters login details, this dependency creates a web token and returns it to the browser. 'AuthenticationToken' is created as middleware to verify the token. It is sent to the authentication header and jet.verify() matches the data entered by user to verify and allow user to get the response.

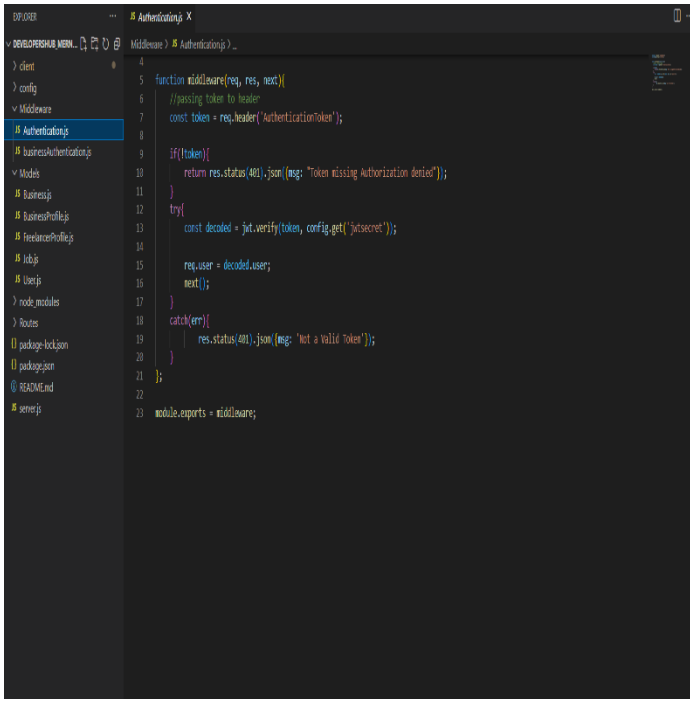

Figure 7. Authentication.js

5) Routes: The interface of the web application is defined using the routes. It connects the request made on the client side to the correct backend resources. Four different types of HTTP methods are used in the routes are

post(): to send data to the server,
get(): to request data from various resources,
delete(): to remove the data from the database,
put(): to update the data.

The routes customized while implementing the applications are:

- User.js: Implements post method while freelancers register themselves on the application. In the registration process, a web token is generated, and the entered details are posted on the server.
- Business.js: Implements post method while businesses register themselves on the application. In the registration process, a web token is generated, and the entered details are posted on the server.
- Auth.js: Implements post method when a user enters username and password for signing into the server which then generates jwt token. It also implements a get method that authenticates the user by jwt token and sends the user details to the client side.
- FreelancerProfile.js: It is implemented to get or delete the details of all freelancers or any specific freelancer details by using their id from the server. It is also used as a method of updating experience and education details.
- BusinessProfile.js: It is implemented to get the details of all businesses or any specific business details by using their id from the server. It also implements a post method which is used to create and update the profile details of a registered business.
- Job.js: It is implemented to post or delete jobs by businesses and freelancers can get all of them or by any specific id. Freelance can put or delete their interests in various jobs.
- Recommendation.js: Implements get method by showing freelancers which could be suitable for jobs on basis of the skills a freelancer has.

B. Client-side Implementation

The project's client side is implemented using only React.js using third-party libraries from npm. React.js optimized the output by allowing live editing in CSS and JavaScript in the early phases of the project. The "npx create-react-app app-name" command is used to set up the development environment. Here, npx is a react tool that is used to run packages. The packages installed throughout the projects are "moment", "React redux ", and "React bootstrap". The moment is used to format the dates like when a freelancer applied for a particular job.

1) React Components: React components are the building blocks of this application. React has built-in components of independent codes which could be reused in order to make the whole code data reductant and easy to update whenever needed. Every page in the application whether it is the landing page, create a profile page, or the page to find jobs is built on the react components. Every page has its own DOM structure. For instance, the Landing page uses <carousel> component to create the banner. Under this <carousel> tag it uses <carousel.caption>, <carousel.item> to add heading and images on the banner.

2) React-router-dom: In order to gain security only authenticated users are allowed to visit any page of the application. If any person whether a freelancer or businessman, the app forces that person follows a particular route to have access to any feature. For this purpose, **react-router-dom** is installed next in the process. To have access to the routes, all components of a particular react file needs to be rendered using the <Route> tag between <Routes> which is updated with version 6 of react-router-dom.

