**University of Plymouth**

School of Engineering,

Computing, and Mathematics

PRCO304

Final Stage Computing Project

2024/2025

EDUNEST

BSc (Hons) Computer Science

Acknowledgements  
 I would like to sincerely thank my supervisor, **Muhammad Asad**, for their constant support and guidance throughout this project, **EduNest**. Their advice and encouragement helped me stay focused and motivated throughout.

I truly appreciate the time and effort they dedicated to helping me, and I’m grateful for the valuable learning experience this project has given me.

# Abstract

EduNest is an easy-to-use Learning Management System (LMS) designed to create a smooth and engaging online learning experience. The platform supports three main users: students, teachers, and admins. Teachers can create, manage, and update courses, while students can browse, enroll, and access course materials. The system also allows students and teachers to interact in real-time through features like chat and a collaborative whiteboard. Notifications keep users updated on new materials and activities, ensuring they stay informed. Admins can manage users and courses, ensuring smooth operations. EduNest ensures secure login and role-specific dashboards, providing a customized experience for each user. Built with Django for the backend and ReactJS for the frontend, EduNest offers a flexible and responsive design. It is easy to maintain and scale, allowing for future updates and improvements. With a strong foundation, EduNest aims to deliver a high-quality learning experience for all users.

Contents

[Statement of Word Count: 9](#_Toc195134263)

[1. Background 10](#_Toc195134264)

[2. Introduction: 12](#_Toc195134265)

[2.1 Project Aims and Objectives 14](#_Toc195134266)

[2.2 Project Vision 14](#_Toc195134269)

[2.3 Problem statement 15](#_Toc195134270)

[2.4 Project Deliverables 15](#_Toc195134271)

[2.5 Report Overview 15](#_Toc195134272)

[3. Literature Review 17](#_Toc195134273)

[3.1 Learning Management System in Education 17](#_Toc195134274)

[3.2 Learning Management System in Workplace 19](#_Toc195134275)

[3.3 Learning Management System in Hospitals 20](#_Toc195134276)

[Hospitals that are using LMS: 21](#_Toc195134277)

[3.4 Different LMS Systems 21](#_Toc195134278)

[360 Learning: 21](#_Toc195134279)

[Moodle 22](#_Toc195134280)

[Skilljar 22](#_Toc195134281)

[Schoology 23](#_Toc195134282)

[Litmos 23](#_Toc195134283)

[Canvas by Instructure 24](#_Toc195134284)

[3.5 Legal, Social, Ethical, and Professional Issues 24](#_Toc195134285)

[4. Project management 27](#_Toc195134286)

[4.1 Flow that I followed 27](#_Toc195134287)

[Project Initiation & Kickoff 27](#_Toc195134288)

[Planning & Risk Management 27](#_Toc195134289)

[Execution: Research & Development 27](#_Toc195134290)

[Quality Control & Testing 27](#_Toc195134291)

[Documentation & Reporting 28](#_Toc195134292)

[Stakeholder (Supervisor) Communication 28](#_Toc195134293)

[Final Review & Submission 28](#_Toc195134294)

[4.2 Tool used for management: 28](#_Toc195134295)

[Weekly Self-Check and Progress Updates 29](#_Toc195134296)

[4.3 Division of Project into Modules 29](#_Toc195134297)

[Module 1: Project Planning and Initiation 29](#_Toc195134298)

[Module 2: User Management 30](#_Toc195134299)

[Module 3: Course Management 31](#_Toc195134300)

[Module 4: Communication and Collaboration Tools 31](#_Toc195134301)

[Module 5: Testing and Quality Assurance 32](#_Toc195134302)

[Module 6: Deployment and Post-Launch Activities 32](#_Toc195134303)

[5. Requirements analysis 33](#_Toc195134304)

[5.1 Functional Requirements 33](#_Toc195134305)

[User Registration and Authentication 33](#_Toc195134306)

[Course Management 33](#_Toc195134307)

[Content and Resource Management 34](#_Toc195134308)

[Assignments and Assessments 34](#_Toc195134309)

[Discussion and Communication 35](#_Toc195134310)

[Progress Tracking and Reporting 35](#_Toc195134311)

[Notifications and Alerts 36](#_Toc195134312)

[User Roles and Permissions 36](#_Toc195134313)

[Attendance 37](#_Toc195134314)

[5.2 Non-Functional Requirements 37](#_Toc195134315)

[Performance 37](#_Toc195134316)

[Security 38](#_Toc195134317)

[Usability 38](#_Toc195134318)

[Reliability 39](#_Toc195134319)

[Scalability 39](#_Toc195134320)

[Maintainability 39](#_Toc195134321)

[Compatibility 40](#_Toc195134322)

[Data Integrity 40](#_Toc195134323)

[Portability 41](#_Toc195134324)

[5.3 Development requirements 41](#_Toc195134325)

[5.4 Software Development Lifecycle Model 42](#_Toc195134326)

[5.4.1 Software Development Methodology 42](#_Toc195134327)

[5.4.2 Reasons for Choosing Agile 43](#_Toc195134328)

[5.5 Risk assessment 44](#_Toc195134329)

[6. System Design 46](#_Toc195134330)

[6.1 Use Cases 46](#_Toc195134331)

[4.2 Sequence Diagrams 51](#_Toc195134332)

[6.3 UML Diagrams 57](#_Toc195134333)

[7. Implementation 58](#_Toc195134334)

[7.1 Code organization and Structure 58](#_Toc195134335)

[Backend Code Structure and Component Descriptions 58](#_Toc195134336)

[Frontend Code Structure and Component Descriptions 60](#_Toc195134337)

[7.2 Designed Application 61](#_Toc195134338)

[8. System Testing 71](#_Toc195134339)

[8.1 Test Cases 71](#_Toc195134340)

[User Registration and Login 71](#_Toc195134341)

[Course Enrollment and Management 71](#_Toc195134342)

[Content Management 72](#_Toc195134343)

[User Notifications 73](#_Toc195134344)

[Chat and Collaboration Tools 74](#_Toc195134345)

[9. Outcome Evaluation 76](#_Toc195134346)

[10. Project Post-Mortem 78](#_Toc195134347)

[10.1 Overview: 78](#_Toc195134348)

[10.2 Technologies Evaluation 79](#_Toc195134349)

[10.3 Developer Performance Evaluation 79](#_Toc195134350)

[11.Conclusion 81](#_Toc195134351)

[12.References 82](#_Toc195134352)

[13. Appendix 84](#_Toc195134353)

[13.1 Requirements for system: 84](#_Toc195134354)

[13.2 Getting started with EDUNEST: 85](#_Toc195134355)

[Troubleshooting 93](#_Toc195134356)

[13.3 Application Deployment 93](#_Toc195134357)

[A. Unzipping and installing requirements: 93](#_Toc195134358)

[For Frontend: 93](#_Toc195134359)

[For Backend: 93](#_Toc195134360)

[B. List of Packages and Versions 93](#_Toc195134361)

[Frontend Packages (React) 93](#_Toc195134362)

[Backend Packages (Django) 95](#_Toc195134363)

[13.4 Development Environment Details 96](#_Toc195134364)

[Operating System Options: 96](#_Toc195134365)

[Code Editors I Used: 96](#_Toc195134366)

[Version Control: 96](#_Toc195134367)

[Django Admin Login Instructions 97](#_Toc195134368)

[Admin Panel Usage 97](#_Toc195134369)

[Login Credentials for Teacher and Student Accounts 97](#_Toc195134370)

[13.5 Edu Nest Software Evaluation Questionnaire 98](#_Toc195134371)

[Questionnaire Results: 104](#_Toc195134372)

[13.6 Unit testing 105](#_Toc195134373)

[How I Ran Unit Tests in Edu Nest 105](#_Toc195134374)

[Django Backend 105](#_Toc195134375)

[React Frontend 105](#_Toc195134376)

13.7 [Future improvements 106](#_Toc195134377)

# List of Tables

[Table 1 Legal, social, ethical and professional issues 24](#_Toc196350019)

[Table 2 FR User registration 33](#_Toc196350020)

[Table 3 FR Course Management 33](#_Toc196350021)

[Table 4 FR Content and Resource Management 34](#_Toc196350022)

[Table 5 FR Assignments 34](#_Toc196350023)

[Table 6 FR Communication 35](#_Toc196350024)

[Table 7 FR Progress Tracking and Reporting 35](#_Toc196350025)

[Table 8 FR Notifications and Alerts 36](#_Toc196350026)

[Table 9 FR User Roles and Permissions 36](#_Toc196350027)

[Table 10 FR Attendance 37](#_Toc196350028)

[Table 11 NFR Performance 37](#_Toc196350029)

[Table 12 NFR Security 38](#_Toc196350030)

[Table 13 NFR Usability 38](#_Toc196350031)

[Table 14 NFR Reliability 39](#_Toc196350032)

[Table 15 NFR Scalability 39](#_Toc196350033)

[Table 16 NFR Maintainability 39](#_Toc196350034)

[Table 17 NFR Compatibility 40](#_Toc196350035)

[Table 18 NFR Data Integrity 40](#_Toc196350036)

[Table 19 NFR Portability 41](#_Toc196350037)

[Table 20 Development Requirements 41](#_Toc196350038)

[Table 21 Risks and their mitigation strategies 44](#_Toc196350039)

[Table 22 UC for registration and login 46](#_Toc196350040)

[Table 23 UC for course creation and management 46](#_Toc196350041)

[Table 24 UC for courses enrollment 47](#_Toc196350042)

[Table 25 UC for accessing course content 47](#_Toc196350043)

[Table 26 UC for providing feedback and reviews 48](#_Toc196350044)

[Table 27 UC for chat 48](#_Toc196350045)

[Table 28 UC for collaborative whiteboard 49](#_Toc196350046)

[Table 29 admin management of users 49](#_Toc196350047)

[Table 30 Test cases for user registration and login 71](#_Toc196350048)

[Table 31 Test cases for course enrollment and management 71](#_Toc196350049)

[Table 32 Test Cases for content management 72](#_Toc196350050)

[Table 33 Test Cases for user notifications 73](#_Toc196350051)

[Table 34 Test Cases for testing chat and collaboration 74](#_Toc196350052)

[Table 35 planned vs achieved outcomes 76](#_Toc196350053)

[Table 36 Packages list used in frontend development 93](#_Toc196350054)

[Table 37 Backend packages used 95](#_Toc196350055)

# List of Figures

[Figure 1 SDLC Life cycle 43](#_Toc196350056)

[Figure 2 User Login and Registration 51](#_Toc196350057)

[Figure 3 Admin Management of Users and Courses 52](#_Toc196350058)

[Figure 4 Collaborative whiteboarding 52](#_Toc196350059)

[Figure 5 Real-Time Communication (Chat) 53](#_Toc196350060)

[Figure 6 Providing Course Feedback 53](#_Toc196350061)

[Figure 7 Accessing Course Materials 54](#_Toc196350062)

[Figure 8 Course Creation and Management 54](#_Toc196350063)

[Figure 9 Course Enrollment 55](#_Toc196350064)

[Figure 10 UML Diagram 57](#_Toc196350065)

[Figure 11 Project Directory 58](#_Toc196350066)

[Figure 12 Code structure 60](#_Toc196350067)

[Figure 13 Teacher authentication and signup 61](#_Toc196350068)

[Figure 14 Teacher Login 62](#_Toc196350069)

[Figure 15 Student signup 62](#_Toc196350070)

[Figure 16 Student dashboard 63](#_Toc196350071)

[Figure 17 Feature to add courses 64](#_Toc196350072)

[Figure 18 My courses 64](#_Toc196350073)

[Figure 19 Course information 65](#_Toc196350074)

[Figure 20 Teachers information 66](#_Toc196350075)

[Figure 21 Ability to add course rating and feedback 67](#_Toc196350076)

[Figure 22 Real-time communication 68](#_Toc196350077)

[Figure 23 My teachers 68](#_Toc196350078)

[Figure 24 Enrolled students 68](#_Toc196350079)

[Figure 25 Add course material 69](#_Toc196350080)

[Figure 26 latest courses 70](#_Toc196350081)

Statement of Word Count: 10400  
  
Code Submission:   
**Web:**

# Background

During the late 1990s, rapidly developing technologies have altered the methods of instruction and learning in educational establishments (Pishva et al., 2010). The 1990s witnessed a revolution in teaching and learning due to the growing capabilities of multimedia computers, broadband networks, and notable advancements in the creation and distribution of educational materials through electronic channels. The industry transitioned from basic synchronous learning applications and computer-based training (CBT) to advanced e-learning platforms that integrated the best features of both. The connection between students and teachers has been made easier by the incorporation of technology into the educational environment. (Pishva et al., 2010).

The majority of universities are located in urban regions, which have easy access to the internet in this era of information technology. A large computer network that connects smaller computer networks all over the world is known as the internet. It consists of governmental, commercial, educational, and other networks that all follow to the same set of communications protocols. Nowadays, the majority of university students have access to the internet as well because their university usually provides it, and there are typically student-focused internet cafes close to the campus that serve to students. Some people have internet connection at home as well since they have an internet service provider subscription.

Students at universities learn primarily independently because they are not taught in a one-way fashion like they are in primary and secondary schools. Instead, lecturers typically provide lecture notes and leave it up to the students to find out additional information for themselves. University learning is a two-way process in which students contribute their ideas or opinions to a topic during a class discussion and lecturers offer their knowledge. As a result, university students must continuously seek out new information to expand their expertise.

It is essential to transition from traditional, in-person instruction to online instruction. This calls for an LMS platform that can handle all of the requirements for interaction between teachers and students.

# Introduction:

In a digital environment where web technology is used for the analysis, regulation, and distribution of knowledge, values, skills, and assessments, the Learning Management System (LMS) functions as an educational tool. It can facilitate information sharing between educators and learners by providing resources, administering instruction, and providing information in a cohesive setting. According to Abu Shawar (2009), LMSs are "internet-based software allowing instructors to manage materials distribution, assignments, communications, and other aspects of instruction for their courses." With the growing popularity of hybrid techniques that combine in-person and virtual learning, learning management systems (LMSs) are now an essential part of the curriculum at the majority of universities (Pishva et al., 2010). Brown (2020) asserts that the LMS offers a way to track and deliver e-learning projects centrally. According to E-learning Basics (2021), LMS is a platform for digital learning with the following salient characteristics:   
**Learning:** It makes it possible to create a single source for training materials and online courses.   
**Management:** It makes it possible to manage learners as well as courses.   
**System**: A computer system is utilised.

Designed and managed by Blackboard Inc., Blackboard is one of the most popular web-based learning management systems (LMS). It is a fully web-based learning environment that facilitates communication between educators and learners while also acting as a repository for different kinds of data. A variety of administrative tools are also included in Blackboard to help teachers and students in their work (Linnaeus University, 2011a).

A wide range of functions are provided by learning management systems (LMS), which are essential for supporting distance learning. These functions include the presentation of learning materials, the sharing and discussion of presented materials, course management, coursework completion, exam attendance, assignment and exam feedback, organization of learning materials, and record-keeping for students, teachers, and systems. For students and teachers located in different places, the Learning Management System (LMS) makes use of all the online resources available to create a collaborative learning environment.

LMSs have advantages and disadvantages as platforms for online learning in educational settings. Lessening educational expenses and giving students more organized access without affecting their competitiveness are two benefits of implementing Learning Management Systems (LMS) in the classroom. The ability for teachers and students to communicate outside of the classroom without regard to time constraints or location makes information flow faster and more affordable. Teachers and students can develop interactions, collaborative replies, and collaboration through learning using LMS. Through the utilisation of current learning resources, it also enables them to create fresh, captivating forms of communication, which has been demonstrated to boost motivation and academic success.

