Wahiq Iqbal & Viney Jain

Principles of Computer Systems- COMP 312

Prof. Chiranjib Sur

Final Project - Part A

24th January 2025

Library Management Software: A Project by Viney Jain and Wahiq Iqbal

---

1. Software Requirement Specification (SRS)

1.1 End-User Version

This library management software aims to streamline library operations by facilitating book issuance, returns, inventory management, and membership records while ensuring user-friendly access for librarians, students, and faculty.

Key Features of the management software:

1. **User Registration & Login:**
    - The software will have separate portals for administrators, librarians, and users (students, faculty).
    - The software will also support password recovery, two-factor authentication, and role-based access (for students, administrators and librarians).
2. **Catalog Management:**

- Stakeholders, exclusively administrators, and librarians can add, update, and delete books, journals, and digital media.
- Administrators and librarians can also categorise and assign books by genre, author, and publication year.
- Administrators and librarians can also batch import/export functionality for bulk updates.

3. **Search Functionality:**
    - All the stakeholders, including students,should be able to search any book or publication by title, author, ISBN, genre, or keyword tags.
    - They can also search using advanced filters for availability, user reviews, and date of addition.
4. **Transaction Management:**
    - Students can issue and return books with automatic due date calculations.
    - The system will also have fine management for overdue items or late returns, with online payment integration (potentially).
    - The system will push notifications to the students for upcoming due dates and overdue fines to avoid the fine promptly.
5. **Membership Management:**

- The system should be able to track the user borrowing history and subscription status of each book and publication.
- The student will receive notifications for expired memberships and renewal options, and the administrator and the librarians can cancel the subscription and renew it also.
- The system should be able to generate membership cards with QR codes for each student for easy scanning.

6. **Reporting:**
    - The system will generate overdue book lists, popular books, and inventory status reports, which can be visible to the administrator and the librarian. This is to provide a general overview of the publications so they can track the inventory and update it accordingly
    - The system will have a dashboard that will provide visual analytics for user activity and book trends. These reports will be primarily charts (histograms, trend charts, etc)
    - The system should also let the administrators and the librarians download reports in various formats (PDF, Excel) for easier sharing and collaboration.

7. **Integration:**
    - The student will receive email/SMS alerts for overdue books, due dates, and new arrivals to be updated with the library accordingly.
    - It will also have API support for integrating with external libraries or academic systems like Bartleby, Internet Archive, Google Books, etc.
    - It will also have third-party tools for payment gateways like Razorpay, PhonePe and other providers and digital reading platforms like Zotero or Google Books.

1.2 Developer Version

- **Functional Requirements:**
- The system should have a comprehensive database schema for books, users, transactions, and notifications, and all this should be mapped to each other through dependencies.
- It should also have RESTful API endpoints for user authentication, book search, and transaction processing.
- The software should have a dynamic admin dashboard with CRUD (Create, Read, Update, Delete) functionalities for catalogue, users, and system settings.

Non-Functional Requirements:

- **Scalability:** It should be capable of supporting 10,000+ concurrent users with minimal latency. Also, in given future, if the system is scaled further, it should be able to scale the system with less to minor changes when shifting to new protocols or servers for handling
- **Performance:** The query response times should be at max 2 seconds
- **Security:** The System should implement encryption (SSL/TLS), secure database connections, and robust authentication measures. It should have RSA for encryption decryption and also SHA used for storing authentication details and other sensitive information.
- **Maintainability:** The system should be based on modular architecture for effortless updates and scalability in the near future.
- **Compliance:** The system should adhere to the data protection laws of India especially the Digital Personal Data Protection Act (DPDP Act) of 2023.

Technology Stack:

- Research and Designing: Figma for easier collaboration and easy to use and present
- Frontend: React.js or Angular for interactive and responsive UI.

- Backend: Node.js or Python (Django/Flask) for scalable and efficient server operations.
- Database: MySQL/PostgreSQL for relational data, Redis for caching.
- Deployment: AWS or Azure using Docker containers for streamlined CI/CD.
- Testing: Selenium and Pytest for automated UI and backend testing.

1.3 Acceptance Document

Criteria for Acceptance:

- The system supports seamless addition, updating, and removal of library resources.
- The system should be user-friendly and intuitive interface with robust search and filtering options.
- The system should have real-time notifications for transactions and membership updates.
- The system should have a reliable method for generating detailed reports and analytics.
- The system should be cross-platform compatible (at least desktop and mobile).
- The System should meet the basic security and compliance standards.

