EDUSynergy: Comprehensive Digital Learning Ecosystem

Document Version: 1.0
Date: 2025-04-19
Prepared by: Technical Team
Project Owner: momaws232

1. Introduction

1.1 Purpose

EDUSynergy is a comprehensive digital learning ecosystem designed to unify and streamline online education, collaborative project management, and AI-driven personalized learning. This Software Requirements Specification (SRS) document captures the functional and non-functional requirements for EDUSynergyThe primary focus of this document is to detail the requirements for core modules including user management, course management, collaborative workspaces, personalized recommendations, virtual classroom integration, and analytics. Future versions may extend into additional features and integrations as identified by stakeholder needs.

1.2 Document Conventions // iEEE

1.2.1 Priority Conventions

In this document, requirement priorities are indicated as **High**, **Medium**, or **Low**. These priorities are explicitly stated mainly in **Sections 3** Numbered as (`REQ-X.Y`)**, 4 and 5** Numbered by category (`ABC-X`) e.g.(`PRF-1`) where detailed feature requirements are described. Higher-level summaries elsewhere inherit the priorities of their referenced detailed requirement

1.2.2 Formatting Conventions

- **All user roles** are written in *italics*
- **Technical terms defined in the Glossary** are written in ***bold and italic***
- **Other significant terms** are presented in **bold**

1.3 Intended Audience and Reading Suggestions

This document is intended for:

- **Stakeholders and Clients:** For understanding the overall vision, scope, and key capabilities Focus on Sections 1 (Introduction) and 2 (Overall Description).
- **Project Managers:** For planning and scope definition (see sections 1, 2, and 6).
- **Development Team:** For technical implementation details (see sections 3–5).
- **QA Team:** For testable system requirements (see sections 3–5).
- **UX/UI Designers:** For user interface requirements (see section 4.1).
- **DevOps Engineers:** For cloud architecture and deployment See Sections 2.4 (Operating Environment) and 5.4 (Quality Attributes)

Readers are advised to start with the Introduction and Overall Description (sections 1 and 2), then proceed to the requirements sections most relevant to their specific role.

1.4 Product Scope

EDUSynergy is developed to revolutionize digital learning by providing a unified platform that integrates:

- **AI-powered personalized learning experiences**,
- **Collaborative project management tools** for real-world skill development,
- **Virtual classroom environments** with video conferencing integration,
- **Comprehensive analytics** for tracking student and institutional progress,
- **Role-specific dashboards** for *students*, *instructors*, and *administrators*,
- **Robust security and compliance** for data management across all platform operations.

The initial release targets deployment in educational institutions and corporate training environments, supporting traditional courses, project-based learning, and self-paced professional development. The goal is to enhance engagement, streamline administrative tasks, and ensure data security and compliance, resulting in improved learning outcomes and operational efficiency.

1.5 References

The following standards, frameworks, and technical documents have been referenced in the preparation of this SRS:

- **IEEE Std 830-1998**

  IEEE Recommended Practice for Software Requirements Specifications
- **W3C Web Content Accessibility Guidelines (WCAG) 2.1**

  International standards for web accessibility
- **General Data Protection Regulation (GDPR)**

  EU regulation on data protection and privacy
- **Family Educational Rights and Privacy Act (FERPA)**

  US law governing access to educational information
- **Health Insurance Portability and Accountability Act (HIPAA)**

  US legislation for healthcare data security
- **Learning Tools Interoperability (LTI) v1.3 Specification**

  Standard for integrating educational applications
- **Experience API (xAPI) Specification v1.0.3**

  eLearning data tracking standard
- **AWS Well-Architected Framework, 2024**

  Cloud infrastructure best practices
- **React.js Technical Documentation v18.0**

  Frontend JavaScript library documentation
- **Django Framework Documentation v4.2**

  Backend web framework documentation

2. Overall Description

///////////////// describe

2.1 Product Perspective

EDUSynergy is a standalone system designed as a comprehensive digital learning ecosystem. It is built to integrate seamlessly with external systems such as video conferencing platforms, calendar applications, and learning content providers through standardized APIs. The system leverages a **microservices architecture** to ensure scalability, maintainability, and robust integration capabilities.

Key contextual points include:

- **System Context:** EDUSynergy operates independently but can integrate with third-party services (e.g., Zoom, Google Calendar) and external learning platforms (e.g., LinkedIn Learning, Coursera).
- **External Interfaces:** The system interacts with APIs for video conferencing, authentication (OAuth 2.0, SAML), and learning tool interoperability (LTI 1.3).
- **Dependencies:** The platform relies on cloud infrastructure (AWS/Azure), modern browsers, and stable internet connectivity for optimal performance.
- **Architecture:** The microservices-based architecture ensures independent scaling and deployment of modules, supported by Kubernetes orchestration and a global CDN.
- **Integration:** EDUSynergy supports external integrations for calendar synchronization, video conferencing, and skill gap analysis, enhancing the user experience and operational workflow.

