**functional requirements:**

1. **User Authentication:**

User initiates registration by providing details and system validates.

User logs in using credentials or third-party accounts.

User resets password if forgotten.

1. **User Actions:**

Students enrolled in courses, access content, and track progress.

Teacher manages assignments, materials, quizzes, and courses.

1. **Platform Features:**

User upload, downloads, and manages learning materials.

Users set preferences for language, accessibility, and appearance.

1. **Course Management:**

Teacher generates enrollment codes and manages courses.

Student self-enrolls using provided codes.

1. **User Settings:**

User updates profile, manages preferences, and data.

1. **Feedback and Assessment:**

Users provide feedback, ratings, and engage in assessments.

1. **Mobile App and Accessibility:**

Users access the platform via mobile app with optimized design.

User utilizes accessibility features.

1. **Manager Functions:**

Manager manages users, courses, platform, and configurations.

1. **Reports and Analytics:**

Manager reviews reports and analytics for decision-making.

1. **User Support and Maintenance:**

Manager supports users and oversees platform maintenance.

**non-functional**:

**U**sability:

Accessibility: Ensure compliance with accessibility standards (e.g., WCAG) to make the LMS usable for individuals with disabilities.

User Experience (UX): Define guidelines for the overall user experience, such as intuitive navigation, consistent layouts, and responsive design.

**P**erformance: To ensure speed and Efficiency, the actual system will be built with low-resources servers in mind and the front-end will be made with platform specific technology to ensure efficiency. By relying on scalable hosting services like AWS, or ORACLE to scale for more customers. and use a microservices architecture to enable scalability, near 100% uptime, and easy integration of new features.

**R**eliability: Reliability is concerned with ensuring our services goes down as little as possible, this can be done by ensuring our code robustness and how it deals with errors and using high quality services to manage the servers.