# Project Proposal

## 1. Overview

The eLearning platform is a web-based system designed to facilitate the delivery of educational content in a secure and scalable manner. It enables users to access and manage courses, track learning progress, and engage in interactive learning experiences. The platform is tailored to meet the needs of diverse users, including students, educators, and administrators.

## 2. Objectives

- Develop a user-friendly and responsive eLearning system.

- Implement a secure authentication system for users.

- Enable course management functionalities for educators.

- Provide progress tracking for learners.

- Integrate interactive learning tools to enhance engagement.

- Ensure scalability to accommodate growing user demand.

## 3. Scope

### In-Scope:

- User authentication and authorization.

- Course creation, management, and enrollment.

- Progress tracking and reporting.

- Interactive learning features such as quizzes and discussions.

- Secure storage and management of educational content.

- Administrative controls for managing users and content.

### Out-of-Scope:

- Offline access to learning materials.

- Integration with third-party learning management systems.

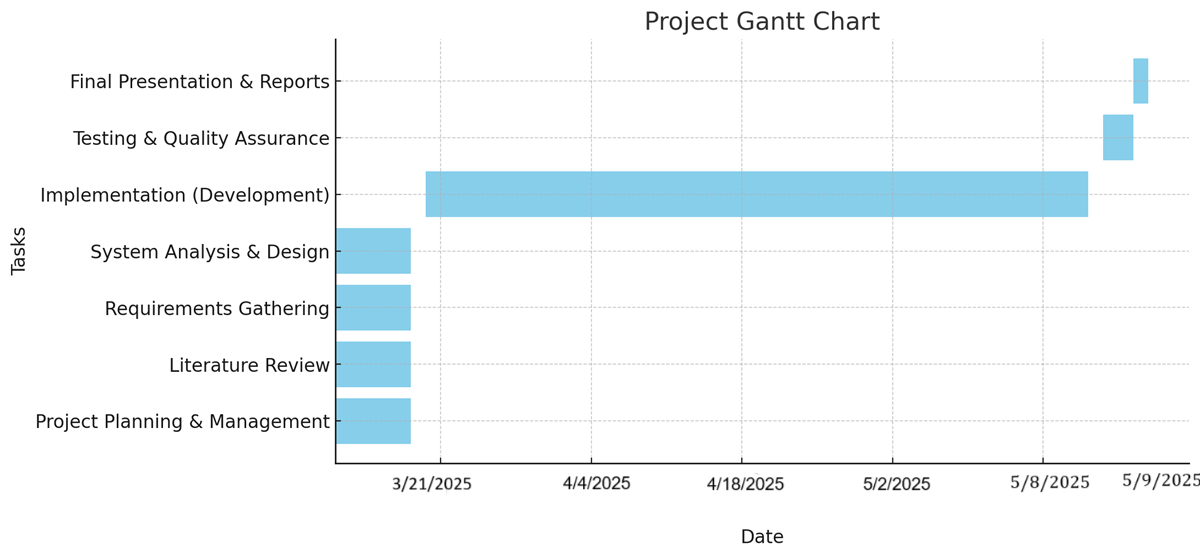
- Advanced AI-based tutoring systems.

This proposal serves as the foundation for the project's planning and execution, ensuring that all stakeholders have a clear understanding of its goals and deliverables.

# Project Plan

## 1. Timeline (Gantt Chart)

Below is the Gantt chart illustrating the project timeline:



## 2. Milestones & Deliverables

- \*\*3/21/2025\*\*: Completion of Project Planning, Literature Review, Requirements Gathering, and System Design.

- \*\*5/5/2025\*\*: Completion of Development (Implementation).

- \*\*5/8/2025\*\*: Completion of Testing & Quality Assurance.

- \*\*5/9/2025\*\*: Final Presentation and Documentation Submission.

## 3. Resource Allocation & Roles

- \*\*Member 1\*\*: Project Manager & Backend Developer – Oversees project progress, manages tasks, and develops backend APIs.

- \*\*Member 2\*\*: Frontend Developer – Designs and implements the user interface (UI/UX).

- \*\*Member 3\*\*: Database Administrator – Manages database schema, queries, and optimizations.

- \*\*Member 4\*\*: Full-stack Developer & QA – Assists in full-stack development and performs testing.

This plan ensures that the project progresses smoothly while meeting the required deadlines.

# Task Assignment & Roles

Each team member is assigned specific responsibilities to ensure smooth project development and timely completion. Below is the task distribution among the four Full-Stack .NET developers.

## Member 1 - Project Manager & Backend Developer

- Oversees project progress and ensures deadlines are met.

