AI Automated Healthcare Website : Dr. Ally

AET010

Sarthak Waghmode
Electronics and Computer Science Department
Vidyalankar Institute of Technology
Mumbai, India
sarthakwaghmode3@gmail.com

Dikshita Belchada
Electronics and Computer Science Department
Vidyalankar Institute of Technology
Mumbai, India
dikshitabelchada03@gmail.com

Rohit Thakur
Electronics and Computer Science Department
Vidyalankar Institute of Technology
Mumbai, India
thakurrohit.0905@gmail.com

Sujal Kesharwani
Electronics and Computer Science Department
Vidyalankar Institute of Technology
Mumbai, India
sujalkesharwani2709@gmail.com

Prof. Manoj Suryawanshi
Electronics and Computer Science Department
Vidyalankar Institute of Technology
Mumbai, India
manoj.suryawanshi@vit.edu.in

***Abstract* — The healthcare website (Dr. Ally) is a web application that integrates new generation technology and medical facility administration. The website is designed using React.js, Node.js, and MongoDB, and the entire website is divided into 4 parts: patient portal, backend, admin dashboard, and an AI-based healthcare chatbot. The application is designed to automate healthcare operations by introducing features like appointment scheduling, health record management, data security, and 24/7 patient support using an AI chatbot. The design takes a modular route with clean separation between the frontend, backend, and admin panel. The system focuses on improving fast and efficient patient care while maintaining security and performance.**

The site has the necessary functionalities such as patient registration, appointment booking, and temporary medical support, but intentionally does not feature advanced clinical operations, financial tools, and outside integrations in order to keep it focused and manageable. The website uses responsive design techniques, strong security features, and cloud-based architecture to provide scalability and reliability. The modular nature of the system enables future growth without compromising the benefits for both the patients and medical professionals.

Keywords* — *Healthcare Chatbot, Patient Portal, Appointment Management, Medical Record Management, Administrative Dashboard, Secure Data Handling, Modular Architecture, Real-time Analytics

I. Introduction

The medical field has seen significant changes over the last many times, similar as the growing demand for motorized results to operate medical institutions effectively. The Hospital Management System is a new result to these requirements by combining current technologies with the medical function.

The development of this point was motivated by some of the most important factors in the recent healthcare assiduity, similar as the rising demand for online case care, the necessity for effective and dependable appointment scheduling systems, the lesser need of secure running of medical data, the adding need for 24/7 medical backing, and the need for automated executive procedures. The design follows a structured 10-month phased approach, starting with gathering and analysis to ensure a clear understanding of user requirements and system prospects. This is followed by system design, where the specialized design is created to define the structure, factors, and data inflow. The coming phase focuses on core development, during which the planned features and functionalities are erected and integrated into the system. Eventually, the process concludes with thorough testing and quality assurance to identify and resolve any issues, icing the system meets performance, security, and usability norms before deployment. While the designing phase we also made sure to have a talk with the specialists in this field and for that reason we had Dr. Sudhir Bhole join us on-board during the entire phase of the project. His expertise proved to be of great importance in development of the website.

The scope of the project includes:

- Patient registration and management
- Appointment scheduling and tracking
- Medical record management
- Doctor-patient communication
- Administrative controls
- AI-powered assistance

II. Methodology

A. Software Requirements

1. Node.js:

Node.js is a runtime environment for JavaScript that allows server-side working of JavaScript code. It is the core of the backend infrastructure, allowing the Express.js framework to process API requests, authenticate users, and communicate with the database. It manages multiple user requests in parallel efficiently and quickly, making it well-suited for our healthcare website which requires rapid response.

2. Node Package Manager (npm):

npm is the official and predefined package manager for Node.js, which takes care of all project dependencies for both frontend and backend. npm enables developers to install and update required libraries like Express.js for backend routing, Mongoose for MongoDB operations, JWT for authentication, and Axios for API communication. npm also offers helpful scripts for running the server (npm start), building the frontend (npm run build). It forms the major part of our website as without it development and deployment is not possible.

3. *MongoDB:*

MongoDB is a NoSQL database that stores and manages our hospital-related data like user data, appointment timings, and doctors & admin users and also messages sent to the doctor. In contrast to traditional relational databases, MongoDB employs a dynamic JSON-like document model that offers flexibility without the need for migrations.