- The system should be easier to navigate, and the user experience should be seamless, with user achieving their goals with minimal clicks and effort

2. Project Planning

2.1 Planner and Timelines

- **Phase 1: Requirement Analysis, Research, and Ideation (2 weeks)**

1. The team should conduct stakeholder interviews with key users, managers, and other relevant parties to identify core needs, pain points, and expectations for the system.
2. The team should gather functional and non-functional requirements through these discussions and document them comprehensively.
3. The team should review the initial Software Requirements Specification (SRS) that outlines the technical and business requirements for the system.
4. The team should refine the SRS by incorporating feedback from stakeholders, ensuring the document is aligned with user needs and project goals.

- **Phase 2: Design (3 weeks)**

1. The team should create detailed wireframes for the system's user interface, ensuring all critical workflows are represented, and the layout is user-friendly.

2. The team should design the database schema to support the data needs of the system, ensuring it is scalable, efficient, and secure.
3. The team should create API specifications that define the interactions between the front end and back end, including data formats, endpoints, and expected responses.
4. The team should develop a working prototype of the system based on the wireframes and specifications to enable early-stage user testing and gather initial feedback.

- **Phase 3: Development (6 weeks)**

1. The team should build the frontend components of the system, focusing on user experience, accessibility, and responsiveness across devices and browsers.
2. The team should simultaneously build the backend components, ensuring the logic, performance, and scalability meet the project requirements.
3. The team should integrate necessary APIs to ensure smooth communication between different parts of the system. This includes third-party services such as payment gateways, notification systems, or analytics tools.
4. The team should integrate and configure a payment gateway to facilitate secure transactions and handle different payment methods, such as credit/debit cards and digital wallets.

5. The team should ensure that notification systems are properly integrated to send alerts and updates to users in a timely manner through channels like email, SMS, or web-based notifications.
6. The team should conduct weekly code reviews to ensure the codebase is aligned with project goals, maintaining quality, consistency, and adherence to best practices.

