1. **Introduction:**
2. **Overall Description**

**2.1 Product Perspective**

The hospital management software is designed to operate within the context of a larger healthcare system. It will integrate with the central database maintained by the health ministry, which contains the patient's medical records. Our software will have a local database that stores relevant information for efficient and quick access. The local database will synchronize with the central database periodically to ensure data consistency.

**2.2 User Classes**

The software is designed to cater to multiple user classes within the hospital setting. These user classes include:

* **Administrators:** Manage user accounts, handle billing and invoicing, and oversee inventory management.
* **Doctors**: Access patient records, schedule appointments, prescribe medications, and order tests.
* **Patients**: Access their medical records, book appointments, receive test results.
* **Laboratory Technologists**: Enter and access test results, manage laboratory data.
* **Staff:** View work schedule, inventory management
* **Nurse:** Access Patient Information, Update Patient Condition, View Work Schedule
* **Pharmacy:** Store Medicines, Accept Prescription from Doctor, Sell Medicine

**2.3 Operating Environment**

The hospital management software will operate in a networked environment within the hospital premises. It will run on a server infrastructure with appropriate hardware specifications and operating systems. The software will be accessible through web browsers, and it will be compatible with common operating systems such as Windows, macOS, and Linux. Mobile access through dedicated applications will also be provided.

**2.4 Constraints**

* The software requires a stable and reliable network infrastructure within the hospital premises to ensure uninterrupted access and data synchronization.
* The local database should have sufficient storage capacity to handle the anticipated volume of patient records and related data.
* The software must adhere to relevant privacy and security regulations to protect patient data, including the utilization of the security code mentioned earlier.

**2.5 Assumptions:**

* + It is assumed that the health ministry will provide timely and accurate updates to the central database for patient medical records.
  + The availability and delivery of security codes to patients via phone messages will be implemented effectively to maintain data security.
  + Users of the software will have a basic level of computer literacy and training to operate the system efficiently.

