Journal of Advances in Developmental Research (IJAIDR)
E-ISSN: 0976-4844 ● Website: www.ijaidr.com ● Email: editor@ijaidr.com

Smart Web Based Hospital Management System

Mrs. Manimala Veeraiyah[1], Dr. S. Prabakaran[2], Ms. L. Rakasri[3], Ms. Vijetha P[4], Ms. E. Vinotha[5]

[1]Asst. Prof/ Department of CSE, V.S.B Engineering College, Karur, Tamil Nadu
[2] Prof. Department of CSE, V.S.B Engineering College, Karur, Tamil Nadu
[3,4,5] Department of CSE, V.S.B Engineering College, Karur, Tamil Nadu

Abstract

The effective management of hospitals is crucial for the timely provision of health care services and the reduction of workloads to health care workers. Traditional manual hospital management systems (e.g., for patient registration, doctor scheduling, bed allocation, etc.) are not very effective as they take much time and involve many errors and mismanagements. In recent years, web-based hospital management systems have been developed to make the management process easier; however, many of these systems do not come with options to manage for urgent conditions, or the update of resources is on real-time. In this paper, we describe the design and implementation of a Smart Web-Based Hospital Management System that gives hospital administrators and patients a system that is practical and secure to use. The system was made with Java (Spring Boot framework) on the back end, a MySQL database, and HTML/CSS/JavaScript on the front end. Once logged in, administrators are able to manage hospital-related resources (availability of doctors, allocation of beds, etc.) while patients can register online, schedule appointments with doctors, and check available services. A unique aspect of the system was the emergency button, which automatically checks resource availability, thus providing patients with recommendations to proceed during a critical situation. The Proposed system increases efficiency, enables accurate documentation, and yet positively impacts the which is ultimately a benefit to the patients' entire experience over the current process. The pilot implementation shows that the system decreases manual involvement, offers accessible information in real-time, and may be scaled up for use at multiple hospitals and/or city-wide in the future.

1. Introduction

Hospitals are very important institutions in human society, as they provide acute, emergent, and long-term medical care, monitoring, response, and management for patients suffering from numerous climatic conditions, stress, workload and other maladies. Good administration of hospitals is important for not only improving patient care but also reducing under-productivity in a hospital. Traditionally, hospitals have relied heavily on either paper-based or semi-automated record systems to keep patient details, doctor schedules, and appointment records. Any manual routine is likely to introduce data redundancy, errors, and delays, which ultimately hinder the quality of healthcare services delivered to patients. A Hospital Management System (HMS) is an automated system designed to manage digital healthcare records related to patients, doctors, and hospital resources in a secure and reliable manner. Typical features of an automated Hospital Management System include patient registration, logins for different roles (admin,

doctor, user), appointment booking, and secure storage of medical details. The databases and web-based applications are in a centralized format and enable a rapid retrieval of information and reduce the use of paper records.

Moreover, the latest technologies, including the real-time web socket integration, can be used to issue timely alerts concerning the appointment or any emergency, and thereby the responsiveness and provision of healthcare services in a timely manner is guaranteed. Some of the works reviewed have examined web-based hospital management systems. However, many of them do not have sophisticated features like integrated emergency response functionality, real-time bed availability management, or multi-admin access. To mitigate these limitations, this paper proposes the design and implementation of a Smart Web-Based Hospital Management System using modern Java frameworks (Spring Boot, Restful API) for backend, MySQL for the database, and the usual suspects - HTML/CSS/JavaScript for frontend. By implementing software engineering principles of modular design, object-oriented programming concepts, and rigorous database management models, the system provides scalability, security, and maintainability. The system has different login modules for patients and administrators which gives you role-based access control to secure, web-enabled applications. Patients register digitally to book appointments, check doctor availability, and view bed availability. Administrators can manage records by inserting, updating and deleting doctors and bed data. A unique aspect of this project is the emergency button component which analyzes the available resources in real time and recommends the hospital's immediate capacity for dealing with emergencies. This goes hand in hand with the current requirements of smart health information systems that can not only store the information but also aid in decision support and quick reaction to problems in services.