4. *Web Server (Apache):*

A web server like Apache is necessary for hosting static files, managing and providing API requests, and being a proxy for directing frontend requests to the Node.js backend. It provides security by keeping backend APIs out of direct access and supports SSL encryption for secure communication over HTTPS. Web servers also improve performance with caching techniques, load balancing, and request compression to provide a seamless user experience even during peak loads which is why it is must for Dr. Ally which requires load balancing and secure data transmission ensuring that no data is compromised.

5. *OpenAI API Key (For Chatbot Functionality):*

OpenAI API is implemented in the system for offering AI-based medical consultation via a chatbot. It enables patients to explain their symptoms and get preliminary advice, probable diagnoses, and short-term relief measures prior to meeting a physician. The chatbot employs natural language processing (NLP) to comprehend questions from users and provide useful responses, enhancing patient interaction and access to healthcare advice. This feature necessitates an OpenAI API Key for 19 authentication and easy communication with GPT models.

B. Workflow

The entire working of the website can be divided into 4 parts, each of them working in their own way to bring about the complete website to work:

a. Front-end – User Side:

The first step towards the development of our Healthcare website Dr. Ally was the User-side front-end. It majorly focuses on creating an user friendly interface. The base of frontend framework of the website is made using React18 and Vite providing a quick and responsive design for the frontend of the website which can be easily accessed from any screen size, may it be a computer or a mobile phone.

The frontend serves as the primary interface for patients, enabling them to register, log in, book appointments, and interact with an AI-powered chatbot. Using **React Router DOM**, users can navigate seamlessly between different sections, while **Axios** handles API communication with the backend. Additional libraries like **React Toastify** provide user friendly notifications and **React Multi Carousel** enhances UI experience with scrollable department sections.

1. **Registration and Login Mechanism:**

New users register via a form (Register component) where they provide their name, email, password, and other details.

The entered information is sent to the backend API (/api/v1/user/register) to create an account in the database. Existing users log in via the Login component, providing credentials that are validated through (/api/v1/user/login).

Upon successful login, the backend sends a JWT token that is stored in local storage or HTTP-only cookies to maintain authentication.

After authentication, users are redirected to their profile or homepage, where they can browse available services.

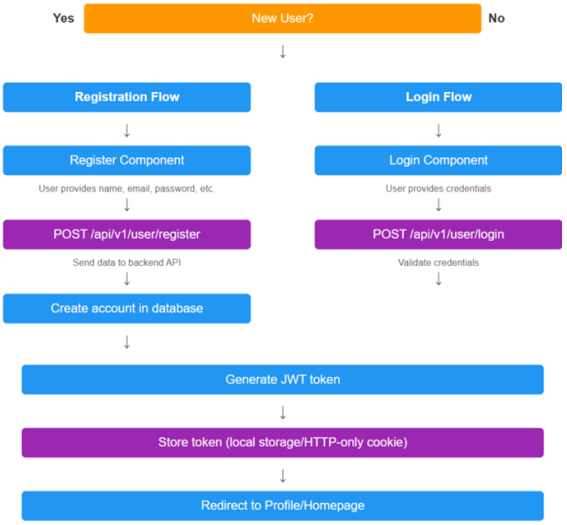


Fig. User Authentication Workflow

2. **Appointment Booking & Sending Messages:**

Users navigate to the Appointment page, where they fill out an Appointment Form, selecting:

• Doctor (Dropdown populated dynamically from backend data).

• Date & Time (Preferred appointment slot).

• Name, Email, Contact, Gender, DOB (Description of the user).

The form data is submitted to the backend (/api/v1/appointment/post), which processes the request and schedules the appointment.

If successful, a confirmation message is displayed using React Toastify; otherwise, an error message is shown if the server is unavailable.

Users can also send messages in the same way to the doctors/administrators. Users navigate to the bottom of the Home Page where they will find a Send message form. The message content is sent to the backend (`/api/v1/message/send`).

