Queue Management System for Clinics – Project Report

Full Stack Development (CA II)

MAHATMA EDUCATION SOCIETY'S

PILLAI COLLEGE OF ARTS, COMMERCE &

SCIENCE

(Autonomous)

NEW PANVEL

PROJECT REPORT ON

"**Queue Management System for Clinics**"

IN PARTIAL FULFILLMENT OF

BACHELOR OF <u>COMPUTER APPLICATION</u>

SEMESTER IV: 2025-26

PROJECT GUIDE

Dr. Nikita Bahaley

SUBMITTED BY: Harshal Pabale
ROLL NO: 5851

**3.	Table of Contents No. Section**

Clinic Queue Management System

1. Introduction

In many small clinics and hospitals, patient queue handling is still managed manually using registers or verbal calls. This traditional approach often leads to confusion, longer waiting times, duplicate entries, and inefficient patient management. As the number of patients increases, manual systems become unreliable and difficult to manage.

To overcome these issues, the **Clinic Queue Management System** has been developed as a web-based application that digitally manages patient tokens and queue flow. The system automates token generation, tracks patient status, and displays queue information in a clear and organized manner.

This project is built using **React.js**, a modern JavaScript library for building user interfaces. The application focuses on simplicity, efficiency, and usability. It works completely in **offline mode using browser localStorage**, making it suitable for small clinics that may not have constant internet connectivity.

---

2. Objectives of the Project

The main objectives of this project are:

- To provide a digital and automated solution for managing patient queues
- To reduce manual effort and errors in token handling
- To display real-time queue status in an organized manner
- To improve patient experience by reducing confusion and waiting uncertainty
- To demonstrate the practical use of React concepts such as components, hooks, and routing

---

3. Scope of the Project

The scope of the Clinic Queue Management System includes:

- Registration of patient tokens through a web interface
- Display of current token and waiting queue
- Management of token status (Waiting, In Progress, Completed)
- Doctor/Admin dashboard to control queue flow
- Offline data persistence using browser localStorage

The project does not include online payment systems, multi-branch clinics, or cloud databases. These features can be added in future versions of the system.

---

4. Technology Stack

Frontend Technologies

- **React.js** – Used for building the component-based user interface
- **JavaScript (ES6+)** – Used for application logic and state handling
- **HTML5 & CSS3** – Used for page structure and styling
- **React Router** – Used for navigation between different pages

Storage

- **Browser localStorage** – Used to store patient tokens and queue data in offline mode

Development Tools

- **Node.js & npm** – Used for managing dependencies and running the project
- **VS Code / Cursor IDE** – Used as the development environment
- **Web Browser (Chrome)** – Used for testing and execution

---

5. System Architecture

The system follows a **client-side architecture** where all processing happens in the browser:

- The user interacts with the React-based user interface
- Application state is managed using React hooks such as `useState` and `useEffect`
- Queue data is stored and retrieved from browser localStorage
- UI components automatically re-render when the state changes

Since there is no backend server, the application is lightweight, fast, and easy to deploy.

---

6. Functional Requirements

The system should be able to:

- Register a new patient and generate a unique token number
- Display the current token being served
- Show the list of waiting and completed tokens
- Move token status from waiting to in-progress and completed
- Preserve queue data even after page refresh

---

7. Non-Functional Requirements

- The system should be easy to use and understand
- The interface should be clean and visually clear
- The application should respond quickly to user actions
- The system should work without an internet connection

---

8. Development / Implementation

This section explains how the Clinic Queue Management System was developed and how its main modules function. The project is implemented using React functional components and hooks. The application logic ensures that queue flow is maintained correctly and that data is persisted using localStorage.

The implementation is divided into three main modules.

8.1 Home Page (Queue Display)

The Home Page acts as the public-facing interface of the application. It displays important queue-related information such as the current token being served, total number of patients waiting, and basic clinic status. This page helps patients track queue progress without directly interacting with staff.