Also, the proposed system was made conscious in its design to be flexible and modular. The entity of Users, Doctors, Appointments, Beds and Emergency is not connected to any other major entity and can therefore be expanded easily in future like a merger with the Io T medical devices, or the hospitals through a city-wide network. The use of relational database management concepts and structured design methodologies has made a strong base for future development. The remainder of the paper is organized as follows: Section II discusses related research and existing systems. Section III explains the proposed system and its important modules. Section IV discusses the methodology and process of implementation. Section V evaluates performance and results. Finally, Section VI concludes the work and describes future directions.

2. Related Work

Ru-chi Dumb-re et al. (2016): The authors developed a healthcare administration system that implemented a domain search feature to allow patients to quickly locate the local health service. The authors aimed at ensuring the patients experienced less delay in accessing care, thus, reducing the burden during the access process. Common interface characteristics were described by the authors, to improve ease of use; however, the assessment of general interface usability was limited, because the authors were primarily focused on service accessibility based on geographical location, outlying important aspects of including other functionality within the hospital modules (i.e., bed availability), appointment scheduling, and physician class allocation. Consequently, the incorporation of the interface and the mentioned limitations rendered the system less effective as a constructive hospital management system.

Dig Vijay H. Gadhari et al. (2016): In this research, a web-based hospital management system was proposed and designed. The hospital management system facilitated core processes such as patient registration, login, and secure storage of medical records, which helped reduce manual paperwork and increased accuracy. The benefits of managing data with digital systems vs paper-based systems were also highlighted. Even though this system was proficient at managing hospital records, it did not incorporate automated processes (e.g. real-time doctor schedule updates, automatic notifications for patients regarding appointments). As a result, the hospital management facilitated basic processes only without introducing dynamic hospital processes.

Olusanya Olamide et al. (2015) - The researchers proposed a system for management in a hospital setting, using Java technologies to successfully build a well-structured and functional system that provided for accurate medical-record maintenance for doctors and patients. Their solution significantly improved data maintenance transparency, and improved health- related information sharing while limiting redundancy in tasks completed by hospital administrators. Furthermore, their solution provided a simple mechanism for booking appointments that may facilitate some small clinic operations. Unfortunately, Olusanya's design did not have any advanced functionalities such as emergency management, bed allocation, or provision for multiple administrators, which would be essential in larger hospitals. This limited the system's capacity for scalability or real-time usability.

Gunjan Yadav et al. (2016): This paper proposed a more sophisticated hospital data management system with a stress on secure storage, structured retrieval, and access to patient and doctor information. The reliability of the health care applications as discussed in this study was critical and the researchers showed how the use of digital records would enhance the working processes in the hospital. However, the study still focused mainly on the backend of the data management system without sufficient modules on the patient-facing aspects. As an instance, features such as the ability to verify doctor availability, register patients online, and/or integrate emergency functions were not featured making this model less usable in a real-world hospital management context.

Adebisi O.A. et al. (2015): The authors created and implemented a web-based hospital management framework that featured modules for patient registration, doctor management, and appointment booking. The framework offered scalability and the option to be customized for different hospital situations. In addition, convenience of access to medical services for patients improved, and doctor allocation processes became easier. Nevertheless, the system was not linked with real-time updates of hospital resources, such as monitoring number of beds, or immediate response to emergencies. This rendered this system less efficient for fast-changing medical situations requiring real-time decision-making.

Anwarul Abedin and colleagues, (2018) proposed a cloud-based hospital information system to improve data access and collaboration among hospital departments. The authors included patient data on a cloud server to enable access to a patient's medical records remotely, this feature could be useful when working with patients across multiple locations when healthcare. The hospital information system was also cloud based, hence allowing data redundancy and backup security. The problems are found however in dependence on cloud infrastructure in terms of privacy of data, high dependence on the internet and

adherence to regulations on medical data. In addition, emergency medical needs, sure doctor, and patient needs were not factored into the system, and priority needs, which reduced the opportunity of their use.

