|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Software Requirement Specifications**  **e-Medicare**  **Version: 1.0**   |  |  | | --- | --- | | Supervisor | Romasha Khurshid | | Project Team | 22K-4626 Asfandyar Khanzada  22K-4231 Munnazzar Shahzad | | Submission Date | 9th May 2025 | |

[Instructions]

* No section of template should be deleted. You can write ‘Not applicable’ if a section is not applicable to your project. But all sections must exist in the final document.
* All comments/examples mentioned in square brackets ([]) are in the template for explanation purposes and must be replaced / removed in final document.
* This’ Instruction’ section should also be removed in final document.
* MS-Word Reviewing feature must be used to get the document reviewed by supervisors or co-supervisors.

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Name of Person | Date | Description of change |
| 0.1 | Munnazzar Shahzad | 05-May-2025 | Document Created |
| 0.2 | Asfandyar Khanzada | 06-May-2025 | Added System Architecture |
| 0.3 | |  | | --- | |  |  |  | | --- | | Munnazzar Shahzad | | 07-May-2025 | |  | | --- | |  |  |  | | --- | | Added Database Design & Diagrams | |
| 1.0 | |  | | --- | |  |  |  | | --- | | Munnazzar & Asfandyar | | 09-May-2025 | |  | | --- | |  |  |  | | --- | | Finalized for submission | |

Distribution List

|  |  |
| --- | --- |
| **Name** | **Role** |
| Ms. Romasha Khurshid | Supervisor |
|  |  |

Document Sign-Off

[Following table will contain sign-off details of document. Once the document is prepared and revised, this should be signed-off by the sign-off authority.

Any subsequent changes in the document after the first sign-off should again get a formal sign-off by the authorities.]

|  |  |  |
| --- | --- | --- |
| **Version** | **Sign-off Authority** | **Sign-off Date** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Table of Contents**

[1. Introduction 7](#_Toc178130213)

[1.1. Purpose of Document 7](#_Toc178130214)

[1.2. Intended Audience 7](#_Toc178130215)  
1.3 Abbreviations ………………………………………………………………………………………...7

[1.4. Document Convention 7](#_Toc178130216)

[2. Overall System Description 8](#_Toc178130217)

[2.1. Project Background 8](#_Toc178130218)

[2.2. Project Scope 8](#_Toc178130219)

[2.3. Not In Scope 8](#_Toc178130220)

[2.4. Project Objectives 8](#_Toc178130221)

[2.5. Stakeholders 8](#_Toc178130222)

[2.6. Operating Environment 8](#_Toc178130223)

[2.7. System Constraints 8](#_Toc178130224)

[2.8. Assumptions & Dependencies 8](#_Toc178130225)

[3. External Interface Requirements 9](#_Toc178130226)

[3.1. Hardware Interfaces 9](#_Toc178130227)

[3.2. Software Interfaces 9](#_Toc178130228)

[3.3. Communications Interfaces 9](#_Toc178130229)

[4. Functional Requirements 10](#_Toc178130230)

[4.1. Functional Hierarchy 10](#_Toc178130231)

[4.2. Use Cases 10](#_Toc178130232)

[4.2.1. [Title of use case] 10](#_Toc178130233)

[5. Non-functional Requirements 11](#_Toc178130234)

[5.1. Performance Requirements 11](#_Toc178130235)

[5.2. Safety Requirements 11](#_Toc178130236)

[5.3. Security Requirements 11](#_Toc178130237)

[5.4. User Documentation 11](#_Toc178130238)

[6. References 12](#_Toc178130239)

[7. Appendices 13](#_Toc178130240)

1. Introduction

* 1. Purpose of Document

The purpose of this document is to outline the Software Design Specification (SDS) for the ICU Hospital Management System. This system is aimed at improving ICU operations by automating and streamlining patient care tasks, enhancing staff communication.

* 1. Intended Audience

This document is intended for the development team, project supervisor and evaluators, and any stakeholder involved in the design, development, and deployment of the system

**1.3 Abbreviations**

| **Term** | **Description** |
| --- | --- |
| HMS | Hospital Management System |
| DD | Design Specification |
| ICU | Intensive Care Unit |
| CRUD | Create, Read, Update, Delete (DB operations) |
| SPA | Single Page Application (React-based frontend) |
| DFD | Data Flow Diagram |
|  |  |

* 1. Document Convention
* Font: Calibri
* Font Size: 11pt
* Headings: Bold
* Sections follow IEEE SRS guidelines

1. Overall System Description
   1. Project Background

Most modern ICUs face challenges in coordinating patient care, resource utilization, and communication among healthcare professionals. This project proposes an autoamted system that helps to perform key ICU operations more efficiently such as doctor assignment, room allocation medication administration.

