Design and Development of Dashboard using React Redux Architecture for Data Management and Visualization

Kotha Sahaj
Department of Electronics and Communication
R.V. College of Engineering
Bangalore, India
kothasahaj.tc16@rvce.edu.in

Sudha R Karbari
Department of Electronics and Communication
R.V. College of Engineering
Bangalore, India
sudhark@rvce.edu.in

***Abstract*— Dashboards allow managers to keep track of various activities and developments with regard to various activities that function within the company. A dashboard is a data management tool that helps in visually tracking, analyzing and displaying real time data, metrics and various key data points which are key to monitor the health of a business or a specific process. They can be customized to meet the specific needs of a company. The proposed system is a statistical dashboard that not only keeps track of all the operations that are related to the company but also takes into consideration security of client data by enforcing various security mechanisms. The various operations that can be tracked and monitored include the real time information about the state and condition of over twenty thousand organization owned bikes, managing the working and movement of over a thousand agents, tracking the status of over fifteen thousand tasks every day that are created to ensure normal working of bikes. The Dashboard is designed by establishing a client server communication using various layers of TCP as in behind the scenes, the dashboard needs to connect to various files, attachments, services and API's, but on the surface displays all this data in varied forms. This is enabled with various security features like OTP based Authentication to ensure only valid people use it and designing a proxy server to protect the end points from being exposed. React is used along with the redux architecture as software tools that are used for development to ensure efficient state management. The various advantages and disadvantages that are involved in using react with and without Redux architecture is compared and discussed. This dashboard aims to bring all the data and information at one place that in turn helps in better coordination, management and enables better decision making process.**

Keywords— Dashboard, Client-Server Communication, Data Management, React Redux, Application program interface, Network security, Data visualization

I. INTRODUCTION

Businesses involve a wide variety of managing different types of operations and information that can potentially impact its overall performance. The key to enable higher performance is to ensure there is optimal management of operations at every level. Hence, a business organization irrespective of its size and structure requires a management tool to monitor, manage and take a decision at every level [1]. This management tool is facilitated by designing and developing a dashboard that can meet all the requirements and make the process of management easier and efficient [2]. The dashboard is often displayed on a web page which is linked to a database that allows the report to be constantly updated and modified with changing real time events [3].

Transmission Control Protocol/Internet Protocol (TCP/IP) is a communication protocol that is used for the communication between the client and the server over a network. Various TCP protocol layers are involved in managing the exchange of data between the client and the server. Development of a dashboard involves various layers of the TCP as the dashboard connects to various files, attachments, services and API's, but on the surface displays all this data in varied forms [4]. All protocols concerned with the communication between client and server operate in the application layer of the protocol. To formalize the data exchange, the server implements the application programming interface (API). The API is the abstraction layer that is used for accessing a service and it is the messenger that delivers the request to the client that it's requesting it from and then delivers the response back to the client [5]. Dashboard involves development of client side applications and the main goal of the client is to interpret the response based on the well-known application protocol.

A dashboard is considered to be one of the most efficient ways to track data sources. For any organization, with growing operations and entities, management and decision making becomes an absolute challenge [6]. The success of the organization lies in the best management of its entities and taking the correct decision at the right time to enable growth. The data around the organization keeps changing in real time and there is an absolute need for a visualization tool that presents the user with various types of data and information [7]. The organization needs to keep track of the status and condition of over twenty thousand bikes, managing the working and movement of over a thousand agents, tracking the status of over fifteen thousand plus tasks every day that are created to ensure normal working of bikes. This enormous amount of data needs to be collected and converted into valuable information that helps various departments within the organization to work collectively in the same direction without any constraints. The dashboards also help to gain insights into the most important aspects of real time data and use them to identify items that require urgent action, streamlining workflows and properly purposing resources.

