Project Name: Library Management System (LMS)

➢ Description:

The Library Management System (LMS) is a comprehensive software solution designed to streamline and automate the management of libraries. This system provides librarians and library staff with powerful tools to efficiently manage library resources, patrons, and administrative tasks. With intuitive user interfaces and robust features, LMS simplifies library operations, enhances patron experiences, and ensures the smooth functioning of library services.

➢ Scope:

User management: Enable user registration and login with differentiated access levels.

Book management: Facilitate adding, editing, and deleting books with basic information.

Borrowing system: Allow users to borrow and return books, with notifications for overdue items.

Admin functions: Provide administrators with capabilities to manage users, view overdue books, and generate basic reports.

Help documentation: Include resources to assist users in navigating and using the system effectively.

➢ Functional requirements:

- User Registration and Login: Allow users to register for an account and log in using their email address and password.
- Search Books: Provide a simple search feature that allows users to search for books by title or author.
- View Book Details: Display basic details about each book, such as title, author, and availability status.
- Borrow Books: Allow users to borrow books by selecting them from the catalog and confirming the transaction.
- Return Books: Enable users to return borrowed books by marking them as returned in their account.
- View Borrowing History: Allow users to view their borrowing history, including the books they've borrowed and their due dates.
- Receive Notifications: Send email notifications to users when their borrowed books are nearing their due dates.
- Admin Login: Provide a separate login for administrators to access the admin dashboard.
- Add Books: Allow administrators to add new books to the catalog by entering basic details manually.
- Edit Books: Enable administrators to edit existing book details, such as title, author, and availability status.
- Delete Books: Allow administrators to delete books from the catalog if necessary.
- Manage Users: Enable administrators to view and manage user accounts, including resetting passwords and deactivating accounts.
- View Overdue Books: Provide administrators with a list of overdue books and their respective borrowers.
- Generate Reports: Allow administrators to generate basic reports, such as the total number of books borrowed each month.
- Help Documentation: Include a help section or FAQ page to provide users with assistance on using the system and resolving common issues.

➢ Non-Functional requirements:

- Performance: Ensure the system responds quickly to user interactions, with minimal loading times for pages and actions.
- Accessibility: Ensure the system is accessible to users with disabilities, following WCAG guidelines for color contrast, keyboard navigation, and screen reader compatibility.
- Usability: Design the user interface to be intuitive and easy to navigate, with clear labels and instructions.
- Reliability: Ensure the system is stable and does not experience frequent crashes or downtime.
- Security: Implement basic security measures such as encryption of sensitive data, secure password storage, and protection against common vulnerabilities like SQL injection and cross-site scripting (XSS).
- Scalability: Design the system to handle a growing number of users and library resources without performance degradation.
- Compatibility: Ensure the system is compatible with a variety of web browsers and devices, including desktops, laptops, tablets, and smartphones.
- Data Backup: Implement regular backups of the database to prevent data loss in case of system failure or corruption.
- Error Handling: Provide informative error messages to users when they encounter issues, guiding them on how to resolve the problem or contact support.
- Documentation: Document the system architecture, codebase, and configuration settings to facilitate maintenance and troubleshooting.
- Logging: Implement logging of system activities and errors to aid in debugging and auditing.
- Performance Monitoring: Set up monitoring tools to track system performance metrics such as response time, server load, and database queries.
- User Feedback: Collect feedback from users to identify areas for improvement and prioritize feature enhancements.
- Backup and Restore: Provide a mechanism for administrators to perform backups and restores of the system data.
- Version Control: Utilize version control systems like Git to manage changes to the codebase and collaborate with other developers effectively.