UNIVERSITY OF HERTFORDSHIRE

School of Physics, Engineering and Computer Science

MSc Advanced Computer Science

7COM1039-0109-2022 - Advanced Computer Science Masters Project

Project: AI based E-learning management System using Django

Student ID: 20057870

Student Name: Priyanka Ghadai

Supervised by: Jerry Bent

Declaration

This report is submitted in partial fulfilment of the requirement for the degree of Master of Science in Advanced Computer Science Masters Project at the University of Hertfordshire (UH). It is my own work except where indicated in the report. I did not use human participants in my MSc Project.

SIGN: -Priyanka Ghadai Date:- 28/08/2023

Acknowledgement

My guide, supporter and supervisor Jerry Bent. I want to say thank you to her for huge support and contribution to my project. I want to pay gratitude to the whole staff of the MSc Computer Science course at University, CAMPUS, for providing huge support and great advice, motivation to cheer me up when I felt de-motivated. The experience they shared and the knowledge you gave were perfect and were so helpful throughout my whole course.

Further, I would like to express my gratitude to my parents for their support and timely encouragement throughout the entire course.

Abstract

The Learning Management System (LMS) presented in this thesis is painstakingly built and delicately combines Python, Django, Machine Learning (ML), HTML, CSS, JavaScript, SQLite, and Jinja2. The LMS design is built on the robust Model-View-Controller (MVC) architecture of Django and the adaptability of Python, allowing user administration, course management, and content delivery. JavaScript, CSS, and HTML work together to produce a complete user experience that increases participation and user involvement. Dynamic learning paths and tailored content suggestions are made possible by the introduction of machine learning, which is powered by ML algorithms that analyse user behaviour to provide adaptive learning experiences. The database management system's interface with SQLite enables effective data storage and retrieval, while Jinja2's support for dynamic content rendering improves the system's responsiveness. The seamless integration of Python, Django, Machine Learning, and web technologies is fundamental to the project's technological relevance since it gives instructors the capacity to offer customised information and lets students participate in a unique, interactive learning environment. The LMS's modular structure allows for scalability and paves the way for further integrations and improvements, highlighting its capacity for expansion and adaptation. The proposed LMS is an example of a technological feat accomplished by the seamless fusion of Python, Django, Machine Learning, and web technologies. By enhancing online learning dynamics and encouraging adaptable, effective learning experiences, it is positioned as a transformational tool by its complex design and cutting-edge features.

Table of Contents

Declaration 1

Acknowledgement 2

Abstract 3

Chapter 1: Introduction 7

Background Information 7

Problem Statement 7

Statement of Purpose 8

Research Question 8

Overview of Methodology 8

Rationale and Significance 9

Role of the Researcher 10

Researcher Assumption 11

Definition of key terminology: 11

Chapter 2: Literature Review 13

Django and its practices in e-learning website 13

Challenges and Solutions for E-learning Management System 15

Artificial Intelligence for Education 17

Use of Chatbot in Learning Management System 19

Comparison Table 20

Chapter 3: Research Methodology 22

Introduction 22

Rationale for Research Purpose 22

State of the Art 24

Research Setting 25

Project Workflow 27

Ethical Considerations: 28

Issues of Trustworthiness: 28

Data Security and Client Assurance: 28

System Reliability: 28
Accuracy and Suitability Endorsement: 29
Limitations and Delimitations 29
Summary 31
Chapter 4: Implementation 32
Introduction 32
Database schema 32
Site-Map 33
Admin Management 34
Teacher Management 35
Student Management 37
Overall System Functionality 39
Analysis of Implemented Features 40
Summary 49
Chapter 5: Discussion and Results 51
Introduction 51
Analysis of Implemented Features 51
Limitations and Challenges 52
Future Enhancements and Directions 54
Summary 54
Chapter 6: Conclusion and Recommendation 56
Conclusion 56
Recommendation 57
References: 58
APPENDIX: 61
Admin Management: 61
validate the user: 61
Manage Teachers by Admin: 67
Manage student by Admin 69