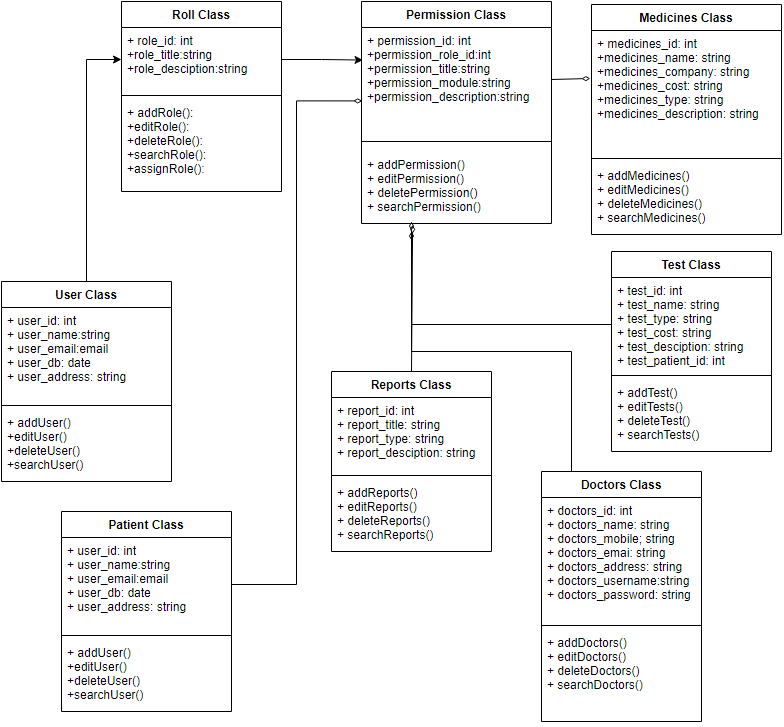
**2.6 Dependencies:**

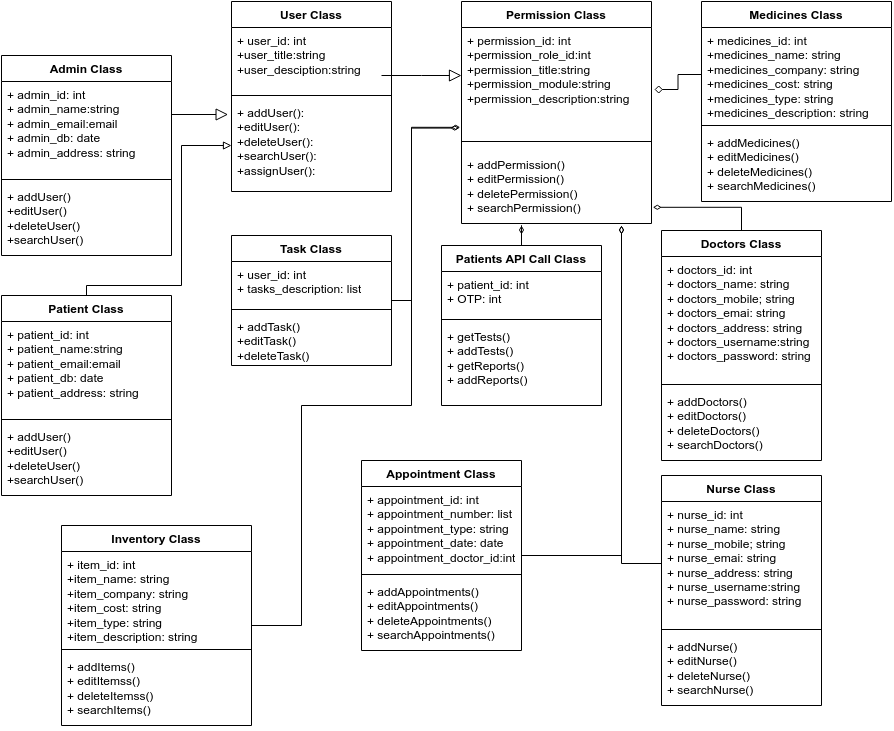
* + The hospital management software relies on the availability and functionality of the central database maintained by the health ministry.
  + The software's security code generation and validation process depend on the successful integration with the messaging system used to deliver codes to patients.
  + The successful implementation of the software relies on the cooperation and support of hospital staff and administration for user training and data input.

1. **Functional Requirements**
2. **Data Requirements**

* 1. **Medical Records**: The system should maintain electronic health records (EHR) containing details of a patient's diagnoses, treatments, medications, laboratory results, radiology reports, and other medical information.
  2. **Personal Information**: The system should capture and store patient, doctor, administrators’ data such as name, age, gender, contact information, medical history, and any relevant personal details.
  3. **Appointments and Scheduling**: The system should manage appointments and scheduling for patients, doctors, nurses, and other staff members. This includes recording appointment details, availability of healthcare professionals, and facilitating appointment booking and rescheduling.
  4. **Billing and Financial Information**: The system should handle billing and financial aspects of the hospital, including patient billing information, insurance details, payment records, and invoicing.
  5. **Inventory Management**: If the hospital manages its inventory internally, the system should track and manage inventory items such as medical supplies, equipment, medications, and their availability.
  6. **Staff and User Management**: The system should maintain a database of staff members, including doctors, nurses, administrators, and their relevant details like names, roles, contact information, and access privileges.
  7. **Reporting and Analytics**: The system should provide reporting and analytics capabilities, allowing administrators to generate various reports, such as patient statistics, financial summaries, operational metrics, and performance indicators.
  8. **Security and Privacy**: The document should specify the security and privacy requirements for protecting patient data.
  9. **Data Backup and Recovery**: The system should have provisions for regular data backups and a plan for data recovery in case of system failures or disasters.