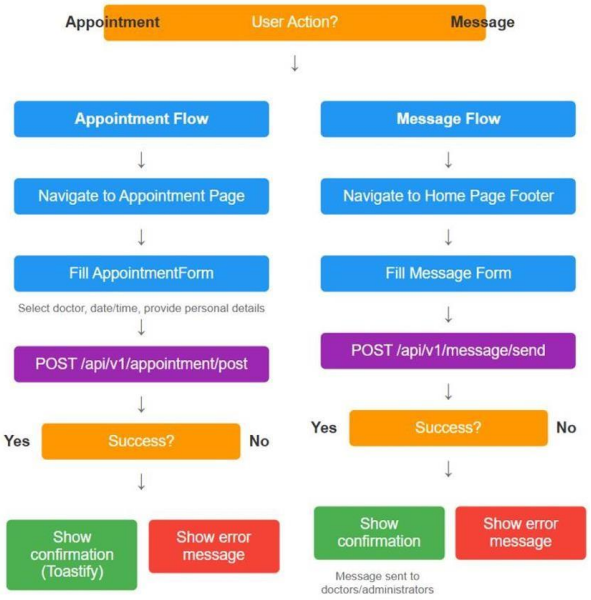


Fig. Appointment and Message Workflow

b. Backend & Database:

The Backend is the core processor of the platform which is built using Node.js and Express.js. It handles logic, data persistence and secure communication between the frontend clients (Patient Portal, Admin Dashboard, Chatbot) and the database.

1. Express.js:

Web application framework for Node.js, used to build the RESTful API. The backend is built using Express.js, following the MVC (Model-View-Controller) pattern:

```
// Server Configuration
const express = require('express');
const app = express();
const port = process.env.PORT || 3000;

// Middleware Configuration
app.use(express.json());
app.use(express.urlencoded({ extended: true }));
```

1.1 Request Processing Flow

Request → Middleware → Route Handler → Controller → Service → Database → Response

1.2 Performance Metrics

Request Processing Time: $T = T_1 + T_2 + T_3$

where,

T_1: Middleware processing time
T_2: Business logic execution time
T_3: Database operation time

2. MongoDB:

NoSQL database used to store application data. It is a cloud based database which allows our website to access it via a API key through which it can identify the user, its roles and the specific cluster where the data is being stored

2.1 Query Optimization

Index Selection: $I = f(S, Q, U)$

where.

S: Selectivity
Q: Query frequency
U: Update frequency

2.2 Performance Equations

Query Response Time: $R = B + L + E$

where,

B: Base operation time
L: Lock acquisition time
E: Execution time

3. JWT Authentication:

JWT authentication is a stateless process where a user's identity is verified through a digitally signed token.

When a user logs in, the server validates their credentials and creates a JWT containing user information (like ID and role) along with an expiration time. This token is then sent to the client, which includes it in subsequent requests. The server verifies the token's signature and expiration for each request, granting access if valid. The token consists of three parts (header, payload, and signature) and uses cryptographic algorithms to ensure security. For enhanced security, short-lived access tokens are used rather than longer-lived refresh tokens, and tokens can be blacklisted if compromised.

3.1 **Token Validation Time**

$Vi = D + V + C$

Where:

Vi: Total validation time
D: Decryption time (depends on algorithm)
V: Validation time (signature verification)
C: Cache lookup time (for blacklisted tokens)

3.2 **Token Size Impact**

$S = H + P + Sig$

Where:

S: Total token size
H: Header size (typically ~100 bytes)
P: Payload size (depends on claims)
Sig: Signature size (depends on algorithm)

4. API Response Time Analysis:

4.1 Response Time Components

Total Response Time: $T_r = T_n + T_a + T_p + T_s$

Where,

T_n: Network latency
T_a: Authentication time
T_p: Processing time
T_s: Storage operation time