Manage All Course Work 73
Chat Bot 84
Manage Results of the coursework 86

Table of Figure

Figure 1 State-of-the-art figure for the AI-LMS 24
Figure 2 Project workflow 27
Figure 3 Entity Relationship diagram 33
Figure 4 Sitemap of the entire website 33
Figure 5 Add Teacher by Admin 34
Figure 6 Add Student by Admin 35
Figure 7 Add assignment. 36
Figure 8 Quiz 36
Figure 9 Student Attendance 37
Figure 10 Student results 37
Figure 11 Quiz questions 38
Figure 12 Discussion 39
Figure 13 Admin management 41
Figure 14 Home page of Admin management 42
Figure 15 Admin manages Teachers. 42
Figure 16 Admin manages Students 43
Figure 17 Teacher Dashboard 43
Figure 18 Courses name of the Teacher dashboard 44
Figure 19 Teacher discussion 44
Figure 20 Faculty model 45
Figure 21 Student Dashboard 46
Figure 22 Subject details on the Student dashboard 46
Figure 23 Student Quiz 47
Figure 24 Student model 48
Figure 25 Chatbot application: Teachers Name displaying 48
Figure 26 Chatbot application: Courses displaying 49

Chapter 1: Introduction

Background Information

AI plays an important role in Learning Management Systems. To enhance the delivery and management of education through online services, learning platforms provide educators, students, and other stakeholders with tools, information, and resources. It is a specific management program that is focused on staff training. It is the solution for improved administration process, management of documents, and result tracking, as same as advance reporting. It combines three words: Learning, Management, and System, also known as LMS. LMS is being broadly used in colleges, schools, universities, and organisations. With a learning management system, training and e-learning are managed by software allowing users and administrators to access courses and activity reports easily. This includes web-based online assessment, computer-based collaborative learning and sharing applications, online discussions with each other, and web seminars are also included in e-learning platforms. Using an e-learning platform has many benefits, such as being easy to use, very effective, cost-effective, lightweight, compatible, and low-cost training interface, and allowing the organisation to deliver the quality content of eLearning to the entire team without any hurdles. With the help of LMS methods, students can access LMS everywhere and learn easily and quickly (Wang et al., 2021).

The system can provide various learning materials, such as videos, simulations, and interactive quizzes, to cater to different learning styles. The system can also track student progress and provide certificates upon completion, ensuring that students have a record of their achievements.

Problem Statement

In the eLearning platform, many types of problems can occur with students, such as infrastructure problems, technical issues, digital literacy, interaction with lack of person, abundant distraction, and lack of discipline, and also some students are confused about course quality and structure, and many types of challenges in this platform (Wang and Wang, 2022).

eLearning is a new generation innovation that helps us teach students using electronic gadgets because, in this platform, all facilities are available for students' learning objectives. Still, many students are not satisfied with this learning platform, and many students require physical interaction between them because there is a lot of communication and collaboration between the teachers and the students in online classes. Some infrastructural problems are

also there, like not requiring large buildings, chairs, tables, big classrooms, chalk or marker, and board is not there in online classrooms. Online classes require high-speed internet, but some students don't have it because network issues are not equal everywhere. In online courses, computers or laptops, otherwise any computer devices, require appropriate software and stable electricity is quite a big demand.

Some students have been completely ignoring online learning methods in the evaluation of innovations who students with special needs. Many students feel bored in online classes and not interested in this learning type and complain of a lack of motivation to make it through (Malik et al., 2020).

Statement of Purpose

The e-learning platform creates a successful and robust learning experience that feels like a classroom experience that offers staff-student interaction, discussion, quizzes, collaborative projects, etc. It is processed with a collection of face-to-face and digital content that can be provided. With the help of e-learning, one can evaluate the quality of learning and teaching, improve efficiency and effectiveness, and improve the flexibility and accessibility of users to engage learners in the learning process. It has a huge perspective in higher education with an expanding and vast platform and many challenges in making learning effective. The main goal of this project is to enhance learning and teaching for educators and students (Sarker et al., 2019).

Research Question

"What are the key benefits, challenges, and potential impacts of integrating AI-driven features into eLearning platforms within Learning Management Systems for educational institutions?"

