**COVER PAGE**

A logo of a university

Description automatically generated

EEC 521/CIS 534: Software Engineering

Professor: Dr. Yongjian Fu

|  |  |  |
| --- | --- | --- |
| **Team Members** | | |
| Team Member 1 | LIKITH REKAPALLI | CSU ID: 2845684 |
| Team Member 2 | KRISHNA SAI SALLA | CSU ID: 2887720 |

**TEST PLAN**

**1.0 Introduction**

**1.1 Purpose and Scope**

**Purpose**:

1. **Patient Management**: Automate and streamline patient registration, appointment scheduling, medical history management, and treatment tracking.
2. **Staff Management**: Manage hospital staff information, schedules, roles, and payroll.
3. **Billing and Invoicing**: Facilitate accurate and efficient billing for patients, including room charges, treatments, consultations, and other medical services.
4. **Room and Bed Management**: Optimize the assignment and management of hospital rooms and beds, ensuring effective utilization.
5. **Medical Record Management**: Safely store and manage patient medical records for quick access by healthcare providers.
6. **Data Security and Privacy**: Ensure the security of sensitive patient data, including medical records, personal information, and financial details.

**Scope**:

1. **Patient Management**: Include features for patient registration, appointment booking, tracking medical history, and managing treatments.
2. **Staff Management**: Implement staff scheduling and role assignments.
3. **Billing and Financial Management**: Handle patient billing and invoicing
4. **Medical Record Management**: Provide tools to store, update, and retrieve patient records, including medical history.
5. **Room and Bed Management**: Assign and track room availability for patients, manage bed occupancy, and allocate rooms based on patient needs.
6. **Communication and Collaboration**: Facilitate communication between healthcare providers, patients, and administrators.
7. **Security and Privacy**: Ensure compliance with data protection laws (e.g., HIPAA, GDPR) by securing patient information and enabling role-based access control.
8. **Scalability and Integration**: Design the system to handle increasing hospital data, patient records, and user volumes.

**1.2 Product Overview**

The **Hospital Management System** is a comprehensive software solution designed to streamline the operations of hospitals, enhance the quality of healthcare delivery, and ensure efficient management of hospital resources.

**Key Capabilities:**

1. **Patient Management**:
   * Registration and appointment booking.
   * Tracking of patient medical history and treatment records.
2. **Staff Management**:
   * Manage hospital staff details and scheduling
   * Role-based access control to ensure confidentiality of sensitive data.
3. **Billing and Financial Management**:
   * Generate bills based on services rendered (room charges, consultations, treatments).
   * Integration with insurance systems for claim management and verification.
   * Track patient payments and outstanding balances.
4. **Room and Bed Management**:
   * Allocate rooms to patients based on available space and medical needs.
   * Track room occupancy, discharges, and transfers.
5. **Medical Record Management**:
   * Store and retrieve patient medical records, including diagnoses, treatments, prescriptions, and test results.
   * Generate medical reports and summaries for quick reference.
6. **Inventory and Supply Management**:
   * Track the availability and usage of medical supplies and equipment.
   * Automate restocking alerts to maintain sufficient inventory.
7. **Communication and Collaboration**:
   * Enable communication between patients, medical staff, and administrators through messaging systems, notifications, and alerts.
8. **Regulatory Compliance**:
   * Ensure adherence to healthcare regulations and standards.
   * Generate reports for audits and compliance checks.

**Scenarios for Using the Product:**

1. **Patient Registration and Appointment Booking**:  
   medical staff can register, book appointments, and access medical records securely through the system.
2. **Medical Record Management**:  
   Doctors and medical staff can easily update patient medical histories, diagnoses, prescriptions, and test results in the system.
3. **Billing and Insurance Verification**:  
   Staff can generate patient bills and view previous bills.
4. **Room Allocation and Management**:  
   The system helps allocate rooms to patients based on their medical needs, monitor room occupancy, and ensure efficient space utilization.
5. **Staff Scheduling**:  
   Administrators can manage staff shifts.

**Benefits:**

1. **Operational Efficiency**:  
   Automating manual processes like patient registration and billing leads to greater operational efficiency.
2. **Improved Patient Care**:  
   Healthcare providers have instant access to patient records, enhancing the accuracy and speed of decision-making in patient care.
3. **Cost Savings**:  
   Optimizing hospital resource usage, such as bed management helps reduce costs and improve profitability.
4. **Enhanced Collaboration**:  
   Integrated communication tools improve collaboration among medical staff, patients, and administrators, leading to faster response times and better service delivery.
5. **Data Security and Privacy**:  
   Role-based access controls, encryption, and secure data storage ensure that patient data remains protected, complying with healthcare regulations.