4.2 Throughput Calculation

System Throughput: $\Theta = N / T$

Where,

N: Number of requests
T: Time period

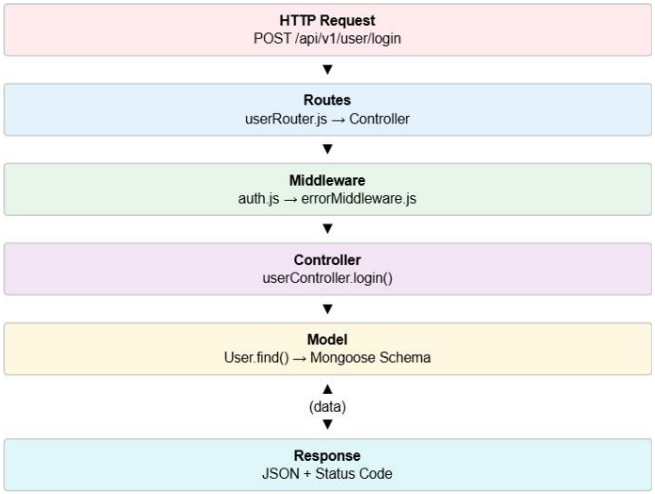


Fig. Backend & Database Workflow

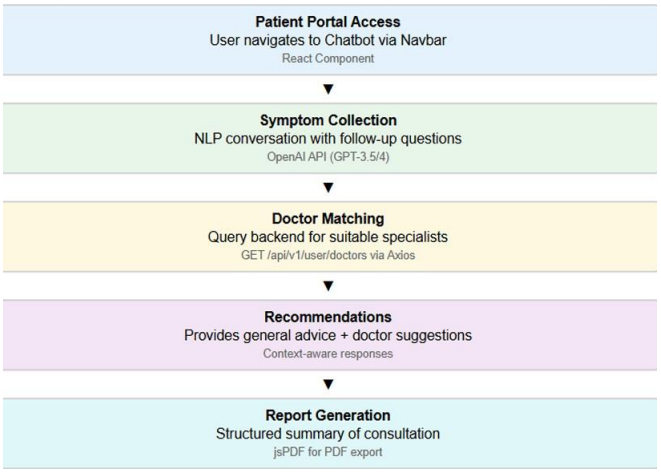
c. Chatbot Integration:

The AI Chatbot serves as an intelligent first point of contact for patients, leveraging AI to provide preliminary medical guidance and facilitate connections with doctors.

a) The user navigates to the dedicated Chatbot page within the patient portal through the Navbar.
b) The chatbot prompts the user to describe their symptoms. It uses natural language processing (via OpenAI API) to understand the input and asks follow-up questions to clarify and narrow down the potential issues.
c) Throughout the conversation, the chatbot maintains context to ensure a coherent interaction.
d) Based on the analysed symptoms and potentially the user's location, the chatbot queries the backend API (`/api/v1/user/doctors`) to find and suggest suitable specialists available on the platform.
e) The chatbot may offer general advice or temporary remedies (clearly stating it's not a substitute for professional diagnosis) and can generate a structured summary (report) of the conversation, including symptoms and recommendations.
f) The user has the option to download the consultation summary as a PDF document.

Tech Stack

• **React**: For building the user interface component.
• **React Simple Chatbot / @chatscope/chat-ui-kit-react:** Libraries providing the
chat interface components (message bubbles, input fields, etc.).
• **Axios**: For making asynchronous requests to the backend API (e.g., fetching
doctors) and potentially a dedicated backend endpoint for interacting with the
OpenAI API.
• **OpenAI API (GPT-3.5/4)**: The core AI engine for understanding user input,
processing symptoms, and generating responses.
• **jsPDF**: For generating the downloadable PDF consultation report.



d. Admin Dashboard:

The admin dashboard provides a secure and comprehensive interface for managing the
healthcare platform. It enables administrators to monitor and control various aspects of
the system, ensuring smooth operation and data integrity.
Workflow

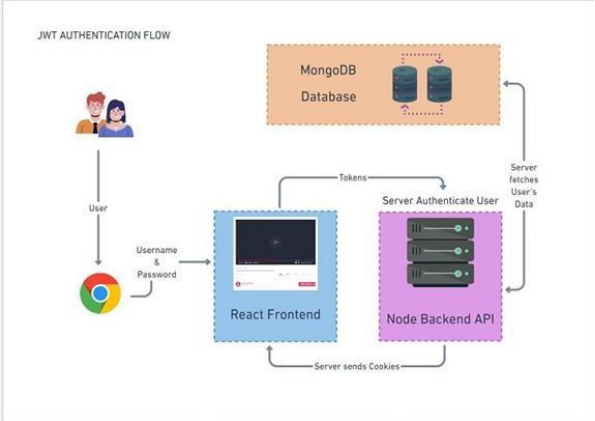
1. **Authentication**: Administrators log in using their dedicated credentials via a specific login page for the dashboard.