2.2 Product Features

1. **User Management:**
    - Role-based dashboards for *students*, *instructors*, and *administrators*.
    - Authentication via OAuth 2.0, SAML SSO, and multi-factor authentication (MFA).
    - Role-specific dashboards and permissions
    - User profile management
2. **Course Management:**
    - Course creation, scheduling, and catalog management

- Module organization and content distribution
- Progress tracking and adaptive content release.

3. **EDU Recommender (CLRS):**
    - AI-driven course recommendations leveraging collaborative and content-based filtering.
    - Personalized learning paths and skill gap analysis.
    - Personalized UI adaptations based on user behavior

4. **Collaborative Workspaces:**
    - Real-time project management tools with Kanban boards and task tracking.
    - Team formation and communication tools
    - Document collaboration and version control

5. **Virtual Classroom Hub:**
    - Integrated video conferencing (Zoom, Google Meet), session recording, and interactive tools (polls, whiteboards).
    - Accessibility features like real-time captions and keyboard navigation, Multilingual support
    -

6. **Assessment and Grading**
    - Various assessment types (quizzes, assignments, projects)
    - Automated and manual grading with rubrics
    - Performance analytics and feedback mechanisms

7. **Reporting and Analytics:**
    - Customizable dashboards for progress tracking and institutional reporting.
    - Exportable reports in multiple formats (PDF, CSV, Excel).
    - Compliance and accreditation reporting

8. **Security and Compliance:**

    - End-to-end encryption (AES-256, TLS 1.3) and role-based access control (RBAC).
    - GDPR, FERPA, and HIPAA compliance with comprehensive audit logging.

9. **Scalable Cloud Infrastructure:**
    - Auto-scaling microservices architecture with Kubernetes orchestration.
    - High availability through global CDN and multi-region redundancy.

2.3 User Classes and Characteristics

The EDUSynergy platform is designed to serve multiple user classes, each with distinct characteristics, technical proficiency, and usage patterns. Below is a detailed description of the primary user roles:

2.3.1 *Students/Learners*

- **Characteristics:**
  Students and learners include individuals ranging from K-12 *students* to adult learners in corporate training environments.
- **Technical Proficiency:**
  Varying levels, from basic computer literacy to advanced technical skills.
- **Usage Frequency:**
  Daily to weekly, depending on academic or training schedules.
- **Primary Functions:**
    - Enrolling in courses.
    - Accessing and consuming learning materials.
    - Collaborating on projects and assignments.
    - Completing assessments and receiving feedback.

2.3.2 *Instructors/Teachers*

- **Characteristics:**
  Includes educators, corporate trainers, and subject matter experts responsible for course delivery.
- **Technical Proficiency:**
  Moderate to high, with familiarity in creating and managing digital content.

- **Usage Frequency:**
  Daily, to manage courses, evaluate student progress, and provide feedback.
- **Primary Functions:**
  - Creating and organizing courses and modules.
  - Monitoring student performance using analytics.
  - Providing feedback and grading assessments.

2.3.3 *Administrators*

- **Characteristics:**
  Includes IT *administrators*, institutional directors, and training managers overseeing platform operations.
- **Technical Proficiency:**
  High, with experience in managing user roles, system configurations, and compliance standards.
- **Usage Frequency:**
  Daily to weekly, depending on operational demands.
- **Primary Functions:**
  - Managing user roles and permissions.
  - Configuring the platform for institutional needs.
  - Monitoring audit logs and ensuring regulatory compliance.

2.3.4 *Content Creators*

- **Characteristics:**
  Instructional designers and subject matter experts responsible for developing learning materials.
- **Technical Proficiency:**
  Moderate to high, with skills in creating engaging, interactive content.
- **Usage Frequency:**
  Weekly, based on content development cycles.
- **Primary Functions:**
  - Designing course content, including multimedia elements.

    - Creating assessments and learning activities.
    - Ensuring content accessibility and alignment with learning objectives.

2.3.5 *System Integrators*

- **Characteristics:**
  IT professionals and developers involved in integrating EDUSynergy with external systems.
- **Technical Proficiency:**
  Very high, with expertise in APIs, system architecture, and custom development.
- **Usage Frequency:**
  Occasional, depending on integration or customization projects.
- **Primary Functions:**
    - Integrating external tools (e.g., video conferencing, learning platforms).
    - Customizing modules or workflows for institutions.
    - Troubleshooting and maintaining system interoperability.