Additionally, LMS gives teachers and students the freedom to research, examine, and obtain information on assignments and learning activities at their own pace, guaranteeing that every student has equal access to material at the same time.  
However, there are disadvantages to using LMS, including the difficulty to adequately monitor student activities during learning, the somewhat higher initial cost of equipment procurement, the constraints on technology adoption for everyone, and inefficiencies in communicating values and norms. Furthermore, there's a chance that LMS-based learning's instructional design doesn't always suit instructors' and students' needs.

By keeping in view the importance of Learning Management Systems (LMS) and the drawbacks of previous systems, we aim to design EduNest as a comprehensive and innovative platform that addresses the limitations of traditional e-learning solutions. EduNest will enhance the educational experience by providing a user-friendly interface that fosters effective communication and collaboration between educators and students. Our focus will be on integrating advanced features such as multimedia course management, real-time interactions through web sockets, and a robust RESTful API for seamless data management.

## 2.1 Project Aims and Objectives

### Aims To develop EduNest, an innovative LMS that enhances online learning by providing a user-friendly, interactive, and adaptable platform for educators and students.

### Objectives

The objectives of this project is to:

* Give the entire training process more control. Setting up the timetable for training sessions, giving team members assignments, and keeping an eye on their performance.
* To Aid in developing a well-rounded educational experience. For learning to be effective, every learner needs a unique strategy. One can tailor the training session to the needs of the learners using our effective learning management system.
* To Facilitate communication with students. So, you may communicate with students using chat and other channels with an LMS. This allows you to maintain the learners' interest throughout the training course.
* To enhance the cooperation between administrators and trainers. Although each person has a separate account, roles can be assigned to maximize teamwork.
* Incorporate multimedia course management, real-time interactions, and support for various teaching methodologies to cater to diverse learning styles.

## 2.2 Project Vision

The vision of this project is to is to explore the role of Learning Management Systems (LMS), integrated classroom education from the users' perspectives and try to identify and explain the patterns of learning and teaching activities. After analyzing all our aim is to develop EduNest that will transform online learning into a highly engaging and collaborative experience. By addressing the limitations of existing systems, EduNest aims to empower educators to deliver effective instruction and provide students with the tools they need to succeed.

## Problem statement

Covid-19 told us a lot of things. During this period, the use of Learning Management Systems (LMS) for the delivery of teaching became popular. According to research, learning management system (LMS) platforms have the potential to improve student engagement and learning outcomes. However, there are issues with the quality of instruction provided by these systems, which are primarily due to issues with infrastructure deficiencies and Internet connectivity as well as inadequate training for educators and students on how to use the platform successfully. Furthermore, research indicates that a number of variables, including gender, years of study, and specialty, have a big impact on how well students embrace and use LMS.

## Project Deliverables

The project deliverable is a web application that will gives a teacher the ability to produce and distribute content, track student engagement, and evaluate student achievement. Additionally, the aim is to provide students access to interactive tools like discussion boards, video conferencing, and threaded discussions.

## Report Overview

**Project Definition & Planning**

* Task Definition
* Project Planning

**Context Review & Subject Knowledge**

* Literature Review
* Citations & References
* Subject Matter Knowledge
* Legal, Social, Ethical, and Professional (LSEP) Issues

**Project Methodology & Implementation**

* Methodology Justification
* Project Engagement & Coverage
* Implementation Skills
* Agile Implementation
* Coding Skills & Complexity

**Critical Evaluation & Conclusions**

* Data Analysis
* Software Testing & Validation
* Critical Evaluation
* Conclusions & Recommendations

**Structure & Presentation**

* Report Structure
* Grammar & Language
* Mathematical Presentation
* Visuals & Figures

# Literature Review

Over time, information technology's (IT) role in organisations has evolved dramatically (Laurindo, 2009; Rosini, 2013). In this case, it supports Laurindo (2009) by taking a broad view of IT, which includes automation, telecommunications, and information systems (IS), as well as a wide range of hardware and software technologies that businesses utilise to supply data, information, and knowledge. The term "digital convergence" has been used in the technology sector to describe an all-encompassing approach to IT.

IT may play a significant role in innovation not only through its direct use but also as a conduit for other innovations that support, improve, and, emphasise e-learning (Laurindo, 2009). There are high expectations for the potential of information technology in the context of globalisation, where virtual organisations and e-businesses are growing (Schlemmer, Saccol & Garrido, 2007).

## 3.1 Learning Management System in Education

Software called an LMS is made to distribute and oversee the delivery of educational materials. Gaining proficiency with an LMS provides the benefits of using different systems. The learning system's modules each offer unique opportunities. Thus, an assessment of the standard module's value for efficient learning is required (Kakasevski et al., 2008). LMS can assist teachers in organising and managing instructional materials and other requirements associated with the teaching and learning process, such as articles, e-books, animated films, noises, and engaging instructional videos to add entertainment and creativity to the learning process. Teachers that teach online classes must be knowledgeable about technology or be able to employ new technologies that may be given in an effective and efficient manner, according to Alias & Zainuddin (2005). Every technological skill the instructor has acquired is surely linked to the improvement of his instruction, both in-person and online. The online learning process is supported by six LMS features, which include:

* Easy-to-use and attractive user interface.
* Online registration with various payment methods.
* Virtual classes enabling distance learning with video conferencing.
* Online quizzes and exams, supporting question creation and distribution.
* Discussion room for students to independently review subject matter.
* Report feature to help teachers track student progress.

Because online teaching takes place in this setting, technology is deemed essential for both the background and the teaching context. Applications for online learning play a wide range of roles (Dube et al., 2022; Tsakeni, 2021). Facebook (Wang et al., 2011) for example, serves as a platform for learning management, practical communication, and information distribution. However, it still has privacy constraints that could be disclosed (Wang et al., 2012).

For a higher education institution, putting in place a learning management system is a big decision. The financial cost is a major factor in this decision. Although the majority of providers provide a strong learning management system, most demand one-time payments and annual site licensing. If only one department or even a small university is thinking about investing in a learning management system, these expenses can be extremely costly. Some schools have created their own open source learning management system, like OpenUSS, to get around these problems (Grob, Bensberg & Dewanto, 2004). Before purchasing a system, institutions should think carefully about the goals they hope to accomplish with the LMS. The following elements are recommended by Iqbal and Qureshi (2011) as the most crucial ones to take into account when choosing a learning management system:

Technical specifications and support, design specifications, clear and easy-to-use graphical interface, well-designed course repository, course administration capability, user interaction capability, evaluation and feedback, student profiles, pedagogy, and organizational goals and objectives. These considerations influence the choice of learning management system, whether it be building an internal system, using an open source solution, or investing in a huge system.

Finding efficient ways to use a learning management system (LMS) to improve learning and provide the best possible experience for both staff and students is a common difficulty in higher education. Studies have looked into ways to use learning management systems (LMSs) to encourage active learning, but the typical design of WebCT and Blackboard frequently runs counter to these goals by emphasizing information delivery over promoting deeper learning (Herrington, Reeves, & Oliver, 2005, cited in Herrington, 2006).

## 3.2 Learning Management System in Workplace

LMS offers a clear method for organising, carrying out, and evaluating certain training programs. Online training has grown in significance in recent years, whether for business skill-based learning or education. This modern approach to teaching not only does away with the need to physically access course materials, but it also fosters greater communication between the instructor and the student.

Employees can acquire the knowledge and skills necessary to complete their work assignments from any location at any time through the LMS, which makes it possible for training materials to be distributed via the internet and intranet. While usability is sometimes overlooked, it is necessary to investigate in order to improve the use of LMS for staff training. Usability is the degree to which a system can be used by specific users to accomplish specific goals in a specific context in an effective, efficient, and satisfactory manner. If e-learning systems are not usable, users may spend more time learning the system than the content they are supposed to be learning.

Job satisfaction can be measured by raising employee motivation, and while there are many ways to do so, good training programs are essential as they help employees learn new skills and expand their knowledge. Because employee training can be conducted both on and off-site, LMS technologies are helpful. For example, Hewlett-Packard emphasises their philosophy that "one size does not fit all" [4] by letting regional trainers in various nations choose the most effective training delivery methods. Managers in the corporate sector see LMS as an essential resource for meeting their staff members' training demands, which is why many businesses worldwide are attempting to use LMS for training.

An exploratory study looked at 106 businesses opinions regarding the usability of online training since businesses need an LMS that is easy to use. According to Kimiloglu et al., adopting an LMS involves a number of factors, including affordability, usefulness, adaptability, and sustainability. The authors discovered that online training is a versatile and effective method of enhancing workers' abilities and expertise.

Corporate learning management systems (LMSs) are different from educational LMSs, according to Dodson et a. A study on MOOCs (massive open online courses), which are off-the-shelf LMSs for online and distance learning, found that LMSs can be customised to the needs of the audience. An education LMS's main focus is solely on information transfer, but a corporate LMS's main function is to guarantee that employees have the abilities and knowledge to support an organization's growth and development.

## 3.3 Learning Management System in Hospitals

A learning management system called a healthcare LMS was created to satisfy the educational, training, and development requirements of individuals working in the healthcare industry. Hospitals, medical device firms, private clinics, health tech companies, and anybody else working in this field can use healthcare learning management systems (LMS) for healthcare training.

HR teams and managers can simply manage user data and course enrolments with the help of healthcare learning management systems (LMS), which also enable anytime, anywhere learning. To handle the ongoing changes in the healthcare environment, training teams can develop and distribute extensive training and learning programs, and learning materials can be arranged into formats that are simple to understand. By providing them with the ability to evaluate learner progress, gather data, and conduct in-depth analysis to find and promote development opportunities, these systems assist healthcare organisations in improving training results.

Businesses in the healthcare industry can also use a learning management system to train employees on subjects including upskilling, onboarding, sales, best practices, compliance, regulatory standards, guidelines, and emerging technology. The healthcare LMS market is expected to reach US$3.5 billion by the end of 2030 due to the advantages that it offers.

### Hospitals that are using LMS:

* To ensure that all staff members are kept up to date on important regulations and procedures, Johns Hopkins Medicine employs an LMS to offer continuous compliance training to its employees across several departments.
* In order to ensure effective adoption and use of their medical equipment in hospitals across the globe, Medtronic, a leader in medical technology, trains healthcare personnel on how to utilise them using an LMS.
* The Mayo Clinic gives patients access to an online portal that offers videos and instructional courses on a variety of health-related subjects, making them more capable of handling their own treatment.
* Kaiser Permanente ensures that new hires are up to date on pertinent certifications and training materials by using its LMS for role-specific training and onboarding.
* An LMS was put in place by Mount Sinai Health System to teach its doctors how to conduct telehealth consultations, guaranteeing a seamless shift to virtual treatment.
* The World Health Organisation (WHO) ensures that medical personnel are equipped to handle emergencies by using an LMS to deliver emergency preparedness training in various geographical areas.

## 3.4 Different LMS Systems

### 360 Learning:

360Learning is a comprehensive learning solution that integrates collaborative learning with the capabilities of an LXP, LMS, and Academies.

360Learning is an excellent option for companies looking for a hybrid platform that can be used for both internal and external staff training. It is also a perfect option for companies who wish to manage and deliver external training, such as partner enablement and customer training, as well as internal employee training.

360Learning is renowned for its ease of use, integrated certification engine, extensive data and analytics, and user-friendliness. It turns internal specialists into L&D collaborators using AI and collaborative features, allowing them to upskill continually and rapidly from within the company.

### Moodle

Moodle is arguably the most well-known learning management system example. This learning platform has been available for free and open-source for a while. Even while it doesn't have the same slick or contemporary appearance as its paid rivals, its powerful features make it an excellent choice for any educational setting. The LMS can be expanded and customised to meet your exact needs. You may also add new features with the thousands of community-developed plugins. Additionally, it easily connects with programs like Microsoft Office 365, Google Apps, NextCloud, and others.

Moodle used to terrify a lot of administrators since it required technical expertise to set up. Configuring it might be complex, especially if you require your own server. But now that MoodleCloud has been released, it is simple to get started without the server or setup. You can begin producing and posting lessons as soon as you sign up, just like with premium choices. 50 students are permitted with 200 MB upload limits under the Free Forever plan. For Starter, the annual cost of the paid plans is $80.

### Skilljar

A user-friendly learning management system called Skilljar is used to train partners and external clients.  
For managing and offering partner, customer, and compliance training, this LMS solution is a great choice.   
By offering learners a configurable and aesthetically pleasing interface, Skilljar encourages the use of educational resources. It offers extensive data and analytics, a built-in certification engine, and other authoring features.

### Schoology

Another feature-rich learning platform is Schoology, which strives to give you all the resources you need to create classes, interact with students, and work with teachers. Building and connecting your learning community from students to administrators is its main strength. Additionally, it enables you to connect your entire school in addition to class-level channels. In this manner, you can establish virtual communities where students can interact with their instructors outside of the classroom.   
Additionally, Schoology facilitates communication and resource sharing amongst teachers in different districts and schools.

Additionally, it includes a professional learning community (PLC) where your academic members can exchange resources and ideas with others worldwide. Additionally, the platform easily interfaces with third-party programs, such as Dropbox and Google Drive. This implies that you can use the Schoology system with your current teaching resources.  
There are two packages for Schoology: Basic and Enterprise. You can use the learning management system's free version with the former. The latter, on the other hand, has all the advanced capabilities plus classroom integrations, communication channels, teacher resources, and more.

### Litmos

 Litmos is a cloud-based learning management system that offers a user-friendly platform for professional development.

Litmos provides comprehensive content libraries, workflow tool integrations, and support services, making it the ideal choice for corporate training and training programs. It is renowned for being user-friendly, compatible with mobile learning, and offering personalised branding.

By offering a wide variety of educational resources that students can access from any location, at any time, and on any device, this learning platform facilitates the skill development of staff members.

### Canvas by Instructure

Instructure's Canvas is a well-liked online learning management system and learning ecosystem in colleges and universities. The University of Utah and the University of Central Florida are among its customers. It claims to be the learning management system with the quickest rate of growth in the world. The LMS is a component of its digital learning solutions, which also include test engines, dashboards, and strong course assemblers.

Teachers can arrange course material and assignments into units using the platform's modules. Administrators can conform with current standards by using the Outcomes function, which integrates state-wide assessment and grading rubrics. It even has a thorough assessment management feature designed especially for school districts in grades K–12. Two of the most adaptable and effective grading tools available are Gradebook and SpeedGrader, which are also included in the LMS.