II. DEVELOPMENT

A. Authentication

The dashboard is equipped with potential sensitive and confidential data and hence for this reason it is very important to maintain security of this data and ensure the data doesn't fall into the hands of the wrong people [8]. To prevent this, a login based authentication is designed for the dashboard so that only verified users whose data used for authentication is stored in the database with appropriate role. The dashboard uses mobile OTP based authentication. The user enters the mobile number and if the user's phone number is present in the database, they will receive an OTP to login. After successful authentication, the user is logged in and is now allowed to use the functionalities of the dashboard.

B. Design Flow

The dashboard is a data management tool that is designed to collect and display raw data in the most efficient ways. To visualize the working of the dashboard, primarily the user who is using the dashboard will have the power and authority to control various actions with just a click of a button or even as simple as scrolling [9]. Whenever, a user clicks on a button, an action dispatches and makes a request to the front end proxy server, this proxy server decodes the end points received in the request and request for the appropriate API to the backend server, the backend server after receiving the request modifies the database accordingly and sends a response back to the front end which again triggers an appropriate action call based on the response received. In the flow of events, a lot of elements are involved in ensuring data flow and also the client and server communicate appropriately and efficiently. Fig. 1 shows the various elements involved in the complete process flow of client server communication.

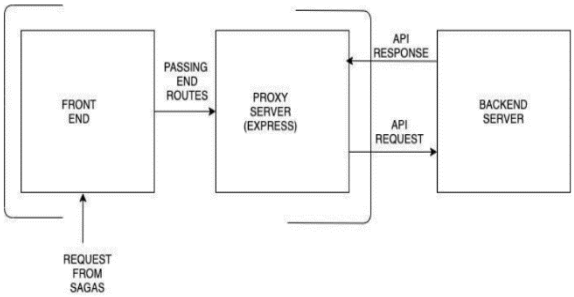

Fig. 1. Data flow representation

I. Client Side Application

The front end element mainly deals with the development of the client side applications and responsible for dispatching the API requests to the back-end. Client side application development is mainly concerned with the UI development. The UI is built to make the dashboard user friendly and an easy to use tool. Various function UI components are developed that provide the user with seamless functionality. UI components developed can range from as small as developing a Button and adding various functionalities attached to it after a click on it to the most complex of adding Maps to track specific items on the Dashboard. With the use of React software tools, we make use of sagas that help in dispatching an API call. This call is made to the proxy server along with some endpoints.

II. Proxy Server

This Proxy server is the server that is developed for the client side and its main purpose is to communicate with the backend server with API requests and receiving back the responses. The purpose of developing this proxy server lies in the fact to maintain Network Security. When a user is using the dashboard in the production, there is a risk of exposing the endpoints which can become a serious threat as any unauthorized person with the endpoint can misuse it and can gain access to the API response leading to breach of date security. Hence, the client sends the data and end points to the proxy server wherein the proxy server decodes the request body and end points requested and makes a suitable call to the backend server.

III. Application Program Interface

The dashboard needs to retrieve data from the database and display it on the website. This is possible with the co-working of the front end and the back end together. Whenever the front end requires some data to be displayed or processed to produce a result, it sends a body of data collected from the user as the request data to the API and this API delivers this request body as parameters to search in the database in order to retrieve back the required data and the same data retrieved from the database is sent as response data back to the front end application which is used as per the requirement. In simple terms, an API is like the messenger that carries the request and response data.

IV. Backend Server

The backend server listens to the request API and runs the appropriate database queries to fetch the required data. There is always a suitable format like the JSON that the backend server will expect the request body to be else it will consider it as a bad request. After retrieving the expected appropriate data from the database, the server sends back the response data in a suitable format that is received by the client side applications.

C. React Redux Architecture

Every dashboard needs to be designed in such a way that there is exceptional optimization in the memory and time for execution. React is a component based library and the components need to communicate with each other at regular intervals to perform a specific function. It is to be noted that

React is based on the principle of a single way of data flow i.e, data flows only from the parent component to the child component. In React, for accessing the state, the root component needs to send state objects as props to the child components and this again needs to send the props to its child components and goes on until it reaches the leaf component.