2.4 Operating Environment

EDUSynergy is designed to operate efficiently across various environments, ensuring compatibility, scalability, and accessibility for a diverse range of users. Below are the key aspects of the operating environment:

2.4.1 Web Application

- **Supported Browsers:**
  The platform SHALL support modern browsers including:
    - Google Chrome (latest 2 major versions)
    - Mozilla Firefox (latest 2 major versions)
    - Safari (latest 2 major versions)
    - Microsoft Edge (latest 2 major versions)
- **Technologies Used:**
    - HTML5, CSS3, and JavaScript ES6+ for responsive design.
- **Device Compatibility:**

    - Desktop, tablet, and mobile devices with a minimum resolution of 320px (mobile) to 4K displays.
- **Performance Requirements:**
    - Optimized for responsive layouts and real-time functionalities.

2.4.2 Mobile Application

- **Supported Platforms:**
    - iOS 15.0+
    - Android 11.0+
- **Features:**
    - Native application experience with offline capabilities.
    - Push notification support for real-time updates.
- **Optimization:**
    - Battery and resource-efficient, ensuring smooth performance on low-end devices.

2.4.3 Server Environment

- **Deployment:**
    - Cloud-based deployment on AWS or Azure.
- **Orchestration:**
    - Kubernetes for container management, ensuring scalability and fault tolerance.
- **Architecture:**
    - Microservices-based to support independent scaling and maintainability.

2.4.4 Database Systems

- **Relational Database:**
    - PostgreSQL v14+ for structured data.

- **Caching:**
  - Redis for real-time features and caching.
- **User Interaction Data:**
  - MongoDB for storing and processing user interaction data for AI/ML modules.

2.4.5 Network Requirements

- **Bandwidth:**
  - Minimum bandwidth: 1 Mbps for basic functions.
  - Recommended bandwidth: 5+ Mbps for video conferencing and collaborative tools.
- **Protocols:**
  - WebSocket support for real-time collaboration and updates.

2.5 Design and Implementation Constraints // deadline and budget hardware and software

The design and development of EDUSynergy are subject to several constraints to ensure compliance with technical, organizational, and regulatory standards. Below are the key design and implementation constraints:

2.5.1 Technology Stack

- **Frontend:**
  - The system SHALL be implemented using React.js with Material Design UI for consistency and responsiveness.
  - Redux SHALL be used for state management across the application.
- **Backend:**
  - Python/Django SHALL be used for RESTful APIs.
  - Node.js SHALL handle real-time collaboration features.
- **Database:**
  - PostgreSQL SHALL be used for relational data storage.

- Redis SHALL be used for caching and real-time functionality.
- MongoDB SHALL store user interaction data for AI/ML modules.

2.5.2 Accessibility and Internationalization

- The system SHALL comply with **WCAG 2.1 Level AA** accessibility standards.
- The system SHALL support internationalization (i18n) and localization, including right-to-left (RTL) languages.

2.5.3 Security and Authentication

- **Authentication Standards:**
  - OAuth 2.0 SHALL be used for third-party authentication.
  - JWT SHALL be used for secure session management.
  - SAML 2.0 SHALL support enterprise single sign-on (SSO).
- **Encryption:**
  - All data in transit SHALL use TLS 1.3, and sensitive data at rest SHALL use AES-256 encryption.

2.5.4 Development and Deployment

- **Coding Standards:**
  - All code SHALL adhere to organizational coding best practices and guidelines.
- **Containerization:**
  - All deployments SHALL be containerized using Docker to ensure consistency across environments.
- **DevOps Practices:**
  - CI/CD pipelines SHALL be used for automated testing and deployment.
- **Scalability:**
  - Kubernetes SHALL be used for container orchestration, allowing independent scaling of microservices.

2.5.5 API Standards

- APIs SHALL follow RESTful design principles and include standardized error handling.
- OpenAPI/Swagger documentation SHALL be provided for all exposed APIs.

2.5.6 Regulatory Compliance

- The system SHALL comply with GDPR, FERPA, and HIPAA for handling sensitive data.

2.6 User Documentation

The EDUSynergy platform SHALL include comprehensive user documentation to assist all user roles in understanding and utilizing the system effectively. The documentation covers various aspects tailored to the needs of *students*, *instructors*, *administrators*, *content creators*, and *system integrators*.

2.6.1 Categories of Documentation

- **Interactive Tutorials:**
    - Step-by-step video tutorials for onboarding and feature demonstrations.
- **Knowledge Base:**
    - A searchable repository of articles with detailed guides, FAQs, and troubleshooting tips.
- **Context-Sensitive Help:**
    - Inline tooltips and help sections accessible within the platform for quick guidance.
- **Role-Specific Guides:**
    - User manuals segmented by role, focusing on core tasks and responsibilities.
- **Technical Documentation:**
    - API references, system configuration guides, and integration manuals for developers and IT teams.
- **Implementation Guides:**
    - Detailed instructions for institutional integration, including setup, configuration, and customization to align with organizational needs.