## 3.5 Legal, Social, Ethical, and Professional Issues

Table Legal, social, ethical and professional issues

|  |  |  |
| --- | --- | --- |
| Category | Issues | Description |
| Legal Issues | Data Privacy and Protection | Compliance with data protection laws (e.g., GDPR, COPPA) to secure personal data and protect students’ privacy. |
|  | Intellectual Property Rights | Respect for copyright and intellectual property when including educational materials (e.g., text, images, videos) on the platform. |
|  | Accessibility Standards | Ensuring LMS platforms meet accessibility laws and standards (e.g., ADA, WCAG) so that content is accessible to all students, including those with disabilities. |
| Social Issues | Digital Divide | Addressing inequities that arise from unequal access to the internet and digital devices, which may affect low-income or remote students. |
|  | Impact on Student Interaction | Understanding how LMS platforms can reduce face-to-face interactions, potentially impacting social skills and engagement, and designing features to foster virtual interaction. |
|  | Cultural Sensitivity | Ensuring platform design and content are culturally inclusive, accommodating diverse backgrounds and learning styles, and providing language and culturally relevant content options. |
| Ethical Issues | Data Security and Confidentiality | Implementing security measures (e.g., encryption, secure protocols) to protect sensitive student data and prevent unauthorized access. |
|  | Surveillance and Student Privacy | Balancing the use of monitoring tools with privacy considerations to avoid excessive surveillance and maintain student autonomy. |
|  | Fair Assessment and Bias | Ensuring that LMS algorithms and automated assessments are free from biases to provide fair grading and equitable learning opportunities for all students. |
| Professional Issues | Transparency and Communication | Being transparent about data collection, privacy policies, and data use, and communicating these clearly to students, parents, and educators. |
|  | Professional Development and Training | Providing educators and administrators with training on ethical use of the LMS, including privacy settings and responsible data usage. |
|  | Continuous Improvement and Accountability | Developers should be accountable for updating the LMS regularly to address security issues, improve accessibility, and adapt to changing educational standards, ensuring the platform is current and reliable. |

Table 1 shows the different legal, social, ethical, and professional issues that need to be considered when developing and using Learning Management Systems (LMS). These issues cover important topics like following data protection laws, making sure the platform is accessible to everyone, addressing unequal access to technology, protecting student privacy, and ensuring fair assessments.

# Project management

## 4.1 Flow that I followed

### Project Initiation & Kickoff

I began by defining the project scope and objectives, clearly outlining what I wanted to achieve with this project. I set up the project tracker using Trello. I also established a timeline with major milestones, such as completing the research phase, starting the development, and finishing the final report. I met with myself (a personal checkpoint) to finalize these plans and make sure everything was aligned.

### Planning & Risk Management

For risk management, I identified key risks early on, such as time constraints and potential difficulties with coding certain features. I added these to my task list as items to monitor closely. I also defined contingency plans, like allowing extra time in case I encountered issues or needed to adjust deadlines. I updated my tracker with these risks and prioritized them so that I would always have a backup plan in place.

### Execution: Research & Development

I moved forward with the research phase, gathering resources and reading key papers on the project’s topic. Once I had a strong foundation, I started working on the development aspect. I wrote the initial code for the main module and then regularly updated my project tracker to reflect the progress. At each stage, I checked my timeline and adjusted if I was falling behind or making quicker progress than expected.

### Quality Control & Testing

After completing each coding milestone, I dedicated time to testing my work. I ran tests on the code to ensure that it was functioning as expected and fixed any bugs that came up. I also reviewed my research and writing for quality, making sure everything was accurate and well-organized. In my self-meetings, I ensured that each component of the project was up to the standard I aimed for, documenting any issues found and addressing them right away.

### Documentation & Reporting

I worked on drafting the report simultaneously with the development. After completing the first draft, I reviewed it for clarity and completeness, checking that all sections (e.g., introduction, methodology, results) were well-organized. I used the tracker to manage my documentation tasks, like drafting sections and revising them based on feedback from my supervisor. I kept the supervisor in the loop with weekly updates, asking for feedback on specific sections.

### Stakeholder (Supervisor) Communication

Throughout the project, I maintained regular communication with my supervisor. I updated them on my progress and sent parts of my work for review, like the research findings and code. After receiving feedback, I made necessary changes and updated my tasks in the tracker. This helped me stay aligned with my supervisor’s expectations and ensured the project was on the right track.

### Final Review & Submission

As the project neared completion, I reviewed everything one last time. I checked all deliverables, including the report and code, to ensure they met the project requirements. I completed the final revisions and finalized the report. After making sure everything was perfect, I submitted the project and ensured that all files were uploaded to the designated platform on time.

## 4.2 Tool used for management:

For this project the tool I used is Trello.

* **Backlog (To Do):**
  + I added all the tasks I needed to complete (research, coding, report writing) into the "Backlog" section of my project tracker.
* **In Progress:**
  + As I moved through different stages of the project, I moved tasks to the "In Progress" column, like when I started coding or writing specific parts of the report.
* **Ready for QA:**
  + When I finished coding or writing, I would move tasks to the "Ready for QA" section to check them for quality.
* **Completed:**
  + Once I finished testing the tasks I moved them to the "Done" column.

### Weekly Self-Check and Progress Updates

Once a week, I held a personal check-in to review my progress against the project timeline. During this time, I:

* Reviewed what was going well and what wasn’t.
* Adjusted deadlines if necessary to stay on track.
* Added any new tasks that came up during the process.

I kept track of everything in my Trello and ensured that no task was overlooked.

## 4.3 Division of Project into Modules

I divided the project into modules for easier management.

### Module 1: Project Planning and Initiation

**Objective**: Define the project scope, plan tasks, and set up the initial infrastructure for the LMS.

**Tasks**:

1. **Define Project Scope**
   * Break down requirements into actionable tasks.
2. **Create Project Roadmap**
   * Develop a high-level project timeline.
   * Set key milestones and deadlines.
   * Prioritize the project features based on requirements.
3. **Set Up Trello for Task Management**
   * Create a Trello project board.
   * Set up project backlog and categorize tasks by sprints.

### Module 2: User Management

**Objective**: Implement user authentication, roles, and permissions.

**Tasks**:

1. **User Registration and Login**
   * Develop backend API for user registration (Django).
   * Implement login functionality with user session management.
   * Create frontend UI for registration and login (ReactJS).
2. **User Role Management**
   * Create user roles (admin, student, teacher).
   * Define permissions for each user role (e.g., admins can delete courses, teachers can create courses).
   * Ensure only authorized roles can access specific features.
3. **Password Reset Functionality**
   * Set up email-based password reset.
   * Develop backend to securely handle password resets.
   * Implement frontend UI to request and set new passwords.

### Module 3: Course Management

**Objective**: Allow teachers to create, edit, and manage courses, and students to enroll in them.

**Tasks**:

1. **Course Creation and Enrollment**
   * Develop backend API for creating and managing courses (Django).
   * Implement frontend for course listing and enrollment (ReactJS).
   * Create an enrollment system where students can join courses.
2. **Course Material Management**
   * Allow teachers to upload, edit, and delete course materials (videos, PDFs).
   * Develop file upload functionality and associate materials with specific courses.
   * Ensure students can only access materials of courses they are enrolled in.
3. **Course Deletion and Management by Teachers**
   * Implement course deletion functionality for teachers.
   * Ensure only the teacher who created a course can delete it.
   * Create a user-friendly interface for managing courses.

### Module 4: Communication and Collaboration Tools

**Objective**: Enable real-time chat and whiteboard functionalities for users to collaborate.

**Tasks**:

1. **Real-Time Chat System**
   * Develop backend for user-to-user chat functionality.
   * Implement real-time messaging between students and teachers.
   * Design UI for chat within the course dashboard.
   * Implement a collaborative whiteboard for enrolled students and teachers.
2. **Notifications System**
   * Set up notifications for course updates, new materials, and enrollments.

### Module 5: Testing and Quality Assurance

**Objective**: Perform thorough testing to ensure that the LMS works as expected and all features are functional.

**Tasks**:

1. **Unit Testing**
   * Write unit tests for backend models, views, and serializers (Django).
   * Test frontend components for functionality (ReactJS).
2. **Integration Testing**
   * Perform end-to-end testing of core functionalities such as registration, course enrollment, and chat.
   * Ensure that all modules work seamlessly together.
3. **User Acceptance Testing (UAT)**
   * Conduct testing with a sample group of users to validate features and usability.
   * Collect feedback and make adjustments as needed.

### Module 6: Deployment and Post-Launch Activities

**Objective**: Deploy the LMS to a live server and provide ongoing support.

**Tasks**:

1. **Production Deployment**
   * Set up a production environment
   * Provide support and updates based on user feedback.

# Requirements analysis

## 5.1 Functional Requirements

### User Registration and Authentication

Table FR User registration

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR1.1 | The system shall allow new users (teachers, students, and admins) to register by providing necessary information, including name, email, password, and role selection. |
| FR1.2 | The system shall allow registered users to log in with their email and password. |
| FR1.3 | The system shall validate user credentials and direct users to their respective dashboards based on their role (teacher, student, or admin). |
| FR1.4 | The system shall support password reset functionality by sending a reset link to the user’s registered email. |
| FR1.5 | The system shall allow users to update their profile information, including profile picture and contact details. |

### Course Management

Table FR Course Management

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR2.1 | The system shall allow teachers to create, edit, and delete courses with details such as course title, description, syllabus, schedule, and prerequisites. |
| FR2.2 | The system shall allow teachers to organize courses into modules or sections to structure the content effectively. |
| FR2.3 | The system shall allow students to enroll in courses that are open for enrollment. |
| FR2.4 | The system shall provide an enrollment approval mechanism if the course requires teacher approval for student enrollment. |
| FR2.5 | The system shall allow teachers to set enrollment limits for each course. |
| FR2.6 | The system shall notify teachers when a student enrolls in or leaves a course. |
| FR2.7 | The system shall notify students when new materials are added to the courses they are enrolled in. |
| FR2.8 | The system shall allow teachers to archive courses, making them accessible only to past participants for reference. |

### Content and Resource Management

Table FR Content and Resource Management

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR3.1 | The system shall allow teachers to upload course materials, including documents, videos, presentations, and other multimedia resources. |
| FR3.2 | The system shall organize uploaded materials by module or section within each course. |
| FR3.3 | The system shall allow students to view, download, and access course materials as per the teacher’s permissions. |
| FR3.4 | The system shall support version control for course materials, allowing teachers to update content while keeping track of previous versions. |
| FR3.5 | The system shall allow teachers to create and share links to external resources relevant to their course. |

### Assignments and Assessments

Table FR Assignments

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR4.1 | The system shall allow teachers to create assignments and assessments, specifying due dates, instructions, and submission formats. |
| FR4.2 | The system shall allow students to submit assignments through the LMS platform by uploading documents or inputting responses directly. |
| FR4.3 | The system shall provide automatic grading options for objective questions (e.g., multiple-choice, true/false). |
| FR4.4 | The system shall notify students of upcoming deadlines for assignments and assessments. |
| FR4.5 | The system shall allow teachers to review, grade, and provide feedback on student submissions within the platform. |
| FR4.6 | The system shall allow teachers to set grading criteria or rubrics for subjective assessments. |

### Discussion and Communication

Table FR Communication

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR5.1 | The system shall provide a discussion forum for each course, allowing students and teachers to post questions, answers, and engage in discussions. |
| FR5.2 | The system shall enable real-time chat functionality for teachers and students within the same course. |
| FR5.3 | The system shall allow users to update their availability or status in the chat module (e.g., online, offline, busy). |
| FR5.4 | The system shall support private messaging between teachers and students for one-on-one communication. |
| FR5.5 | The system shall allow users to receive notifications for new messages, forum replies, and chat updates. |

### Progress Tracking and Reporting

Table FR Progress Tracking and Reporting

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR6.1 | The system shall track student progress for each course, including assignment submissions, quiz scores, and module completion. |
| FR6.2 | The system shall display progress reports for students, showing their performance, grades, and feedback for each completed module. |
| FR6.3 | The system shall allow students to view their overall course progress and grades within their dashboard. |
| FR6.4 | The system shall provide teachers with analytics on student performance, such as average scores and completion rates. |
| FR6.5 | The system shall allow teachers and admins to export student progress reports as CSV or PDF files for record-keeping. |

### Notifications and Alerts

Table FR Notifications and Alerts

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR7.1 | The system shall send email notifications to users for key events, including enrollment confirmation, assignment deadlines, and new material uploads. |
| FR7.2 | The system shall display in-app notifications for new messages, course updates, and deadlines within the user dashboard. |
| FR7.3 | The system shall allow users to customize their notification preferences, such as opting in or out of specific types of notifications. |

### User Roles and Permissions

Table FR User Roles and Permissions

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR8.1 | The system shall assign different permissions based on user roles (student, teacher, admin), restricting access to specific features accordingly. |
| FR8.2 | The system shall allow admins to manage user accounts, including creating, suspending, or deleting users. |
| FR8.3 | The system shall provide admins with an overview of active courses, enrolled students, and teacher activities. |
| FR8.4 | The system shall restrict access to course management and grading features to teachers and admins only. |

### Attendance

Table FR Attendance

|  |  |
| --- | --- |
| ID | Functional Requirement |
| FR9.1 | The system shall allow teachers to record attendance for each course session. |
| FR9.2 | The system shall provide students with a view of their attendance record for each course. |
| FR9.3 | The system shall allow teachers to view attendance reports and calculate attendance-based participation scores. |
| FR9.4 | The system shall notify students if they are at risk of low attendance based on course requirements. |

## 5.2 Non-Functional Requirements

### Performance

Table NFR Performance

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR1.1 | The system shall support concurrent access for at least 1,000 users without performance degradation. |
| NFR1.2 | The system shall load the main dashboard page within 2 seconds under normal load conditions. |
| NFR1.3 | The system shall handle large file uploads (up to 500MB) without crashing or timing out. |
| NFR1.4 | The system shall optimize data queries and caching mechanisms to ensure fast retrieval of course data. |

### Security

Table NFR Security

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR2.1 | The system shall enforce SSL encryption to secure all data transmitted between clients and the server. |
| NFR2.2 | The system shall implement user authentication mechanisms to prevent unauthorized access to restricted features. |
| NFR2.3 | The system shall automatically log out inactive users after 15 minutes of inactivity. |
| NFR2.4 | The system shall store user passwords in a hashed format to prevent unauthorized access in case of a data breach. |
| NFR2.5 | The system shall comply with GDPR and data protection regulations, allowing users to delete their accounts and personal data upon request. |

### Usability

Table NFR Usability

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR3.1 | The system shall provide a user-friendly interface with intuitive navigation for students, teachers, and admins. |
| NFR3.2 | The system shall ensure consistent layout and design across all pages to improve user experience. |
| NFR3.3 | The system shall support accessibility features, such as screen reader compatibility and keyboard navigation, to accommodate all users. |
| NFR3.4 | The system shall provide a responsive design that adapts to various screen sizes, including desktop, tablet, and mobile devices. |

### Reliability

Table NFR Reliability

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR4.1 | The system shall have an uptime of 99.9% to ensure high availability. |
| NFR4.2 | The system shall perform regular data backups daily to prevent data loss in case of system failure. |
| NFR4.3 | The system shall log all errors and provide alerts for critical issues to ensure timely troubleshooting. |
| NFR4.4 | The system shall support graceful error handling and display meaningful error messages to users when issues occur. |

### Scalability

Table NFR Scalability

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR5.1 | The system shall be able to scale horizontally by adding more servers to handle increased user load. |
| NFR5.2 | The system shall allow for the easy addition of new features or modules without significant changes to the existing architecture. |
| NFR5.3 | The system shall be capable of handling an increase in enrolled courses and resources without impacting performance. |

### Maintainability

Table NFR Maintainability

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR6.1 | The system shall be developed using modular code to facilitate maintenance and updates. |
| NFR6.2 | The system shall conform to standard coding practices, including adherence to the PEP8 style guide for Python. |
| NFR6.3 | The system shall include detailed documentation for developers to simplify future development and troubleshooting. |
| NFR6.4 | The system shall implement version control for tracking changes to code and ensuring compatibility in updates. |

### Compatibility

Table NFR Compatibility

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR7.1 | The system shall be compatible with major web browsers (e.g., Chrome, Firefox, Safari, Edge) to ensure accessibility for all users. |
| NFR7.2 | The system shall integrate with external systems (e.g., payment gateways, analytics) via APIs for extended functionality. |
| NFR7.3 | The system shall support multiple databases (e.g., PostgreSQL, MySQL) to allow flexibility in deployment. |

### Data Integrity

Table NFR Data Integrity

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR8.1 | The system shall ensure accurate data synchronization across modules to prevent inconsistencies. |
| NFR8.2 | The system shall use input validation to prevent incorrect data entry from users. |
| NFR8.3 | The system shall maintain data integrity by enforcing referential integrity within the database. |

### Portability

Table NFR Portability

|  |  |
| --- | --- |
| ID | Non-Functional Requirement |
| NFR9.1 | The system shall be deployable on various operating systems, including Windows, Linux, and macOS. |
| NFR9.2 | The system shall be portable across cloud environments (e.g., AWS, Azure) to support different hosting preferences. |

## Development requirements

Table Development Requirements

|  |  |  |
| --- | --- | --- |
| Requirement ID | Category | Description |
| DR1 | Backend Framework | Utilize Django as the backend framework to manage server-side logic and data processing. |
| DR2 | Database Design | Create a well-structured and normalized database schema using Django's ORM to efficiently store account information, course details, and relationships. |
| DR3 | REST API | Implement a RESTful API using Django Rest Framework (DRF) to facilitate communication between the frontend and backend, following standard REST principles. |
| DR4 | Unit Testing | Develop comprehensive unit tests for backend logic and API endpoints, ensuring both positive and negative test cases are covered to validate functionality. |
| DR5 | Code Quality | Maintain adherence to best practices, including clean code principles and documentation, to ensure a readable and maintainable codebase. |
| DR6 | Modular Architecture | Ensure that the application is designed with modularity in mind, facilitating easier maintenance and scalability. |
| DR7 | Dependency Management | Utilize a package manager (like pip) to handle dependencies and ensure compatibility across different environments. |
| DR8 | Configuration Management | Implement environment-based configurations (development, testing, production) for the application to ensure appropriate settings are applied in each context. |
| DR9 | Error Handling | Establish robust error handling mechanisms throughout the application to gracefully manage and log exceptions. |
| DR10 | Performance Optimization | Optimize database queries and API responses to improve performance and reduce latency for end users. |

* 1. Software Development Lifecycle Model

### 5.4.1 Software Development Methodology

In this project the methodology that will be used is Agile. Flexibility, iteration, and continuous improvement are key components of the Agile method, a modern approach to software development and project management. Agile was first used in the software sector in the early 2000s, but it has subsequently spread to other sectors as a successful approach to managing complex tasks with quickly shifting needs.