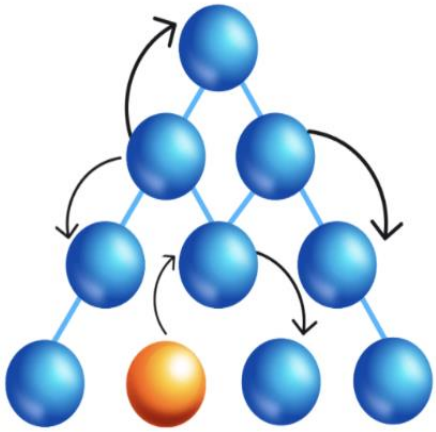

Fig. 2. Data flow representation without Redux

While this is easily managed in a small application, stage management becomes complicated to manage for numerous components of a large application. Hence, the project is designed in a way such that stage management becomes easy with no hindrance. This is achieved by using Redux which is mainly used for state management and it can also improve rendering performance.

The store used in Redux is similar to the JavaScript object and it also allows all components to be connected to the store, and extract all the data that is required. It is possible to put all the application's data into the redux store, and it is also possible to keep some data in the components local state if required, in the end it all depends on the use case and hence the most appropriate method is chosen to store the data. Using Redux, developers can easily manage these states of the application easily by the help of global accessing features. When any change is made in the state, it disrupts the child components, thereby affecting the performance. However, Redux library centralizes the state management of the application. This allows the developer to use significant features of development like undo/redo, state persistence, and much more. This is being illustrated in Fig. 2

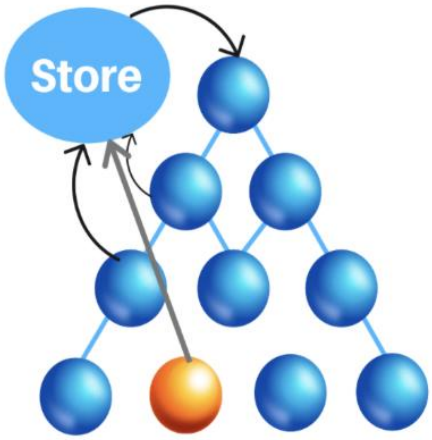

Fig. 3. Data flow representation with Redux having a central store

In React, it is very difficult to track the state of the application at the time of debugging process. But Redux provides a great developer experience that allows time travel debugging, and even sends complete error reports to a server. React has a complex UI, where the data flow would be difficult while we are using more components to share the same data. However, Redux is flexible with all the UI layers and has a large ecosystem of add-ons to fit the requirements. It is very difficult to reuse the components in React because it is tightly coupled with the root component. Redux reduces this complexity and provides global accessibility that helps to build applications that work frequently and are easy to test and run in different environments like the client, server, and native. Fig. 3 illustrates the Redux Architecture with Redux elements.

III. RESULTS AND DISCUSSION

The Client side applications have been developed to perform various functions required by the organization to manage the day to day operations. The enormous amount of data is now presented to the stakeholders in the best visualized way that makes the process of decision making a simple and effective process. The client side is designed with a proxy server that is used to communicate with the backend server to retrieve data as a Network Security feature. The dashboard is equipped with a login based OTP Authentication that allows only valid and registered users to log into the dashboard and work with the various functionalities the dashboard is equipped with. Data flow representation using redux architecture is compared with the architecture without Redux, the various advantages are illustrated and explains how redux architecture helps in better state management and ease of data flow. The Dashboard has been equipped with the best possible User Interface that is user friendly and an easy to use platform with the complexity reduced to as minimum as possible.

IV. CONCLUSION

A data dashboard to manage large amounts of data and information is developed to facilitate the process of data management and decision making, a simple process. Businesses involve a wide variety of flow of day to day operations and hence enormous amounts of data and information is generated and this data can potentially impact its overall performance. Hence, there was a need to have a visualization tool that can process and display the data in the best possible way by highlighting all the Key Performance Indicators and for this reason a dashboard is required. The development of a dashboard is the key to enable higher performance is to ensure there is optimal management of operations at every level.