- **Security Best Practices Guide:**
    - Documentation outlining security protocols, safe usage tips, and compliance requirements for all users and *administrators*.

2.6.2 Accessibility and Availability

- The documentation SHALL be accessible directly from the platform and available in multiple formats, including PDF and web-based versions.
- It SHALL support multilingual content to cater to users in different regions.

This documentation ensures that all users can effectively utilize EDUSynergy, regardless of their technical proficiency, enhancing overall user satisfaction and system adoption.

2.7 Assumptions and Dependencies

The development and deployment of EDUSynergy rely on several assumptions and external dependencies. These factors are critical for the system's functionality and success.

2.7.1 Assumptions

- **User Proficiency:**
    - Users possess basic computer literacy appropriate to their role (e.g., navigating a web application, using video conferencing tools).
- **Internet Connectivity:**
    - Educational institutions and corporate clients have stable and sufficient internet bandwidth to support video conferencing and real-time collaboration.
- **Device Compatibility:**
    - Users have access to compatible devices (e.g., modern browsers, smartphones, tablets, or laptops).
- **Content Ownership:**
    - *Content creators* have the necessary rights to upload and share materials on the platform.

2.7.2 Dependencies

- **Third-Party APIs:**
  - The stability and availability of APIs for video conferencing (e.g., Zoom, Google Meet), calendar synchronization, and external learning platforms (e.g., LinkedIn Learning).
- **Cloud Services:**
  - Uptime and performance of cloud infrastructure providers (AWS, Azure).
- **Network Infrastructure:**
  - Adequate institutional network infrastructure to support concurrent user access and system performance.
- **Browser Compatibility:**
  - Continued support for required features in modern browsers (e.g., WebSocket, HTML5).
- **Regulatory Consistency:**
  - Legal and compliance requirements (e.g., GDPR, FERPA, HIPAA) remain stable during the development lifecycle.

These assumptions and dependencies ensure the successful implementation and operation of EDUSynergy while highlighting areas of risk that require monitoring and mitigation.

3 . System Features /// for user req describe in detail the system req propriety of each one

3.1 User Management and Authentication

3.1.1 Description and Priority

This module handles user registration, authentication, authorization, and profile management. It ensures secure access to the platform through role-based controls and compliance with data protection standards.

Priority: High and Medium for platform security and user accountability.

3.1.2 Stimulus/Response Sequences

1. **User Registration Flow**

- *Stimulus:* User initiates registration
  Response: System displays form with email, password, and profile detail fields
- *Stimulus:* User submits valid registration form
  Response: System sends verification email and displays pending confirmation message

1.1 *Alternative Flow:*

- *Stimulus:* User submits invalid email format
  Response: System displays error message without sending verification

2. **Email Verification**

- *Stimulus:* User clicks verification link
  Response: System activates account and redirects to login page

3. **Login Authentication**

- *Stimulus:* User logs in via OAuth/SAML/MFA
  Response: System validates credentials and grants role-based access

4. **Role Management**

- *Stimulus:* Admin creates custom role
  Response: System updates RBAC policies and notifies affected users

5. **Profile Updates**

- *Stimulus:* User changes privacy settings
  Response: System saves preferences and syncs across active sessions

6. **Cross-Device Sync**

- *Stimulus:* User logs in from new device
  Response: System applies stored preferences and sends security notification

3.1.3 Functional Requirements

High Priority:
REQ-1.1: The system shall support user registration with email verification and bulk CSV imports.
REQ-1.2: The system shall implement OAuth 2.0, SAML 2.0, and MFA for authentication.
REQ-1.3: The system shall enforce role-based access control (RBAC) with customizable permissions.
REQ-1.4: The system shall allow *administrators* to create custom roles and hierarchical access policies.
REQ-1.5: The system shall provide user profile management with privacy controls.

Medium Priority:
REQ-1.6: The system should track user preferences and sync settings across devices.

3.2 Course Management

3.2.1 Description and Priority

This module enables course creation, content organization, enrollment management, and adaptive content delivery.
Priority: High and Medium for educational content delivery and learner progression.