The foundation of the Agile methodology is the division of huge projects into smaller, more manageable units known as "sprints," which are addressed through ongoing feedback loops and cross-functional teamwork. Teams can swiftly adjust to changes, provide value progressively, and make sure the finished product satisfies stakeholders' changing needs due to this iterative approach.

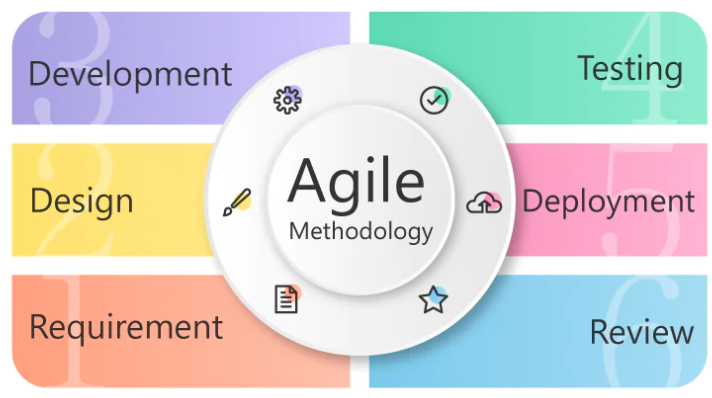


Figure SDLC Life cycle

Figure 1 displays the basic software development life cycle that we have followed in our project.  
Agile places a strong emphasis on people and relationships rather than procedures and equipment. Agile teams are cross-functional and self-organising, bringing together experts with a range of expertise to jointly plan, carry out, and evaluate each project iteration. Agile teams stay connected and adaptable to change through frequent reevaluations of priorities and regular in-person communication.

### 5.4.2 Reasons for Choosing Agile

The main reason for using Agile in my project is because it helps to guarantee that the development of the project is on schedule and under budget. Additionally, it facilitates better communication between the product owner and the development team.   
Furthermore, the Agile development process might assist in lowering the hazards connected to intricate projects. It enables rapid and simple adjustments to be made in the project without impacting the project's overall timeframe.

The agile development process has numerous advantages, some of which are as follows:

**Greater adaptability:** Compared to traditional project management approaches, agile development offers greater adaptability. Changes can be made more quickly by development teams.

**Better communication:** Agile development facilitates better communication between the product owner and the development team. As a result, cooperation and feedback are given more attention.

**Decreased risks:** Complex projects can have fewer hazards because of agile development. Project managers can analyse complex projects and meet shareholder needs by segmenting them into smaller sprints.

**Enhanced client satisfaction:** Agile development settings frequently result in happier clients. This is because the client participates in the development process and offers input at every project level.

## Risk assessment

Table Risks and their mitigation strategies

|  |  |  |  |
| --- | --- | --- | --- |
| Risk ID | Risk Description | Impact | Mitigation Strategy (What I Did / Would Do) |
| R1 | Requirement changes after development starts | Medium – causes rework and delays | I kept requirements flexible and had weekly check-ins with my supervisor to confirm each feature. |
| R2 | Integration issues between frontend and backend | High – system might fail to function | I used Django REST API with proper versioning and tested endpoints thoroughly during development. |
| R3 | Limited time to complete all features | High – incomplete or rushed features | I divided the work into sprints and prioritized core modules like registration, courses, and chat. |
| R4 | Bugs in production during deployment | Medium – disrupts user experience | I tested each module separately before final integration and fixed errors early. |
| R5 | Security concerns like duplicate registrations or data leaks | High – data integrity issues | I implemented validations, used hashed passwords, and handled error cases like duplicate emails. |
| R6 | Lack of proper UI/UX feedback | Medium – poor user experience | I reviewed the frontend during each sprint and kept the design simple, responsive, and user-focused. |
| R7 | Version control conflicts | Low – could lose code | I used Git consistently, committed frequently, and kept backup copies for each milestone. |
| R8 | Deployment problems due to tech stack configuration | Medium – downtime or setup delays | I used proper environment setup and documented all steps for backend and frontend deployment. |
| R9 | Single developer workload | High – burnout or missed deadlines | I managed time using a sprint plan and tracked all tasks in Trello to stay on schedule. |
| R10 | Testing gaps due to limited QA | Medium – some bugs might go unnoticed | I created test cases for each module and manually verified them during sprint reviews. |

# System Design

## 6.1 Use Cases

**1. User Registration and Login**

Table UC for registration and login

|  |  |
| --- | --- |
| Use Case ID | UC-01 |
| Use Case Name | User Registration and Login |
| Actors | Student, Teacher, Admin |
| Description | Users can register and log in to the platform. |
| Preconditions | The user has access to the LMS application. |
| Postconditions | User is authenticated and redirected to their dashboard. |
| Main Flow | 1. User opens the registration or login page. 2. User enters required credentials (e.g., email, password). 3. System verifies credentials and grants access. 4. User is redirected to their role-specific dashboard. |
| Alternative Flow | 2a. If the user already exists (during registration), the system displays an error message. 3a. If credentials are incorrect (during login), the system shows an "Invalid credentials" message. |

**2. Course Creation and Management**

Table UC for course creation and management

|  |  |
| --- | --- |
| Use Case ID | UC-02 |
| Use Case Name | Course Creation and Management |
| Actors | Teacher |
| Description | Teachers can create, edit, and delete courses. |
| Preconditions | Teacher is logged in. |
| Postconditions | Course is available in the course list for students to view and enroll in. |
| Main Flow | 1. Teacher navigates to the course creation page. 2. Teacher fills in details (e.g., course title, description, materials). 3. Teacher submits the course. 4. System saves the course and makes it visible to students. |
| Alternative Flow | 2a. If the course title is missing, the system prompts the teacher to enter all required details. |

**3. Course Enrollment**

Table UC for courses enrollment

|  |  |
| --- | --- |
| Use Case ID | UC-03 |
| Use Case Name | Course Enrollment |
| Actors | Student |
| Description | Students can browse and enroll in available courses. |
| Preconditions | Student is logged in and able to view available courses. |
| Postconditions | Student is enrolled and can access course materials. |
| Main Flow | 1. Student navigates to the course catalog. 2. Student selects a course to enroll in. 3. Student clicks on the "Enroll" button. 4. System enrolls the student and updates their dashboard with the course. |
| Alternative Flow | 3a. If the course is full or enrollment is closed, the system displays an error message. |

* 1. **Accessing Course Materials**

Table UC for accessing course content

|  |  |
| --- | --- |
| Use Case ID | UC-04 |
| Use Case Name | Accessing Course Materials |
| Actors | Student |
| Description | Enrolled students can access and download course materials. |
| Preconditions | Student is enrolled in the course. |
| Postconditions | Student can view or download materials as needed. |
| Main Flow | 1. Student navigates to their course dashboard. 2. Student selects a course. 3. System displays available course materials. 4. Student views or downloads the materials. |
| Alternative Flow | 2a. If the student is not enrolled, the system prevents access to materials. |

* 1. **Providing Course Feedback**

Table UC for providing feedback and reviews

|  |  |
| --- | --- |
| Use Case ID | UC-05 |
| Use Case Name | Providing Course Feedback |
| Actors | Student |
| Description | Students can leave feedback on courses they have completed. |
| Preconditions | Student is enrolled in the course. |
| Postconditions | Feedback is saved and visible to the teacher. |
| Main Flow | 1. Student navigates to the completed course. 2. Student clicks on "Leave Feedback". 3. Student enters feedback comments and submits. 4. System saves the feedback and notifies the teacher. |
| Alternative Flow | 3a. If the feedback form is incomplete, the system prompts the student to complete all required fields. |

* 1. **Real-Time Communication (Chat)**

Table UC for chat

|  |  |
| --- | --- |
| Use Case ID | UC-06 |
| Use Case Name | Real-Time Communication (Chat) |
| Actors | Student, Teacher |
| Description | Students and teachers can chat in real time within the LMS. |
| Preconditions | Both users are logged in and have access to the chat feature. |
| Postconditions | Message is delivered in real time to the intended recipient(s). |
| Main Flow | 1. User navigates to the chat section. 2. User selects a chat thread or starts a new one. 3. User types a message and sends it. 4. System delivers the message to the other user in real time. |
| Alternative Flow | 2a. If the user does not have permission, the system prevents access to the chat. |

* 1. **Collaborative Whiteboarding**

Table UC for collaborative whiteboard

|  |  |
| --- | --- |
| Use Case ID | UC-07 |
| Use Case Name | Collaborative Whiteboarding |
| Actors | Student, Teacher |
| Description | Allows users to collaborate in real time on a shared whiteboard. |
| Preconditions | Both users are logged in and have access to the whiteboard tool. |
| Postconditions | Real-time updates are displayed to all users on the whiteboard. |
| Main Flow | 1. User opens the whiteboard for a course. 2. User draws or adds notes on the whiteboard. 3. System synchronizes changes in real time. 4. Other users in the session see updates instantly. |
| Alternative Flow | 2a. If the user is not enrolled in the course, the system restricts access to the whiteboard. |

* 1. **Admin Management of Users and Courses**

Table admin management of users

|  |  |
| --- | --- |
| Use Case ID | UC-08 |
| Use Case Name | Admin Management of Users and Courses |
| Actors | Admin |
| Description | Admin can manage users (students and teachers) and oversee course content. |
| Preconditions | Admin is logged in and has administrative privileges. |
| Postconditions | Changes are saved and reflected in the LMS system. |
| Main Flow | 1. Admin navigates to the management dashboard. 2. Admin selects a user or course to manage. 3. Admin makes necessary updates (e.g., deactivates a user or deletes a course). 4. System saves and applies changes. |
| Alternative Flow | 2a. If the admin attempts to make unauthorized changes, the system displays an error message. |

## 6.2 Sequence Diagrams

A diagram of a system

Description automatically generated

Figure User Login and Registration

A diagram of a system

Description automatically generated

Figure Admin Management of Users and Courses

A screenshot of a diagram

Description automatically generated

Figure Collaborative whiteboarding

A diagram of a chat

Description automatically generated

Figure Real-Time Communication (Chat)

A diagram of a system

Description automatically generated

Figure Providing Course Feedback

A diagram of a student

Description automatically generated

Figure Accessing Course Materials

A diagram of a course

Description automatically generated

Figure Course Creation and Management

A diagram of a student

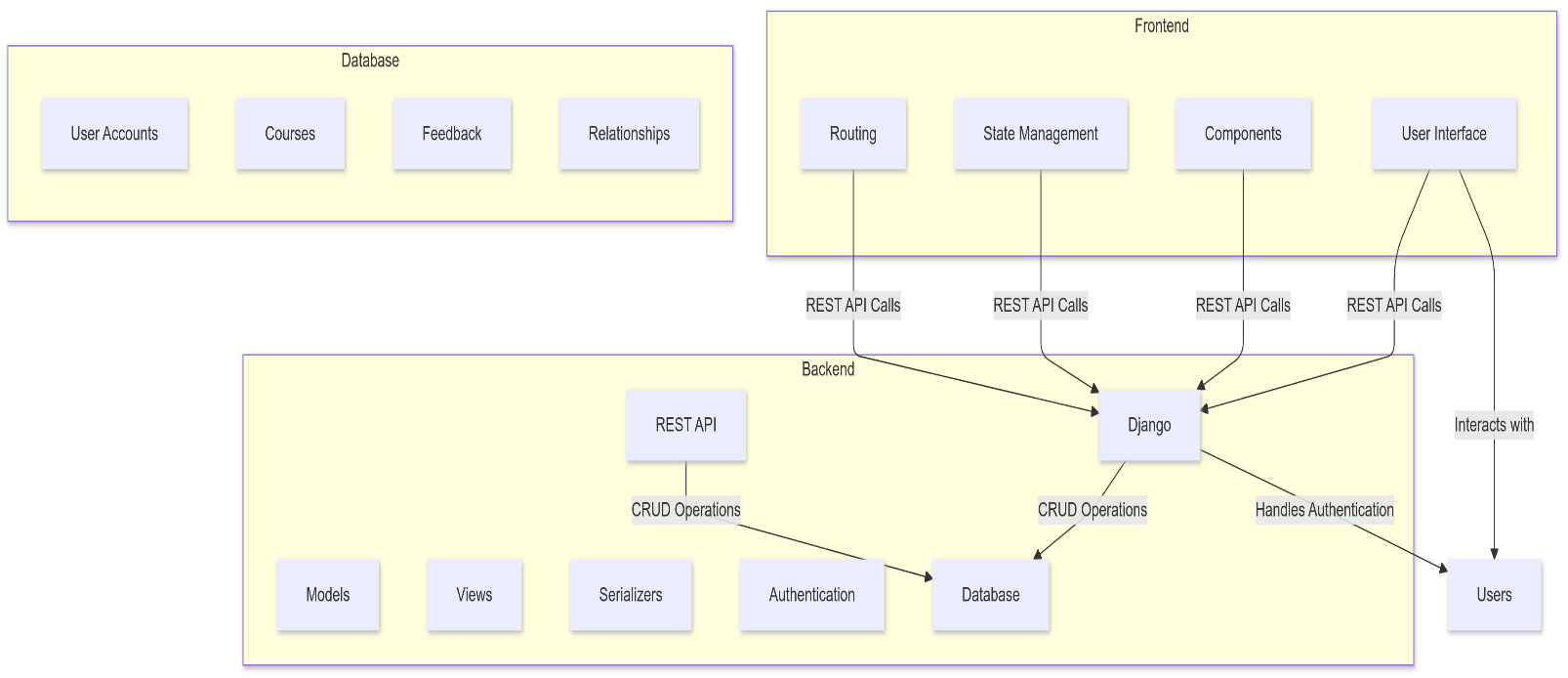
Description automatically generated

Figure Course Enrollment

A diagram of a software application

Description automatically generated

## UML Diagrams



A diagram of a company

Description automatically generated

Figure UML Diagram

# Implementation

## 7.1 Code organization and Structure

### Backend Code Structure and Component Descriptions

The code for the LMS\_API backend has been organized following best practices for Django applications. Below is an explanation of the chosen code organization scheme:

1. **Project Root Directory**: 'Lms\_api' serves as the root directory containing all necessary directories and files to operate the application.