Adepoju et al. (2019): The authors created an electronic medical record (EMR) management system rather than the manual one. The authors worked out a system according to which the history of a patient, details of his or her prescriptions, and diagnostic findings could be safely stored, which resulted in fewer administrative errors and higher efficiency. The communication experience of the patient-doctor was more satisfied by the EMR system that allowed sharing the records virtually. Although useful, the EMR management system lacked more advanced features like doctor on-call, automated appointments, and resource tracking. Overall, the EMR management system was an effective tool for digitization of existing processes, but not as a comprehensive automation of hospital management.

S. Kumar et al. (2020): The research offered a hospital resources management system which aimed to consolidate various hospital resources such as bed allocation, staff scheduling, and appointment scheduling on one platform. The study argued that centralized systems would create unified hospitals to reduce inefficiency in complex hospital operations. This system was capable of managing complex workflows, but such a centralized operating system required high computational resources and high technical resources, which small, low-budget hospitals could not afford. Furthermore, the system does not have lightweight emergency modules for specific emergency use, reducing adaptability to allow timely access to resources in a real-world situation.

3. EXISTING SYSTEM

In the majority of hospitals, managing patient records, appointments, and bed assignments is done by hand using paper-based files or by using simple apps that are not integrated. Patients have to come to the hospital personally to register, check to see whether a doctor is available, and book an appointment. Likewise, administrators update doctor schedules and bed availability using handwritten logs or basic spreadsheets. These systems work reasonably well for smaller clinics, but they each have a number of limitations:
Manual dependency: Higher risk of errors and mismanagement.
Delayed updates: Doctor schedules and bed status are not changing in real-time.
No emergency handling: Critical cases require staff to check and confirm resources manually, wasting time for those in need of emergency care.
Limited accessibility: Patients must go to the hospital; there is no online access to data.
Single point of control: Often managed by one staff member, allowing the system to bottleneck. The existing system is thus inefficient, time-consuming, and unfeasible for handling larger patient volumes or emergency cases; this inspires a need for a Smart Web-Based Hospital Management System with enhanced real-time, securely and saleable features.

4. PROPOSED SYSTEM

The Smart Web-Based Hospital Management System (SWHMS) proposed in this paper aims to ease and digitalize hospital activities by managing patients, doctors, appointment schedules and bed vacancy through an online centralized platform. The system ensures accessibility in real-time, accuracy in data

management and effective response during emergent situations. The system is designed using Spring Boot (Java), MySQL, HTML/CSS and JavaScript. Hence, it is economical, secure and able to be sellable.

A. System Architecture:

The structure consists of the following major elements:

Front-end (User Interface): The front end was created using HTML, CSS, and JavaScript to develop user-friendly dashboards for patients and the administrator.

Back-end (Application Logic): The application logic was developed using Spring Boot and REST API to process requests (registration, appointment booking, and data updates).

Database: The database was developed to use MySQL to store user details, doctors' schedules, appointments, and bed availability.

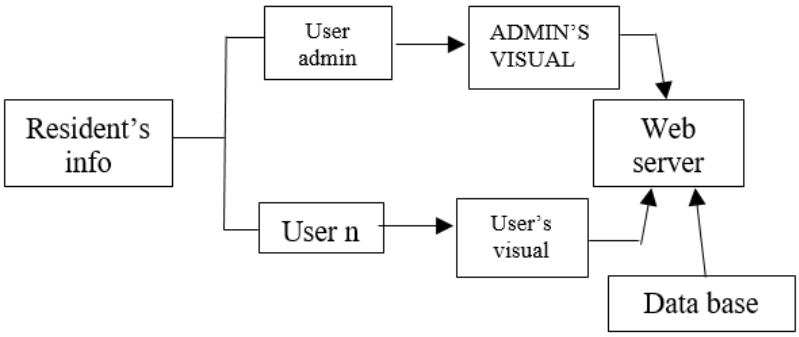
Modules: User Management (registration, login, and profile). Doctor Management (availability, specialization, and scheduling). Appointment Booking (selecting and confirming the doctor). Bed Management (bed allocation and available updates in real-time). Emergency Module (quick analysis on available doctors/beds).