- Develops backend APIs and business logic using ASP.NET Core.

- Handles authentication and authorization mechanisms.

- Coordinates meetings and documentation management.

## Member 2 - Frontend Developer

- Designs and implements the user interface using HTML, CSS, JavaScript, and Razor Views.

- Ensures responsive and user-friendly UI/UX.

- Integrates frontend components with backend APIs.

- Works on interactive learning features like quizzes and discussions.

## Member 3 - Database Administrator

- Designs and implements the database schema using SQL Server.

- Manages database migrations and optimizations.

- Ensures data consistency and security.

- Implements progress tracking and reporting mechanisms.

## Member 4 - Full-Stack Developer & QA

- Assists in both frontend and backend development.

- Performs system testing and debugging.

- Writes test cases and ensures quality assurance.

- Handles deployment and server-side configurations.

This role distribution ensures balanced workload and efficient development throughout the project.

# Risk Assessment & Mitigation Plan

This document outlines potential risks that could impact the eLearning platform project and the strategies to mitigate them. The risks are categorized by their impact levels and have corresponding solutions to minimize disruptions.

|  |  |  |
| --- | --- | --- |
| Risk | Impact Level | Mitigation Strategy |
| Requirement Changes | Medium | Maintain clear documentation and conduct regular meetings with stakeholders to align expectations. |
| Delays in Development | High | Use a task management system, track progress weekly, and adjust schedules if needed. |
| Technical Challenges | Medium | Encourage team collaboration and research solutions early when encountering issues. |
| Security Issues | High | Implement secure authentication, encryption, and regular security checks. |
| Insufficient Testing | Medium | Allocate dedicated time for testing and bug fixes before deployment. |
| Team Communication Gaps | Low | Hold regular stand-up meetings and use collaboration tools like Slack or Teams. |

By proactively addressing these risks, the team can ensure smoother project execution and minimize disruptions.

# Key Performance Indicators (KPIs)

This document defines the key performance indicators (KPIs) that will be used to evaluate the success of the eLearning platform. The selected KPIs focus on system performance, user engagement, and reliability.

|  |  |
| --- | --- |
| KPI | Measurement Method |
| System Uptime | Ensure 99%+ uptime by monitoring server availability. |
| Page Load Speed | Maintain an average response time below 3 seconds. |
| User Adoption Rate | Track the number of registered users over time. |
| Course Completion Rate | Measure the percentage of users who complete courses. |
| Bug Resolution Time | Resolve reported issues within 48 hours. |
| User Satisfaction | Collect feedback and maintain a 4+ star rating in surveys. |

These KPIs will help track the project's progress and ensure the platform meets performance and user expectations.

# Literature Review

## 1. Feedback & Evaluation

This section will include the lecturer’s feedback on the project once received. The evaluation will cover aspects such as project clarity, technical execution, and overall feasibility.

## 2. Suggested Improvements

Based on feedback, potential improvements to enhance system performance, usability, or functionality will be documented here.

## 3. Final Grading Criteria

|  |  |
| --- | --- |
| Criteria | Description |
| Documentation | Marks will be allocated based on the completeness and clarity of project documentation. |
| Implementation | Quality of code, functionality, and adherence to project requirements will be assessed. |
| Testing | Effectiveness of testing strategies and issue resolution will contribute to grading. |
| Presentation | The clarity and professionalism of the final project presentation will be evaluated. |

This section ensures the team is aware of the evaluation aspects and can refine the project accordingly.

# Requirements Gathering

## 1. Stakeholder Analysis

The following stakeholders are identified for the eLearning platform along with their roles and expectations:

|  |  |
| --- | --- |
| Stakeholder | Role & Expectations |
| Students | Access courses, track progress, and participate in interactive learning. |
| Instructors | Create and manage courses, track student progress, and provide feedback. |
| Administrators | Manage users, monitor system performance, and maintain security. |
| System Developers | Ensure platform functionality, security, and scalability. |

## 2. User Stories & Use Cases

User stories define how different users will interact with the system.

- As a student, I want to enroll in a course so that I can start learning.

- As an instructor, I want to create and update courses so that students can access new materials.

- As an administrator, I want to manage users so that I can control access and permissions.

- As a student, I want to track my progress so that I can see my learning achievements.

## 3. Functional Requirements

- User authentication and role-based access control.

- Course creation, enrollment, and management.

- Progress tracking and reporting.

- Interactive learning features like quizzes and discussions.

- Secure data storage and content management.

## 4. Non-Functional Requirements

- System should have 99%+ uptime.

- Pages should load within 3 seconds on average.

- Platform should be accessible on both desktop and mobile devices.

- User data should be encrypted for security.