Overview of Methodology

The methodology is a main part of any project; what method will be used in this project, and how can it be proven better these approaches from others? The eLearning platform typically involves a combination of principles, design strategies, and technical tools to facilitate effective learning experiences for users. Let's talk about some common methods used in e-learning platforms (Shaiba, John, and Meshoul, 2023):

First, we need to analyse to identify the target audience, their learning goals, and any specific requirements in the e-learning platform. The platform adopts an academic approach, which could be based on theories like behaviours, cognitive, constructive, or connective. Then, the

chosen path, how content is presented, designs and activities, and assessments are conducted. Structural designers organise the content into modules, units, or lessons and determine the progression and sequence of learning activities by creating a structured plan for e-learning content. The content development of the e-learning platform involves creating or curating instructional material such as text, images, videos, quizzes, assessments, and interactive simulations, and it is designed to be engaging and suitable for online delivery. The eLearning platform aims to promote active learning by incorporating interactive elements. Some e-learning platforms utilise adaptive learning algorithms for personalising learning experiences. These algorithms analyse learner data, such as performance, preferences, and progress, to provide tailored content, feedback, and recommendations. An LMS is used as the underlying technology infrastructure for eLearning platforms, and it manages the delivery of content, tracks learner progress, records assessment results, and provides administrative features such as reporting, enrolment, and analytics. eLearning platforms gather feedback from learners and instructors to identify areas for improvement. The e-learning platforms may have differences according to their methodology (Luque and Francisco José García-Peñalvo, 2020).

Rationale and Significance

The study presented a logical argument for the role of the rationale; significance predicts the benefits of using e-learning platforms during the survey. There are several factors of rationale significance as given below:

- ✓ LMS provides access to education and training material anytime, anywhere, as long as learners have an internet connection. This accessibility breaks down barriers to education, enabling individuals to learn according to their schedule. LMS also offers flexibility in course selection, learning pace, and study location, and learners can choose a wide range of courses and programs that suit their interests and goals. These platforms can be used to analyse learner data, provide personalised recommendations assessment, and ensure about content that learners receive target instruction and support.
- ✓ eLearning platforms can reduce the cost of education compared to traditional classroom-based learning. It also offers affordable or even free courses, making education more accessible to a broader range of learners. It connects learners from different geographic locations, fostering cross-cultural interaction and collaboration. Learners can also participate in online group discussions, forums, and virtual group

projects to enhance their communication and teamwork skills while benefitting from diverse perspectives.

- ✓ It offers many opportunities for continued learning and professional development. Learners can access various courses, certifications, and upskilling programs to expand their knowledge and acquire new skills using LMS.
- ✓ It offers flexibility, accessibility, continuous learning experience, cost-effectiveness, global collaboration, and data-driven insights, making it a significant and valuable tool in the field of education (Zuhairi, Karthikeyan and Priyadarshana, 2019).

Role of the Researcher

The researcher's role is to gather, identify, analyse, and synthesise relevant information and data to enhance the platform's effectiveness. Researchers investigated current educational trends, instructional design models, and emerging ones to inform the development and improvement of learning platforms. They will study maximum theories, collect user feedback, and evaluate the platform's impact on learning outcomes. The researcher thoroughly reviews current literature, reads research papers, and uses existing methods related to the eLearning platforms. Using technological advancement and educational strategies, researchers contribute to the quality of content, user experiences, and effectiveness updated to the media. It helps to identify approaches, techniques, and gaps in knowledge, which can guide the development of an effective learning methodology. Researchers can identify the methods for the eLearning platform based on the problem outcoming and literature review. This includes designing the framework, defining learning collection and processing techniques, and selecting appropriate algorithms in the development process and evaluation procedure. In the analysing and collecting data process researcher is taking all the responsibilities related to gathering compulsory data related to e-learning platforms from relevant sources. The researcher also applies analytical techniques for analysing the data and extracting meaningful insights into data that can be contributed to the learning platforms. They carefully examine the answers and draw conclusions based on their output during the experiment. They can also check the limitations and improvement areas and suggest future directions for the research. After completing all the processes, they can share and communicate with other experts to improve it, such as exchanging ideas with security professionals to gather insight and validate the research findings. Researchers find their recommendations help shape the platform's instructional materials, assessment methods, and user support system. Ultimately, they optimise learners' learning experiences (Alenezi, 2020).

Researcher Assumption

In the e-learning platform, researchers may make certain assumptions based on their investigation and findings to guide their work. These assumptions can provide a foundation for the research and help define the study's scope and directions. Here will discuss some common beliefs given while researching and finding it. Analysis may assume learners have access to the necessary technological infrastructure, such as computers, internet connectivity, and compatible devices, to engage with e-learning platforms effectively. They also assume learners are motivated and capable of self-regulated learning. They have the discipline and intrinsic motivation to engage with the platform, complete assignments, and actively participate in education. They also assume that learners process basic digital literacy skills required to navigate and interact with the eLearning platform effectively. These include skills like web browser, accessing and submitting online assignments, and participating in discussions.