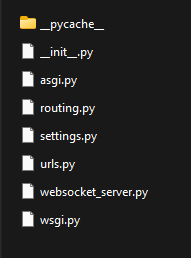


Figure Project Directory

1. **ASGI Configuration**: In Figure 11 the 'asgi' directory holds the file responsible for handling asynchronous tasks.
2. **URL Routing Configurations**: This file is used to manage the mapping between URL patterns and corresponding view functions.
3. **Main Package**: The 'main' package comprises 'serializers', 'models', 'views', 'urls', 'routing', 'consumers', and 'tests' subdirectories.

**5. WSGI Script:** The WSGI script defines entry points for the application.

1. **Serializers**: Serialisation is the process of writing an object's state to a file, but in a strict sense, it's the process of transforming an object from a Java-supported form into a file-supported or network-supported form. Serializer classes reside in this directory, facilitating conversion between native Python data types and JSON representations compatible with REST frameworks.
2. **Models**: The 'models' directory contains the data structures represented in the application, using Django's Object-Relational Mapper (ORM).
3. **Views**: The code of views defines various Django REST API views for a learning management system, handling CRUD operations for entities like teachers, students, courses, categories, quizzes, and enrollments. Pagination is used in certain views, and WebSocket support enables real-time messaging.
4. **View URL Pattern Definitions**: Aligned with respective view functions, URL patterns are defined in this directory, enhancing modularity and separation of concerns.
5. **WebSocket Consumers**: Handling WebSocket connections, the 'consumers' directory stores specialized classes managing bi-directional message passing.
6. **Unit Tests**: Important for quality assurance, the 'tests' directory accommodates unit tests confirming the integrity and accuracy of the application.

### Frontend Code Structure and Component Descriptions

**A screenshot of a computer

Description automatically generated**

Figure Code structure

1. **"components"**: Figure 12 showing essential folders (src, public, node\_modules) and configuration files (package.json, .gitignore) used in the Edunest application .It includes the majority of visual elements and interactive pieces, this folder houses numerous subfolders, each representing unique aspects of the application. Examples include:
   * **"Teacher"**: Accommodates components specifically geared toward the instructor role, such as "TeacherDashboard", "TeacherProfileSetting", and "TeacherChangePassword".
   * **"User"**: Encapsulates components targeting regular users, comprising "Login", "Register", "Dashboard", "MyCourses", "FavoriteCourses", and many others.
   * **"About", "CourseDetail", "Faq"**: Store domain-specific components focusing on conveying course particulars, FAQs, and general application info.
   * **"Footer", "Header"**: Hold shared layout components rendering common headers and footers found on every screen.
2. **"public"**: Provides a location for publicly accessible static assets, such as image files, fonts, and global stylesheet declarations.
3. **"node\_modules"**: Preserves third-party libraries and modules downloaded via Node Package Manager (npm) or yarn.

## 7.2 Designed Application

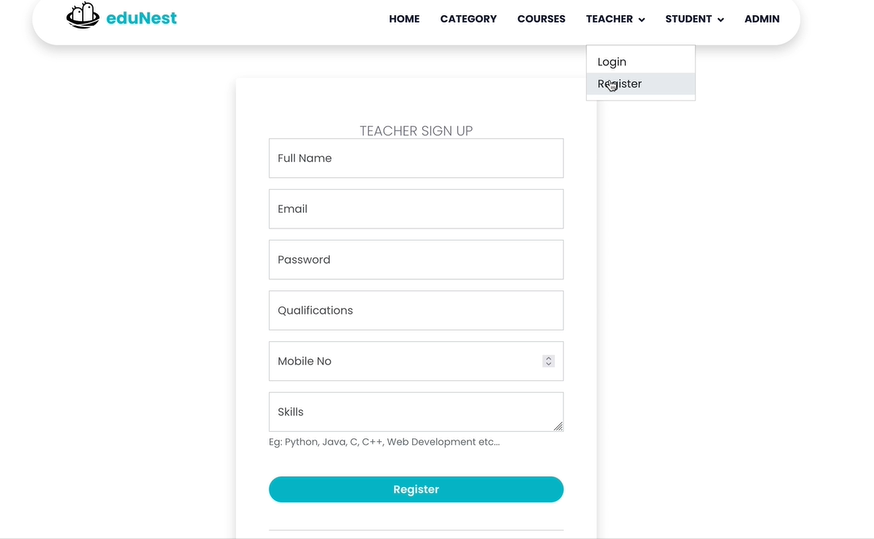


Figure Teacher authentication and signup

A screenshot of a computer

Description automatically generated

Figure Teacher Login

A screen shot of a computer

Description automatically generated

Figure Student signup

A screenshot of a computer

Description automatically generated

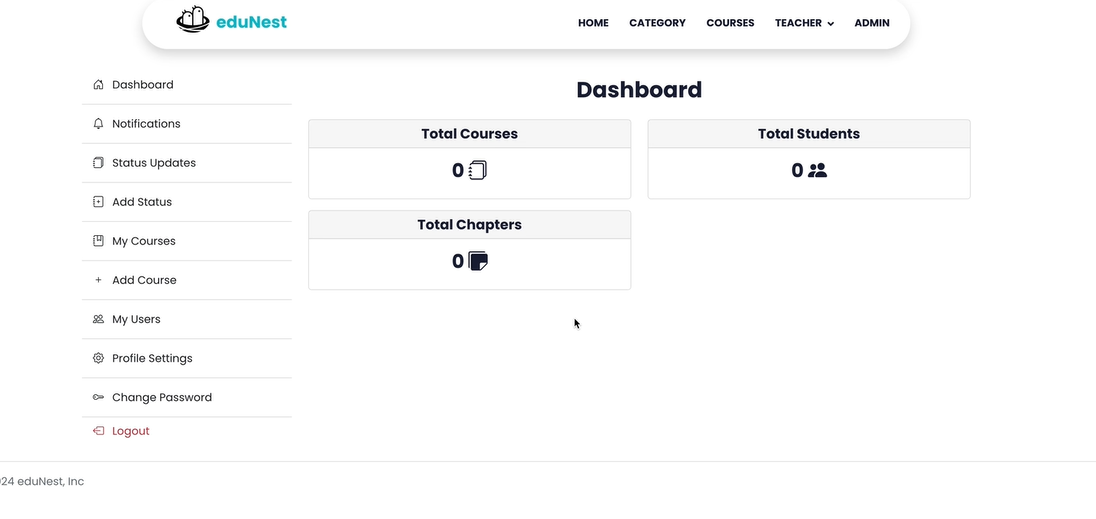


Figure Student dashboard

A screenshot of a computer

Description automatically generated

Figure Feature to add courses

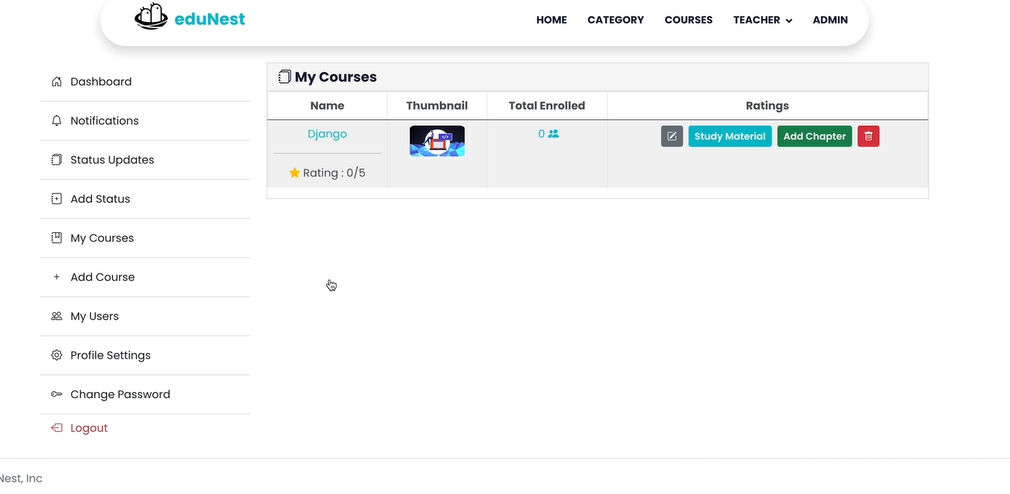


Figure My courses

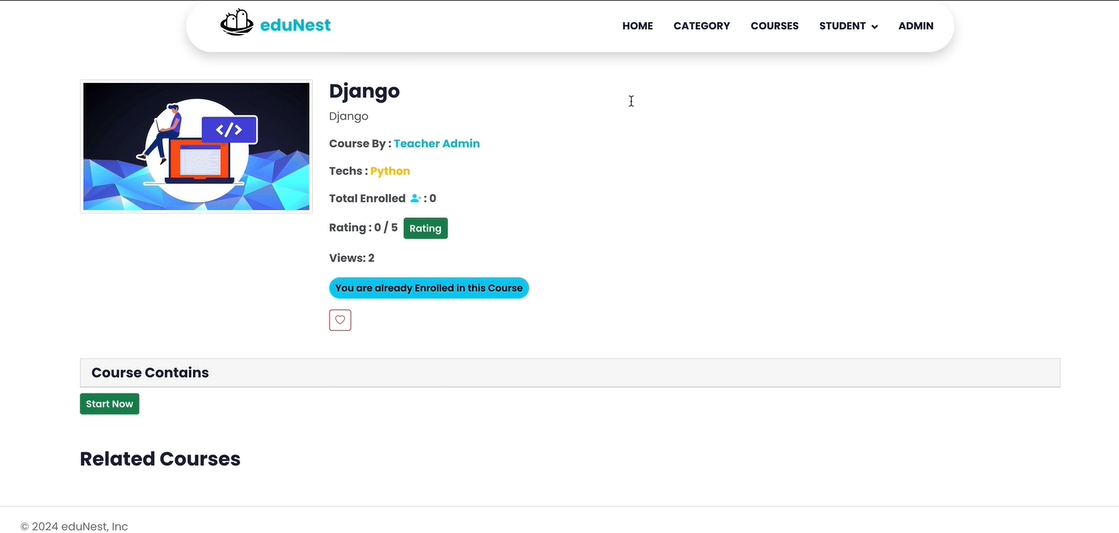
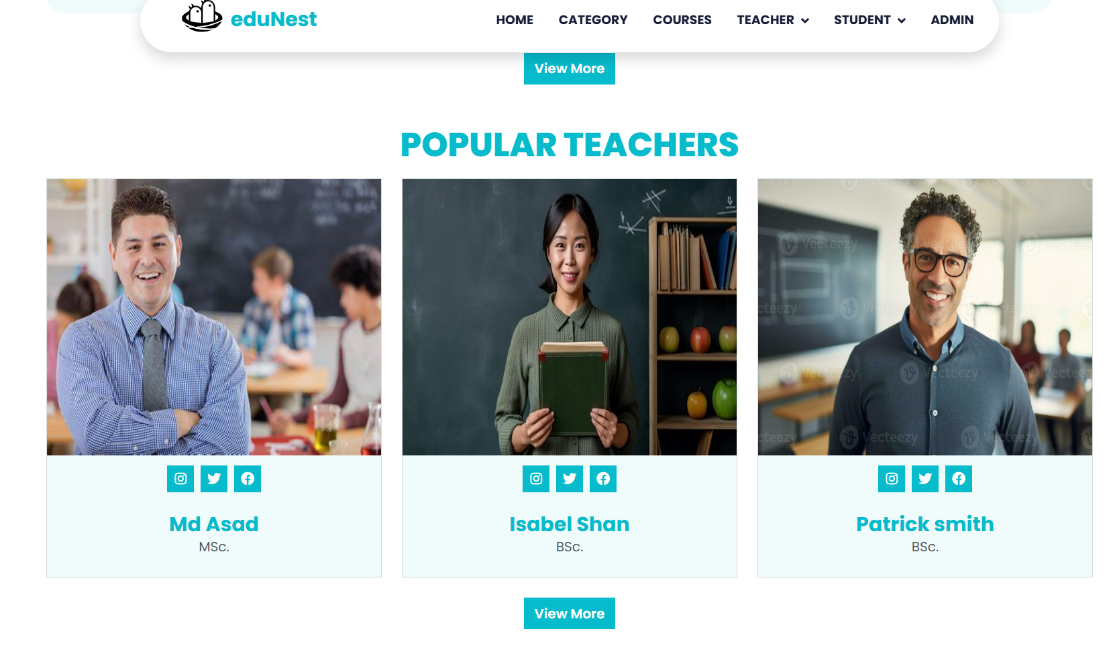