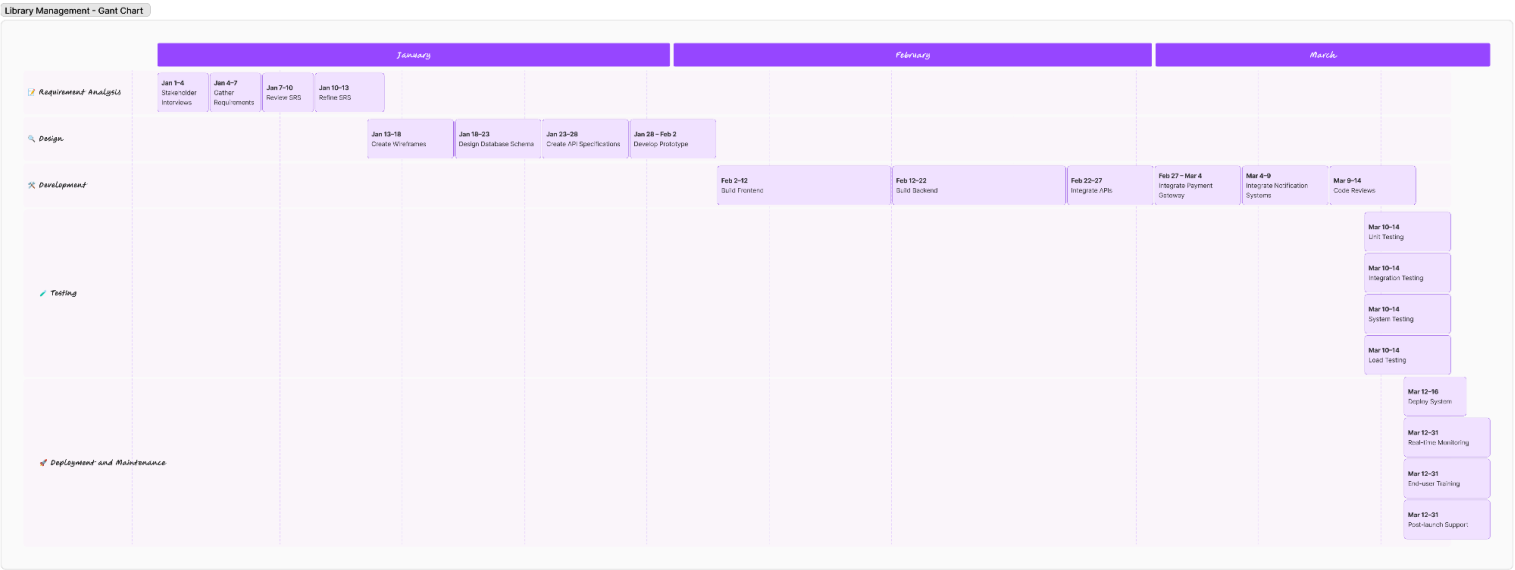
- **Phase 4: Testing (3 weeks)**

1. The team should perform unit testing on individual components of the system to ensure each part works as expected in isolation.
2. The team should conduct integration testing to verify that different components of the system interact correctly and data flows properly between them.
3. The team should perform system testing to validate the overall functionality of the system, ensuring that all requirements are met and the system performs as expected in real-world conditions.
4. The team should conduct load testing to simulate high traffic and verify that the system can handle peak usage without performance degradation or downtime.

- **Phase 5: Deployment and Maintenance (3 weeks)**

1. The team should deploy the system on cloud infrastructure, ensuring that the deployment is smooth and the system is set up for high availability and scalability.
2. The team should monitor the system in real-time to track performance, identify potential issues, and ensure uptime is maximised. Any issues discovered during monitoring should be addressed immediately to minimise downtime.
3. The team should provide end-user training sessions to ensure that users understand how to navigate and use the system effectively. The team should also create user-friendly documentation for reference.
4. The team should offer post-launch support, addressing any issues that arise after deployment and making necessary updates or improvements based on user feedback.

2.2 Gantt Chart



View the complete chart here:

https://www.figma.com/board/pm79HS1rgK4Wz3HvQCf9Uf/Library-Management-%7C-Wahiq-Iqbal-%26-Viney-Jain?node-id=1-1538&t=mSQSO1OlXKGjnLtF-1

2.3 Roles and Responsibilities

- **Project Manager**

1. The Project Manager will oversee the entire project lifecycle, ensuring all phases are executed according to the project plan, timeline, and budget.
2. The Project Manager will coordinate between different teams, manage resources, and ensure that stakeholders are regularly updated on progress and milestones.

3. The Project Manager will ensure that project goals are met, resolve any issues or roadblocks that arise, and facilitate communication across all involved parties.

- **System Analyst**

1. The System Analyst will define and document the system requirements, ensuring that all business and technical needs are captured accurately.
2. The System Analyst will map out workflows and business processes, identifying key interactions and dependencies within the system to ensure smooth integration.
3. The System Analyst will work closely with stakeholders to clarify any requirements, prioritise them, and ensure they are aligned with the project's goals.

- **UI/UX Designer**

1. The UI/UX Designer will develop user-friendly interfaces that prioritise ease of use, accessibility, and overall user experience, following system thinking principles.
2. The UI/UX Designer will create detailed mockups and wireframes for the system's user interfaces, ensuring they align with the product vision and are intuitive for users.

3. The UI/UX Designer will conduct usability tests with real users to gather feedback on the designs, iterate on these designs based on feedback, and ensure the final design meets user needs and expectations.

- **Developers**

1. The developers will implement the system's frontend and backend functionalities, ensuring that all features are developed according to the specifications provided in the SRS and design documents.
2. The Developers will focus on building a robust, scalable, and maintainable codebase, using best practices for code quality, performance, and security.
3. The Developers will ensure effective integration between different components of the system, such as the frontend, backend, APIs, and third-party services.