* 1. Project Scope

The ICU Hospital Management System focuses on streamlining and automating ICU operations for enhancing efficiency, including doctor assignment, patient record management, room allocation, medication prescription, and dosage administration. It is a web-based Node application with a React frontend, Express.js backend, and PostgreSQL database.

* 1. Not In Scope

 Integration with non-ICU departments and IOT devices

 Advanced AI analytics and care suggestion

 Dedicated mobile application

* 1. Project Objectives

The primary objective of the e-Medicare System is to help effectively perform ICU operations by automating critical tasks such as patient admission, doctor assignment, room allocation, medication prescription, and dosage logging. The end result of the project will be a secure, scalabale and user-friendly web-based platform that enhances collaboration, accountability, and the overall standard of care in ICU settings.

* 1. Stakeholders

[This section will describe stakeholders of the system. This will include different business user classes that are expected to interact with system and similarly the technical people who are going to be involved in software development/management]

The stakeholders of the ICU Hospital Management System include:

* **Doctors**: Access patient records, receive automated assignments, and prescribe medications.
* **Nurses**: Receive medication schedules, log dosage administration, and interact with the system for daily patient management.
* **Hospital Administrators (Admin)**: Monitor ICU operations, manage staff assignments, and add new patients/doctors/nurses. Also to monitor medication and its dosages
* **Patients (Indirectly)**: Benefit from efficient and timely treatment facilitated by the system.
* **Development Team**: Responsible for designing, implementing, testing, and deploying the system.
  1. Operating Environment

The e-Medicare system will be web application and is expected to operate in the following environment:

* **Hardware Platform**: Standard desktop computers, tablets or mobile devices used by hospital staff within ICU wards.
* **Operating System**: Any platform that supports modern web browsers (e.g., Chrome, Firefox, Edge).
* **Network Environment**: Requires stable internet connectivity within the hospital's internal network or secure VPN access for authorized remote use.
* **Software Components**:
  + Frontend: React.js:
  + Backend: Node.js with Express:
  + Database: PostgreSQL:
* **Web Server**: Deployed via cloud infrastructure or on local server depending on hospital IT policies.
  1. System Constraints

[Describe the constraints imposed on the system by the external environment. External environment may be caused by the stakeholders, business conditions, technical issues, academic requirements etc and may include the following:

o The ICU Hospital Management System is subject to the following constraints:

* **Software Constraints**:
  + Must be developed using React, Node.js, Express, and PostgreSQL.
  + Compatibility of third-party libraries and frameworks with the hospital’s IT security policies.
* **Hardware Constraints**:
  + The system must run efficiently on standard desktop computers, tablets or mobile devices used by hospital staff within ICU wards.
* **Legal Constraints**:
  + The system must comply with healthcare data protection regulations (e.g., HIPAA or relevant local laws).
* **Environmental Constraints**:
  + Designed for use in ICU environments, the system must support high readability and low interaction complexity due to the critical nature of operations.
* **User Constraints**:
  + The system must provide an intuitive UI with minimal training requirements for medical staff.
  1. Assumptions & Dependencies
* It is assumed that all users (doctors, nurses, administrators) will have access to modern web browsers and reliable internet connectivity.
* Node.js and PostgreSQL will be supported in the deployment environment, either on-premises or via a cloud-based infrastructure.
* An up-to-date list of the staff schedules, patient records and inventory would be provided for system initialization.
* The system depends on timely and accurate input from staff, especially for dosage logging and inventory updates.
* Integration with biometric or IoT-based monitoring systems is currently out of scope but may be considered for future releases.

1. External Interface Requirements
   1. User Interfaces

The software will provide a user-friendly interface, including clean layouts, intuitive navigation, standard buttons and error. Detailed UI designs will follow standard guidelines and be documented separately

* 1. Hardware Interfaces

The system will interact with standard computer systems, mobile devices, tables and any devices allowed to be used in an ICU environment.

* 1. Software Interfaces

The system integrates with PostgreSQL for data management, Node for backend operations, and APIs for third-party tools. Inputs and outputs will follow structured formats to ensure seamless integration.

