**School of Computer Science**

**UNIVERSITY OF PETROLEUM AND ENERGY STUDIES**

**DEHRADUN, UTTARAKHAND**



**Software Engineering**

**&**

**Project Management Lab**

**Lab File**

**(2023-2024)**

**for**

**4rd Semester**

**Submitted To: Submitted By:**

Dr Ram Kumar Akshat Negi

Assistant Professor-SS Btech CSF [4rd Semester]

School of Computer Science 500106533

Batch 2

**Experiment 1**

Perform requirement analysis and to find the requirement specification (both functional and nonfunctional) of a given Problem.

**FUNCTIONAL REQUIREMENTS:**

1. Patient Management:

- Registration: The system should allow for the registration of new patients, including capturing their personal details, contact information, and medical history.

- Appointment Scheduling: Patients should be able to schedule appointments with doctors, and the system should manage these appointments efficiently.

- Admission and Discharge: Facilitate the admission and discharge process for patients, including bed allocation and discharge summaries.

2. Doctor Management:

- Profile Management: Maintain profiles for doctors, including their specialties, contact details, and availability.

- Appointment Management: Allow doctors to view their appointments, reschedule if necessary, and manage their availability.

3. Medical Records Management:

- Electronic Health Records (EHR): Store and manage patient medical records securely, including diagnosis, treatment plans, medications, lab results, and imaging reports.

- Record Retrieval: Provide quick access to patient records for healthcare providers as needed during consultations or treatments.

4. Billing and Invoicing:

- Billing and Invoicing: Generate bills for services provided, including consultations, procedures, medications, and other healthcare services.

- Insurance Management: Handle insurance information for patients, including verification, claims processing, and coordination of benefits.

5. Inventory Management:

- Medication and Supply Management: Keep track of medications, medical supplies, and equipment stock levels, and automate reordering processes when inventory levels are low.

6. Laboratory and Imaging Management:

- Test Orders and Results: Facilitate the ordering of laboratory tests and imaging studies, as well as the recording and retrieval of results.

- Integration: Integrate with laboratory and imaging systems for seamless transfer of data and results.

7. Reporting and Analytics:

- Reporting: Generate reports on various aspects of hospital operations, including patient demographics, financial performance, resource utilization, and quality metrics.

- Analytics: Provide analytical tools to identify trends, patterns, and areas for improvement in patient care and operational efficiency.

8. Security and Compliance:

- Data Security: Ensure the confidentiality, integrity, and availability of patient information through robust security measures, including user authentication and access controls.

- Compliance: Adhere to regulatory requirements such as HIPAA (Health Insurance Portability and Accountability Act) for patient data protection and privacy.

**NON FUNCTIONAL REQUIREMENT:**

1. Performance:

- Response Time: The system should respond to user requests within an acceptable timeframe, ensuring smooth and efficient user interaction.

- Throughput: It should be able to handle a large number of concurrent users and transactions without significant degradation in performance.

- Scalability: The system should be scalable to accommodate growth in data volume, user base, and system load over time.

2. Reliability:

- Availability: The system should be available for use whenever required, with minimal downtime for maintenance or upgrades.

- Fault Tolerance: It should be resilient to failures, with mechanisms in place to detect and recover from errors gracefully to prevent data loss or service disruptions.

3. Security:

- Data Security: Ensure the confidentiality, integrity, and privacy of patient information through encryption, access controls, and secure transmission protocols.

- Authentication and Authorization: Implement robust mechanisms for user authentication and authorization to prevent unauthorized access to sensitive data and system functionalities.

- Audit Trails: Maintain detailed logs of user activities and system events for auditing and forensic purposes.

4. Usability:

- User Interface: The user interface should be intuitive, easy to navigate, and visually appealing, catering to users with varying levels of technical expertise.

- Accessibility: Ensure that the system is accessible to users with disabilities, complying with accessibility standards and guidelines.