- System should be scalable to handle increased user load.

These requirements ensure the system meets both user expectations and technical standards.

# System Analysis & Design

## 1. Problem Statement & Objectives

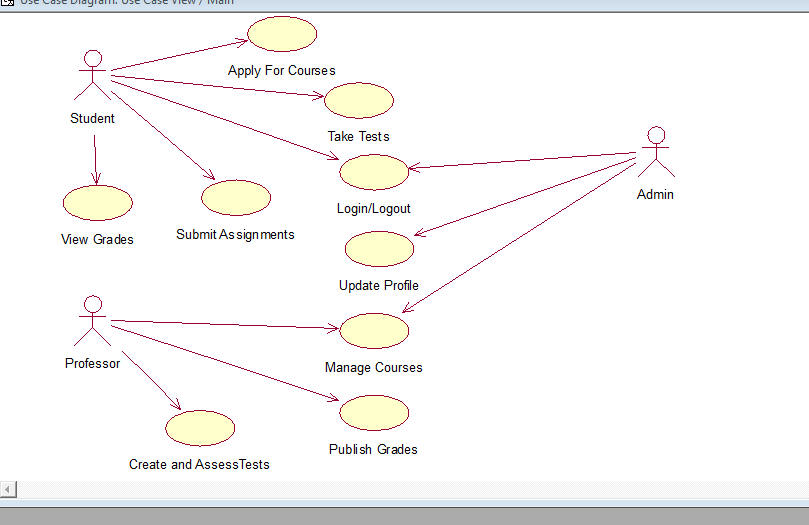
### Problem Statement:

The current e-learning environment lacks an efficient, user-friendly platform that provides seamless interaction between students and professors. Challenges include ineffective course management, limited tracking of student progress, and inadequate communication tools.

### Objectives:

• Develop a structured e-learning system with course management, student tracking, and assessment features.  
• Ensure secure authentication and user role management.  
• Provide an intuitive user experience with clear navigation and accessibility.

## 2. Use Case Diagram & Descriptions

• **Actors:** Student, Professor, Admin  
• **Use Cases**:  
 - Students can register, log in, enroll in courses, submit assignments, and take quizzes.  
 - Professors can create courses, upload materials, evaluate assignments, and track student progress.  
 - Admins can manage users, handle system configurations.

## 3. Functional & Non-Functional Requirements

### Functional Requirements:

1. User authentication and role-based access control.  
2. Course management: creating, updating, and deleting courses.  
3. Assignment submission and evaluation.  
4. Quiz and assessment module.  
5. Communication system (messages, announcements).  
6. Progress tracking and reporting.

### Non-Functional Requirements:

1. Performance: Fast response time (< 2s per request).  
2. Security: Encrypted data storage and secure login.  
3. Usability: Intuitive UI and mobile responsiveness.  
4. Reliability: 99.9% uptime for learning resources.

## 4. Software Architecture

• **Architecture Style:** MVC (Model-View-Controller) using .NET  
• **Components:**  
 - **Frontend:** ASP.NET MVC for web interface.  
 - **Backend:** .NET Core with RESTful APIs.  
 - **Database:** MySQL for structured data storage.  
 - **Authentication:**ASP.NET Identity for user management.

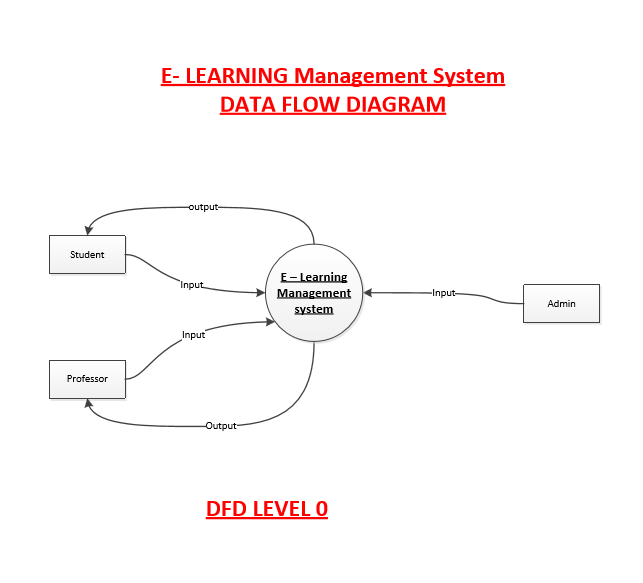
## 5. Database Design & Data Modeling

• **Entity-Relationship Diagram (ERD):** Defines relationships between entities like Users, Courses, Assignments, and Quizzes.  
• **Logical & Physical Schema:**Tables include Users, Roles, Courses, Enrollments, Assignments, Submissions, Quizzes.A diagram of a diagram

AI-generated content may be incorrect.