- **Testers**

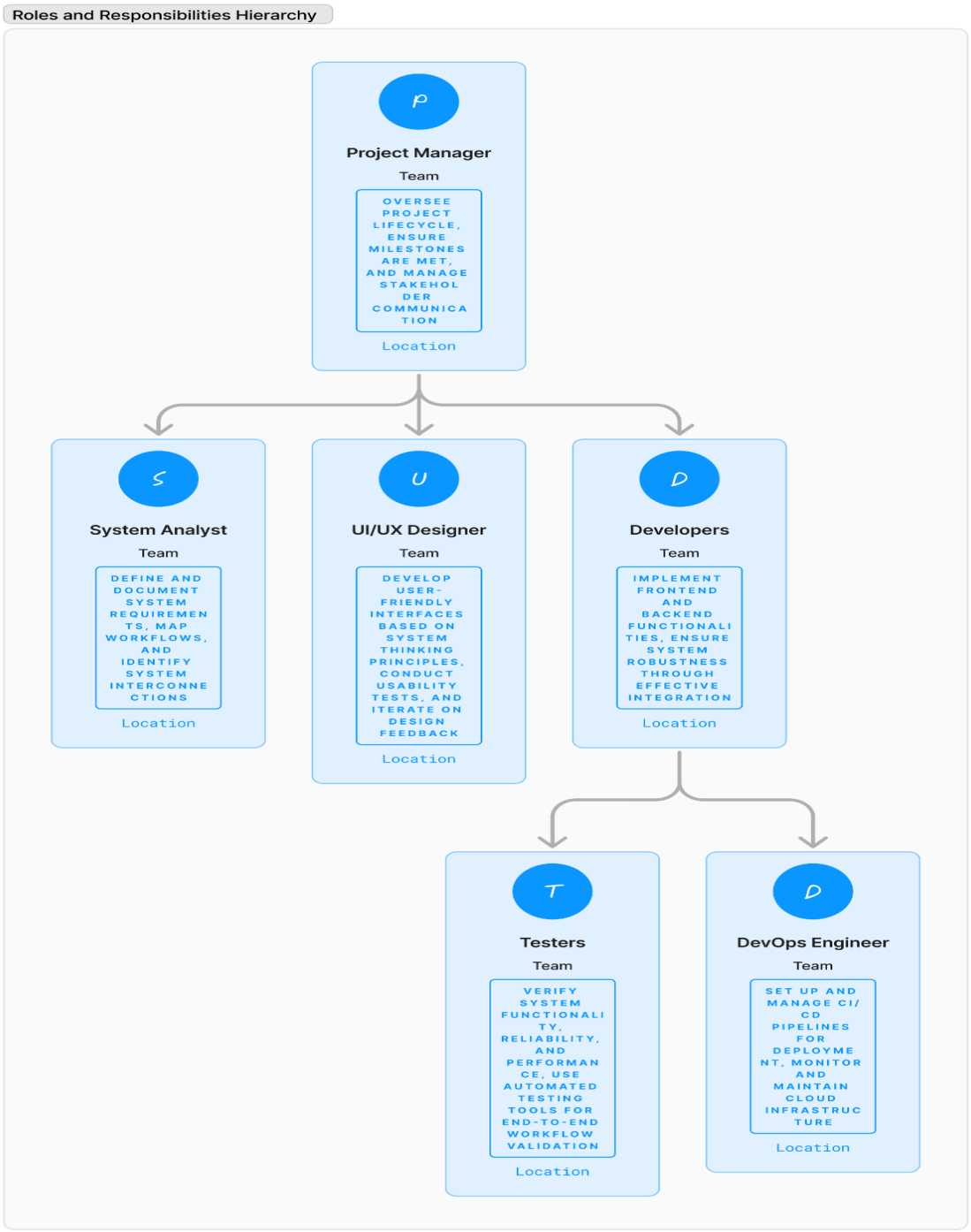
1. The Testers will verify the functionality, reliability, and performance of the system by running various types of tests, including functional, integration, and system tests.
2. The Testers will use automated testing tools to validate end-to-end workflows, ensuring that all components of the system work together as intended and that the system meets the defined requirements.

3. The Testers will document any bugs or issues discovered during testing and work closely with the development team to ensure timely resolution.

- **DevOps Engineer**

1. The DevOps Engineer will set up and manage continuous integration/continuous deployment (CI/CD) pipelines to streamline the process of building, testing, and deploying code.
2. The DevOps Engineer will ensure that the CI/CD pipelines are optimised for efficiency, reliability, and scalability, minimising downtime and manual intervention during deployment.

   The DevOps Engineer will monitor and maintain cloud infrastructure, ensuring the system runs smoothly in a scalable and secure environment and addressing any performance or availability issues as they arise.