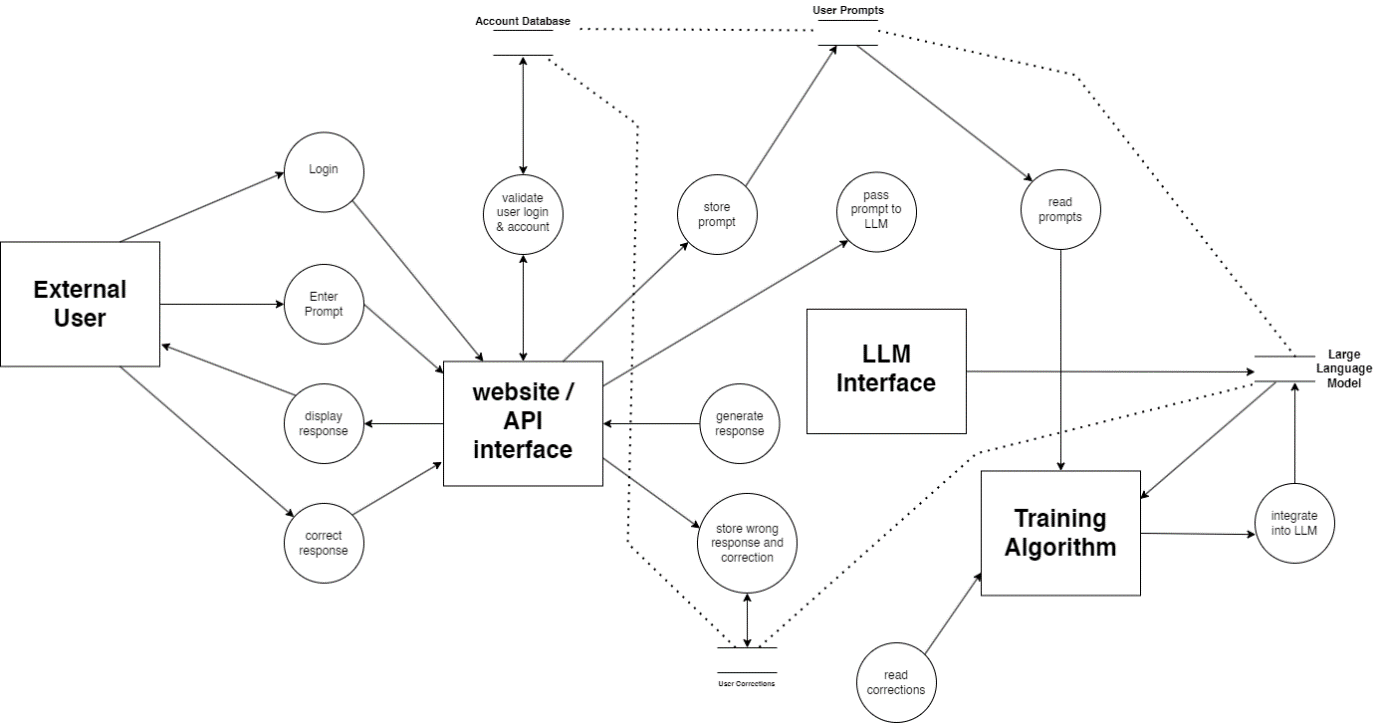
5. Maintainability:

- Modularity: Design the system with modular components that can be easily maintained, updated, or replaced without affecting the overall functionality.

- Documentation: Provide comprehensive documentation, including system architecture, configuration instructions, and troubleshooting guides, to support system maintenance and administration.

**Experiment 2**

To perform the function-oriented diagram: Data Flow Diagram (DFD) and Structured chart.

****

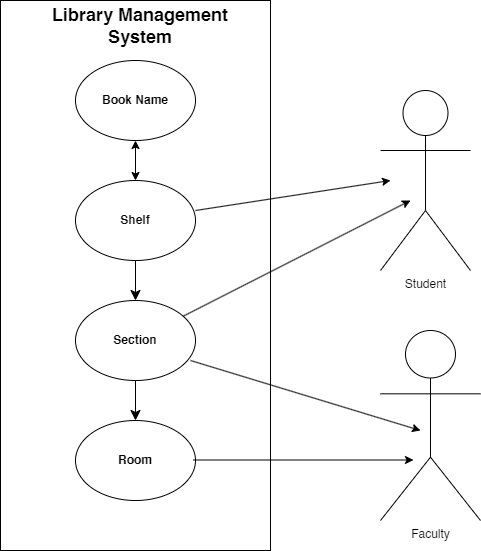
**Experiment 3**

To perform the user ‘s view analysis for the suggested system: Use case diagram.

