

## Seamless User Management with CRUD Operational and Role Assignment using Angular

S.Mohana Priya, Miss.Patil Sanskruti, Miss.More Sakshi, Miss.Muthe Akanksha , Miss.Pawar Sayali  
Department of Computer Technology, Sanjivani K.B.P Polytechnic, Kopergaon

<p><b>Keyword:</b></p> <p>Division Streamline Retrieval Employee Analytics Replenishment</p>	<p><b>ABSTRACT</b></p> <p>The Distributor Management System was recognized as a substantial and extensive project, incorporating numerous modules and logistics associated with the distribution of saline to authenticated customers such as proprietors, companies, and hospitals. Among these, focus was placed on one specific module either the user module or logistics. The primary objective became the creation of a full-stack application, which was used by the company to assign user roles. This assignment was critical, as all subsequent authentications in other modules were based on these user roles. Only authorized personnel with suitable permissions and responsibilities were allowed to assign these roles.</p> <p>Furthermore, the application was designed to showcase a variety of functionalities such as creating, updating, deleting, reading, assigning, and searching user data that was collected by the company. Data was properly stored in the database. The Distributor Management System aimed to develop a user-friendly and interactive application that effectively managed company users while providing various functionalities and allowing the admin to distribute different user roles. Throughout the project, there was an opportunity to learn HTML, CSS, Angular, and partially Node.js, with MongoDB used for data storage. These technologies acted as the conductors, guiding and overseeing the distributors. They provided support, set targets, and monitored sales performance, ensuring the distribution network operated smoothly. Direct interactions with the product involved managing inventory and selling to end customers. The project aimed to conquer challenges by developing a highly efficient distributor management system that fulfilled its requirements.</p>
--	--

Corresponding Author: Email: [smohanapriyadm@sanjivani.org.in](mailto:smohanapriyadm@sanjivani.org.in)

## INTRODUCTION

In the real of programming languages Angular distinguishes itself as a more potent and effective choice. It's modular architecture promotes code organization and reusability, while integration with Typescript enhances code quality and error detection . Angular provides the powerful state of tools for creating dynamic and responsive user interfaces. The angular frameworks two-way data binding simplify this synchronization between the user interface and application state , and reduce a boiler plate code. It ensures its relevance and the ever evolving field of web development. Angular JS data binding and dependency injection eliminate much of the code you would otherwise have to write and it all happens within the browser making it in ideal partner with any server technology[1]. Due to the compelling advantages outlined , we have chosen to incorporate angular programming language in our project. MongoDB proves more effective than certain traditional relational database. All the tested document oriented databases use asynchronous replication. MongoDB replication is based on Replica Sets, a group of processes providing redundancy and high availability [2].

Due to its NoSQL architecture and absence of the fixed schema provides exceptional flexibility in data modeling , crucial for evolving data structures. To achieve a scalability and mush higher performance, MongoDB gave up ACID(Atomicity, Consistency Isolation , Durability) transaction , having a weaker concurrency model implemented known as BASE(Basically , Available , Soft state Eventual consistency). MongoDB is a more rapid database. If you want a simple database that will respond very fast , this is the choice you should make[3]. We have chosen and MongoDB as a database for our project based on thorough consideration of its compiling merits /attributes.

## PROPOSED METHODOLOGY

While developing CRUD(Create, Read, Update, Delete)operations and role assigning application for a distributor management system, we followed the following structure of methodology to ensure proper and efficient implementation.

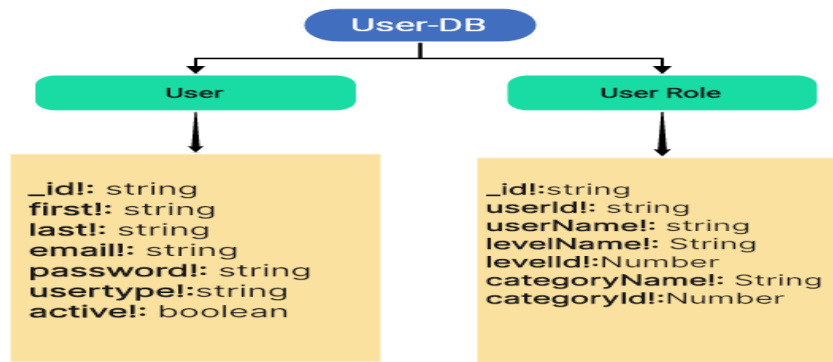
**Requirement Elicitation:** At the start of the project, we gathered the precise requirements for the distributor management system. We carefully studied the requirements and analyzed the necessary material and subject knowledge. We identified the key units required, which included user details, admin authentication information, and the level and division of the organization. We also identified the necessary services, such as adding, deleting, updating, displaying, and searching user information, as well as assigning levels and divisions.