**Target Users:**

1. **Hospitals and Healthcare Providers**:  
   Streamline operations, improve patient care, and ensure regulatory compliance.
2. **Healthcare Administrators**:  
   Efficiently manage hospital resources, including staffing, billing, and inventory.
3. **Doctors and Medical Staff**:  
   Access patient records, track treatments, and collaborate with other medical professionals to provide optimal care.

**2. Project Management Plan**

**2.1 Project Organization**

**Project Roles and Responsibilities**

1. **Project Manager**: Oversee project planning, execution, and delivery.
2. **System Analyst**: Conduct requirements analysis and documentation.
3. **Software Developers**: Design, develop, and implement the HMS system.
4. **Quality Assurance (QA) Team**: Test the system to ensure functionality, performance, and security.
5. **Database Administrator (DBA)**: Design and manage the database schema and performance.
6. **IT Support Team**: Ensure system stability during and after deployment.
7. **Stakeholders**: Provide feedback and approve key project deliverables.

**2.2 Lifecycle Model Used**

**Waterfall Model with Phased Implementation**

**Phases**

1. **Requirements Analysis**:
   * Gather detailed requirements from hospital administrators, staff and other stakeholders.
   * Document functional and non-functional specifications.
2. **System Design**:
   * Create system architecture and database schema.
   * Plan phased implementation starting with patient management and appointment scheduling.
3. **Implementation (Phase 1)**:
   * Develop core functionalities such as patient registration, appointment management, and basic reporting.
4. **Testing (Phase 1)**:
   * Perform unit, integration, and system testing on core functionalities.
   * Address and resolve identified issues.
5. **Implementation (Phase 2)**:
   * Expand functionalities to include staff scheduling and billing.
   * Integrate with external systems like payment gateways.
6. **Testing (Phase 2)**:
   * Test integrated systems with advanced functionalities and data exchanges.
7. **User Acceptance Testing (UAT)**:
   * Include hospital administrators, healthcare providers, and technical staff in UAT.
   * Refine the system based on feedback.
8. **Deployment and Maintenance**:
   * Deploy the HMS system across the hospital.
   * Provide ongoing maintenance and support.

**Considerations**

* **Regulatory Compliance**: Adhere to healthcare data protection regulations.
* **Stakeholder Involvement**: Involve end users at each stage for alignment with their needs.
* **Integration Challenges**: Mitigate complexities in integrating with hospital legacy systems.

**2.3 Risk Analysis**

**Identified Risks and Mitigation Strategies**

1. **Regulatory Compliance**
   * **Risk**: Non-compliance with healthcare data protection laws.
   * **Mitigation**: Regular monitoring of regulatory changes, involve legal experts, and design a flexible system.
2. **Data Security and Privacy**
   * **Risk**: Breaches or unauthorized access to patient data.
   * **Mitigation**: Implement encryption, access controls, regular audits, and adhere to HIPAA/GDPR standards.
3. **Healthcare Integration Challenges**
   * **Risk**: Technical challenges in integrating with legacy hospital systems.
   * **Mitigation**: Collaborate with hospital IT teams and use standardized integration protocols.
4. **User Adoption and Training**
   * **Risk**: Difficulty in system adoption by healthcare staff.
   * **Mitigation**: Conduct training sessions and create user-friendly interfaces.
5. **Scalability Issues**
   * **Risk**: System struggles to handle increased data or user load.
   * **Mitigation**: Design for scalability, including load testing and future-proofing architecture.
6. **Operational Downtime**
   * **Risk**: System failures or downtime during critical hospital operations.
   * **Mitigation**: Implement redundancy and backup systems, and schedule maintenance during low-impact periods.
7. **Incomplete Requirements**
   * **Risk**: Missing or ambiguous requirements lead to rework.
   * **Mitigation**: Conduct detailed requirements analysis with stakeholder involvement.

**2.4 Hardware and Software Resource Requirements**

**Hardware Requirements**

1. **Web Server**: Scalable server for hosting the HMS application.
2. **Database Server**: High-performance server for managing large datasets.
3. **Application Server**: Middleware to ensure smooth communication between web and database servers.
4. **Bandwidth**: High-speed internet for uninterrupted operations.
5. **Redundancy**: Backup servers for failover support.