Figure Course information



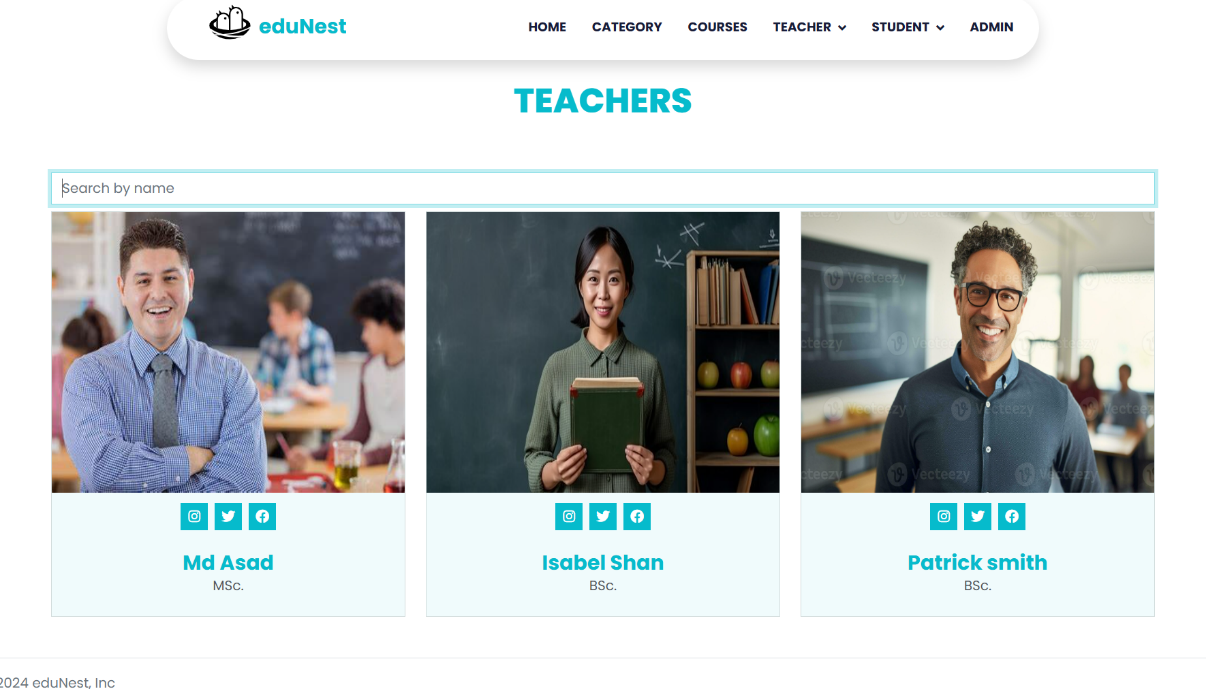


Figure Teachers information

A screenshot of a web page

Description automatically generated

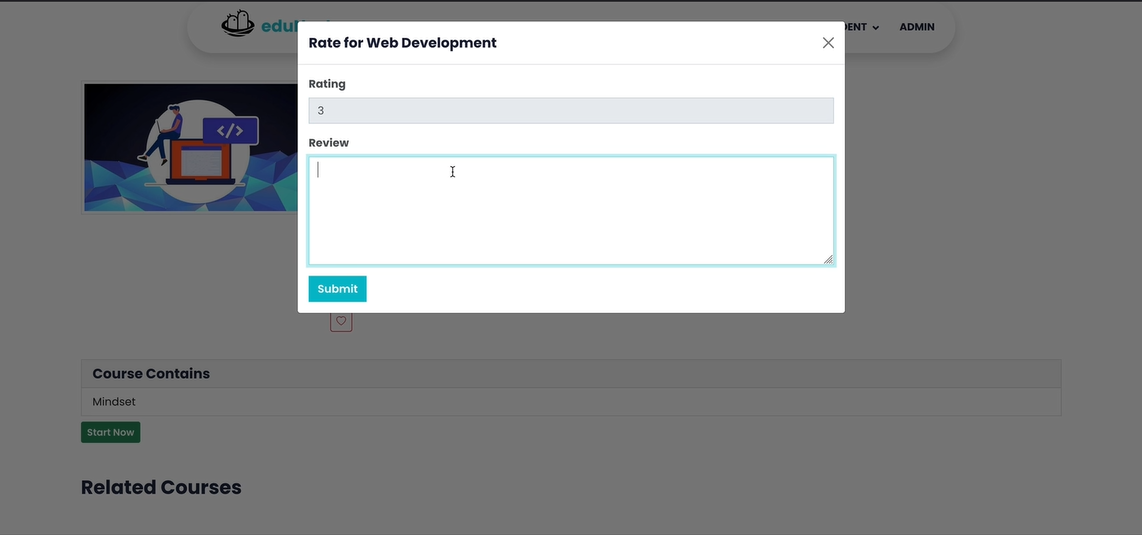


Figure Ability to add course rating and feedback

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figure Real-time communication

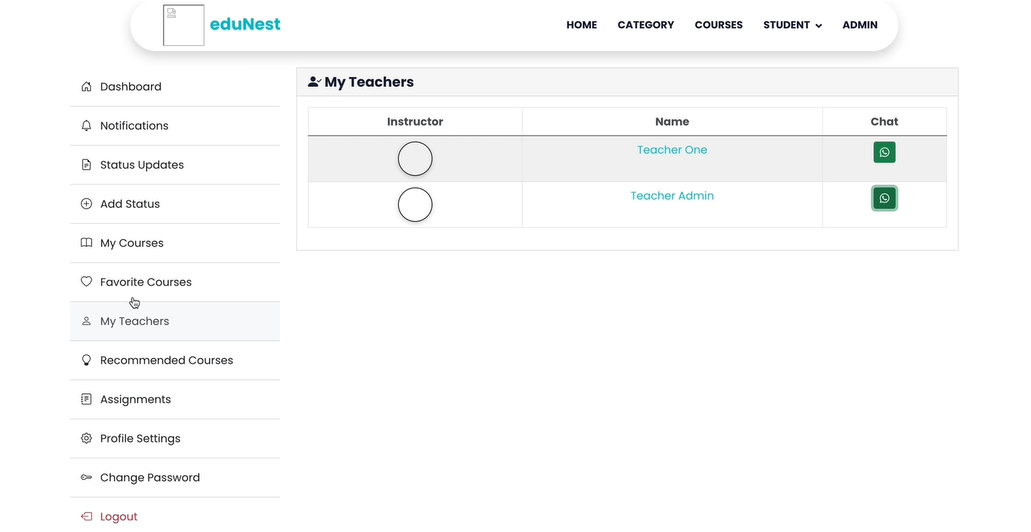


Figure My teachers

A screenshot of a computer

Description automatically generated

Figure Enrolled students

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Figure Add course material

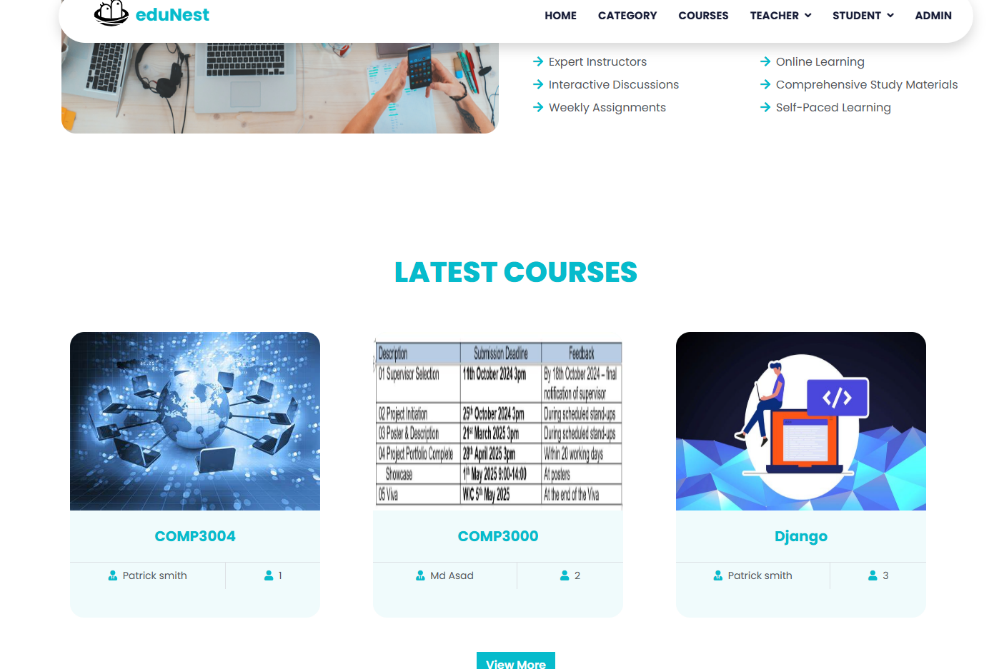


Figure latest courses

As shown in Figure 13, teachers begin by signing up and authenticating their accounts, followed by logging in as illustrated in Figure 14. Students register through a dedicated signup process (Figure 15) and gain access to their personalized dashboard (Figure 16). Figure 17 demonstrates the feature that allows teachers to add new courses, which are then listed under "My Courses" as shown in Figure 18. Detailed course information is available in Figure 19, alongside teacher profiles (Figure 20).

Figure 21 highlights the functionality for students to submit course ratings and feedback. Real-time communication between users is supported as shown in Figure 22. The "My Teachers" section (Figure 23) allows students to view assigned instructors, while teachers can manage enrolled students, as displayed in Figure 24. Course material can be added by instructors using the feature in Figure 25, and students can browse newly available content under "Latest Courses" (Figure 26).

# System Testing

## 8.1 Test Cases

### User Registration and Login

Table Test cases for user registration and login

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Description | Preconditions | Test Steps | Expected Result |
| TC-UR-01 | Verify successful user registration | User is not registered | 1. Submit registration form with valid details. | User account is created; success message is displayed. |
| TC-UR-02 | Verify error on duplicate registration | User already registered | 1. Submit registration form with the same email as an existing user. | Error message indicating email is already in use. |
| TC-UR-03 | Verify successful user login with valid credentials | Registered user | 1. Enter valid email and password. 2. Submit login. | User is logged in and redirected to the dashboard. |
| TC-UR-04 | Verify error on login with invalid credentials | Registered user | 1. Enter invalid email/password. 2. Submit login. | Error message indicating incorrect credentials. |
| TC-UR-05 | Verify password reset functionality | Registered user | 1. Request password reset. 2. Click on the reset link received via email. 3. Set a new password. | Password is reset, and user can log in with the new password. |

### Course Enrollment and Management

Table Test cases for course enrollment and management

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Description | Preconditions | Test Steps | Expected Result |
| TC-CE-01 | Verify successful course enrollment by a student | Student logged in, course exists | 1. Select a course. 2. Click enroll. | Student is enrolled; course appears in the student’s dashboard. |
| TC-CE-02 | Verify error on enrolling in a full course | Student logged in, course is full | 1. Select a full course. 2. Click enroll. | Error message indicating the course is full. |
| TC-CE-03 | Verify successful course creation by a teacher | Teacher logged in | 1. Go to the "Create Course" page. 2. Enter course details and submit. | Course is created; appears in the list of available courses. |
| TC-CE-04 | Verify course deletion by a teacher | Teacher logged in, course created | 1. Go to the "Manage Courses" page. 2. Select a course and click delete. | Course is deleted; no longer visible to students. |
| TC-CE-05 | Verify that only the teacher who created a course can delete it | Teacher logged in, course created by another teacher | 1. Try to delete a course created by another teacher. | Error message indicating unauthorized action. |

### Content Management

Table Test Cases for content management

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Description | Preconditions | Test Steps | Expected Result |
| TC-CM-01 | Verify adding new material to a course by a teacher | Teacher logged in, course exists | 1. Go to the "Add Material" page. 2. Upload a file and save. | Material is added; visible to students enrolled in the course. |
| TC-CM-02 | Verify editing existing material | Teacher logged in, material exists | 1. Go to the "Manage Materials" page. 2. Select material and edit details. 3. Save changes. | Material details are updated as per changes. |
| TC-CM-03 | Verify successful deletion of course material | Teacher logged in, material exists | 1. Go to the "Manage Materials" page. 2. Select material and click delete. | Material is deleted; no longer visible to students. |
| TC-CM-04 | Verify that students cannot edit or delete course materials | Student logged in | 1. Attempt to edit or delete a course material. | Error message indicating insufficient permissions. |
| TC-CM-05 | Verify material visibility for enrolled students only | Student enrolled in course | 1. Go to course materials. | Student can view materials for enrolled courses only. |

### User Notifications

Table Test Cases for user notifications

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Description | Preconditions | Test Steps | Expected Result |
| TC-UN-01 | Verify that teachers receive notification upon student enrollment | Teacher logged in, course exists | 1. Student enrolls in a teacher's course. | Teacher receives notification of student enrollment. |
| TC-UN-02 | Verify that students receive notification on new materials added | Student logged in, enrolled in course | 1. Teacher adds new material to a course. | Student receives notification for new material. |
| TC-UN-03 | Verify unread notifications count displays accurately | User has multiple unread notifications | 1. Log in as user. | Unread notifications count displays correctly on dashboard. |
| TC-UN-04 | Verify notifications disappear after being read | User has unread notifications | 1. Go to notifications page. 2. Read all notifications. | Notifications are marked as read and disappear from unread count. |
| TC-UN-05 | Verify notification persistence after logout and login | User has unread notifications | 1. Log out with unread notifications. 2. Log back in. | Unread notifications remain upon re-login. |

### Chat and Collaboration Tools

Table Test Cases for testing chat and collaboration

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Test Case ID | Test Description | Preconditions | Test Steps | Expected Result |
| TC-CC-01 | Verify chat functionality between users | Both users are logged in | 1. User A sends a message to User B. | User B receives the message in real-time. |
| TC-CC-02 | Verify whiteboard access and editing for enrolled students | User logged in, enrolled in course | 1. Access whiteboard tool for an enrolled course. 2. Add drawing/notes. | Changes appear in real-time for other users in the session. |
| TC-CC-03 | Verify that non-enrolled users cannot access whiteboard | User logged in, not enrolled in course | 1. Attempt to access whiteboard tool for a course the user is not enrolled in. | Access is denied with appropriate error message. |
| TC-CC-04 | Verify chat history persistence after logout | Both users have active chat history | 1. Log out as User A. 2. Log back in. 3. Go to chat window with User B. | Chat history is retained and viewable after re-login. |
| TC-CC-05 | Verify maximum file size restriction in whiteboard uploads | User logged in | 1. Attempt to upload a file exceeding the maximum size limit to the whiteboard. | File upload is blocked with an error message about size limits. |

# Outcome Evaluation

Table planned vs achieved outcomes

|  |  |  |
| --- | --- | --- |
| Objective | Planned Goal | Achieved Outcome |
| User Registration & Login | Allow users to register, log in, and access role-based dashboards | Developed complete authentication system with login, signup, and role-based access |
| Course Creation & Management | Enable creation, editing, and assignment of courses | Successfully built course module with forms to create, update, and manage courses |
| Assignment and Quiz Features | Let teachers upload assignments/quizzes and allow students to submit responses | Developed both modules, with quiz timer, automatic checking, and assignment upload |
| Chat Feature | Add real-time chat for communication between users | Integrated real-time chat system with message sync and user support |
| Whiteboard Tool | Add an online whiteboard for visual collaboration during sessions | Integrated third-party whiteboard tool and ensured smooth operation |
| Notifications | Send alerts and updates to users for tasks and changes | Implemented real-time notifications using Firebase |
| Responsive and Clean User Interface | Ensure UI works well on desktop and mobile with clear navigation | Used ReactJS to design a responsive and clean front-end across all screen sizes |
| Role-Based Access Control | Provide different access to admins, teachers, and students | Role-based permission system added, restricting views and actions based on role |
| Real-time Features Performance | Ensure chat and whiteboard work smoothly under different conditions | Completed but further optimization is needed for better performance at scale |
| Documentation | Provide clear documentation for future use and maintenance | All features documented with setup steps, code explanation, and future notes |
| Testing and Bug Fixing | Check for errors and fix issues in each module through testing | Conducted manual and functional testing regularly, fixed bugs during development |

# Project Post-Mortem

## 10.1 Overview:

* The project followed an agile method and was divided into several sprints.
* Trello was used to manage tasks, track progress, and ensure everything stayed on schedule.
* Milestones were planned at the start of each sprint and adjusted when necessary to handle changes or challenges.
* Each module was completed step by step with focused attention on quality and timely delivery.

**What went well:**

* Each sprint had clear goals, which helped maintain steady progress.
* Adjustments were made successfully when new requirements came up, such as adding the chat and whiteboard features.
* According to table 35, Detailed documentation was written throughout the project to support future updates and maintenance.

**Challenges:**

* Some features like whiteboard and chat faced delays due to technical integration issues.
* Overlapping tasks sometimes caused a slowdown in completing certain parts.
* Bugs in real-time features were only noticed in later testing phases, leading to additional debugging time.

**What was learned:**

* Testing complex features like chat and whiteboard earlier can help prevent last-minute issues.
* Planning extra time for integrating third-party tools is important to reduce development delays.

## 10.2 Technologies Evaluation

**Overview:**

* Django and Django REST Framework were used for backend development.
* ReactJS was used for building the frontend.
* These technologies helped create a responsive, scalable, and well-structured learning management system.

**What went well:**

* Django’s built-in tools made backend development faster and more efficient.
* The REST API allowed for smooth connection between frontend and backend.
* ReactJS helped build a clean and dynamic user interface.
* The modular structure made it easier to maintain and expand the system.

**Challenges:**

* Integrating Django and React required time to understand and set up properly.
* Adding real-time features like chat and whiteboard brought extra complexity.
* The system needed additional optimization to handle more users during testing.