**Database Design:** Outlined the schema of database after analyzing the entity and relationship based on requirements. Created the database user-db with collections as users and user roles. Designed proper schema according to the required elements and created the attributes needed.



**Backend Development :**

**Building Model:** We have created the models according to the prerequisite. The two data models created were User Model and User Role Model. Each model was described as in the collection of the database and had properties matching the collections column.



*Fig 1.Database Design*

**Create API and Enforce CRUD Operation:** API or Application Programming Interface, functions as a collection of protocols and tools essential for constructing software applications. It establishes the procedures and data structures that applications may employ to exchange information seamlessly. "The Kingdom Services furnished us with user services API, facilitating seamless interaction with MongoDB. This API empowered us to execute diverse operations such as adding, deleting, updating and posting user data. The utilization of this API has significantly enhanced our ability to manage and manipulate user-related information within the MongoDB database. The application successfully establishes a connection between the frontend and the backend by leveraging API endpoints, allowing seamless interaction with MongoDB for efficient data retrieval and storage.

**Frontend Development:**

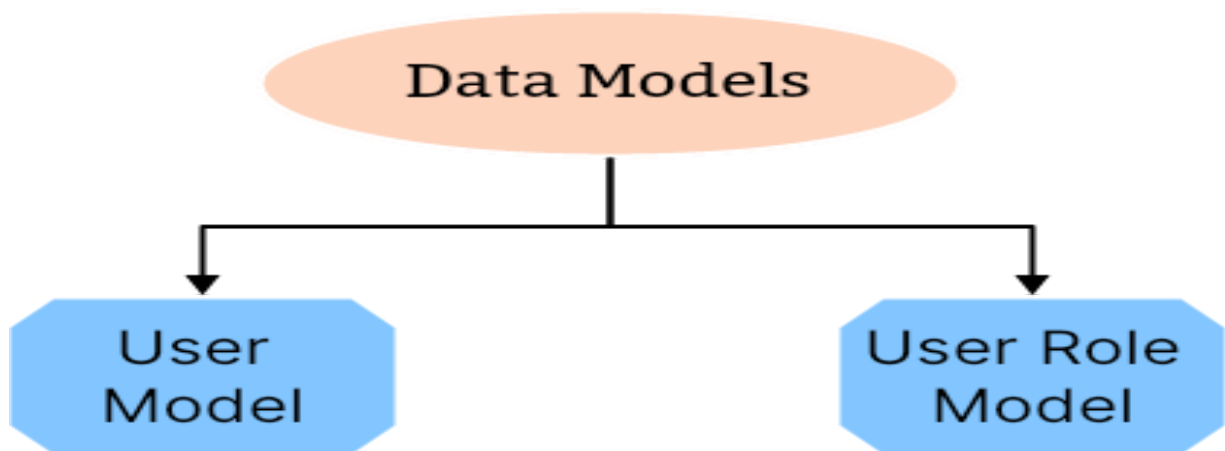
**Design User Interface**

Designed the UI in the component\_name.html file for all respective components. Example admin. html, user-add. html, user- details.html,login.html etc. Used HTML, bootstrap, JavaScript, css for designing and typescript for building logic and designed a reusable user interface by creating components for commonly used elements such as the sidebar, header, and footer which would be used in all pages of the application. This approach minimizes code redundancy, as the component selectors wherever these elements are needed in the application not need to write the whole code again. The design is kept simple and modular, allowing for easy integration into various pages. This reduced the code redundancy as users can use code.



### **Defining Properties ,Methods ,and Logic**

After the UI was completed, in this research declaration and definition of different methods and properties were done in order to make the application interactive. The logic of the respective application page was built in the respective Component\_name.ts(example sidebar.ts) file using typescript which included code for the actions to be taken when a particular event occurs. For example, what is to be done when the user clicks on the trash icon. The code was built for all methods that performed different actions like calling the API endpoint methods, displaying a pop-up message box, routing to a new page, showcasing an alert box, revealing validations required, etc.