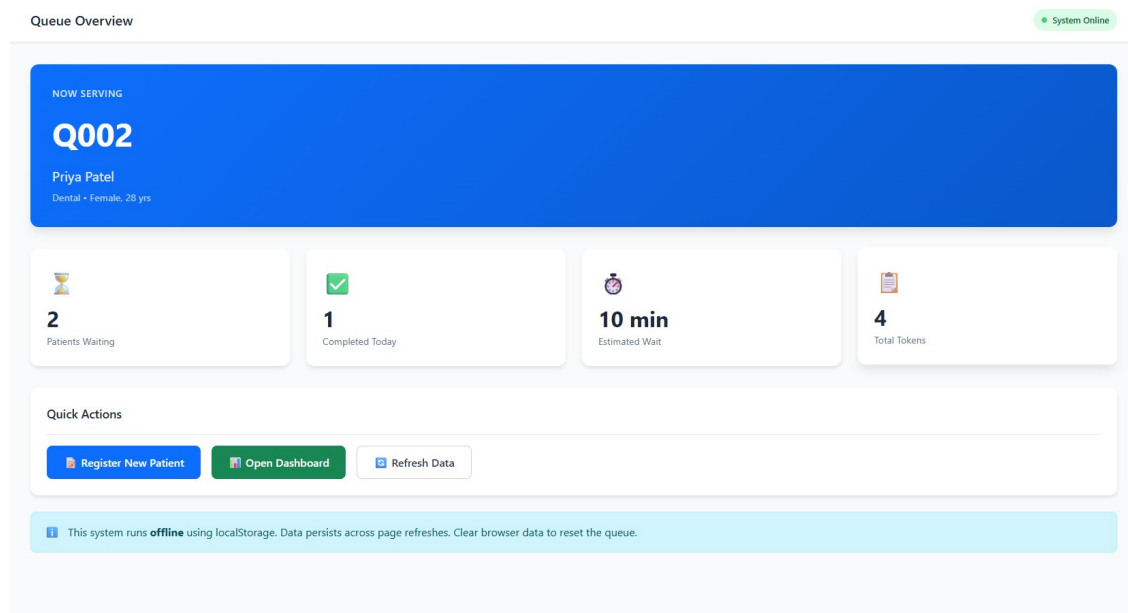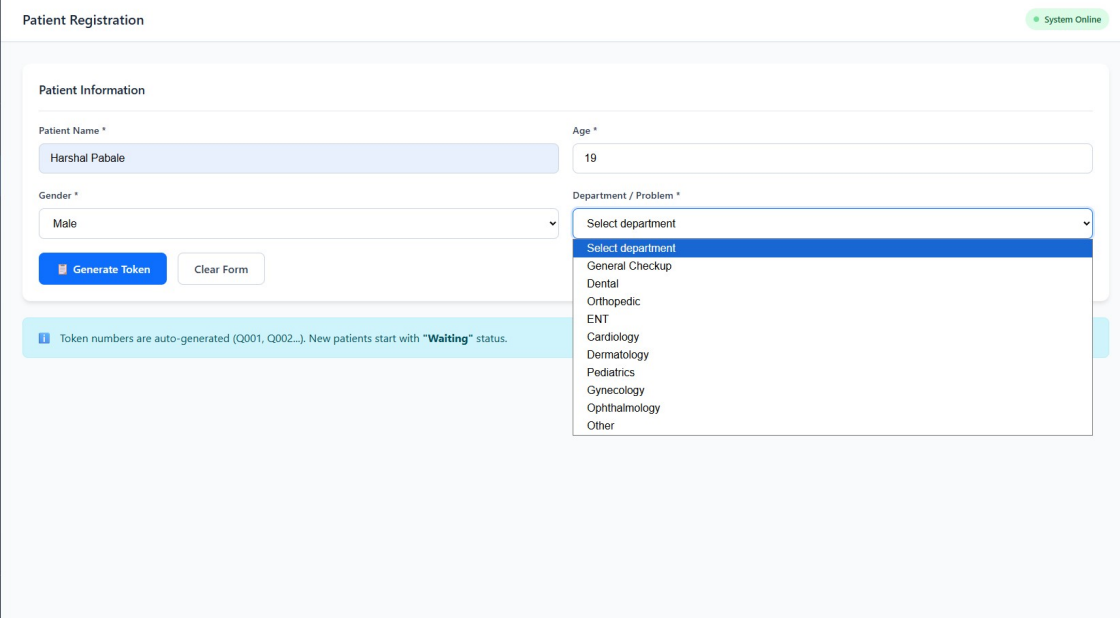The data displayed on the Home Page is dynamically updated based on application state and localStorage data.