**What was learned:**

* Starting integration and testing of real-time features early saves time later.
* Reviewing and testing third-party tools early in the project helps avoid future problems.

## 10.3 Developer Performance Evaluation

**Overview:**

* Development was done using an agile approach with tasks organized into sprints.
* Each task was tracked using Trello, and progress was reviewed regularly.
* Code quality, documentation, and meeting deadlines were given high priority throughout the project.

**What went well:**

* Sprint goals were mostly completed on time.
* Code was written to be clean, understandable, and easy to maintain.
* Problems were resolved quickly when they appeared, and testing was done regularly.

**Challenges:**

* Managing multiple tasks and responsibilities sometimes caused delays.
* Some bugs, especially in real-time modules, appeared late and required extra effort to fix.

**What was learned:**

* Prioritizing early testing can help catch bugs sooner.
* Focusing on high-risk modules early in the process leads to smoother development later on.

# 11. Conclusion

In summary, the LMS project is a well-designed online learning platform that balances flexibility, ease of maintenance, and user-friendliness. The backend, built with Django and Django REST Framework, handles data and user interactions smoothly through a reliable API. The Reactjs frontend offers a responsive, engaging experience that makes it easy for users to interact with courses and other features.

The project’s organized structure and careful planning allow for easy updates and new features. With a strong foundation and thoughtful design, this LMS is set up to provide an effective and enjoyable learning experience for users while being straightforward to maintain and scale over time.

# 12.References

1. Wang, Y. S., Wang, H. Y., & Shee, D. Y. (2007). Measuring e-learning systems success in an organizational context: Scale development and validation. *Computers in Human Behavior, 23*(4), 1792-1808.
2. Wang, W. S., & Wang, C. C. (2009). An empirical study of instructor adoption of web-based learning systems. *Computers & Education, 53*(3), 761-774.
3. Goyal, E., & Purohit, S. (2011). Using Moodle to enhance student satisfaction from ICT. *2011 IEEE International Conference on Technology for Education*, 191-198.
4. Pandey, S. R., & Pandey, S. (2009). Developing a more effective and flexible learning management system (LMS) for the academic institutions using Moodle. In *ICAL 2009 Technology, Policy and Innovation*.
5. Naveh, G., Tubin, D., & Pliskin, N. (2010). Student LMS use and satisfaction in academic institutions: The organizational perspective. *The Internet and Higher Education, 13*(3), 127-133.
6. Al-Busaidi, K. A. (2012). Learners’ perspective on critical factors to LMS success in blended learning: An empirical investigation. *Communications of the Association for Information Systems, 30*(2), 11-34.
7. Sharifov, M., & Mustafa, A. S. (2020). Review of prevailing trends, barriers, and future perspectives of learning management systems (LMSs) in higher institutions. Online Journal of New Horizons in Education, 10(3), 166.
8. Kraleva, R., Sabani, M., & Kralev, V. (2019). An analysis of some learning management systems. International Journal of Advanced Science, Engineering and Information Technology, 9(4), 1190–1198.
9. Forum, W. E. (2022). These 3 charts show the global growth in online learning. Retrieved April 6, 2024, from https://www.weforum.org/agenda/2022/01/online-learning-courses-reskill-skills-gap
10. Linder, K. E., & Hayes, C. M. (2023). High-impact practices in online education: Research and best practices. Taylor & Francis.
11. Salama, R., & Hinton, T. (2023). Online higher education: Current landscape and future trends. Journal of Further and Higher Education, 47(7), 913–924. https://doi.org/10.1080/0309877X.2023.2200136
12. Martin, F., & Bolliger, D. U. (2022). Designing online learning in higher education. Singapore: Springer. https://doi.org/10.1007/978-981-19-0351-9\_72-1
13. Alomari, M. M., El-Kanj, H., Alshdaifat, N. I., & Topal, A. (2020). A framework for the impact of human factors on the effectiveness of learning management systems. IEEE Access, 8, 23542–23558.
14. Fearnley, M. R., & Amora, J. T. (2020). Learning management system adoption in higher education using the extended technology acceptance model. IAFOR Journal of Education, 8(2), 89–106.
15. Sánchez Padilla, L. V., & Peñarreta Guevara, J. D. (2023). Estudio comparativo de los learning management systems (LMS) (B.S. thesis). Riobamba.
16. Cavus, N. (2010). The evaluation of learning management systems using an artificial intelligence fuzzy logic algorithm. Advances in Engineering Software, 41(2), 248–254
17. Momani, A. M. (2010). Comparison between two learning management systems: Moodle and Blackboard. Behavioural and Social Methods Journal, 2(54).
18. Cavus, N., & Zabadi, T. (2014). A comparison of open source learning management systems. Procedia - Social and Behavioral Sciences, 143, 521–526.
19. Al-Ajlan, A. S. (2012). A comparative study between e-learning features. In Methodologies, Tools and New Developments for E-Learning (pp. 191–214).
20. Kasim, N. N. M., & Khalid, F. (2016). Choosing the right learning management system (LMS) for the higher education institution context: A systematic review. International Journal of Emerging Technologies in Learning, 11(6).

# Appendix

## 13.1 Requirements for system:

asgiref==3.7.2

async-timeout==4.0.3

attrs==23.2.0

autobahn==23.6.2

Automat==22.10.0

cffi==1.16.0

channels==4.0.0

channels-redis==4.2.0

constantly==23.10.4

cryptography==42.0.5

daphne==4.1.0

Django==5.0.2

django-cors-headers==4.3.1

djangorestframework==3.14.0

hyperlink==21.0.0

idna==3.6

incremental==22.10.0

msgpack==1.0.8

mysqlclient==2.2.4

pillow==10.2.0

pyasn1==0.5.1

pyasn1-modules==0.3.0

pycparser==2.21

pyOpenSSL==24.0.0

pytz==2024.1

redis==5.0.2

service-identity==24.1.0

setuptools==69.1.1

six==1.16.0

sqlparse==0.4.4

Twisted==24.3.0

txaio==23.1.1

typing\_extensions==4.10.0

websockets==12.0

zope.interface==6.2

# 13.2 Getting started with EDUNEST:

* 1. **System Overview**

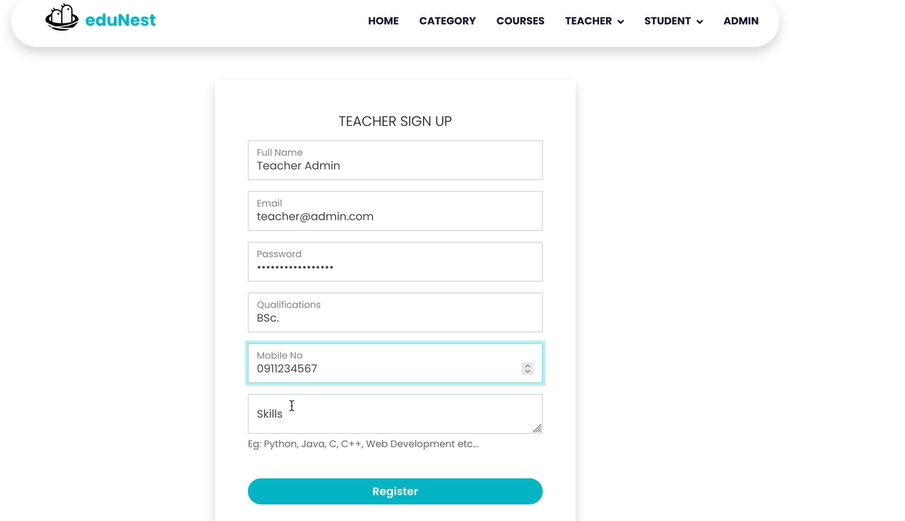
Edunest is built with two main types of users:

* **Admin** – Manages the platform, users, courses, and content.
* **Instructor** – Creates and manages course material, interacts with students.
* **Student** – Enrolls in courses, participates in discussions, views lectures, and submits assignments.
  1. **Getting Started**

**Logging In**

1. Go to the login page.
2. Enter your registered email and password.
3. Click **Login** to access the dashboard.

**Note:** New users must be registered by the admin.





* 1. **Main Features**

**A screenshot of a computer

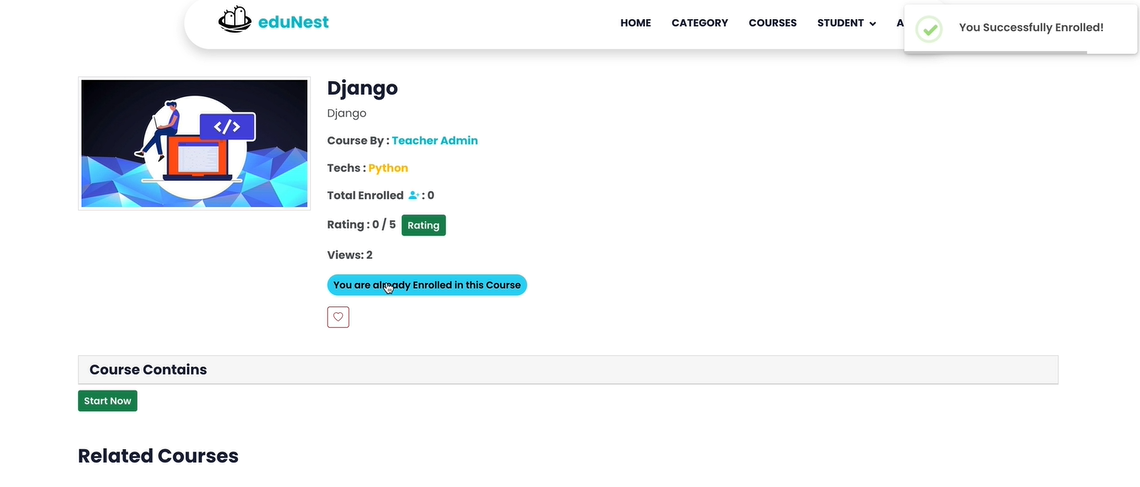
AI-generated content may be incorrect.**

**3.1 Course Enrollment (Students)**

* Navigate to the list of available courses.
* Click on the course you want to enroll in and press **Enroll**.
* A notification will be sent to the respective teacher.

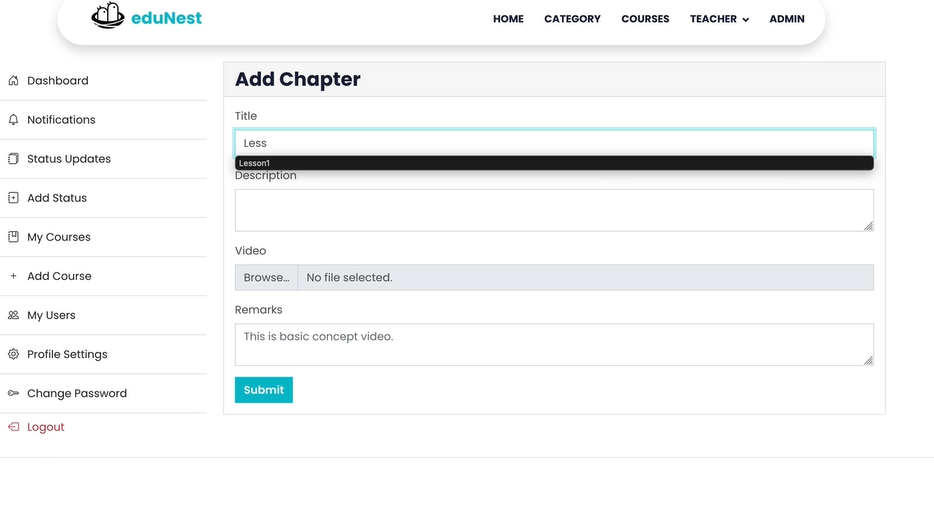
A screenshot of a computer

AI-generated content may be incorrect.



**3.2 Adding Course Material (Teachers)**

* After logging in, navigate to “My Courses.”
* Select a course and click **Add Material**.
* Upload files (PDF, DOC, PPT, etc.) and click **Submit**.

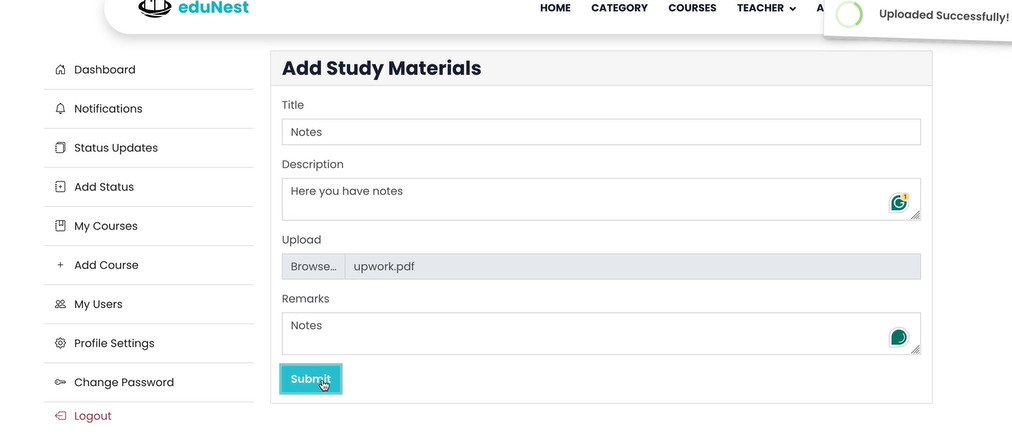


A screenshot of a computer

AI-generated content may be incorrect.

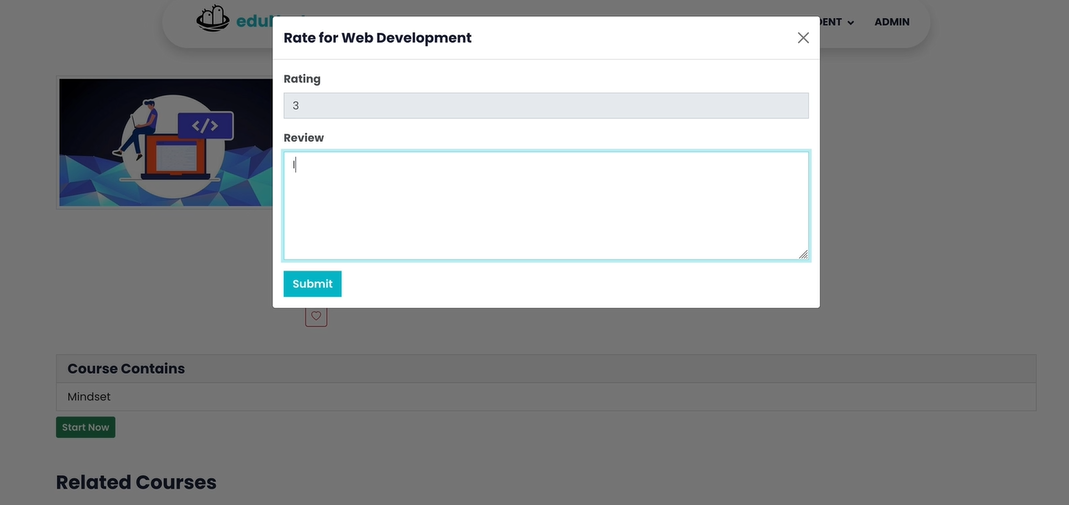
A screenshot of a computer

AI-generated content may be incorrect.



**3.3 Rating (Students)**

* Open a course from your dashboard.
* Scroll to the **Feedback** section and submit your comments or suggestions.