**Software Requirements**

1. **Server OS**:
   * Linux (e.g., Ubuntu) or Windows Server.
2. **Web Application Server**:
   * Apache or Nginx for hosting the HMS application.
3. **Relational Database**:
   * MSSQL for handling structured hospital data.
4. **Programming Languages**:
   * Backend: PHPMyadmin
   * Frontend: HTML, CSS, JavaScript.
5. **Integrated Development Environment (IDE)**:
   * Visual Studio for development.
6. **Security Tools**:
   * SSL/TLS certificates for HTTPS communication.

**2.5 Deliverables and Schedule**

**Key Deliverables**

1. Requirements Document
2. System Architecture and Design Documents
3. Core Functionalities Implementation (Phase 1)
4. Advanced Features and Integration (Phase 2)
5. User Acceptance Testing Results
6. Deployment and Maintenance Plan

**Project Schedule**

|  |  |  |  |
| --- | --- | --- | --- |
| **Phase** | **Start Date** | **End Date** | **Duration** |
| Requirements Analysis | 01/01/2024 | 15/01/2024 | 15 days |
| System Design | 16/01/2024 | 30/01/2024 | 15 days |
| Implementation (Phase 1) | 01/02/2024 | 29/02/2024 | 1 month |
| Testing (Phase 1) | 01/03/2024 | 15/03/2024 | 15 days |
| Implementation (Phase 2) | 16/03/2024 | 15/04/2024 | 1 month |
| Testing (Phase 2) | 16/04/2024 | 30/04/2024 | 15 days |
| User Acceptance Testing | 01/05/2024 | 15/05/2024 | 15 days |
| Deployment and Maintenance | 16/05/2024 | Ongoing | Ongoing |

**3. Requirement Specifications**

**3.1 Stakeholders for the System**

1. **Hospital Administrators**
   * Oversee hospital operations and ensure the system supports efficient management.
   * Provide requirements related to hospital workflows and compliance.
2. **Doctors and Nurses**
   * Use the system for managing patient records and appointments.
   * Provide feedback on user interface and ease of use.
3. **System Administrators**
   * Configure the HMS system, manage user accounts, and ensure system security and uptime.
4. **IT Support Team**
   * Provide technical support and maintenance for the HMS.
5. **Government and Regulatory Bodies**
   * Oversee compliance with healthcare regulations and standards (e.g., HIPAA, GDPR).

**3.2 Use Cases**

**3.2.1 Graphic Use Case Model**

**A diagram of a diagram

Description automatically generated**

**3.2.2 Textual Description for Each Use Case**

1. **Patient Info:**

**Actors:** Staff & Admin

* Both Staff and Admin can view the patient info

1. **Edit Patient:**

**Actors:** Admin

* This action can only be performed by Admin

1. **Staff Info:**

**Actors:** Staff & Admin

* Both Staff and Admin can view the details of the staff

1. **Edit Staff:**

**Actors:** Admin

* This action can only be performed by Admin

1. **Room Allocation:**

**Actors:** Staff & Admin

* Room allocation can be done by both staff and admin

1. **Generate Invoice:**

**Actors:** Staff & Admin

* Both staff and admin can generate invoices.

1. **Admitted Patient Info:**

**Actors:** Staff & Admin

* Staff and admin both can view the admitted patient info

**3.3 Rationale for Use Case Model**

1. **User-Centric Approach**
   * Focuses on key factors such as patients, healthcare staff, and administrators, ensuring the system meets their needs.
2. **Hospital Operations**
   * Captures core hospital operations like medical records management, and billing.
3. **System Administration**
   * Highlights functionalities needed for system administrators to maintain and support the HMS effectively.

**3.4 Non-Functional Requirements**

1. **Performance**
   * **Response Time**: All system actions, including record retrieval and appointment scheduling, should be completed within 2 seconds.
   * **Throughput**: The system should support up to 500 concurrent users without performance degradation.
2. **Scalability**
   * **User Scalability**: Accommodate increasing numbers of patients, staff, and administrators.
   * **Data Scalability**: Handle growing volumes of patient and hospital data efficiently.
3. **Reliability**
   * **Availability**: Ensure 99.9% system uptime for uninterrupted hospital operations.
   * **Fault Tolerance** : Implement mechanisms for detecting and recovering from system failures.
4. **Security**
   * **Data Encryption**: Encrypt sensitive data during storage and transmission.
   * **Access Control**: Role-based access control for sensitive functionalities.
   * **Audit Trails**: Log all critical actions and changes to sensitive data for security and compliance monitoring.
5. **Usability**
   * **User Interface Design**: Provide an intuitive interface tailored to the needs of hospital staff and patients.
   * **Accessibility**: Ensure compliance with accessibility standards (e.g., WCAG).
6. **Compatibility**
   * **Browser Compatibility**: Support major web browsers, including Chrome, Firefox, and Edge.
7. **Maintainability**
   * **Code Maintainability**: Write modular, clean, and well-documented code to simplify updates.
   * **System Documentation**: Provide comprehensive documentation for users, administrators, and developers.
8. **Data Backup and Recovery**
   * **Regular Backups**: Schedule daily automated backups of critical data.
   * **Recovery Procedures**: Establish protocols for rapid data recovery in case of system failures.
9. **Regulatory Compliance**
   * **Legal Requirements**: Ensure compliance with healthcare regulations, such as HIPAA for patient data privacy.