Fig 8.1: Home Page displaying current token and queue status

8.2 Register Token Module

The Register Token page allows clinic staff to register new patients into the queue. The form collects basic patient details and automatically generates a unique token number. Once submitted, the token is added to the queue with a default status of "Waiting".

This module uses controlled components and state management to handle form inputs and submissions efficiently.

Fig 8.2: Register Token page showing patient registration form

8.3 Dashboard Module

The Dashboard is designed for doctors or clinic administrators. It provides a complete view of all tokens categorized by their status: Waiting, In Progress, and Completed. The dashboard includes buttons to call the next patient, mark the current token as completed, and clear the queue if required.

This module controls the main queue logic and ensures that only one token can be in progress at a time.
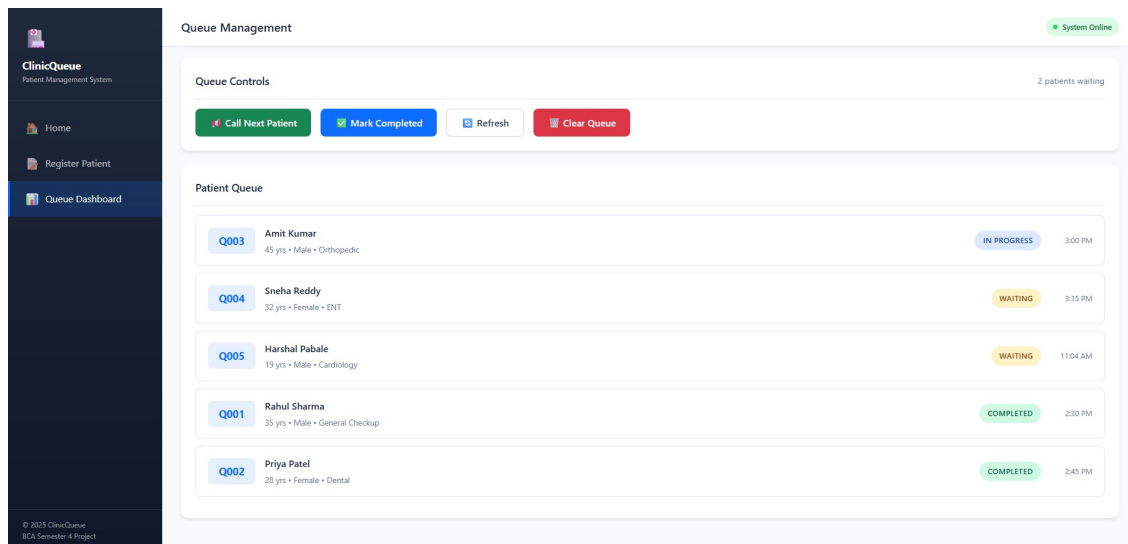
Fig 8.3: Dashboard page showing queue management controls

---

9. Testing

The system was tested under various conditions to ensure reliability:

- Multiple tokens were registered sequentially
- The browser was refreshed to verify localStorage persistence
- Queue flow logic was tested using call-next and complete actions
- UI responsiveness was checked across different screen sizes

All core functionalities performed as expected.

---

10. Limitations

- No backend database or server integration
- Supports only a single clinic setup
- No user authentication or role management

---

11. Future Enhancements

- Integration with backend using Node.js and MongoDB
- Real-time online queue updates
- SMS or notification alerts for patients
- Support for multiple doctors and clinics
- Role-based access control

12. Conclusion

The Clinic Queue Management System demonstrates the effective use of React.js for solving real-world problems. By digitizing the queue management process, the system improves efficiency, reduces errors, and enhances patient experience. The project successfully meets its objectives and provides a strong foundation for future expansion.

---

13. References

- React.js Official Documentation
- MDN Web Docs
- Node.js Official Website

Mahatma Education Society's

PILLAI COLLEGE OF ARTS, COMMERCE & SCIENCE

(Autonomous), New Panvel

Re-accredited "A" Grade by NAAC (3rd Cycle)

Project Completion Certificate

THIS IS TO CERTIFY THAT

Harshal Pabale

of **SYBCA** has completed the project titled

" Queue Management System for Clinics"

Under our guidance and supervision during the academic year
2025-26 in the department of Information Technology

Course Coordinator   Head of Department