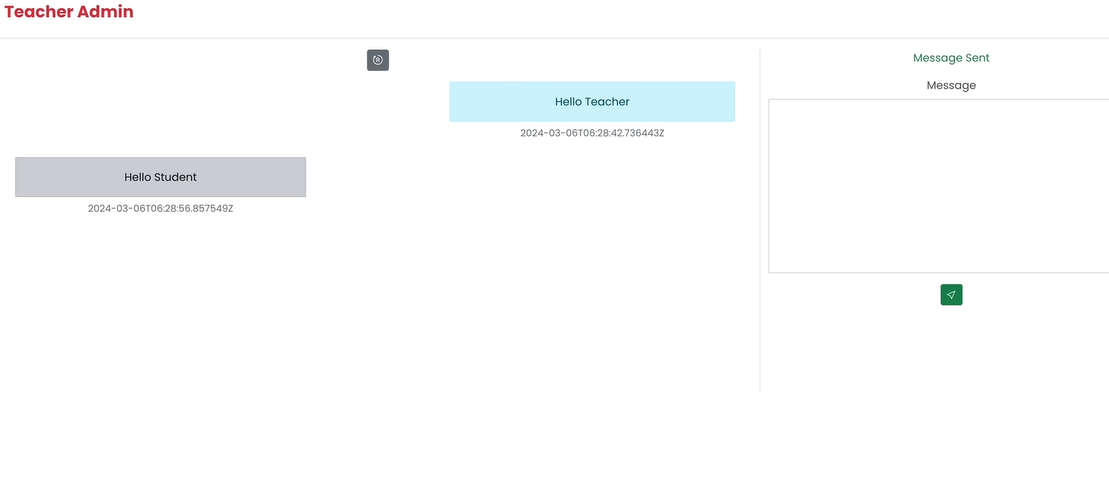


**3.4 Chat System**

* Click the **Chat** button from the navigation bar.
* Select a user (teacher/student) to start a conversation.
* Use the chat window to exchange real-time messages.

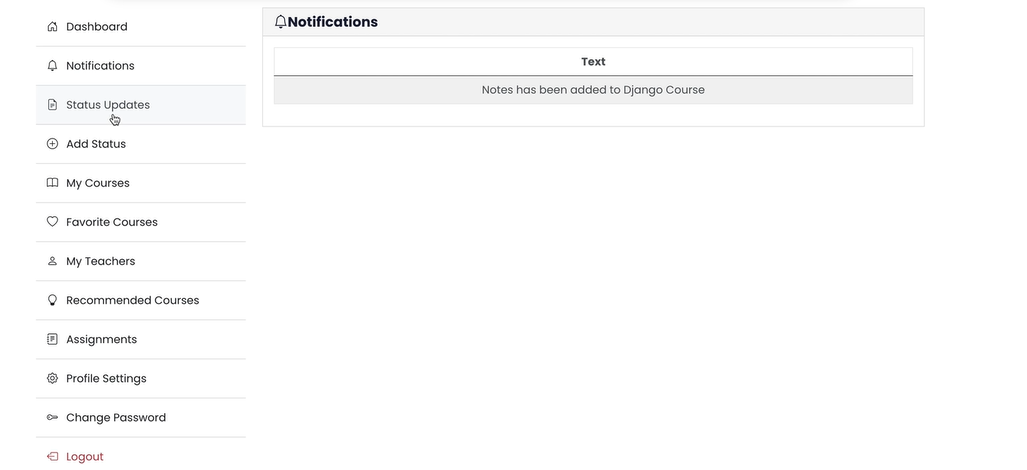
A screenshot of a computer

AI-generated content may be incorrect.



**3.5 Notifications**

* Notifications appear automatically in your dashboard for events such as course updates, new materials, and feedback.



**3.6 Search Functionality**

* Use the search bar on the top navigation to search for teachers, students, or courses.
* You can filter results by name, subject, or role.

**3.7 Status Updates (Teachers and Students)**

* Update your online/offline or active/inactive status from your profile page.
* Your status is visible to other users for real-time interaction tracking.

A screenshot of a computer

AI-generated content may be incorrect.

**3.8 Managing Users (Teachers)**

* Teachers can view a list of enrolled students.
* From their dashboard, they can choose to **Remove** or **Block** any student from a course if needed.

## Troubleshooting

|  |  |
| --- | --- |
| Issue | Solution |
| Cannot log in | Check email/password, or contact admin |
| Upload error | Ensure file is within size limit and format |
| Whiteboard not loading | Refresh page or check internet connection |
| Chat messages not sending | Try logging out and logging in again |

# 13.3 Application Deployment

## Unzipping and installing requirements:

To set up the Edunest project, follow these steps:

1. Unzip the project archive.
2. Navigate to the backend and frontend directories respectively.
3. Run the following commands to install dependencies:

### For Frontend:

npm install

### For Backend:

pip install -r requirements.txt

## List of Packages and Versions

### Frontend Packages (React)

Table Packages list used in frontend development

|  |  |  |
| --- | --- | --- |
| No. | Package Name | Description |
| 1 | @fortawesome/fontawesome-svg-core | Foundation library for Font Awesome SVG icon support. |
| 2 | @fortawesome/free-solid-svg-icons | Includes over 1,500 solid style vector icons. |
| 3 | @fortawesome/react-fontawesome | Integrates Font Awesome with React components. |
| 4 | @testing-library/jest-dom | Adds custom matchers for testing DOM nodes using Jest. |
| 5 | @testing-library/react | Utility for testing React components via the DOM. |
| 6 | @testing-library/user-event | Simulates user interactions for testing. |
| 7 | axios | Sends HTTP requests to servers and APIs. |
| 8 | firebase | SDK for building web and mobile apps with hosting, database, and auth. |
| 9 | react | Core library for building UIs using JavaScript and JSX. |
| 10 | react-dom | Renders React components to the DOM. |
| 11 | react-input-emoji | Allows users to insert emojis into input fields. |
| 12 | react-responsive-carousel | Responsive carousel component for React. |
| 13 | react-router-dom | Routing library for Single Page Applications in React. |
| 14 | react-scripts | Bundled tools for development and testing in React projects. |
| 15 | react-websocket | Connects WebSocket functionality with React components. |
| 16 | sweetalert2 | Customizable, stylish alert popups with animations. |
| 17 | web-vitals | Captures performance metrics like LCP, CLS, and TBT. |
| 18 | websocket | Native browser WebSocket for real-time communication. |

**Additional Resources:**

* **Google Fonts**
* **Font Awesome CDN**
* **Bootstrap Icons**

These are used for enhancing typography and visuals across the UI.

### Backend Packages (Django)

Table Backend packages used

|  |  |  |
| --- | --- | --- |
| No. | Package Name | Description |
| 1 | Django | High-level Python framework with built-in ORM, templating, and authentication. |
| 2 | Django REST Framework | Toolkit for building REST APIs with serializers, viewsets, and more. |
| 3 | Channels | Adds support for asynchronous WebSocket communication in Django. |
| 4 | Redis | In-memory store used with Channels for real-time messaging and background task distribution. |
| 5 | django-cors-headers | Middleware to enable CORS requests across domains. |
| 6 | psycopg2 | PostgreSQL adapter for Python/Django database interaction. |
| 7 | Pillow | Image processing library for handling image uploads and transformations. |
| 8 | PyJWT | Library for encoding/decoding JSON Web Tokens for user authentication. |
| 9 | Celery | Distributed task queue for executing background processes (emails, PDF generation). |
| 10 | Whitenoise | Middleware to serve static files in production environments. |

# 13.4 Development Environment Details

To begin the project, I created a **virtual environment** using tools like venv or conda. This approach ensures that dependencies for this project remain isolated, preventing conflicts with packages from other projects.

Once the environment was ready, I installed all the required packages using:

pip install -r requirements.txt

### Operating System Options:

* **Linux (Ubuntu)**: Known for its performance, stability, and suitability for development.
* **macOS**: Offers a smooth user interface and UNIX-based system compatibility.
* **Windows**: Has broad tool support but may require additional configuration for certain developer tools.

Regardless of the OS, I made sure it was up to date before starting development.

### Code Editors I Used:

* **Visual Studio Code** (my preferred choice)
* Other options include Sublime Text, Atom, or PyCharm — all supporting plugins and extensions that enhance coding efficiency.

### Version Control:

I used git for version control to track changes and push code to remote repositories (like GitHub or GitLab).

**Local Web Server & Database:**

For running the project locally:

* I used **Gunicorn** as the application server.
* For serving static files in production, **WhiteNoise** was integrated.
* A **PostgreSQL** database was set up to store and manage application data.

For load balancing and reverse proxy needs, tools like **NGINX** or **HAProxy** can be configured later for scaling the app in production.

### Django Admin Login Instructions

To log in to the Django admin panel, I followed these steps:

1. Opened the terminal and navigated to the Django project root.
2. Started the development server using:

python manage.py runserver

1. Opened a browser and visited:  
   [**http://127.0.0.1:8000/admin**](http://127.0.0.1:8000/admin)
2. Entered the **admin username and password** I set up during the initial configuration.
3. Clicked the **“Log in”** button to access the Django admin dashboard.

### Admin Panel Usage

From here, I can:

* Add/edit/delete **courses**, **teachers**, and **students**.
* View user activities and manage content dynamically.

**Important:** I always log out after completing tasks to maintain security. The **“Log out”** button is located at the top-right of the admin panel.

### Login Credentials for Teacher and Student Accounts

For demonstration and testing purposes, I created the following accounts:

**1. Teacher Account**

* **Email:** teacher@admin.com
* **Password:** teacher@admin.co

**2. Student Account**

* **Email:** student@admin.com
* **Password:** student@admin.com

These credentials are meant **only for testing** and do not correspond to real users. In production, user authentication will be handled securely.

# 13.5 Edu Nest Software Evaluation Questionnaire

**1. General Information**

1. **Which role do you primarily use in Edu Nest?**
   * Teacher
   * Student
   * Admin
   * Other (please specify)
2. **How frequently do you use Edu Nest?**
   * Daily
   * Weekly
   * Monthly
   * Rarely
3. **How would you rate your overall experience with Edu Nest?**
   * Excellent
   * Good
   * Average
   * Poor

**2. User Interface and Experience**

1. **How easy is it to navigate the Edu Nest platform?**
   * Very easy
   * Easy
   * Neutral
   * Difficult
   * Very difficult
2. **How visually appealing do you find the platform?**
   * Very appealing
   * Appealing
   * Neutral
   * Unappealing
   * Very unappealing
3. **How intuitive do you find the overall layout and design of Edu Nest?**
   * Very intuitive
   * Intuitive
   * Neutral
   * Confusing
   * Very confusing
4. **How responsive is Edu Nest on different devices (e.g., mobile, tablet, desktop)?**
   * Very responsive
   * Responsive
   * Neutral
   * Unresponsive
   * Very unresponsive

**3. Functionality**

1. **How satisfied are you with the following features of Edu Nest? (Rate each feature)**  
   a. **Account creation and login**
   * Very satisfied
   * Satisfied
   * Neutral
   * Dissatisfied
   * Very dissatisfied

b. **Search functionality (finding courses, teachers, students)**

* + Very satisfied
  + Satisfied
  + Neutral
  + Dissatisfied
  + Very dissatisfied

c. **Course enrollment and management**

* + Very satisfied
  + Satisfied
  + Neutral
  + Dissatisfied
  + Very dissatisfied

d. **Feedback submission**

* + Very satisfied
  + Satisfied
  + Neutral
  + Dissatisfied
  + Very dissatisfied

e. **Real-time chat functionality**

* + Very satisfied
  + Satisfied
  + Neutral
  + Dissatisfied
  + Very dissatisfied

f. **Notifications (course updates, new materials, etc.)**

* + Very satisfied
  + Satisfied
  + Neutral
  + Dissatisfied
  + Very dissatisfied

1. **Have you experienced any technical issues or bugs while using Edu Nest?**
   * Yes (please specify)
   * No

**4. Performance**

1. **How would you rate the platform's speed and performance?**

* Very fast
* Fast
* Neutral
* Slow
* Very slow

1. **Have you experienced any downtime or server issues while using Edu Nest?**

* Yes (please specify)
* No

**5. Communication and Support**

1. **How would you rate the quality of support provided (if you’ve used it)?**

* Excellent
* Good
* Average
* Poor
* Never used support

1. **How effective are the notifications and updates you receive?**

* Very effective
* Effective
* Neutral
* Ineffective
* Very ineffective

1. **Have you found the help documentation and tutorials helpful?**

* Very helpful
* Helpful
* Neutral
* Unhelpful
* Very unhelpful

**6. Security and Privacy**

1. **How secure do you feel using Edu Nest in terms of data privacy?**

* Very secure
* Secure
* Neutral
* Insecure
* Very insecure

1. **Have you ever experienced issues related to account security (e.g., unauthorized access)?**

* Yes (please specify)
* No

**7. Overall Satisfaction**

1. **Would you recommend Edu Nest to others (e.g., teachers, students, educational institutions)?**

* Definitely yes
* Yes
* Maybe
* No
* Definitely no

1. **What improvements or features would you like to see in future updates of Edu Nest?**  
   *(Open-ended response)*
2. **Is there anything else you would like to share about your experience with Edu Nest?**  
   *(Open-ended response)*
3. **What is your age group?**

* Under 18
* 18-24
* 25-34
* 35-44
* 45-54
* 55+

1. **What is your highest level of education?**

* High School
* Undergraduate
* Graduate
* Doctoral
* Other (please specify)

## Questionnaire Results:

A graph with blue lines

AI-generated content may be incorrect.

# 13.6 Unit testing

## How I Ran Unit Tests in Edu Nest

### Django Backend

To ensure everything works smoothly in the **Edu Nest backend**, I ran unit tests using Django’s built-in testing framework.

* First, I made sure all necessary dependencies were installed by running:

pip install -r requirements.txt

* I added any required test data in a fixtures directory and used:

python manage.py loaddata <fixture\_name>

* After setting up the environment and test data, I executed the following command to run all backend unit tests:

python manage.py test

This tested core features like authentication, API endpoints, and database operations—ensuring everything in the backend functions as expected.

### React Frontend

For the **Edu Nest front end**, I implemented unit tests using **Jest** and **React Testing Library**. These tools help ensure React components behave correctly when interacted with.

* I confirmed all test-related dependencies like Jest and React Testing Library were included in package.json.
* Each React component had corresponding test files following the \*.test.js or \*.spec.js naming convention.
* To run the tests, I used:

npm test

This allowed me to verify frontend behavior such as rendering, user interaction, and logic execution for various components across the platform.

# Future improvements

The following are some future improvements that I have thought of to enhance the overall experience and functionality of **Edu Nest**. These ideas are based on feedback, potential scalability, and trends in modern e-learning systems:

1. **Enhanced Role-Based Permissions**  
   I plan to add more advanced role management features, so each user (admin, teacher, or student) can have customized access to different parts of the system.
2. **Progress Tracking and Analytics**  
   I want to implement dashboards that visually show each student's progress and allow teachers to track class performance more efficiently.
3. **Video Conferencing Integration**  
   I have thought about integrating video conferencing tools like Zoom or Jitsi, so that teachers can conduct live classes directly from the platform.
4. **Mobile App Development**  
   A mobile app version of Edu Nest is something I’m considering to make the platform more accessible on the go, especially for students.
5. **Gamification Features**  
   I believe adding badges, points, and leaderboards will make learning more fun and motivating for students.
6. **Multilingual Support**  
   To make Edu Nest more globally accessible, I’m thinking of adding language selection options for users.
7. **AI-Based Recommendations**  
   I also plan to use AI to suggest personalized learning paths for students based on their activity and performance.