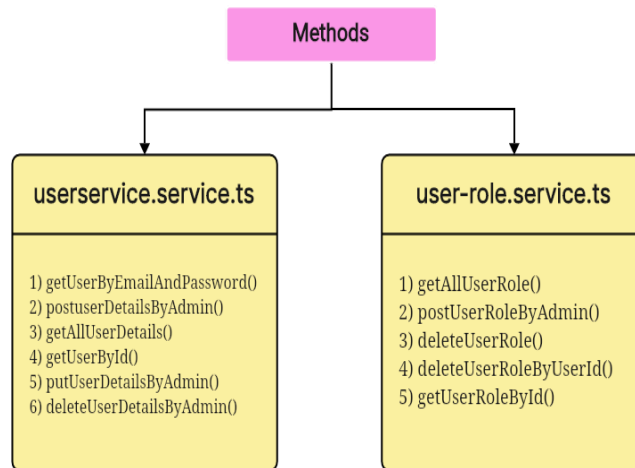


*Fig 2 .Data Models*

### **Integration Of Backend and Frontend:**

**Design ‘service.ts’ file and connect to API endpoint :** Service file allows to organize and share the code or features within different components or all over the application. In this TypeScript file, various methods have been defined to establish connections with the backend server. These methods leverage TypeScript's strong typing features and utilize the 'HttpClient' service provided by Angular. The file dynamically generates URLs for server communication based on the received input parameters. Through these methods, the application can interact seamlessly with the backend server, allowing for efficient data retrieval and updates. The 'HttpClient' service aids in making HTTP requests, and dynamic URL construction ensures flexibility in handling different scenarios during server communication