## 6. Data Flow & System Behavior

**Data Flow Diagrams (DFD):**  
-**DFD Level 0:** Shows overall system inputs, processing, and outputs.  
 -**DFD Level 1 & 2:** Breakdown of course management, assignment processing, and authentication workflows.



A diagram of a course

AI-generated content may be incorrect.

A diagram of a course

AI-generated content may be incorrect.

**Sequence Diagrams:**  
- **Professor Interaction:** Course creation and student progress tracking.  
- **Student Interaction:** Enrollment, assignment submission, and quiz attempts.

A diagram of a course

AI-generated content may be incorrect.

A screenshot of a computer screen

AI-generated content may be incorrect.  
  
 **Activity Diagram:**  
- Illustrates user workflows such as registration, course enrollment, and assignment submission.

A screenshot of a diagram

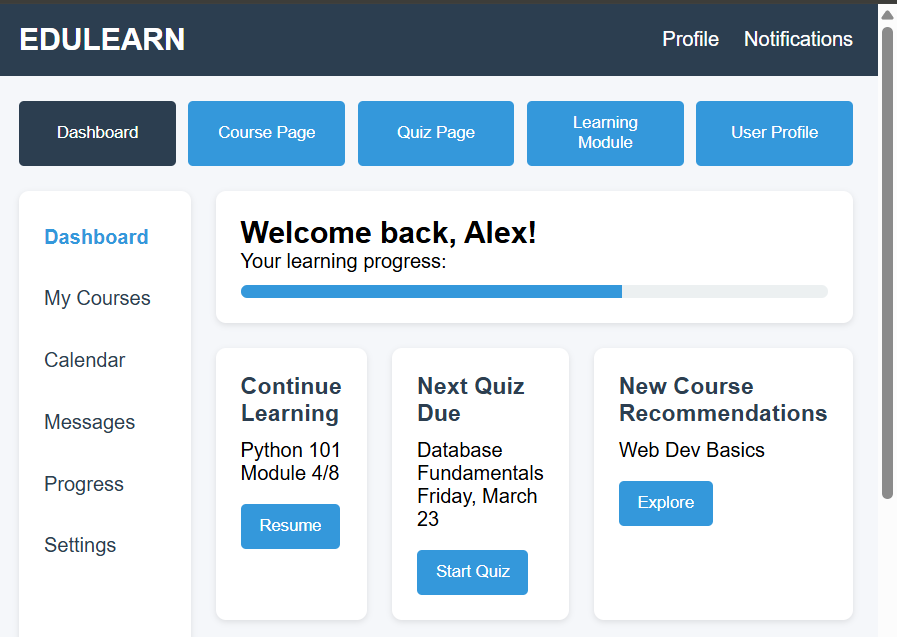
AI-generated content may be incorrect.  
  
 **Class Diagram:**  
- Represents system classes, their attributes, and relationships.

A screenshot of a computer

AI-generated content may be incorrect.

## 7. UI/UX Design & Prototyping

• **Wireframes:** Designed for key user interfaces (Dashboard, Course Page, Quiz Page, etc.).



A screenshot of a computer

AI-generated content may be incorrect.

A screenshot of a computer

AI-generated content may be incorrect.

• **UI/UX Guidelines:**

1. Consistency: Maintain uniform design patterns, typography, and color schemes throughout the application.
2. Responsiveness: Ensure that the UI adapts seamlessly to various screen sizes, including desktops, tablets, and mobile devices.
3. Navigation Simplicity: Design intuitive navigation with clear menus and easy access to key features.
4. Accessibility: Follow WCAG guidelines for usability by providing alt text for images, keyboard navigation, and color contrast adjustments.
5. Performance Optimization: Minimize load times by using optimized assets and asynchronous data fetching.
6. Error Handling: Implement clear and informative error messages to guide users in case of failed actions.
7. User Feedback: Include interactive elements like hover effects, loading indicators, and confirmation messages for better user experience.

## 8. System Deployment & Integration

• **Technology Stack:**  
 - Frontend: ASP.NET MVC, Bootstrap.  
 - Backend: .NET Core, MySQL.  
 - Authentication: ASP.NET Identity.

• **Deployment Diagram:** Illustrates system distribution across hardware components.

## 9. Additional Deliverables

• **API Documentation:** RESTful API endpoints.  
• **Testing & Validation:** Unit tests and user acceptance testing.  
• **Deployment Strategy:** Hosting on cloud services (Azure/AWS) with CI/CD integration.