In the e-learning platform, user engagement is under the research assumption, and it will actively engage with the eLearning platform and utilise its features as intended. This includes assuming learners will participate in the discussion, complete assessments, and seek clarification when needed. Researchers should always validate and update their assumptions through continuous data collection and analysis to ensure the platform's effectiveness.

Definition of key terminology:

It is a webspace or portal for educational resources and content that offers students everything they need in one place: lectures, notes, chats, an opportunity to meet with other students, and more. With the help of the learning platform, teachers and students progress easily by taking and giving lectures, providing notes easily for every student, and more. There are many different types of keys of e-learning platforms, which should provide a good understanding of the key terminology of e-learning as shown below (Karkar, Fatlawi and Al-Jobouri, 2020): -

- ✓ It is also known as electronic learning, which refers to delivering educational content and training through digital technology over the Internet. It provides features like course creation, enrolment, assessment, and reporting with the help of software applications that allow the management, delivery, and tracking of eLearning courses and resources. The digital content compromises an eLearning system, including modules, lessons, videos, and other academic materials.

- ✓ A set of technical standards governs the packaging and delivery of eLearning content that ensures interoperability and reusability of content across different sharable content object reference model-compliant learning systems. The application of game elements and mechanics, such as badges, points, leaderboards, and challenges, to evaluate learner engagement and motivation in learning.
- ✓ In the e-learning platforms, there are two types of learning available one is synchronous and the other is asynchronous. Synchronous is a real-time learning approach where learners and instructors participate in online activities that involve live webinars, video conferencing, or chat-based discussion. At the same time, asynchronous is a self-paced learning approach where learners access course materials and complete all activities at their convenience. It allows learners to access content anytime and anywhere without needing real-time interaction.
- ✓ Collaborative learning and knowledge sharing is an online platform or features within an e-learning platform where learners can interact, ask questions, and discuss with instructors and fellow learners. Also, learning platforms can monitor and record learners' progress throughout a course. It gives instructors and learners insights into completed activities, performance, and scores.
- ✓ The delivery of eLearning content and activities through mobile devices like smartphones and tablets. It allows learners to access and engage with course materials on the go. It also helps to serve badges and certificates as recognition of accomplishment and can be shared to showcase skills and achievements. Analytics helps to identify trends, assess effectiveness, and make data-driven decisions for course improvement (Haghshenas, 2019).

Chapter 2: Literature Review

Django and its practices in e-learning website

To have an easy and convenient life, different digital devices came into use. The college or university does not provide proper notes to students, and they cannot communicate with classmates or seniors because of any reason that can affect their education. So, to make their academic skills better, a system is developed on the Django framework, with HTML, CSS, and JavaScript used in the front end and Python, SQLite, and Jinja2 used as the back end. For having such a working system, the development process followed an iterative model of software development. It takes less time to develop, while it takes more use of resources to find errors easily. The Django Python Web framework has fastened development and realistic design. Python is secure compared to other programming languages, which prevents any unwanted attack on the database. It follows the process in the PCL when functionality is added, then removes it from the PCL. The website consists of web pages such as user registration, login, education categories, notes & and question papers, and many more. A chatbot helps users to communicate with each other. The proposed efficiency is accuracy, planned approach, reliability, no redundancy, and ease of operation (Shyam and Mukesh, 2020).

To have growth in the field of the education system, it is important to adopt new technologies. This helps students to learn in a smart way, which is an easy and effective way of learning. With these technologies, students gain good communication with their teacher; they can ask to doubt any time, and teachers can provide them with notes anytime. Massive Open Online Course (MOOC) has become a part of the education system because they can be accessed from anywhere (laptop or mobile), and they can access different fields that enhance their knowledge. The system is based on HTLM. CSS AND JavaScript for the front-end and Django for the Back-end part. For the system login, both students and faculties must have the logins or register in it. There are different pages in the system, such as Home, About, Services, Contact, and many more. The learner dashboard consists of tutorials, notes, courses, quizzes, and announcements. The live classroom provides classes to students in which the teachers remain in one place and educate many students. The system gives a chance to educators to interact with students, peers, and parents. Video conferencing is also an option for students to communicate with other students (Sameena, 2021).