3.2.2 Stimulus/Response Sequences

1. **Course Creation**
   - *Stimulus:* Instructor creates course structure
     Response: System saves modules/lessons and updates catalog
2. **Enrollment Process**

- *Stimulus:* User requests course enrollment
  Response: System checks prerequisites and either:
  - Enrolls user (if met)
  - Waitlists user (if unmet)

3. **Progress Tracking**
   - *Stimulus:* User completes lesson
     Response: System records completion time and updates progress metrics
4. **Content Unlocking**
   - *Stimulus:* User finishes prerequisite course
     Response: System enables access to dependent content

3.2.3 Functional Requirements

High Priority:
REQ-2.1: The system shall provide course templates and hierarchical content structuring (modules/lessons).
REQ-2.2: The system shall allow self-enrollment, admin enrollment, and waitlisting.
REQ-2.3: The system shall enforce prerequisites for course enrollment.
REQ-2.4: The system shall track student progress and time-on-task metrics.
REQ-2.5: The system shall support adaptive content release based on prerequisites or schedules.

Medium Priority:
REQ-2.6: The system should enable offline access to downloaded course materials.

3.3 EDU Recommender (CLRS)

3.3.1 Description and Priority

The EDU Recommender uses AI/ML to provide personalized course suggestions based on user behavior and skill gaps.
Priority: High and Medium for enhancing learner engagement.

3.3.2 Stimulus/Response Sequences

The recommendation engine interface displays personalized course suggestions with "Why Recommended" tooltips. Each recommendation card shows completion percentage and skill relevance indicators.

1. **Behavior Tracking**
   - *Stimulus:* User browses/completes course
     Response: System logs interactions and updates recommendation model
2. **Recommendation Request**
   - *Stimulus:* User views suggestions
     Response: System displays personalized courses with justification tags
3. **Feedback Incorporation**
   - *Stimulus:* User rates recommendation
     Response: System adjusts future suggestions and confirms update

3.3.3 Functional Requirements

High Priority:
REQ-3.1: The system shall analyze user interaction data for recommendation training.
REQ-3.2: The system shall combine collaborative and content-based filtering algorithms.
REQ-3.3: The system shall display recommendations with contextual explanations.
REQ-3.4: The system shall adjust suggestions based on explicit/implicit feedback.

Medium Priority:

REQ-3.5: The system should allow users to opt out of personalized recommendations.

3.4 Collaborative Workspaces

3.4.1 Description and Priority

This feature provides tools for team-based project management, real-time document collaboration, and communication.

Priority: High and Medium for fostering teamwork and practical skill development.

3.4.2 Stimulus/Response Sequences

The virtual classroom interface shows participant video feeds, interactive controls, and real-time captions. Session tools appear in a collapsible sidebar.

1. **Project Board Creation**

- o *Stimulus:* User creates new project board

  Response: System initializes:

- ▪ Default task columns (To Do/In Progress/Done)
- ▪ Empty contributor list
- ▪ Template selection prompt

2. **Real-Time Editing**

- o *Stimulus:* User edits shared document

  Response: System:

- ▪ Locks document section
- ▪ Broadcasts changes to collaborators
- ▪ Updates version history

2.1 *Conflict Scenario:*

- *Stimulus:* Two users edit same section simultaneously
  Response: System creates conflict resolution interface with change comparison

3. **Deadline Management**

- *Stimulus:* Project deadline approaches (within 24h)
  Response: System:
  - Notifies all team members
  - Updates integrated calendars
  - Highlights overdue tasks

4. **Task Completion**

- *Stimulus:* Team marks task complete
  Response: System:
  - Logs contributor metrics
  - Updates progress analytics
  - Triggers dependent tasks

3.4.3 Functional Requirements

High Priority:

REQ-4.1: The system shall provide Kanban boards with task dependencies.

REQ-4.2: The system shall enable real-time document editing and version control.

REQ-4.3: The system shall integrate with Google Calendar/Microsoft Outlook.

REQ-4.4: The system shall generate team contribution reports.

Medium Priority:

REQ-4.5: The system should support threaded discussions with @mentions.

3.5 Virtual Classroom Hub

3.5.1 Description and Priority

This module integrates live video conferencing with interactive tools for synchronous learning.
Priority: High and Medium for immersive virtual instruction.

3.5.2 Stimulus/Response Sequences

1. **Session Scheduling**
   - *Stimulus:* Instructor schedules class
     Response: System:
     - Creates calendar events
     - Sends participant notifications
     - Reserves virtual room
2. **Session Join Flow**
   - *Stimulus:* User joins session
     Response: System:
     - Verifies enrollment
     - Launches video interface
     - Applies accessibility settings
3. **Interactive Features**
   - *Stimulus:* Instructor starts poll
     Response: System:
     - Distributes poll to participants
     - Tracks responses in real-time
     - Generates instant analytics
4. **Bandwidth Adaptation**

- *Stimulus:* Low bandwidth detected
  Response: System:
  - Reduces video resolution
  - Prioritizes audio stream
  - Disables non-essential features

3.5.3 Functional Requirements

High Priority:

REQ-5.1: The system shall support Zoom/Google Meet integration for live sessions.
REQ-5.2: The system shall generate real-time captions and multilingual subtitles.
REQ-5.3: The system shall track attendance and participation metrics.
REQ-5.4: The system shall provide breakout rooms and interactive polls.
REQ-5.5: The system shall optimize video quality for low-bandwidth users.