**What is a use case diagram?**

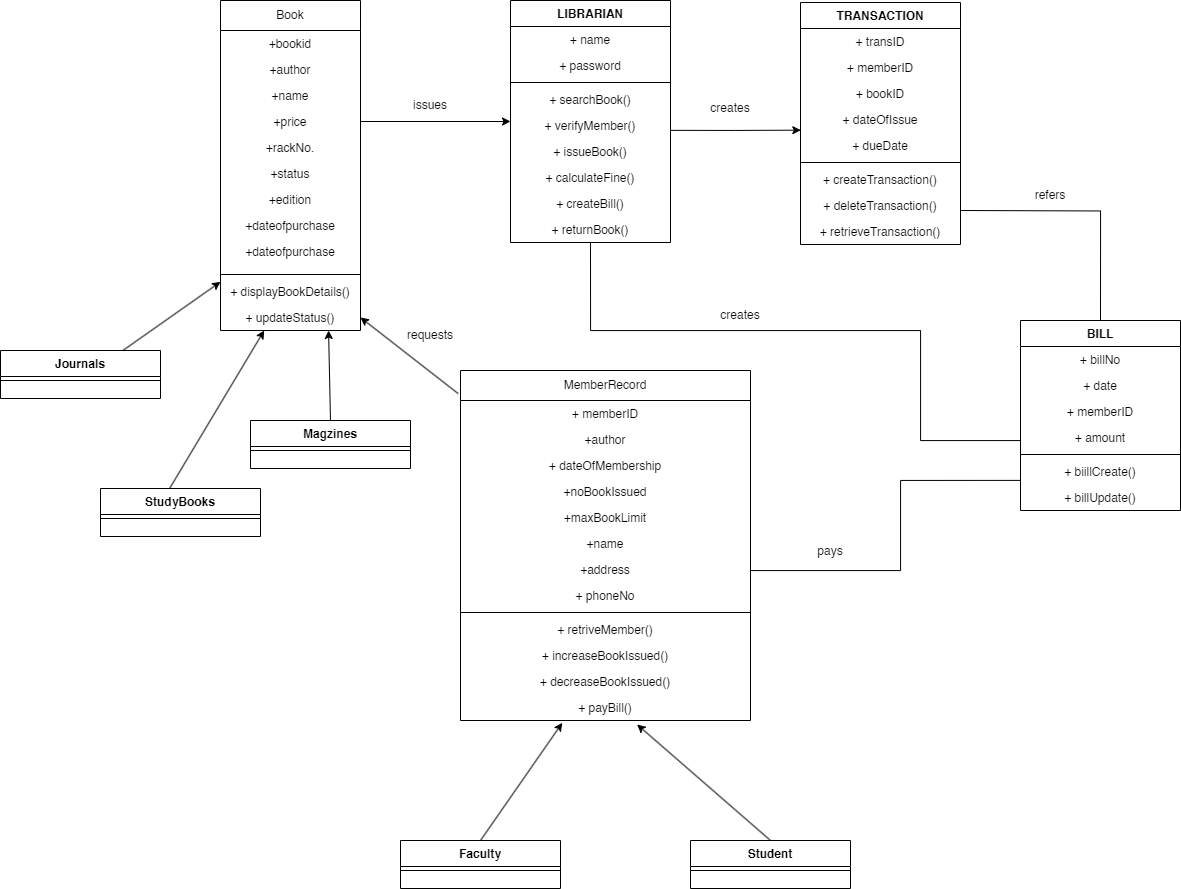
• In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system.

• Use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems.

•

**Experiment 4**

To draw the structural view diagram for the system: Class diagram, object diagram.



**Experiment 5**

To perform the behavioral view diagram for the suggested system: Sequence diagram, Collaboration diagram

**What is an interaction diagram?**

Interaction diagrams are used in UML to establish communication between objects. They do not manipulate the data associated with the particular communication path. Interaction diagrams mostly focus on message passing and how these messages make up one functionality of a system. Interaction diagrams are designed to display how the objects will realize the particular requirements of a system. The critical component in an interaction diagram is lifeline and messages. Various UML elements have their own interaction diagrams. The details of the interaction can be shown using several annotations, such as the sequence diagram, timing diagram, and communication/collaboration diagram. Interaction diagrams capture the dynamic behavior of any system.

**The different types of interaction diagrams defined in UML:**

**• Sequence diagram**

**• Collaboration diagram**

**• Timing diagram**

• The purpose of a sequence diagram in UML diagrams to visualize the sequence of a message flow in the system. The Sequence Diagram in Software Engineering shows the interaction between two lifelines as a time-ordered sequence of events.

• The Collaboration Diagram in UML is also called a communication diagram. The purpose of a collaboration diagram is to emphasize structural aspects of a system, i.e., how various lifelines in the system connects.

• Collaboration Diagram depicts the relationships and interactions among software objects. They are used to understand the object architecture within a system rather than the flow of a message as in a sequence diagram. They are also known as “Communication Diagrams.”