**4.10 Patient Class Diagram:**

****

**4.11 Hospital Class Diagram**:

1. **External Requirements**
   1. **Laboratory Information Systems (LIS):** System needs to interface with a laboratory information system, it specifies the requirements for exchanging lab orders, test results, and other relevant data between the two systems.
   2. **Pharmacy Systems:** System outline the requirements for managing medication orders, dispensing information, medication inventory updates, and drug-drug interaction checks.
   3. **Electronic Health Record (EHR) Systems:** System specifies the requirements for interoperability, including the ability to import/export patient demographics, medical history, and other relevant data.
   4. **Billing and Insurance Systems:** System defines the requirements for submitting claims, receiving payment information, managing insurance eligibility verification, and generating accurate financial reports.
2. **Non-Functional Requirements:**

**6.1 Availability Requirements**

**6.1.1** - System must be available 99% of the time outside of scheduled maintenance.

**6.2 Compatibility Requirements**

**6.2.1** System must be compatible with different operating systems, such as Windows, macOS, Linux, and mobile operating systems like iOS and Android.

**6.2.2** System must be compatible with device types, including desktop computers, laptops, tablets, and smartphones.

**6.3 Reliability Requirements**

**6.3.1** System must install any available software updates nightly to remain up to date.

**6.3.2** System must install any available software updates nightly to remain up to date. The system must have failover mechanisms to ensure continuous functioning of the website in case of software failures.

**6.4 Scalability Requirements**

**6.4.1** System must be able to handle a large volume of concurrent users without affecting performance or availability.

**6.4.2** System must have a caching mechanism that stores frequently accessed data and pages, reducing the load on the database and speeding up page load times.

**6.5 Interoperability Requirements**

**6.5.1** System should be able to integrate with other existing hospital systems such as laboratory equipment, billing systems, and electronic health record (EHR) systems.

**6.5.2** The system should support industry-standard data exchange formats and protocols for seamless interoperability.

**6.6 Security Requirements**

**6.6.1** The system should have robust security measures to protect sensitive patient data and maintain patient privacy.

**6.6.2** System should include authentication mechanisms, access controls, encryption, and audit logs to prevent unauthorized access or data breaches.

**6.7 Performance Requirements**

**6.7.1** The system should be capable of handling a large number of concurrent users and high volumes of data without significant performance degradation.

**6.7.2** Response times should be within acceptable limits to ensure efficient user interactions.

**6.8 Maintainability Requirements**

**6.8.1** The system should be designed in a modular and extensible manner, facilitating easy maintenance, upgrades, and bug fixes.

**6.8.2** Changes to the system should be manageable and should not disrupt the overall functionality.

**7. Supporting Section**

**7.1 Glossary**

* **Central Patient Database:** A database managed by the health ministry, containing the medical records of individuals nationwide.
* **Local Database:** A database specific to the hospital, storing relevant data for efficient access and operations.
* **Class Diagram:** A graphical representation illustrating the data elements and relationships within a system, typically used for high-level modeling.
* **Operating System:** System software that manages computer hardware and software resources, providing essential services for computer programs to run.
* **SQL (Structured Query Language)**: is a standard language for accessing and manipulating databases.
* **Distributed Database:** A database distributed across multiple locations or networks, comprising multiple files stored in different sites
* **Front-end:** The user-facing layer of software that encompasses the design and functionality of the user interface.
* **Back-end:** The underlying components of a computer application or program that handle data processing and functionality inaccessible to the user.

**7.2 References**

* IEEE Recommended Practice for Software Requirements Specification (IEEE-STD-83.50- 1998):

[https://personal.utdallas.edu/~chung/RE/IEEE83.50-1993.5.pdf](https://personal.utdallas.edu/~chung/RE/IEEE830-1993.pdf)

* "Software Requirement Specification (SRS) Format" from GeeksforGeeks website:

<https://www.geeksforgeeks.org/software-requirement-specification-srs-format/>

* "Software Requirement Specifications" from JavaTpoint website:

<https://www.javatpoint.com/software-requirement-specifications>

* Sample SRS Document for Gephi software:

https://gephi.org/users/gephi\_srs\_document.pdf

* "A Practical Guide to Writing Software Requirements" by Ruth Malan and Dana Bredemeyer
* "Requirements Engineering: From System Goals to UML Models to Software Specifications" by Axel van Lamsweerde.
* "Writing Effective Use Cases" by Alistair Cockburn.
* "Software Requirements" by Karl Wiegers and Joy Beatty.