Patient Management System using Next.js

Bhavna Sharma
Department of Computer Science and Engineering
Galgotias University, Greater Noida, Uttar Pradesh, India
mail2bhavnasharma7@gmail.com

Ayushman Singh Rajawat
Department of Computer Science and Engineering
Galgotias University, Greater Noida, Uttar Pradesh, India
rajawatayushman17@gmail.com

Dr. Ajay Shanker Singh
Department of Computer Science and Engineering
Galgotias University, Greater Noida, Uttar Pradesh, India
drajay.cse@gmail.com

***Abstract*— This paper proposes the development of a patient management system (PMS) using modern web technologies such as Next.js, TypeScript, Twilio, and TailwindCSS. The system aims to streamline healthcare by providing an efficient interface to manage patient records, schedule appointments, and enable doctor-patient communication Twilio has been integrated to handle SMS notifications use, and build the interface with TailwindCSS for a more modern design. The program uses TypeScript to ensure type safety and improve development performance. This paper discusses the design, implementation, and benefits of using this tech stack for healthcare applications. The results of this work are important because they provide an efficient, reliable, and user-friendly way to manage patient data, ultimately contributing to better health care and patients' satisfaction.**

Keywords— Patient Management System, Next.js, TypeScript, Twilio, TailwindCSS, Web Application

I. INTRODUCTION

A. General Introduction

Healthcare systems around the world are increasingly turning to digital solutions to improve patient care and streamline business processes. Patient management systems are one such digital platform that provides healthcare professionals with tools to better manage patient information, planning and communication. These programs digitize patient records, schedule appointments, and improve clinical efficiencies through developed web technologies that enable high-performance, flexible and responsive, healthcareenabled PMS services employees are able to deliver better services while reducing operational inefficiencies Patient management systems include Next.js , Modern web technologies such as The combination of TypeScript, Twilio, and TailwindCSS delivers what is essential to improved productivity and user experience These technologies allow for increased productivity, type safety, seamless interaction, and responsive interaction , it's mobile-friendly. The aim of this paper is to discuss such a systematic implementation of this tech stack and its usefulness in a healthcare setting.

B. Problem Statement

Traditional methods of patient management, such as manual records and planning, are often inefficient, error-prone, and inflexible. Many health care professionals still rely on paperbased systems on or on outdated software not accompanied by modern communication tools, resulting in poor patient engagement and increased administrative burden. Appointment reminders and other failures to communicate in a timely manner can lead to missed appointments, which can result in healthcare provider care deterioration and loss of revenue.

C. Objectives of the project

The following are the main goals of a patient management system built with Next.js, TypeScript, Twilio, and TailwindCSS.

1) Enhance patient management: Provide a platform for healthcare professionals to better manage patient records, charts, and medical histories.
2) Integrate SMS-based communication: Use Twilio to send automated SMS notifications for appointment reminders to reduce missed appointments and provide patient autonomy insertion has been improved.
3) Improve user experience: Use TailwindCSS to create responsive, modern and intuitive designs that evolve well for devices including desktop and mobile platforms
4) Ensure scalability and performance: Use Next.js for server-side rendering and static site generation to ensure PMS performance and scalability.
5) Improve code reliability: Use TypeScript for type safety, reduce potential errors and improve system maintainability over time.

D. Current Scope

The current scope of the project focuses on developing a Patient Management System that addresses the core needs of healthcare providers in managing patient data, appointments, and communication. The system will offer:

1. User roles: Different interfaces for patients, doctors, and administrators, each with its own set of functionalities.
2. Appointment management: Patients can book appointments, and doctors can confirm or reschedule them.
3. SMS notifications: Twilio will handle SMS reminders and alerts for patients about their appointments.
4. Medical records: Patients and doctors can securely access and update medical records.
5. Responsive design: The system will be fully responsive, ensuring an optimized user experience across all devices.

Future expansions may include integration with email notifications, enhanced security features, and interoperability with other healthcare systems, depending on the evolving needs of the healthcare sector.