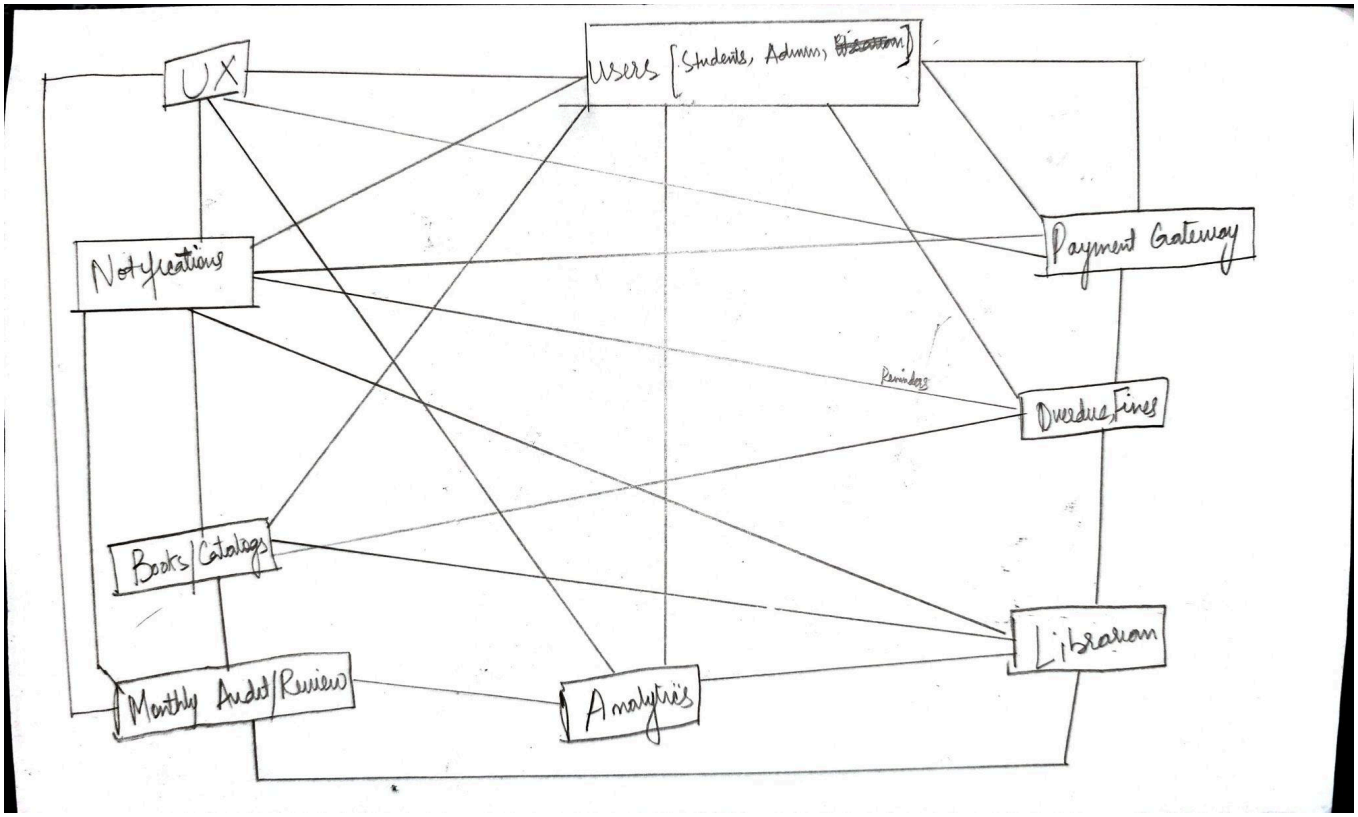


View the complete chart here:

https://www.figma.com/board/pm79HS1rgK4Wz3HvQCf9Uf/Library-Management-%7C-Wahiq-Iqbal-%26-Viney-Jain?node-id=1-1538&t=mSQSO1OlXKGjnLtF-1

3. Knowledge we tried to integrate from Workshops

Mesh Map of Interconnections



Systems Thinking Application

- **Elements**
  - **Resources**: The system will manage essential resources, including books, journals, users, transactions, and notifications.
    - Books and journals will be catalogued for easy access, either for borrowing or purchase, ensuring availability to users.

- User data will include personal details, account settings, transaction history, and activity tracking.
- Transaction records will track all user interactions with the system, including book checkouts, purchases, and returns, allowing for accurate record-keeping.
- Notifications will be used to send alerts and reminders to users and administrators about important events like due dates, new content, or system updates.

➢ **Interfaces**: The system will include various interfaces for both users and administrators.

- The **user portal** will allow users to search for and borrow books, manage their accounts, and view transaction history.
- The **admin dashboard** will provide system administrators with tools to manage content, monitor system performance, and track key metrics like user activity and book circulation.
- The **notification system** will enable real-time communication with users and admins about due dates, system updates, and other relevant events.

➢ **Supporting Systems**: The system will integrate with various supporting systems.

- **Payment gateways** will allow users to securely process payments for transactions such as book purchases, fines, or donations.
- **External library systems** will provide access to shared resources, enabling users to borrow from a larger catalogue of books and journals.
- **Digital platforms** will make digital versions of books and journals available to users, ensuring access across different devices.