Web development is used in many fields, mainly in promotional fields, and there are many uses for it. For the backend part, Django is mainly used to develop the websites. It is used to build an application that has the CRUD functionality. The application can be updated by the administration panel and created in a virtual environment by using Django APIs with the Django Rest Framework (DRF). SQLite is used as the default database in Django, and it is created by using the keyword class, so no code is available for that. A file is created by using the Python extension. With that, an API is built that needs to have a sign-up to have real-time information. It needs to have many parts to be completed, such as view, models, forms, and templates. Here, Python libraries are also being used for getting the actual data. In the research paper, a weather application is created. Functions are also used in the application in every part created. Developing an application in Django requires implementing a proper process, such as creating the app, adding the template and view, using API in the app, need to display data in the template, and creating a form (Chandiramani, 2021).

Django is the most popular web – framework to be used in Python programming languages. It uses secure methods to transform the data, which is also the reason to use it. It has many instruments that are being used to develop the applications, such as CSRF-tokens, XSS-protection, Django-ORM, and many more. SQL queries are used to generate the databases. Using the Django ORM consists of an object-oriented programming paradigm design. The Django web framework does not work with databases directly, and it uses ORM for all database-related operations. As a web framework, it has some functions that will help in making the site secure. Dev versions and production set is made for good Django developers. The most common attacks are XSS and CSRF, from which Django can provide protection. The Django template does that work. To encrypt, all deals that happen between the user and the server are used by setting SSL/HTTPS on a web server. By using HTTPS, Django allows for security techniques such as SECURE_PROXY_SSL_HEADER, HTTP Strict Transport Security, and more (Duisebekova, Khabirov and Zholzhan, 2021).

In the research paper (Patil et al., 2023), The system is integrated to store the student record management by using Django, in which students also have access to their records and know about the marks they obtain. The relevance of efficient student record administration in educational institutions is covered in the study's opening section. It draws attention to the drawbacks of conventional paper-based systems, including redundant data, laborious procedures, and restricted accessibility. To overcome these difficulties and boost the effectiveness of student record administration, the authors suggest using a web-based system

created with Django. The system includes features such as student registration, attendance monitoring, grade administration, course administration, and report generation.

The researcher highlights the advantages of automation, including how it lowers errors associated with manual data entry, makes it easier to get student data, and produces in-depth reports. The created system is integrated into the infrastructure of an educational institution during the implementation phase. The writers go over the technical elements of deployment, such as user authentication, database management, and system requirements. They also offer information on user assistance and training to guarantee a seamless change from the prior record management system. The agile methodology emphasises collaboration, flexibility, and rapid iteration. For system development, it uses the Model-View-Template (MVT) pattern, which is used in Django projects.

Challenges and Solutions for E-learning Management System

In India, students face many problems regarding the education system by not getting notes for courses or assignments that are not completed. During the COVID-19 times, students face many challenges as they cannot go to the premises. There was only one choice for students to distance learning, which is easy in the world of the internet. The e-learning platforms that provide consistent education such as Google Meet, GoToWebinar, Zoom, GoToMeeting, Zoho Meeting, Adobe Connect, and more. The platform features are third-party integration, security, and customer support. The influence of the branch and year of study had a crucial understanding of perception in the data. All these platforms must have a better API to work. Google Meet has better API than other platforms, and Windows does not support platforms like Microsoft Teams and Zoom on the phone app. Google Meet and Zoho Meeting are the most used platforms and are suggested by the users. Every platform provides different features and is ranked differently. However, concern is about the security issues that it faces. The platforms must be secured because they consist of most education data, and cloud architecture has resulted in large data breaches (Thakker, Parab and Kaisare, 2020).

The research paper (Murtaza et al., 2022) includes the issues or challenges faced during the implementation of e-LMS and suggests some solutions for it. For evaluating the effectiveness of AI in e-LMS, different AI-based techniques are integrated. The research is based on the factors building this model, the state of the art of adaptive, and the future scope of the model. The approach is founded on the inherent principles of giving material that demonstrates the issues encountered. Other difficulties include evaluating the discovered

preferred mode, changing user choices, and conducting continual assessments and data collecting. AI-based techniques are used for the e-learning system, such as Clustering of Learner's Level, Classification, Knowledge Tracing, Learning Factor Analysis, and Item Response Theory.