The innumerous operations that are now being managed at one place include the state and condition of over twenty thousand bikes, the functioning and monitoring of over thousand agents who work time to time to ensure the bikes are in suitable and repair free condition to be used and also the fifteen thousand tasks a day that are created and assigned to the agents. The dashboard is developed with various security features that include OTP based Mobile number Login Authentication that ensures only people having prior roles can log into the dashboard. The dashboard is using the method in which the client communicates with the server where a proxy server is designed that receives a request from the client with suitable end points and this server converts these endpoints and sends an appropriate API request to the backend server. This backend server after receiving the API request body, uses this received request body and accordingly retrieves or modifies data in the database as per the request. After this, the backend server sends an appropriate response to the client side application and the client performs certain actions based on the response received from the backend thereby completing the process of communication between the client and server.

The dashboard also uses a Library for certain frequently used components that helps in reducing the redundancy and increasing the efficiency, latency and readability of the code. The dashboard is equipped with some best UI features that provide the user with a seamless and an easy to use interface. The React project is built with several components and hence these components develop the need to communicate with one or every other component. It is observed that the data flows from the top to the bottom i.e, from the parent components to the child components. This architecture is best suited for small applications, but as the size and complexity of the application being developed grows, this architecture proves to be less efficient. Hence, Redux along with React is made use in the development of the dashboard where by using Redux, it provides a central store where all or some of the application's data is stored and hence in this way, any component can access this store to retrieve any data eliminating the need for the data to flow from the parent component to the child component. This method provides a better state management. The dashboard has been developed with the main aim to ensure the user has the best seamless experience while using the dashboard with the complexity reduced to as minimum as possible.

V. ACKNOWLEDGEMENTS

I am indebted to my guides, Mrs. Sudha R Karbari, Assistant Professor, RV College of Engineering and Pramod MG for the wholehearted support, suggestions and invaluable advice throughout my project work and also helped in the preparation of this thesis.

VI. REFERENCES

[1] Siddharth Mahajan, Mitesh Parekh, Hardik Patel, "BRB dashboard: A web-based statistical dashboard",2017 International Conference on Innovations in Information, Embedded and Communication Systems (ICIIECS).

[2] Maëlick Claes,Tom Mens, Philippe Grosjean,"maintaineR: A Web-Based Dashboard for Maintainers of CRAN Packages",2014 IEEE International Conference on Software Maintenance and Evolution.

[3] Wasinee Noonpakdee, Thitiporn Khunkornsiri, Acharaphun Phothichai, Kriangsak Danaisawat, "A framework for analyzing and developing dashboard templates for small and medium enterprises", 2018 5th International Conference on Industrial Engineering and Applications (ICIEA).

[4] Benjamin Mayer, Rainer Weinreich, "A Dashboard for Microservice Monitoring and Management", 2017 IEEE International Conference on Software Architecture Workshops (ICSAW)

[5] Olga Baysal, Reid Holmes, Michael W. Godfrey,"Developer Dashboards: The Need for Qualitative Analytics", 2013, IEEE Software.

[6] Duong Thi Anh Hoang, Thanh Binh Nguyen and A Min Tjoa. "Dashboard by-Example: A Hypergraph-based approach to On-demand Data warehousing systems". International Conference on Systems, Man, and Cybernetics COEX, Seoul, Korea on. IEEE, October 14-17, 2012.

[7] Hideaki Yanagisawa,"Extension of Web-Based Software Development Environment", 2015 IEEE 29th International Conference on Advanced Information Networking and Applications Workshops.

[8] J. L. Herrero, F. Lucio, P. Carmona, "Web services and web components", 2011 7th International Conference on Next Generation Web Services Practices.

[9] Hu Ran, Wang Zhuo, Xu Jianfeng," Web Quality of Agile Web Development", 2009 IITA International Conference on Services Science, Management and Engineering.