3.6 Assessment and Grading

3.6.1 Description and Priority

Handles creation, delivery, and evaluation of assessments.
Priority: High for academic integrity.

3.6.2 Stimulus/Response Sequences

The grading interface displays submission files on the left and rubric tools on the right. Annotations can be added directly to student work.

1. **Assessment Creation**

- *Stimulus:* Instructor builds quiz
  Response: System:
  - Validates question formats

- Sets time limits
- Publishes to target cohort

2. **Submission Handling**

- *Stimulus:* Time limit expires

  Response: System:

- Auto-submits attempt
- Runs automated grading
- Flags suspicious activity

3. **Rubric Application**

- *Stimulus:* Instructor grades essay

  Response: System:

- Calculates weighted scores
- Stores annotated feedback
- Updates gradebook

4. **Result Display**

- *Stimulus:* Student views grades

  Response: System shows:

- Comparative analytics
- Instructor comments
- Improvement resources

3.6.3 Functional Requirements

High Priority:

REQ-6.1: The system shall support multiple question types (e.g., essays, coding).

REQ-6.2: The system shall automate grading for objective assessments.

REQ-6.3: The system shall enforce time limits and proctoring controls.

REQ-6.4: The system shall allow rubric-based manual grading.

REQ-6.5: The system shall generate grade reports and analytics.

3.7 Reporting and Analytics

3.7.1 Description and Priority

Provides institutional insights and compliance tracking.

Priority: High/Medium for data-driven decisions.

3.7.2 Stimulus/Response Sequences

1. **Dashboard Access**
   - *Stimulus:* Admin logs in

     Response: System displays:
     - Real-time enrollment metrics
     - Compliance status indicators
     - Custom visualization widgets
2. **Report Generation**
   - *Stimulus:* Instructor exports data

     Response: System:
     - Applies role-based filters
     - Formats per selected template
     - Delivers download link
3. **Risk Detection**
   - *Stimulus:* Student misses deadlines

     Response: System:
     - Triggers alert rules
     - Suggests interventions
     - Updates predictive models

3.7.3 Functional Requirements

High Priority:

REQ-7.1: The system shall display role-specific dashboards with KPIs.

REQ-7.2: The system shall generate accreditation/compliance reports.

REQ-7.3: The system shall export reports in PDF, CSV, and Excel formats.

Medium Priority:

REQ-7.4: The system should identify at-risk *students* using configurable metrics.

3.8 Security and Compliance Management

3.8.1 Description and Priority

Ensures data protection and regulatory adherence.

Priority: High for sensitive information.

3.8.2 Stimulus/Response Sequences

The audit dashboard displays real-time security events in a timeline view. Each log entry shows actor, action, and compliance status.

1. **Data Handling**

- o *Stimulus:* User submits PII

  Response: System:

- ▪ Encrypts with AES-256
- ▪ Logs access attempts
- ▪ Confirms secure storage

2. **File Uploads**

- o *Stimulus:* New file received

  Response: System:

- Scans for malware
- Validates file type
- Stores in quarantined area

3. **Audit Preparation**

- *Stimulus:* Audit scheduled

  Response: System:

- Compiles access logs
- Generates compliance matrix
- Flags anomalies

3.8.3 Functional Requirements

High Priority:

REQ-8.1: The system shall encrypt data at rest (AES-256) and in transit (TLS 1.3).

REQ-8.2: The system shall maintain audit logs for all security actions.

REQ-8.3: The system shall comply with GDPR, FERPA, and HIPAA.

4. External Interface Requirements

4.1 User Interfaces

The system shall provide intuitive, role-specific interfaces for all user classes (*students, instructors, administrators, content creators, and system integrators*). All interfaces will maintain a consistent layout and adhere to Material Design principles and WCAG 2.1 AA accessibility standards.

Student Interface

- The student interface will feature a dashboard displaying enrolled courses, deadlines, and personalized recommendations. *Students* will be able to navigate courses using module progression indicators and will see visual cues for completion status. The interface will support multiple content types (text, video, interactive components) with suitable viewers.

Instructor Interface

- *Instructors* will have a dashboard showing pending evaluations and student analytics. Collaboration tools such as Kanban-style task boards with drag-and-drop, real-time collaborative editing, and team member availability status will be available.

Administrator Interface

- *Administrators* will access dashboards with system metrics and compliance alerts. They will have management screens for user roles and content oversight.