.

II. RELATED WORKS

The healthcare sector has seen the proliferation of various digital platforms aimed at improving patient care. Several webbased PMS solutions have been developed over the years, which provide comprehensive management of patient data. : EPIC excels providing accurate/connected information virtually in real time with which to adjust medical practice. However, hidden costs are associated with EPIC, including expensive vendor support and add-on programs, “technological somnambulism,” increased data entry “after-hours tax,” and training. Nevertheless, EPIC can enhance patient safety, monitoring, tracking, continuity of care, and patient involvement. It also has promise as a medical education tool [1]. OpenMRS is an open-source, robust electronic health record (EHR) platform that is supported by a large global network and used in over forty countries [2]. Hospital information system (HIS) is used as a comprehensive, integrated information system designed to manage the administrative, financial, and clinical aspects of a hospital in urban India [3].

III. DESIGN

The proposed Patient Management System (PMS) follows a microservice architecture where different components such as patient records, appointments, and communication are modularized for maintainability and scalability.

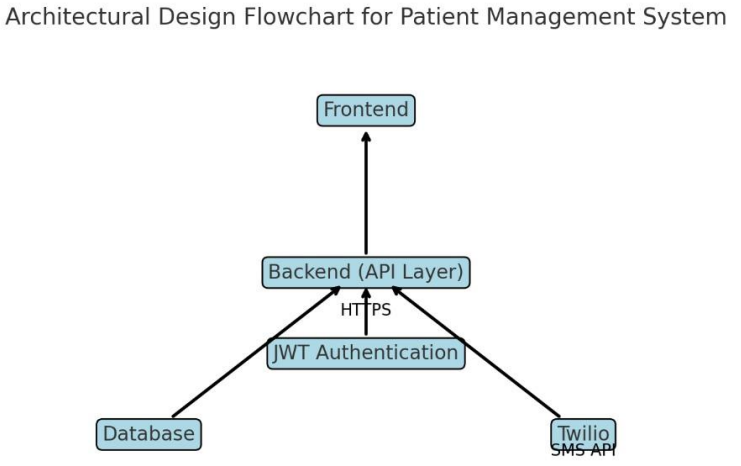
A. Architecture Design

The architecture of the proposed Patient Management System (PMS) follows a modular microservice-based design, ensuring scalability, maintainability, and flexibility. The system is divided into three primary layers: the frontend (client), the backend (API layer), and the communication layer.

- Frontend: The frontend built with Next.js uses serverside rendering (SSR) and static site generation (SSG) throughout TailwindCSS for data-heavy pages like patient records and appointments to make performance and load times dashboards fly effective Integrated devices for responsive, modern UI interface.
- Backend: The backend consists of RESTful APIs that manage patient data, configuration, and user activities. It is built with Node.js and TypeScript to ensure type safety and easy maintenance. The API interfaces with a relational database (e.g., PostgreSQL or MySQL) to store and manage medical records, user profiles, and policies securely.
- Communication Layer: Twilio handles SMS notifications and reminders. This layer is responsible for sending appointment confirmations, reminders, and updates, which are triggered by changes in appointment status or upcoming schedule.

Both front-end and back-end communications are secured using HTTPS using JSON Web Tokens (JWT) for user authentication and authorization. This ensures that sensitive patient information is protected in compliance with HIPAA and other healthcare regulations.

Fig 1. Basic architecture of system



B. System Design

The system design focuses on the core functionalities of the PMS and the interaction between different components.