B. Workflow:

The User/Admin logs into the system using the Login Page.

Admin logs into the system to manage the hospital resources (i.e., doctors, beds, and appointments). Patients are able to log in after registration to book appointments, check availability of doctors or check the status of the beds. Once the Emergency Button is clicked the system will scan the database and propose doctors and beds available within seconds. The data is compiled and updated in the same Central Database to ensure uniformity and at the same time to have a certain real time information which is readily accessible.

At any point, patients and Admin can also view the latest information on their dashboard.



5. METHODOLOGY

The modular and layered techniques are employed in development approach which gives maintainability, scalability and reliability.

A. Requirement Analysis:

The needs were received with the help of the surveys of the published literature and study of the existing hospital systems. Functional requirements included patient registration, appointment scheduling, bed

allocation, and emergency services. Non-functional requirements encompassed security, usability, and scalability.

A. System Design:

The system employs a three-tier architecture:

Presentation Layer (Front-end):

Provides simple-to-read dashboards from the patient's and admin's perspective using web technologies.

Application Layer (Back-end):

Implements the business logic with Spring Boot using REST API.

Data Layer (Database):

MySQL database manages data for patients, doctors, appointments, and bed allocation. UML diagrams (use case, class, and sequence diagrams) were used to model the system's structure.

C. Database Design:

The schema is organized to be normalized in order to avoid redundancy, ensure consistency between tables, and add primary and foreign key constraints. Tables are created for: Users, Doctors, Appointments, Beds, and Emergency Logs.

D. Implementation:

The system was implemented using:

BACKEND: Java (Spring Boot, Hibernate/JPA).
FRONTEND: HTML, CSS, JavaScript.
DATABASE: MySQL.
COMMUNICATION: REST API and Web-socket for real-time updates.
CODE IS ORGANIZED INTO LAYERS: Controller, Service, Repository, Model, to follow the MVC pattern.

E. Testing:

Testing occurs at three levels:

UNIT TESTING:

Individual units or components (i.e. login, registration, appointment, etc.) are tested.

INTEGRATION TESTING:

Interaction between front-end, back-end, and database are verified.

SYSTEM TESTING:

The end-to-end system is tested for functional correctness, performance and security.

F. Deployment:

The system is deployed on a local Tomcat server in the development environment, with capability for cloud deployment in the future. Deployment requires configuration of environment files (application properties), database connections, and the running of the Spring Boot application.

6. TECHNOLOGIES USED

Back-End Framework: Java with Spring Boot:

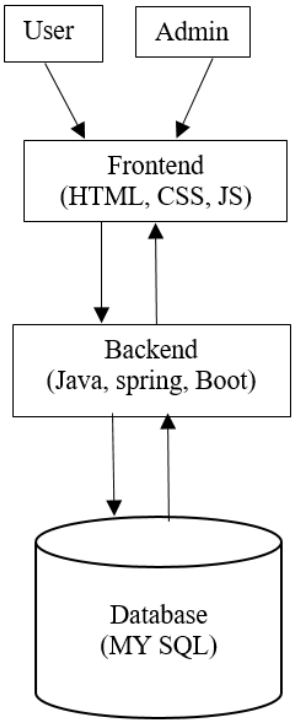
The application logic for the server-side is written in Java, known for being ubiquitous and having solid object-oriented capabilities. The backend is built using the Spring Boot framework to speed up production-grade application development with low configuration. Spring Boot provides a great foundation for development with great security, and it handles a lot of the boilerplate code as well as providing built-in features for transaction context and error handling. The front-end communicates with the backend by using Restful API calls, making it a stateless and saleable interface.

Data Persistence: MySQL with Hibernate/JPA:

Data of the system will be stored in a MySQL relational database which will include patient records, doctor schedules and bed status. The consistency and ability of MySQL to maintain data integrity during transactions or ACID properties are the reasons why it is a good solution to an application that handles sensitive data. Structured hospital datasets are suitable in the relational model. Spring Data JPA and Hibernate will be used to access the database and provide an ORM layer of communication with the relational database. The ORM layer enables the application to interact with the database in Java objects rather than in raw SQL. This adds security and development makes it easier.