The learning theories for better e-LMS are cognitivism, cognitivism, humanism, and behaviourism. Methods like Deep Knowledge Tracing (DKT) use neural network models, and Item Response Theory uses Deep Learning for the system. Some recommendations are hybrid, tag-based, and content-based for a better model. The research paper (Hassen and Aliakbari, 2022) focuses on the development and integration of Artificial Intelligence. E-learning helps the user to learn more and communicate more. In the first part of the study, the author focuses on the merits of e-Learning. They also consider user acceptance and provide insights into the deployment process to ensure a seamless user experience. The digital curriculum and electronic books are created to interact with the students for deeper learning, which provides an experience that is more real-world, interactive, and instructional. It has many motives for obtaining e-LMS, such as raising the return on investment by reducing the cost of education, satisfying the needs of learners by providing unique features, and the necessity of renewal and development in higher education institutions.

In the research paper (Maslov, Nikou, and Hansen, 2021), the learning management system helps many learners, and everyone has different experiences while using it. By taking the interviews with the users, many outcomes have come that can be analysed. Many factors affect e-learning and information technology that facilitate it. The users can measure the issues in the UX of the LMS. University students have identified such issues. The user experience includes the integration of people with the model and how they handle it while using it. LMS has some confusion while working online, insufficiency of social interactions, and is potentially uncomfortable for some users.

The research depends on the understanding of users, how their interaction went, and the user interface, including the evaluation of prototypes and iterative creations. It is important that organisational goals are met and that the product is designed and executed for the tasks. Mean values are used for the descriptive analysis of the user experiences that allowed us to find important features. Some interviewers talked about the features, such as the Chatbot not being available for everyone, and they do not get responses on time. Many platforms use contact information for other platforms, such as WhatsApp, Google Drive, and Email. Many

users use YouTube to learn as compared to other platforms because it gives more information on the topics. Some suggested that communication is easier with teachers and students as being a student is old-school than with Moodle.

Artificial Intelligence for Education

Artificial Intelligence has been a good part of the E-learning management system. Learning new things in the industry of technologies helps to understand the workings of any technologies better. Students can learn more about VR, AR, MR, or XR technologies that enhance their skills. The technologies can store their work, and they use it whenever they need that. It consists of different content, tools, or forms to create learning in the appropriate context. The methods it consists of help to identify patterns and analyse the internal components. The system divided the technology according to the AR, VR, and XR to have a core separation and to train the skills. It saves the learning behavior of all media of learners. Doing practical work with such technology can help learners easily understand the lessons. Both the physical and electrical understanding shown to the learners, they can adjust to the functions that made changes can occur immediately. The E-learning industry is using AI for teachers in e-learning systems. The system has videos, Text, and 3D animation to understand the working of technologies. This can be used many times if students are having doubts. The system stores the points based on skills that focus on the elements of learning (Hirankerd and Kittisunthonphisarn, 2020).

Learning Management System (LMS) helps to provide education in the best way possible. There are many different varieties of Learning Management Systems by use Artificial Intelligence. The system is based on AI, which allows users to view, discuss, control, share, and view online education activities. This system consists of different content for students to learn new every time. The model provides data according to the individual's learning ability. All the advanced assignments can be accessible to the students to prepare for better roles. The learners from the Learning Management System showcase the factors affecting based on the quality of the system, technical, service, and information. The AI approach has sensors to identify the learner's needs to give all factors affecting them. SVM and NN classifiers are used to determine the accuracy. The data can get messy because of the classifier's bad success. MOOCs can show the problems to solve them new theoretical advances in AI can be created. It analyses the data to find how many numbers of query required and the extensive filtering phase that takes place in choosing databases. The system has developed based on experimental studies, developments, and theories for the education sector. The model

captures the gestures of the learners to understand their requirements (Aldahwan and Alsaeed, 2020).

Artificial Intelligence is increasingly used in educational environments and learning processes in schools. The study (Huang, Saleh and Liu, 2021). begins by discussing the importance of AI on campus, as it realises management, teaching, and intelligent learning. It includes image recognition technologies, face recognition technologies, and more for interactive learning. It helps teachers take attendance with the help of face recognition technologies. AI provides virtual simulation laboratories that help students to learn in a great way. The tutoring robots consist of different functions such as robot-directed instructions, robot-managed instructions, and many more. Apps such as BYJU's and ALEKS use features to promote student learning. The paper elaborates on the model's work that includes providing quizzes or even correct assignments and test papers. The research paper concludes by highlighting some issues faced during the implementation of AI in Education and solutions to control it.