Content Creator Interface

- *Content creators* will use intuitive navigation to build and organize learning modules. The interface will support uploading and integrating various content types and display module completion statuses.

System Integrator Interface

- *System integrators* will access configuration dashboards for API integrations, authentication providers, and external services.

Accessibility

- All user interfaces will comply with WCAG 2.1 Level AA standards, support keyboard navigation and screen readers (JAWS, NVDA, VoiceOver), maintain a minimum color contrast ratio of 4.5:1, and allow text resizing without loss of functionality.

4.2 Hardware Interfaces

Our system interface with various hardware components to optimize performance across devices.

- The system will support camera and microphone access for video conferencing and provide touch interface compatibility for mobile/tablet devices.
- Performance will be optimized for low-end devices with fallback mechanisms for those lacking camera/microphone capabilities.
- Special measures will minimize battery consumption on mobile devices.

4.3 Software Interfaces

The platform will integrate with third-party services and platforms through standardized APIs.

- **Video Conferencing:** Integrates with Zoom API v2.0+ and Google Meet SDK for meeting sessions; stores recordings in AWS S3 or Azure Blob storage.
- **Calendar:** Synchronizes with Google Calendar API and Microsoft Graph API for Outlook; supports iCalendar exports.
- **Learning Content:** Integrates with LinkedIn Learning API for content recommendations and supports Coursera course imports via LTI 1.3.
- **Authentication:** Implements OAuth 2.0 for third-party authentication, supports SAML 2.0 for enterprise SSO, and provides multi-factor authentication (MFA) options.

4.4 Communications Interfaces

The system shall implement secure and efficient communication protocols for all data transmissions.

- **Real-time Communication:** Uses WebSocket for real-time collaboration updates and WebRTC for peer-to-peer communications.
- **Notification Systems:** Implements SMTP for email notifications and integrates with SMS gateways for urgent alerts.
- **Data Security:** Enforces HTTPS/TLS 1.3 for all data transmissions.

5. Other Nonfunctional Requirements

Nonfunctional requirements define the quality attributes, constraints, and performance standards of the system. These ensure the system operates efficiently, securely, and reliably while meeting user expectations.

5.1 Performance Requirements /// remove the priority

- **PRF-1:** The system shall support at least 10,000 concurrent users (measured using JMeter load testing).
- **PRF-2:** The system shall load web pages in under 2 seconds for 90% of requests (measured by web analytics tools such as Google Lighthouse or New Relic).

- **PRF-3:** The system should process database transactions in under 100ms (measured through automated database benchmarking scripts).
- **PRF-5:** The system should maintain CDN response times below 100ms for static content (measured using CDN monitoring platforms such as Cloudflare Analytics).

- **PRF-4:** The system would handle file uploads of up to 1GB per file (verified by uploading test files and confirming via system logs).

5.2 Safety Requirements

- **SAF-1:** The system shall validate user inputs to prevent injection attacks (verified by code review and OWASP ZAP penetration testing).
- **SAF-3:** The system shall provide mechanisms to report harmful/inappropriate content (verified through user acceptance testing and feature inspection).
- **SAF-4:** The system shall maintain separation between production and test environments (confirmed by deployment architecture documentation reviews).
- **SAF-5:** The system shall scan file uploads for malware and viruses (tested using EICAR test files and ClamAV or similar antivirus tools).

- **SAF-2:** The system should implement content moderation for user-generated content (measured by reviewing moderation workflow during integration testing).

5.3 Security Requirements

- **SEC-1:** The system shall enforce strong password policies configurable by administrators (measured through admin panel inspection and attempted password registration).

- **SEC-2:** The system shall encrypt data in transit with TLS 1.3 and at rest with AES-256 (verified by SSL Labs scan and database encryption status).
- **SEC-3:** The system shall implement multi-factor authentication (MFA) (measured by login flow testing and support for TOTP/SMS apps).
- **SEC-4:** The system shall maintain audit logs for security-relevant actions (measured by inspecting log entries in ELK or Splunk).
- **SEC-5:** The system shall undergo regular penetration testing (verified by reviewing third-party pentest reports).
- **SEC-6:** The system shall sanitize outputs to prevent XSS attacks (evaluated through static code analysis tools like SonarQube and manual penetration testing).

5.4 Software Quality Attributes/// REMOVE AND PUT THE REQS

Reliability

- **REL-1:** The system shall achieve 99.9% uptime during core operating hours (monitored using uptime monitoring tools like Pingdom or AWS CloudWatch).
- **REL-2:** The system shall implement automated backups with point-in-time recovery (verified by scheduled backup jobs and test restores).
- **REL-3:** The system shall maintain data consistency across distributed components (measured by automated consistency check scripts).