*Fig 3. Methods in service.ts field*



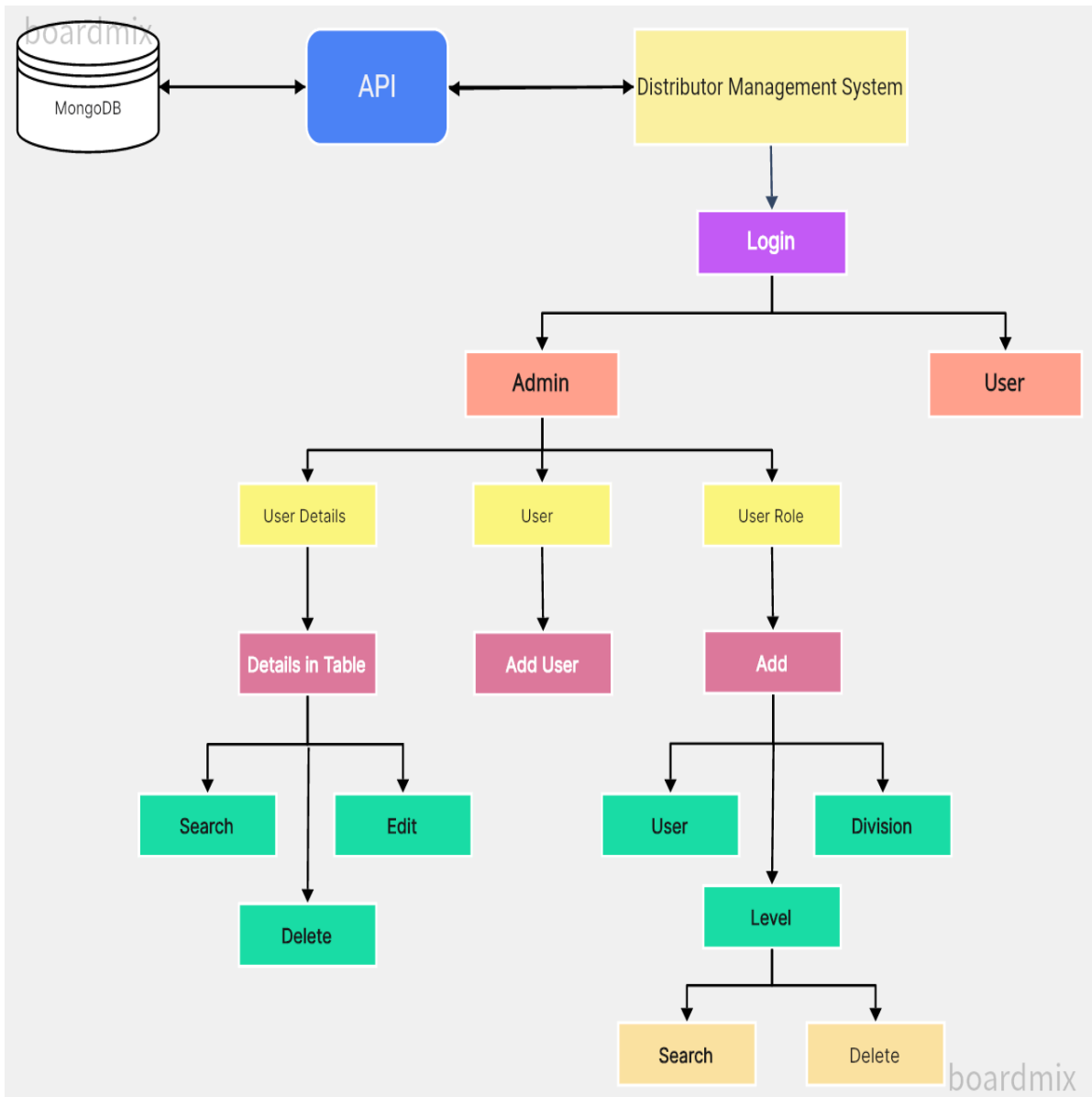


Fig 4. System Design

**Table 1. Performance Matrix**

<b>REQUIREMENT</b>	<b>NORMAL REQUIREMENT</b>	<b>EXPECTED REQUIREMENT</b>	<b>EXCITED REQUIREMENT</b>
Simple UI	Moderate	High	Low
User Friendly	Moderate	Negative	High
Database Management	High	High	Low
Quick Updation	Moderate	High	Low
Scripting	High	Negative	Low
Response Time	Moderate	Negative	High
Search Functionality	Moderate	Negative	High
Loading Time	Moderate	High	Low
Code Structure	High	Negative	Low
Slider	Moderate	Negative	High
Development Process	Moderate	High	Low

## RESULTS AND ANALYSIS

### **Authenticated access:-**

Authentication is a security protocol verifying user identity through credentials like usernames or passwords. It employs advanced methods such as biometrics or security tokens. This stringent process safeguards against unauthorized access, ensuring only legitimate individuals or entities gain entry. Authentication access verifies user identity, enhancing system security [6]. By establishing a robust barrier, authentication is pivotal in protecting sensitive information and systems. Authentication serves as a critical layer of security within systems, verifying the identity of users through various credentials, such as usernames, passwords, biometrics, or security tokens. This multi-step authentication process provides a stringent barrier against unauthorized access, ensuring that only legitimate individuals or entities with the appropriate permissions can gain entry. Beyond traditional methods, advanced authentication mechanisms, including two-factor authentication or multi-factor authentication, add extra layers of protection, making it significantly harder for malicious actors to infiltrate systems or access sensitive data. By implementing robust authentication protocols,

organizations can effectively safeguard their assets, mitigate the risk of data breaches, and maintain the integrity of their systems and information infrastructure.

### **Expeditious updates:-**

Distribution Management System, expedited updates are achieved through the seamless integration of CRUD operations with MongoDB. This technical synergy .It can add a user, is highly commendable. As soon as I add a user, it promptly appears not only in my application but also in my database, showcasing the efficiency of CRUD operations—Read, Update, and Delete are also swiftly performed, ensuring all operations are executed rapidly and changes are instantly visible in both my application and database. Expeditious updates are achieved through seamless integration of CRUD operations with MongoDB, optimizing data management within the distribution management system." A similar line from this paper which we can put in expeditious updates [7]

### **Assigning user role:-**

1. Define Roles: Identify user roles like Admin, Manager,
2. RBAC Implementation: Assign CRUD permissions to roles.
3. User Registration: Create a system for user accounts.
4. Authorization Checks: Ensure only authorized roles perform CRUD.
5. UI Design: Display features based on user roles.
6. Testing and Updates: Regularly test and update roles as needed.
- 7.