1. User Roles: The system supports three types of users—patients, doctors, and administrators—with role-based access control.
    - Patients can book, view, and cancel appointments, view medical history, and receive SMS notifications for upcoming appointments.
    - Doctors can manage their availability, confirm, or reschedule appointments, and review patient medical records.
    - Administrators manage system settings, user roles, and ensure smooth functioning of all communication channels.
2. Database Design: The database includes tables for patients, doctors, appointments, and messages.
    - The patient table stores personal and medical information.
    - The appointment table tracks appointment details such as patient ID, doctor ID, date, and status (e.g., pending, confirmed).
    - The message table logs all SMS notifications sent through Twilio.
3. Appointment Management: Dynamic appointment scheduling is enabled through Next.js dynamic routing and state management. Patients can view available slots based on doctor schedules and book appointments. Doctors can approve or modify these appointments in real time.
4. SMS Notification System: Integrated with Twilio, the PMS automatically sends SMS alerts when patients book or modify appointments. Reminders are also sent 24 hours before the scheduled time to reduce noshows. The system records the status of these notifications for monitoring.

The figure displays the relationship between the entire system. (Fig 2)

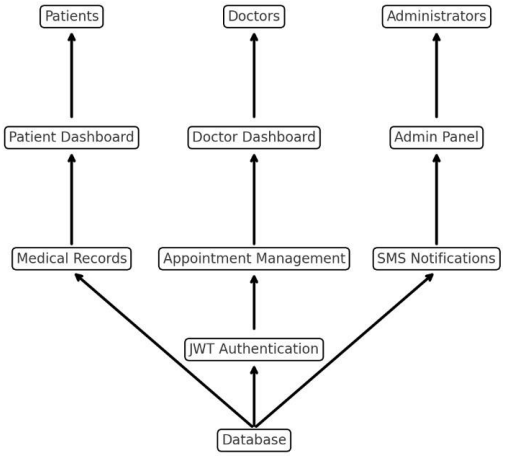


Fig 2. System Design for the platform

C. Data Flow

The data flow in the Patient Management System outlines the processing and transferring of data between various components. Here is an overview of the data flow:

1. User Registration/Login:
    a. Input: User enters credentials (username, password).
    b. Process: The client sends a request to the authentication API.
    c. Output: The server verifies credentials, responds with a success/failure message and a JWT token (for authenticated sessions).
2. Patient Data Management:
    a. Input: Healthcare provider inputs new patient data or updates existing records.
    b. Process: The client sends a request to the patient management API to create or update records in the database.
    c. Output: The server processes the request, updates the database, and returns a confirmation message.
3. Appointment Scheduling:
    a. Input: Patient selects an available appointment slot.
    b. Process: The client sends an appointment request to the scheduling API.
    c. Output: The server verifies the slot, reserves it in the database, and responds with appointment details.
4. Notifications:
    a. Input: Scheduled appointment triggers notification setup.
    b. Process: The server sends an SMS notification via Twilio.
    c. Output: Patients receive reminders about their upcoming appointments.
5. Feedback and Reporting:
    a. Input: Patients and providers can submit feedback.
    b. Process: The client sends feedback data to the feedback API.
    c. Output: The server processes and stores feedback for future analysis and system improvements.

IV. DEVELOPMENT TECHNOLOGIES

A. Next.js

Next.js is a React-based framework that enables server-side rendering (SSR) and static site generation (SSG), improving performance and SEO optimization [4]. In the context of PMS, Next.js helps pre-render pages such as the appointment list and patient dashboard, ensuring faster load times.

B. TypeScript

TypeScript is a statically typed superset of JavaScript that enhances code quality by catching errors at compile time. Using TypeScript in the PMS ensures type safety, reduces bugs, and improves code maintainability. TypeScript is an extension of JavaScript intended for easier development of large-scale
JavaScript applications.[5]

C. Twilio

Twilio provides APIs for SMS, voice, and video communications. Integrating Twilio into the PMS allows us to send SMS reminders for upcoming appointments and other important notifications, thereby improving patient engagement and reducing no-shows.