❖ **Interconnections**

➢ **Real-time updates of book availability after transactions**: Whenever a user borrows or returns a book, the system will automatically update the availability status to ensure real-time accuracy for other users.

➢ **Notifications reduce overdue books and improve user compliance**: Timely notifications about upcoming due dates will encourage users to return books on time, reducing overdue items and improving overall user compliance.

➢ **Analytics inform acquisition decisions and system improvements**: Data gathered from user transactions, resource usage, and system

performance will be analysed to inform decisions regarding new book acquisitions, system optimisations, and user experience improvements.

- **Purpose**
  - **Efficient resource utilisation, user satisfaction, and streamlined operations**: The system will ensure that resources like books and journals are used efficiently, user satisfaction is maximised, and library operations are streamlined, creating a seamless experience for both users and administrators.
  - **Support academic excellence through reliable library services**: The system's goal will be to provide reliable and accessible library services, supporting academic excellence by ensuring that users have access to the resources they need to succeed.

SDLC Principles

- **Agile methodology ensures**:
  - **Frequent iterations and deliverables aligned with user feedback**: The development process will follow agile principles, with frequent iterations and regular feedback loops to ensure that the system meets the evolving needs of users.
  - **Continuous improvement of system functionality and usability**: The system will undergo continuous improvements to enhance functionality, fix bugs, and improve the overall user experience.
- **CI/CD pipeline facilitates**:
  - **Automated deployment for consistent updates**: The continuous integration and continuous deployment (CI/CD) pipeline will automate the deployment process, allowing for regular updates and consistent delivery of new features and bug fixes.
  - **Tools like Jenkins ensure efficient integration and delivery processes**: Tools like Jenkins will be used to streamline integration, automate testing, and ensure that new code is delivered efficiently and without disruption to users.

Final Deliverables

- **Fully functional software system accessible via web and mobile platforms**

  The system will be fully operational across both web and mobile platforms, ensuring users can access the library's resources from anywhere, on any device. This will include all core functionalities, such as browsing, borrowing, and managing accounts, with a seamless experience across platforms.

- **Comprehensive user manuals for administrators, librarians, and general users**

  The project will include detailed user manuals tailored to different user groups.

  - **Administrators** will have documentation outlining how to manage the system, configure settings, handle user issues, and analyze reports.
  - **Librarians** will have guides on managing book checkouts, processing returns, and maintaining the catalogue.
  - **General users** will receive instructions on how to navigate the portal, search for resources, borrow items, and manage their accounts.

- **API documentation for third-party integrations**

  API documentation will be provided to allow third-party developers to integrate external services with the system, such as payment gateways, digital platforms, or analytics tools. The documentation will cover endpoint details, data formats, authentication, and examples for ease of integration.

- **Analytics dashboard for decision-making**

  The system will include an analytics dashboard to help administrators make data-driven decisions. This dashboard will track key metrics like book circulation, user activity, overdue items, payment processing, and resource utilisation, providing actionable insights for system improvements and resource management.

- **Post-deployment monitoring and a post deployment support**

  Post-launch, the system will undergo continuous monitoring to ensure stability, performance, and user satisfaction. The post deployment support will cover troubleshooting, updates, and necessary adjustments, ensuring the system remains fully operational after deployment.

- **Training sessions for library staff**

  Comprehensive training will be provided to library staff, ensuring they understand how to use the system effectively, handle user queries, and

manage daily operations. This training will also cover troubleshooting, system updates, and new feature rollouts.

Anticipated Emergent Properties

- **Usage Trends**

  The system will track usage trends, such as peak borrowing times, most popular genres, and patterns in resource access. This data will help the library make informed decisions about resource allocation, allowing for optimised book acquisitions and improved management of library resources based on user behaviour.

- **Behavioural Insights**

  Notifications and overdue fine management will influence user habits, encouraging timely returns and reducing the number of late items. By providing users with reminders about due dates and implementing effective fine management strategies, the system will foster a culture of timely returns and overall better compliance with library rules.

- **Scalability Needs**

  As the system grows, there may be an increase in the user base or the size of

the collection, which will necessitate modular expansions. This could include features like:

- The integration of **eBooks** to cater to digital reading preferences allows users to access books remotely.
- The addition of **inter-library systems** to enable seamless borrowing across different library networks, increasing resource accessibility for users.

Thank you 🙂