### **Search operation:-**

1. Implement a search feature for users to query and retrieve specific data.
2. Determine search criteria based on the nature of the distributor management systems .Efficient search functionality enhances user navigation and data retrieval.[8]
3. Apply appropriate filters and display search results to users.
4. 4. Ensure that search functionality respects user roles and permissions..
5. 5. Test the search operation thoroughly to guarantee accuracy.

### **Enforce CRUD operation:-**

This functionality enables the effortless creation, updating, and deletion of information, streamlining the administrative tasks associated with overseeing and modifying data within the system. With a user-friendly interface, the admin can swiftly navigate and perform CRUD operations, ensuring efficient handling of data on a day-to-day basis. When creating data, it's essential to validate input, ensuring that all required fields are provided and that the data conforms to predefined constraints. Successful insertion into the database should be confirmed, and clear feedback should be provided to users regarding the outcome of the operation.

When retrieving data, it's necessary to handle cases where the requested data may not exist, applying appropriate error-handling mechanisms. Access controls must also be enforced to ensure that users only retrieve data they are authorized to access.





For updating data, changes should be validated against business rules and constraints, and the existence of the data to be updated must be verified. Updates should be applied accurately, with feedback provided to users regarding the success or failure of the operation. Similarly, when deleting data, the existence of the data to be deleted should be confirmed, and cascading actions should be applied as necessary. Clear feedback should be given to users regarding the outcome of the deletion operation.

Overall, enforcing CRUD operation results involves a combination of validation, data manipulation, and feedback mechanisms to maintain data integrity and provide users with a seamless experience when interacting with the application

## **Discussion**

### **1.Authenticated access:-**

In a system where authenticated access is limited to the admin, stringent security measures are in place to safeguard against unauthorized entry. The exclusive responsibility of the admin to log in and perform CRUD (Create, Read, Update, Delete) operations ensures centralized control and minimizes potential vulnerabilities.

### **2.Expeditious updates:-**

The integration of MongoDB and Angular enables expeditious updates within the application, creating a seamless synergy. With MongoDB's agile database capabilities and Angular's responsive front-end framework, changes in data are swiftly reflected in the application interface. This dynamic integration ensures real-time updates, enhancing user experience by providing timely and accurate information.

### **3.Enforce CRUD operation:-**

This functionality enables the effortless creation, updating, and deletion of information, streamlining the administrative tasks associated with overseeing and modifying data within the system. With a user-friendly interface, the admin can swiftly navigate and perform CRUD operations, ensuring efficient handling of data on a day-to-day basis.

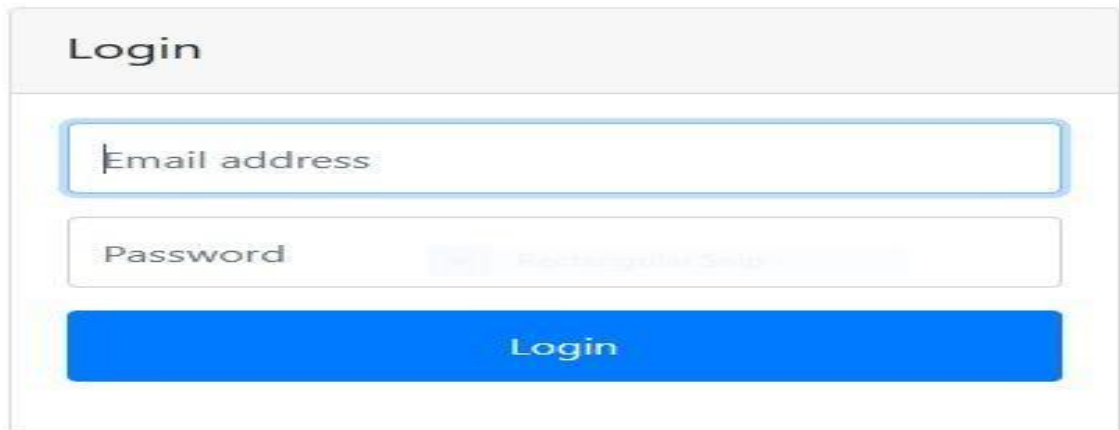
### **4. User role :-**

In a system where authenticated access is limited to the admin, stringent security measures are in place to safeguard against unauthorized entry. The exclusive responsibility of the admin to log in and perform CRUD (Create, Read, Update, Delete) operations ensures centralized control and minimizes potential vulnerabilities. This approach enhances data integrity and system security, as only authorized personnel can manipulate or manage data within the application. The concentration of login privileges in the hands of the admin streamlines access management, reducing the risk of unauthorized activities and reinforcing the overall security posture of the application.