D. TailwindCSS

TailwindCSS is a utility-first CSS framework that allows for rapid UI development with a responsive design. Using this we created an intuitive, modern, and mobile-friendly interface, essential for both patients and healthcare professionals

V. IMPLEMENTATION

A. User Interface Design

The user interface (UI) of the PMS was developed using TailwindCSS, ensuring a clean, responsive, and mobile-friendly experience. The interface is designed to accommodate different user roles—patients, physicians, and administrators—by assigning them specific functionality:

1) Patient Dashboard: Displays upcoming appointments, medical records, and SMS notification history.
2) Physician Dashboard: Allows physicians to manage their appointments, view patient history, and see appointments.
3) Admin Panel: Enables administrators to view system users, appointments, and communication logs.

B. Appointment Management

Built with Next.js' dynamic routing and state management techniques, the appointment management system allows patients to register, and physicians to confirm or reschedule appointments Integration with Twilio ensures patient hand will include an SMS notification of a confirmed appointment.

C. SMS Integration with Twilio

The system integrates Twilio’s SMS functionality to send appointment reminders, confirmations, and follow-up messages. Twilio’s API was used to schedule and trigger

messages based on changes in appointments, reducing the chances of missed appointments.

D. TypeScript Integration

TypeScript was used throughout the application for type safety and development best practices. By using TypeScript, we were able to reduce runtime errors and achieve better maintained code.

VI. INCORPORATING ARTIFICIAL INTELLIGENCE FOR PERSONALIZED PATIENT SUPPORT

Integrating artificial intelligence (AI) into patient management systems (PMS) opens possibilities for creating chatbots that improve patient engagement and support, these AI-powered chatbots can be used to input patient data on to obtain personalized support for various aspects of health care delivery. Key Features and Benefits:

a. Personalized Treatment Guidance: By using a patient's medical history and appointment information, the chatbot can provide personalized advice on treatment regimens, including medications or treatment times including a reminder.

b. Optimized patient appointments: Patients can schedule, change, cancel or interact with the chatbot in real time, streamlining administrative tasks.

c. Round-the-Clock Support: Chatbots can act as virtual assistants, answering patients' questions and providing healthcare advice based on their medical records, improving patient satisfaction, and reducing reliance on human staff.

VI. CONCLUSION AND FUTURE WORK

The proposed system demonstrated efficiency in managing appointments and enhancing patient-doctor communication. Owing to the problems faced by healthcare institutions in accessing and maintaining large amounts of data that they have to deal with [7], Future work will focus on enhancing security measures, integrating more communication channels (e.g., email), and improving growth in larger healthcare facilities.

VIII. REFERENCES

[1] Johnson III RJ. A Comprehensive Review of an Electronic Health Record System Soon to Assume Market Ascendancy: EPIC®. J Healthc Commun. 2016,
1:4.

[2] Mohammed-Rajput NA, Smith DC, Mamlin B, Biondich P, Doebbeling BN; Open MRS Collaborative Investigators. OpenMRS, a global medical records system collaborative: factors influencing successful implementation. AMIA Annu Symp Proc. 2011;2011:960-8. Epub 2011 Oct 22. PMID: 22195155; PMCID: PMC3243141.

[3] Mohapatra, S. (2015). Using integrated information system for patient benefits: a case study in India. *International Journal of Healthcare Management*, *8*(4), 262–271. https://doi.org/10.1179/2047971915Y.0000000007

[4] Lazuardy, Mochammad Fariz Syah, and Dyah
Anggraini. "Modern front end web architectures with react. js and next. js." *Research Journal of Advanced Engineering and Science* 7.1 (2022): 132-141.

[5] Bierman, G., Abadi, M., Torgersen, M. (2014). Understanding TypeScript. In: Jones, R. (eds) ECOOP 2014 – Object-Oriented Programming. ECOOP 2014. Lecture Notes in Computer Science, vol 8586. Springer,
Berlin, Heidelberg. https://doi.org/10.1007/978-3-662-44202-9_11

[6] https://scholar.google.com/

[7] A. Dwivedi, R. K. Bali, A. E. James and R. N. G. Naguib, "Workflow management systems: the healthcare technology of the future?," 2001 Conference
Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, Istanbul, Turkey, 2001, pp. 3887-3890 vol.4, doi: 10.1109/IEMBS.2001.1019689.