Front-end Development: HTML, CSS and JavaScript:

The front-end portion of the system is developed using the standard and commonly used technology to further guarantee compatibility and good user experience. The presentation of the user interface is coded in static HTML5 and CSS3. JavaScript offers client-side logic capabilities to tasks, including validation of forms, dynamic content, and asynchronous data retrieval over an API of a REST implementation. Collectively, this contemporary technique will enable the system to be logged in via various devices and browsers, which is significant when designing an app that is open to society.



Front-end Development: HTML, CSS, and JavaScript:

The front-end of the system is created using a traditional stack of web technologies to support broad compatibility and high-quality user experience. The static content structure and styles for the user interface are completed in HTML5 and CSS3. JavaScript has client-side logic which consists of, but not limited to, form validation, dynamic content updates, and asynchronous calls to the REST API. The combination means that the system can serve a wide range of devices and browsers, which is significant in case of a publicly available application.

7. RESULTS AND DISCUSSION

The Smart Web-Based Hospital Management System (SWHMS) was developed and tried. The most important findings are outlined as follows.

A. Functional Results:

The separation of the endpoints existed between the user and the admin account and the application was functional to log in. The patients were able to enroll themselves in the database and store their data accordingly. During the booking of appointment, it presented the doctors available and confirmed the appointment. The number of beds shown was the right bed in the bed management features. The emergency button used to show the available doctors and beds within a short time.

B. Performance:

The Majority of operations received responses within 2 seconds. Data remained stable with multiple users accessing the system simultaneously. The modular design was easy to test and maintain.

C. Usability:

The interface was straightforward and user-friendly. Users found the emergency button to be very valuable in an urgent context. Administrators were able to easily update and manage the emergency hospital resources.

D. Discussion:

The system eliminated much manual work, updated quickly, and provided secure data storage. The reliability and efficiency of the system outweigh manual record keeping. One limitation was the administrator still needed to update the doctor and bed information. In the future, IT-based automatic updates could be incorporated to "smart" the system further.

8. CONCLUSION

The Smart Web-Based Hospital Management System suggests an easy to use and effective solution to patient registration, doctor appointment, bed management and emergency support. The Smart Web-Based Hospital Management System increases efficiency, reduces manual effort and provides secure data handling for the hospital.

In this version, the updates for doctor and bed availability are still a manual process, however, with changes in a future release, such as potentially adopting IT and implementing mobile application support, the system will become even smarter and more automated.

References

1. R. S. Pressman, Software Engineering: A Practitioner's Approach, 8th ed., McGraw Hill, 2014.
2. H. M. Deitel, P. J. Deitel, Java: How to Program, 11th ed., Pearson Education, 2018.
3. C. J. Date, An Introduction to Database Systems, 8th ed., Addison-Wesley, 2004.
4. Thomas Connolly, Carolyn Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, 6th ed., Pearson, 2015.
5. D. Flanagan, JavaScript: The Definitive Guide, 7th ed., O'Reilly Media, 2020.
6. B. W. Kernighan, D. M. Ritchie, The C Programming Language, 2nd ed., Prentice Hall, 1988.
7. S. Shamkant Navathe, Ramez Elmasri, Fundamentals of Database Systems, 7th ed., Pearson, 2016.
8. Ian Sommerville, Software Engineering, 10th ed., Pearson, 2015.
9. A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts, 7th ed., McGraw Hill, 2019.
10. James Holmes, Java Servlets and JSP, O'Reilly Media, 2018.
11. Anbumani P, Dhanapal R. Enhanced Blockchain-based Key Generation using Butterfly Optimization Algorithm for Efficient Data Sharing in Cloud Computing. Journal of Information Technology Management. 2023 Mar 1;15(Special Issue: Digital Twin Enabled Neural Networks Architecture Management for Sustainable Computing):21-42.