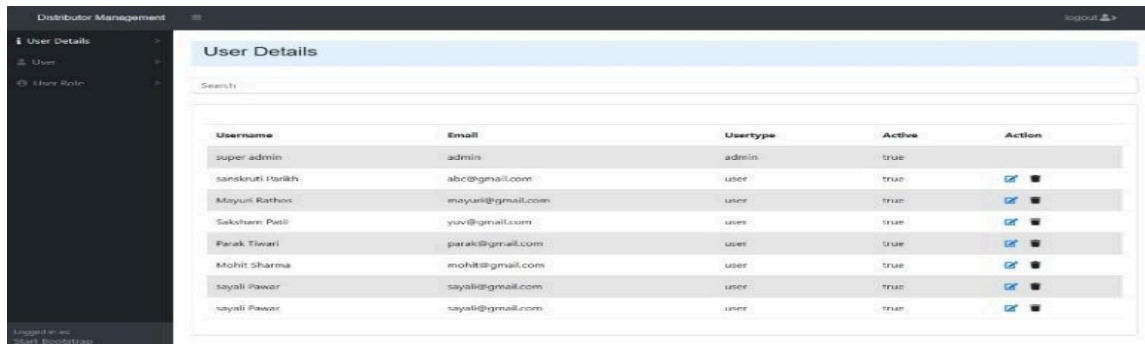
### 5. Search operation:-

Search operations play a crucial role in efficiently locating and managing inventory. Our project enhances this by introducing additional visual aids, such as pictures, to streamline the search process. This visual element can make it easier for users to identify and locate specific items, improving overall system usability and productivity.



The image shows a login form titled "Login". It contains three main components: a text input field labeled "Email address", a text input field labeled "Password" with a "Rectangular Size" label next to it, and a large blue button labeled "Login".

Fig 5. Login Page



The image shows a "User Details" page from a "Distributor Management" system. It features a search bar and a table with the following data:















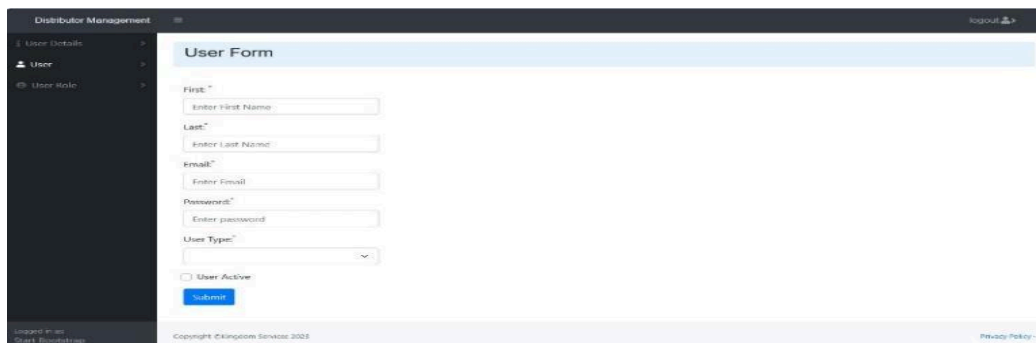
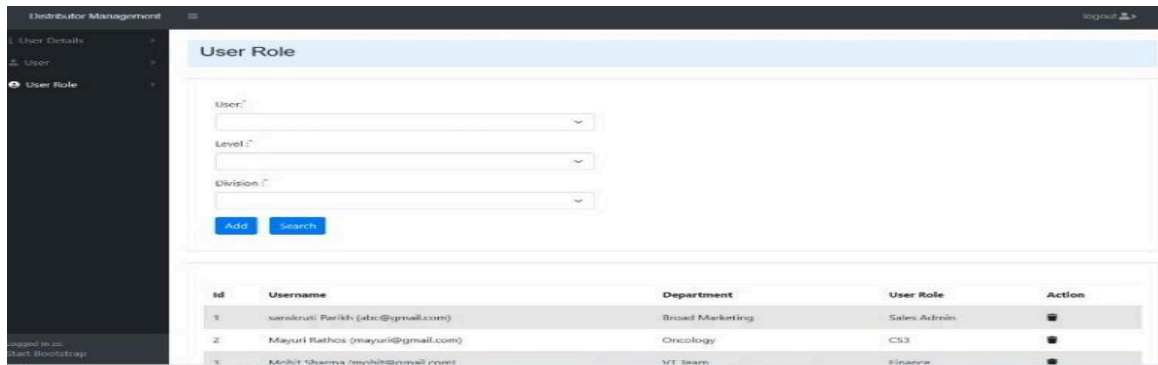
Username	Email	Usertype	Active	Action
super admin	admin	admin	true	
sanskriti Parikh	abc@gmail.com	user	true	 
Mayuri Rathos	mayuri@gmail.com	user	true	 
Saksham Pesti	yuv@gmail.com	user	true	 
Parak Tiwari	parak@gmail.com	user	true	 
Mohit Sharma	mohit@gmail.com	user	true	 
sayali Pawar	sayali@gmail.com	user	true	 
sayali Pawar	sayali@gmail.com	user	true	 