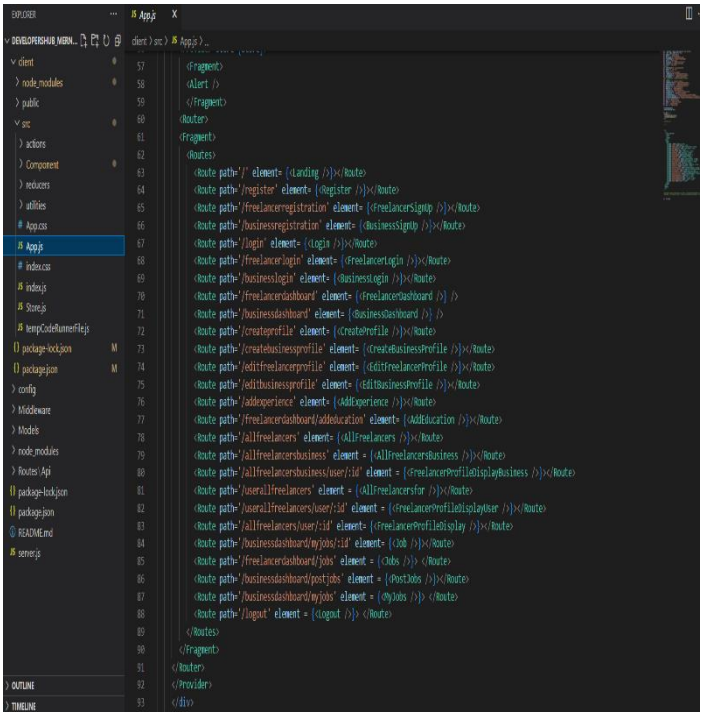

Figure 8. Routes

3) React redux: React redux is an openly sourced JavaScript library that is used in this application for its capability of managing states with a unidirectional flow of data models of the MERN Stack applications. Redux works as a virtual cloud on the client side of the application as it manages multiple states at a time. It reads and updates data for multiple components for making the UI interface easy to render. For instance, Jobs posted by some businesses got stored in the redux cloud which is also known as the redux store. This can be executed with help of Axios by writing a code of actions.

4) Actions: Actions are objects which contain a blank field for the user to fill the information with some event that happened in the application. Action implanted in this application are:

- Alert: It is implemented to create a dialogue box and alert the users whenever needed.
- Authentication: It is implemented for both freelancers and businesses to login into their profiles and allows them to access routes that are protected. Only a user which is registered before can access these routes. It is also used in process of

logging out and created a void for another user's data to log in.

- Profile: It is also implemented for both freelancers and businesses to create or update their profiles. In the case of freelancers, they can use defined functions to add experience and education while in the case of businesses they can add jobs and other details of their business.
- Job: It is implemented for business users to create them by adding different details of the job and an addintrested() method is used by them to view interested freelancers.

There are defined reducers for each action created. Reducers are methods that take the state of the components and actions taken as their parameter and return a new state.

VI. CHALLENGES FACED

- Learned React.js for creating the multipage dynamic front end.
- Learned React Redux for State Management throughout the app
- Managed large pieces of code together to connect frontend with the backend
- Learned Node.js and Express.js for creating and maintaining backend server and API endpoints
- Learned MongoDB database to use it as an alternative to traditional databases with more scalability, security, and accessibility
- Learned React-Bootstrap and Material UI for developing the application's eye-catching user interface.
- Followed modern security measures and middleware to make the application more secure and safe for the users

VII. CONCLUSION

Even after the availability of multiple popular freelancing websites, either the requirements of newbie freelancers are not accomplished, or it is very difficult to gain success on such websites. The developer's hub is opening an easy path for them apart from the basic features a freelancing website must have. The application was developed using MERN Stack. MERN stacks marvellous characteristics, and its highly adaptable capabilities make the project possible in such an easy way. The Stack is handling the client data with mongo DB, the client and application interface is handled by React.js, and a bridge to connect both is developed using express and node.js.

VIII. REFERENCES

[1]. Dr. Santosh Kumar Shukla, Shivam Dubey, Tarun Rastogi, Nikita Srivastava,"Application using MERN stack", International Journal for Modern Trends in Science and Technology,2022.

[2]. Yogesh Baiskar, Priyas Paulzagade, Krutik Koradia, Pramod Ingole, Dhiraj Shirbhate,"MERN: A full stack development", iJRASAT journal for research in applied science and engineering technology, 2022.

[3]. Sourabh Mahadev Malewade, Archana Ekbote, "Performance optimizing using MERN Stack on web application", International Journal of Engineering Research & Technology,2021.

[4]. Fathima Thabassum,"A study on freelancing remote websites", Research gate,2021.

[5]. Joeran Beel, Bela Gipp, Stefan Langer, Corinna Breitinger, "Recommender system, a literature survey", International journal on digital libraries.

[6]. Subramanian, Vasan, "Pro MERN Stack", Apress, 2017.

[7]. Mehra, Monika, Manish Kumar, Anjali Maurya, and Charu Sharma "MERN stack Web Development", Annals of the Romanian Society for Cell Biology.

[8]. P. Thung, C. Ng, S. Thung and S. Sulaiman, "Improving a Web Application Using Design Patterns, A Case Study", International Symposium in Information Technology, 2010.

[9]. Sanja Delcev, Drazen Draskovic, “Modern JavaScript frameworks, A Survey Study”, Zooming Innovation in Consumer Technologies Conference (ZINC), 2018.

Cite this article as :

Karishma Arora, Vaishnavi, Jai Nagpal, "Implementation of MERN : A Stack of Technologies to Design Effective Web Based Freelancing Applications", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 9, Issue 3, pp.23-32, May-June-2023. doi : doi : https://doi.org/10.32628/CSEIT23902104