DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Software Engineering [UE22CS341A]

Project Plan for Hospital Management

Hospital Records Management System

Samarth NN (PES1UG22CS509)

Vidhaan Viswas (PES1UG23CS835)

Table of Contents

1. Lifecycle Selection

- Chosen Lifecycle: Agile methodology
- Justification:
  - A hospital management system requires frequent feedback from medical staff, administrators, and patients to ensure usability, security, and efficiency. Agile's iterative approach allows continuous improvements.
  - The ability to deliver working modules (e.g., patient management, appointments, billing) incrementally minimizes risk and allows quick adaptation to evolving requirements.

---

2. Tools Selection

- Planning Tool: Microsoft Project or Trello for creating and tracking the project schedule.
- Design Tool: Lucid chart or Draw.io for designing system architecture and database schemas.
- Version Control: Git with GitHub for managing code changes and collaboration.
- Development Tool: Eclipse IDE or IntelliJ IDEA for Java development; MySQL Workbench for database management.
- Bug Tracking: JIRA or Bugzilla for tracking issues, tasks, and bugs during development.
- Testing Tool: JUnit for unit testing in Java, and Selenium for functional testing of the application.

---

3. Deliverables

3.1 Develop Components:

- Patient Management: Custom-built to track patient records, admissions, and discharges (features: patient registration, appointment scheduling, history management).
- Staff Management: System for managing staff roles, shifts, and payroll.
- Billing System: Custom-built to handle patient billing, insurance claims, and payment tracking.

---

4. Reuse Components

- Authentication Module: Utilize Java libraries for secure login and role-based access control.
- Report Generation: Use third-party reporting tools or libraries to export reports in PDF/Excel formats (e.g., patient records, billing summaries).
- Justification: Custom development is necessary for critical hospital-specific features, while reusable libraries are used for common functionalities to save time and ensure security.

5. Work Breakdown Structure (WBS)

5.1 Level 1: Major modules

1. User Authentication
2. Patient Management
3. Staff Management
4. Appointment Scheduling
5. Billing System
6. Report Generation
7. Testing

5.2 Level 2: Tasks within each module

- User Authentication: Secure login, role-based access.
- Patient Management: CRUD operations for patient records, medical history, and reports.
- Staff Management: CRUD operations for staff data, shift management, payroll.
- Appointment Scheduling: Scheduling interface, reminders, patient-doctor match.
- Billing System: Invoice generation, insurance claims management, payment tracking.
- Report Generation: Daily, weekly, monthly reports for patient data, staff activities, billing.

---

6. Effort Estimation and Gantt Chart

6.1 Effort Estimation:

- Authentication Module: 1 person for 1 month.
- Patient Management Module: 2 people for 3 months.
- Staff Management Module: 1 person for 1.5 months.
- Appointment Scheduling Module: 1 person for 1.5 months.
- Billing System: 2 people for 2 months.
- Report Generation: 1 person for 1 month.
- Testing & Debugging: 2 people for 1 month.

6.2 Gantt Chart:

- Use Microsoft Project or similar tools to visualize the project timeline.
- Divide the project into sprints for each module, ensuring parallel development where possible (e.g., patient management and billing can be developed concurrently).
- Schedule frequent review meetings for feedback and adjustments.

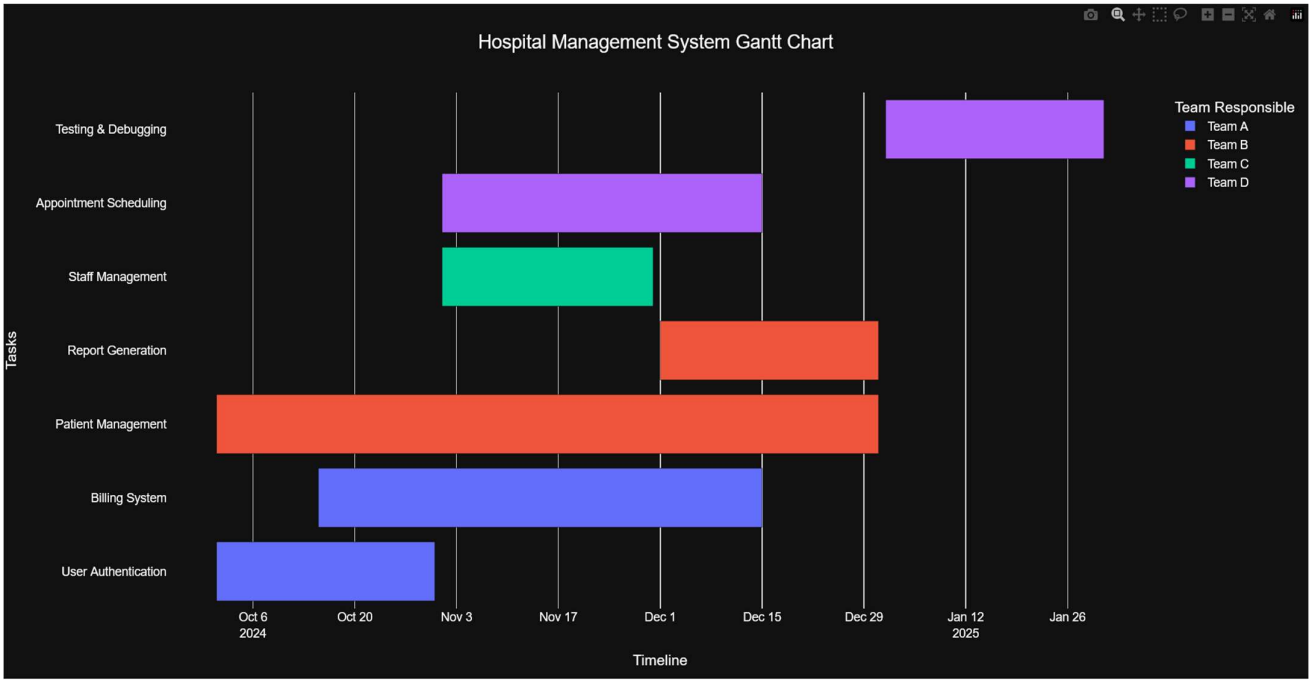

Figure 1 Sample Gant Chart

7. Coding Details

7.1 Language

The project will be coded in Java using Swing or JavaFX for the GUI, and MySQL for database operations.

7.2 Coding Standards

- Follow Java best practices, including proper use of object-oriented principles (encapsulation, inheritance).
- Ensure code modularity to facilitate scalability and ease of maintenance.
- Code comments and documentation will be thorough, especially for complex functions.

7.2 Version Control

Use GitHub for managing all code changes and maintaining different branches for each module. Regular commits are expected to track progress.

End of Report