**4. Architecture**

**4.1 Architectural Style**

**1. Service-Oriented Architecture (SOA)**

**Description**:  
SOA structures the system as a set of loosely coupled and interoperable services. Each service corresponds to a specific business functionality and operates independently.

**Rationale**:

* SOA enables modular development, enhancing scalability and flexibility.
* It supports the distributed nature of HMS, allowing seamless integration.

**4.2 Architectural Model (Components and Interactions)**

**1. User Interface (UI) Layer**

**Description**:  
The UI layer interacts directly with users, presenting information and capturing input.

**Components**:

* **Web-based Interfaces**: For patients, doctors, and administrators to access the system.

**2. Application Layer**

**Description**:  
Contains the core business logic and operational workflows of the system.

**Components**:

* **Medical Records Component**: Handles secure access to patient medical histories.
* **Billing Component**: Manages billing operations, including invoice generation.

**3. Services Layer**

**Description**:  
Encapsulates individual services that provide specific functionalities and interact with other layers.

**Components**:

* **Patient Service**: Manages patient information and interactions.
* **Medical Records Service**: Provides secure access to patient records.

**4. Data Access Layer**

**Description**:  
Manages data persistence and ensures secure interaction with the database.

**Components**:

* **Database Connector**: Establishes secure communication with the database.
* **Data Access Objects (DAOs)**: Manages CRUD (Create, Read, Update, Delete) operations.

**5. Security Layer**

**Description**:  
Ensures secure authentication, authorization, and data protection.

**Components**:

* **Authentication Service**: Verifies user identities.
* **Authorization Service**: Manages role-based access controls.
* **Encryption Module**: Encrypts sensitive data for secure storage and transmission.

**6. Interactions:**

**Internal Component Interaction**:

* + The UI layer communicates with the application layer to perform core business logic.
  + The application layer uses the services layer to execute specific tasks.
  + The services layer interacts with the data access layer to retrieve or store information.

**Security Interactions**:

* + The security layer ensures all interactions are authenticated, authorized, and encrypted.

**4.3 Technology, Software, and Hardware Used**

**Hardware Requirements**

1. **Web Server**:
   * Scalable server to host the HMS application, ensuring reliable access.
2. **Database Server**:
   * High-performance server to handle large datasets and complex queries.
3. **Application Server**:
   * Middleware to connect the web server and database, facilitating communication.
4. **Network Infrastructure**:
   * High-speed internet connectivity and firewalls for secure, uninterrupted operations.
5. **Backup and Redundancy**:
   * Backup systems for regular data backups.
   * Redundant servers for high availability and disaster recovery.

**Software Requirements**

1. **Server OS**:
   * Linux (e.g., Ubuntu, CentOS) or Windows Server for stability and security.
2. **Web Server**:
   * **Apache** or **Nginx** for hosting the application.
3. **Database**:
   * **MSSQL** for structured data storage.
4. **Programming Languages**:
   * Backend:**PHPMyadmin**
   * Frontend: **HTML, CSS, JavaScript**

**4.4 Rationale for Architectural Style and Model**

1. **Modularity and Reusability**:
   * Decomposing the system into independent services enhances reusability and simplifies updates.
2. **Scalability**:
   * SOA supports horizontal and vertical scaling, allowing the system to grow with increasing users and data.
3. **Flexibility and Adaptability**:
   * New services can be added or updated without affecting existing components, ensuring adaptability to changing requirements.
4. **Ease of Maintenance**:
   * Modular services simplify debugging and updates, improving system reliability and reducing downtime.

**5. Design**

**5.1 User Interface (UI) Design**

**1. User Persona’s Rationale**:  
Identify distinct user roles—patients, healthcare staff, and administrators. Design role-specific interfaces to cater to their unique workflows and requirements.