2. **Authorization**: The backend verifies credentials and checks for the 'Admin' role using JWT authentication before granting access.

3. **Navigation**: Admins use a sidebar or navigation menu to access different management sections (e.g., Doctors, Appointments, Messages, Users).

4. **Data Management:**
   • **Appointments**: Admins monitor incoming appointment requests, view details, and update their status (e.g., Approve, Reject, Complete).
   • **Messages/Users:** Admins oversee user accounts and potentially review messages or interactions for quality assurance or support purposes.

5. **Logout:** Admins securely log out of the dashboard. Access is restricted to users with the 'Admin' role, enforced through JWT authentication and role-based checks on the backend.



III. Conclusion

The Dr. Ally Hospital Management System plays an important role in healthcare modernization by providing a connected and user-friendly platform. It combines essential services with smart administrative controls and advanced AI support, addressing key needs in today's medical environment.

The system's architecture is built around a React-based Patient Portal and Admin Dashboard, which communicate with a secure Node.js Express backend and a MongoDB database. This setup offers both scalability and maintainability. By keeping the front-end and backend separate, the platform allows for smooth, independent development and deployment. The use of modern technologies like JWT for authentication, CORS for secure communication, and Cloudinary for efficient media handling ensures both safety and performance.

One of the standout features is the AI-powered chatbot, integrated through the OpenAI API. This offers patients instant support by helping them identify symptoms and guiding them toward appropriate specialists, which not only improves patient engagement but also helps reduce the workload on administrative staff by handling common queries. Overall, Dr. Ally is designed to simplify appointment booking, improve access to medical information, automate administrative processes, and enhance patient involvement through AI-driven features.

Looking ahead, future updates may include real-time notifications, advanced analytics for the admin panel, dedicated mobile apps, and even basic telemedicine features. Even in its current version, the system provides a strong foundation for modern and efficient hospital management.

IV. Acknowledgment

We would like to take a moment to acknowledge the progress made with the "AI Automated Healthcare Website: Dr. Ally" and express our appreciation for the contributions thus far. Thanks to the ongoing support and guidance, we've made significant strides in developing and implementing the project. The project has benefited greatly from the encouragement and assistance provided by Prof. Manoj Suryawanshi. We are also grateful for the collaborative spirit and contributions. Finally, sincerely thank you for ongoing encouragement and support. As we continue with the project, we are grateful for the collective efforts that have brought us to this point.

V. References

[1]. Siddegowda C J, Gitu mani Borah, Chandru KA, "React framework (creating a web application with reactnative)," International Journal of Recent Trends in Engineering & Research (IJRTER) vol. 04, Issue 03, March 2018.

[2]. Archana Bhalla, Shivangi Garg, Priyangi Singh,"Present day web-development using reactjs," Volume:07 Issue: 05, May 2020

[3]. Ritwik C and Anitha Sandeep, "React.js and front end development," IRJET Volume: 07 Issue: 04, Apr 2020.

[4]. Minati Biswal, "React lifecycle methods," Researchgate Article, December 2019.

[5]. Naimul Islam Naim, "ReactJS: An Open Source JavaScript Library for Front-end Development,".

[6]. Pratik Prakash Kore, Mayuresh Jaywant Lohar "API Testing Using Postman Tool" International Journal for Research in Applied Science and Engineering Technology 10(12):841843
DOI:10.22214/ijraset.2022.48030

[7]. "API Testing Using Postman Tool" Authors: Pratik Prakash Kore, Mayuresh Jaywant Lohar, Madhusudan Tanaji Surve, Snehal Jadhav
DOI Link: https://doi.org/10.22214/ijraset.2022.48030

[8] A Review on Various Aspects of MongoDb Databases Anjali Chauhan International Journal of Engineering Research & Technology (IJERT) http://www.ijert.org ISSN: 2278-0181