# Smart Hospital Patient Queue Management System

## Introduction

Hospitals often face difficulties in managing patient queues, especially during emergency situations where critical patients need immediate attention. Traditional first-come-first-served systems fail to prioritize patients based on the severity of their condition. This can lead to delays in treatment and increased health risks. The Smart Hospital Patient Queue Management System is designed to solve this issue by using a priority-based approach. The system ensures that patients with higher medical urgency are treated first. It dynamically updates patient priority based on severity and waiting time. This improves efficiency, reduces patient waiting time, and enhances overall healthcare service quality.

## Selected Data Structure and Justification

The Heap (Priority Queue) data structure is used in this application. A heap always allows quick access to the element with the highest priority, which is essential in emergency healthcare scenarios. In this system, patients are assigned priority values based on severity level, waiting time, and age. The heap ensures that the most critical patient is always placed at the top and treated first. Insert and delete operations in a heap take O(log n) time, making it efficient for real-time hospital environments. Therefore, a heap is the most suitable data structure for managing dynamic and priority-based patient queues.

## Input Requirements

1. Patient ID
2. Patient Name
3. Severity Level (1–5)
4. Arrival Time
5. Patient Age
6. Emergency Type
7. Assigned Department
8. Doctor Availability Status

## Output Requirements

1. Current prioritized patient queue
2. Next patient to be treated
3. Estimated waiting time
4. Emergency alerts for critical patients
5. Doctor assignment details
6. Patient treatment order
7. Queue update notifications

## Process Requirements

1. Accept patient details at reception
2. Calculate patient priority value
3. Insert patient into the heap
4. Reorder heap based on priority
5. Assign highest-priority patient to doctor
6. Update priority based on waiting time
7. Remove treated patient from heap
8. Display real-time queue updates