**2. Clear and Role-Based Dashboards Rationale**:  
Provide customized dashboards for each user role:

* **Patients**: medical records, and billing details.
* **Healthcare Staff**: Access patient histories.
* **Administrators**: Monitor system performance, manage user roles, and review analytics.

**3. Intuitive Navigation Rationale**:  
Implement an intuitive navigation system with a well-structured menu. Users should quickly locate core functionalities such as records management, and billing.

**4. Security Measures Rationale**:  
Implement multi-factor authentication (MFA), encrypted data storage, and secure communication protocols. Regularly audit for vulnerabilities.

**5.2 Components Design**

A diagram of a diagram

Description automatically generated

**5.3 Database Design**

**A diagram of a patient invoice

Description automatically generated**

**6. Test Management**

**6.1 Complete List of System Test Cases**

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case ID** | **Test Scenario** | **Expected Outcome** | **Pass/Fail** |
| TC001 | User login functionality | Users (doctors, nurses, admin, patients) can log in securely. | Pass |
| TC002 | Patient registration | New patients are registered successfully in the system. | Pass |
| TC003 | Medical record creation and retrieval | Medical staff can create and retrieve patient records accurately. | Pass |
| TC004 | Prescription generation and management | Doctors can generate, update, and review prescriptions. | Pass |
| TC005 | Laboratory report management | Lab staff can upload, manage, and retrieve lab reports. | Pass |
| TC006 | Staff role-based access control | System enforces appropriate permissions based on user roles. | Pass |
| TC007 | Data encryption | Sensitive data is encrypted in transit and at rest. | Pass |
| TC008 | System backup and recovery | Backups can be restored without data loss. | Pass |
| TC009 | Load handling for concurrent users | System remains responsive under simultaneous usage by multiple staff members. | Pass |
| TC010 | Error handling and notifications | System provides meaningful error messages and notifications for failures. | Pass |

**6.2 Techniques Used for Test Case Generation**

**1. Regression Testing**

**Description** : Verify that new features or fixes do not negatively impact existing functionalities, such as patient record management or prescription updates.

**2. Compatibility Testing**

**Description** : Ensure the system operates correctly across various devices, browsers, and operating systems used in the hospital.

**3. User Interface Testing**

**Description** : Assess the UI for clarity, accessibility, and ease of navigation for all roles, including doctors and administrators.

**4. Security Testing**

**Description** : Test for vulnerabilities such as unauthorized access, data leakage, or privilege escalation.

**5. Load and Performance Testing**

**Description** : Simulate high user activity to test the system's responsiveness and stability under different loads.

**6. Use Case Testing**

**Description** : Validate key workflows such as medical record creation to ensure they align with functional requirements.

**6.3 Test Results and Assessments**

**To Be Conducted After Implementation**

**6.4 Defects Reports**

**To Be Documented Based on Testing Outcomes**

**7. Conclusions**

**7.1 Outcomes of the Project**

1. **Medical Record Management**  
   **Goal**: Enable efficient creation, retrieval, and management of patient medical records.  
   **Outcome**: Medical staff can easily access and update patient records without errors.
2. **Administrative Functions**  
   **Goal**: Provide administrators with tools for user management, system monitoring, and reporting.  
   **Outcome**: Administrators can efficiently manage the system and ensure smooth operations.

**7.2 Lessons Learned**

1. **Scalability Needs**  
   **Lesson Learned**: The system should scale to handle increased patient data as the hospital grows.  
   **Action Item**: Prioritize scalable architecture in future updates.
2. **Role-Based User Feedback**  
   **Lesson Learned**: Early user feedback highlighted areas for improving role-specific workflows.  
   **Action Item**: Integrate regular feedback loops in the development cycle.

**7.3 Future Development**

1. **Mobile Accessibility for Staff**  
   **Objective**: Develop responsive web interfaces or mobile apps to allow staff access to key features on-the-go.  
   **Actions**:
   * Explore frameworks for mobile-friendly UI development.
   * Test critical workflows on mobile devices.
2. **AI Integration for Diagnostics**  
   **Objective**: Incorporate AI tools to assist in diagnostics and treatment recommendations.  
   **Actions**:
   * Evaluate AI modules for compatibility with existing systems.
   * Train staff in the use of AI-assisted tools.
3. **Interoperability with External Systems**  
   **Objective**: Enhance integration with external healthcare systems for better data sharing.  
   **Actions**:
   * Implement HL7 or FHIR standards for data exchange.
   * Conduct interoperability testing with external platforms.