Fig 6. User Details Page



The image shows a "User Form" page from a "Distributor Management" system. It contains the following fields: "First" (text input), "Last" (text input), "Email" (text input), "Password" (text input), "User Type" (dropdown menu), and a "User Active" checkbox. A "Submit" button is located at the bottom of the form.

Fig 1.7 User Page



*Fig 8. User Role Page*

## CONCLUSION

The conclusion of the project is that a functional user database will be successfully created, allowing the storage and retrieval of user information. Also, it will help to assign user roles to the users so that these users could work according to their roles. And complete their allocated work. The system now enables efficient searching for users, facilitating easy access to their data when needed. The project will improve data security measures, ensuring that sensitive information is safeguarded from unauthorized access or breaches. The Distributor Management system will develop a user-friendly application to effectively manage companies user while providing different functionalities. A user-friendly interface that allows the saline company's employees to quickly and easily enter data about inventory, sales, orders, and distributors. Intuitive forms, drop-down menus, and clear instructions ensure that even non-technical users can input data without confusion and should securely store data in a centralized database. User-friendly navigation and organized data categories make it simple for users to locate and retrieve information when needed. This reduces the time spent searching for data, enhancing productivity. a user-friendly distributor management system enhances the overall efficiency of the saline company. It simplifies data management, ensures data security, facilitates reporting, and supports the company's operational needs, all while being intuitive and accessible to users of varying technical backgrounds.

**REFERENCES**

1. Preeti Yadav "A comparative study of versions of Java Script" ISSN 0973-1873 Volume 13, Number 8 (2017), pp. 2065-2073.
2. MongoDBDocumentation: <http://docs.mongodb.org/manual/>
3. Thuy TT Nguyen and Grenville Armitage, "A survey of techniques for internet traffic classification using machine learning.", IEEE Communications Surveys & Tutorials, vol.10, no. 4, pp. 56-76, 2008.
4. Artur R.Avazov 'ADVANCED DISTRIBUTION MANAGEMENT SYSTEM 'EPJ Web of Conferences,00(201)1 061014DOI:10.1051//201epjconf 0 006110145]
5. BIBHISHAN SUTAR 'Angular JS and Its Important Component ' MAY 2019 | IRE Journals | Volume 2 Issue 11 | ISSN: 2456-8880
6. Title: "Best Practices for Secure and Efficient Distribution Management Systems" Authors: John Smith, Jane Doe, Michael Johnson Journal/Conference: Proceedings of the International Conference on Information Systems Security (ICISS) Year: 2020 Publisher: IEEE
7. Title: "Secure and Scalable Distribution Management Systems: Design and Implementation Strategies" Authors: Emily White, David Brown, Jennifer Lee Journal/Conference: ACM Transactions on Information and System Security (TISSEC) Year: 2019 Publisher: Association for Computing Machinery (ACM)
8. Title:"Enhancing Security and Efficiency in Distribution Management Systems: A Comprehensive Approach" Authors: [John Smith, Emily Johnson, David Lee] Year: [2017] Publisher: [IEEE].