Availability

- **AVL-1:** The system shall be available 24/7/365 with scheduled maintenance windows (measured by system log reports and incident monitoring).
- **AVL-2:** The system shall implement redundancy for critical components (verified by architecture review and failover simulation).

- **AVL-3:** The system should provide a public status page for uptime monitoring (confirmed by checking the status page and uptime feed).

- **MAI-1:** The system shall follow a modular design for independent component updates (measured by architecture documentation and codebase review).

- **MAI-2:** The system should maintain comprehensive technical documentation (verified by documentation coverage audits).
- **MAI-3:** The system should implement feature flags for controlled rollouts (measured by code inspection and rollout simulations).

Portability

- **POR-1:** The system shall be deployable across AWS and Azure cloud platforms (measured by successful deployment on both platforms).

- **POR-2:** The system should use containerization for environment consistency (verified by Docker/Kubernetes deployment testing).

- **POR-3:** The system would minimize platform-specific dependencies (verified by static code analysis and dependency audits).

Usability

- **USA-2:** The system shall provide consistent navigation patterns across all sections (confirmed by usability tests and heuristic evaluation).

- **USA-1:** The system should require ≤3 clicks to access core functions (measured through user journey walkthroughs).
- **USA-3:** The system should maintain UI response times of <100ms for interactions (measured by Chrome DevTools or similar profiling tools).

Scalability

- **SCL-1:** The system shall auto-scale microservices using Kubernetes (measured by Kubernetes HPA metrics during simulated load).
- **SCL-2:** The system shall support global access via CDN with multi-region redundancy (verified through CDN analytics and latency testing).

- **SCL-3:** The system would optimize costs through resource right-sizing (measured by monthly cloud cost reports).

Reusability

- **REU-1:** The system shall expose reusable microservices via well-documented APIs (measured by API documentation inspection and code review).
- **REU-2:** The system shall design UI components with a shared library (e.g., React Storybook) (verified by inspection of the shared component library).
- **REU-3:** The system should achieve ≥70% code reuse for similar functional modules (measured by static code analysis tools).

Robustness

- **ROB-1:** The system shall validate all inputs against OWASP standards (verified by static code analysis and input validation tests).
- **ROB-2:** The system shall maintain functionality during partial network outages (measured by chaos engineering tests using tools like Gremlin).
- **ROB-3:** The system should recover automatically from 95% of runtime exceptions (measured by monitoring error recovery logs).

Testability

- **TST-1:** The system shall provide 85%+ unit test coverage for critical modules (measured by code coverage reports from Jest, JaCoCo, etc.).
- **TST-2:** The system shall expose test hooks for integration testing (verified by executing integration test suites).
- **TST-3:** The system should generate test datasets mimicking production workloads (verified by reviewing test dataset characteristics).

Business Rules

- **BRL-1:** The system shall enforce institutional enrollment policies (verified by policy-based system testing).
- **BRL-2:** The system shall support plagiarism detection for academic integrity (measured by successful integration and test of plagiarism detection service).
- **BRL-3:** The system shall comply with data retention policies aligned with legal standards (verified via compliance audits and policy review).

6. Appendix

6.1 Glossary

- **API** Application Programming Interface - Enables software components to communicate
- **CLRS** Collaborative Learning Recommendation System - EDUSynergy's AI recommendation engine
- **FERPA** Family Educational Rights and Privacy Act - US law protecting student education records
- **GDPR** General Data Protection Regulation - EU data privacy regulation
- **HIPAA** Health Insurance Portability and Accountability Act - US healthcare data security law
- **JWT** JSON Web Token - Secure method for transmitting claims between parties
- **KPI** Key Performance Indicator - Measurable value demonstrating effectiveness
- **LTI** Learning Tools Interoperability - Standard for educational technology integration
- **MFA** Multi-Factor Authentication - Security system requiring multiple verification methods
- **RBAC** Role-Based Access Control - Authorization system based on user roles
- **RPO**
  Recovery Point Objective - Maximum tolerable data loss measured in time
- **RTO** Recovery Time Objective - Maximum tolerable downtime after failure
- **SAML** Security Assertion Markup Language - Standard for exchanging authentication data
- **SSO** Single Sign-On - Authentication allowing access to multiple systems with one login
- **WCAG** Web Content Accessibility Guidelines - International web accessibility standards
- **xAPI** Experience API - eLearning data tracking specification

6.2 Analysis Models UML

[Provided in earlier sections through PlantUML diagrams]

- B.1 Use Case Diagrams
- B.2 Activity Diagrams
- B.3 State Diagrams
- B.4 Sequence Diagrams
- B.5 Data flow Diagrams
- B.6 Class Diagrams

6.3 Issue list