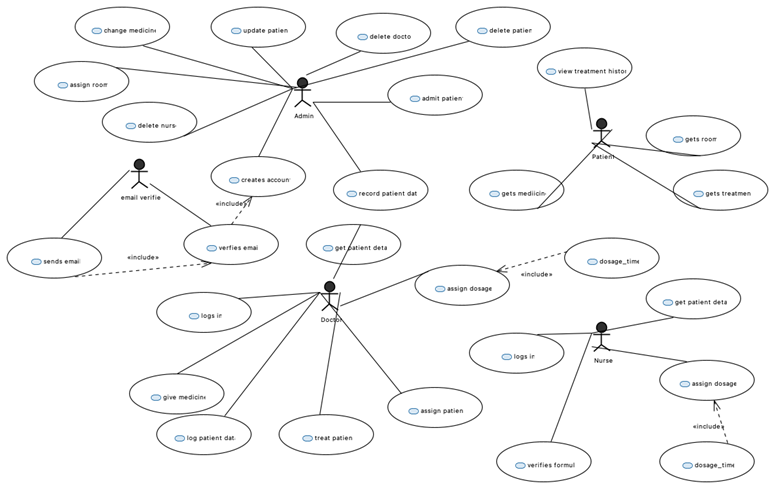
* 1. Communications Interfaces

.The system will use HTTPS for secure communication, email notifications, and potential integration with external communication services like SMS gateways or messaging APIs.

1. Functional Requirements
   1. Functional Hierarchy

The e-Medicare System is composed of several core functional modules, each responsible for managing a critical aspect of ICU operations. Below is the hierarchical breakdown of the system’s main functionalities and their sub-components:

* **Patient Management**
  + Admit new patient
  + Delete patient record
  + View dosage history
  + Assign room to patient
  + Record patient data
  + Retrieve patient data
  + Create staff(doctor/nurse) accounts
  + Assign doctors to patients
  + Delete doctors/nurses
* **Medication and Dosage Management**
  + Prescribe medication
  + Assign dosage
  + Define dosage timing
  + Verify dosage formula
  + Log medication administration
  + Change medication
* **Authentication and Authorization**
  + Login for all user roles (Admin, Doctor, Nurse)
  + Role-based access control
  + Dashboard for each role
* **Notification System**
  + Notify nurses about dosage timings
  + Alert on missing or delayed medication
  1. Use Cases



1. Non-functional Requirements
   1. Performance Requirements

The System must handle up to 1,000 concurrent users, with database queries taking less than 2 seconds.

* 1. Safety Requirements

Ensure Patient data integrity by restricting unauthorized access. Comply with healthcare safety standards to avoid mishandling critical patient data.

* 1. Security Requirements

Implement role-based access, encrypted communication, and secure login with multi-factor authentication. Follow data privacy regulations like GDPR and HIPAA.

* 1. Business Rules

Only authorized users can access specific functions. For example, admins can manage staff schedules, while doctors can only access their patient details.

* 1. Software Quality Attributes

The system must be reliable, maintainable, and scalable. Focus on usability, correctness, and interoperability with external systems.

1. References

* + **[@google/generative-ai](https://www.npmjs.com/package/@google/generative-ai" \t "_new)**Library for accessing Google's Generative AI APIs.

[**Documentation**](https://github.com/google/generative-ai-js)

* + [**bcrypt**](https://www.npmjs.com/package/bcrypt)Library for hashing passwords.

[**Documentation**](https://github.com/kelektiv/node.bcrypt.js)

* + [**cors**](https://www.npmjs.com/package/cors)Middleware for enabling Cross-Origin Resource Sharing in Express apps.
  + [**dotenv**](https://www.npmjs.com/package/dotenv)Loads environment variables from a .env file into process.env.

[**Documentation**](https://github.com/motdotla/dotenv)

* + [**express**](https://www.npmjs.com/package/express)Fast, unopinionated, minimalist web framework for Node.js.

[**Documentation**](https://expressjs.com/)

* + [**jsonwebtoken**](https://www.npmjs.com/package/jsonwebtoken)Implementation of JSON Web Tokens for authentication and authorization.

[**Documentation**](https://github.com/auth0/node-jsonwebtoken)

* + [**node-schedule**](https://www.npmjs.com/package/node-schedule)A cron-like and not-cron-like job scheduler for Node.

[**Documentation**](https://github.com/node-schedule/node-schedule)

* + [**nodemailer**](https://www.npmjs.com/package/nodemailer)Module to send emails from Node.js.
  + [**nodemon**](https://www.npmjs.com/package/nodemon)Utility that monitors for changes in files and restarts your app automatically.

[**Documentation**](https://nodemon.io/)

* + [**pg**](https://www.npmjs.com/package/pg)PostgreSQL client for Node.js.

[**Documentation**](https://node-postgres.com/)

* + [**chai**](https://www.npmjs.com/package/chai)BDD / TDD assertion library for Node and the browser.

[**Documentation**](https://www.chaijs.com/)

* + [**mocha**](https://www.npmjs.com/package/mocha)JavaScript test framework for Node.js programs.

[**Documentation**](https://mochajs.org/)

* + [**supertest**](https://www.npmjs.com/package/supertest)Super-agent driven library for testing HTTP servers.

[**Documentation**](https://github.com/visionmedia/supertest)