In the research paper (Maslov, Nikou, and Hansen, 2021), the learning management system helps many learners, and everyone has different experiences while using it. By taking the interviews with the users, many outcomes have come that can be analysed. Many factors affect e-learning and information technology that facilitate it. The users can measure the issues in the UX of the LMS. University students have identified such issues. The user experience includes the integration of people with the model and how they handle it while using it. LMS has some confusion while working online, insufficiency of social interactions, and is potentially uncomfortable for some users. The research depends on the understanding of users, how their interaction went, and the user interface, including the evaluation of prototypes and iterative creations. It is important that organisational goals are met and that the product is designed and executed for the tasks. Mean values are used for the descriptive analysis of the user experiences that allowed us to find important features. Some interviewers talked about the features, such as the chatbot not being available for everyone and they do not get responses on time. Many platforms use contact information for other platforms, such as WhatsApp, Google Drive, and Email. Many users use YouTube to learn as compared to other platforms because it gives more information on the topics. Some suggested that communication is easier with teachers and students as being a student is old-school than with Moodle.

Use of Chatbot in Learning Management System

Artificial Intelligence is beneficial to be used in LMS for Moodle. The study (Shilowaras and Jusoh, 2022) begins by highlighting the growing importance of AI and chatbot products for learning remotely. The design and development of the chatbot utilising several algorithms, such as machine learning algorithms and natural language processing (NLP), are part of the study approach. Chatbot helps students to get information for the courses they opt for and pertinent educational material to train to ensure accurate responses and fruitful interactions. Developing a system for user needs, system architecture, and chatbot functionalities.

The researchers (Shilowaras and Jusoh, 2022) carried out a pilot study with a group of students using the Moodle LMS to gauge the efficacy of the AI chatbot. The Rational Unified Process (RUP) is used for the methodology, and there are different phases, such as inception, elaboration, construction, and transition. For Moodle architecture, High-Performance Cluster, two server cluster, and single server. The database server limits Moodle's scalability the most. Data on user interactions, user satisfaction, and learning results were gathered. According to the findings, the chatbot dramatically increased student engagement offered prompt assistance, and improved the learning process. The chatbot includes many features, such as greeting the user before the conversation.

For modern information and communication technology applications, chatbot technologies are used. The introduction of the paper (Kovan Mzwri and Márta Turcsányi-Szabó, 2023) discusses how MOOCs are becoming more and more popular to provide online education to a variety of students. However, traditional classroom environments frequently provide more individualised and interactive support than MOOCs do. The MOOC platform uses Web APIs, also known as representational state transfer (RESTful), that are relatively simple, and the APIs work on websites such as Google, Wikipedia, and Facebook. The researchers suggest using a chatbot built within the MOOC platform to overcome this restriction. The system for the chatbot includes GUI for better integration, and it is based on Natural Language Processing (NLP).

The research paper (Mendoza et al., 2022) explores the development and implementation of a chatbot that serves as a multipurpose tool to carry out educational and executive responsibilities and help middle school students and educational staff. It is mainly used in times of COVID-19 when several big enterprises improve their implementation process for all the problems of unforeseen increases in online sales. The paper elaborates on the several

roles of the Chabot such as advising students, making reminders, better communication skills, and information regarding academics. The research methodology has a better user interface for interacting via voice, functional cores are developed with AngularJS. The main goal is to develop tools that help the student learn about different courses and interact with other users. In this system, UX evaluation is important for end-user needs. The results are based on the implementation mechanisms such as hedonic and pragmatic.

Comparison Table

Author	Domain	Tools and Technology	Relevance
Sameena, 2021	A Review of E-learning Websites	Django	Using Django for the implementation of LMS is better; through many e-LMS sites, the educators provide all the study material to them remotely.
Aldahwan and Alsaeed, 2020	AI-based LMS	Artificial Intelligence	It is based on human intelligence concepts, which determine the model can overcome the issues faced.
Duisebekova, Khabirov and Zholzhan, 2021	Web Development	Python (Django)	Mainly, Django is used to develop web applications for managing data transfer and other computational activities. It is becoming more secure after every next version. Django template protects from XSS attacks.
Murtaza et al., 2022	Issues and challenges	Artificial Intelligence ✓ AKT ✓ GIKT ✓ GNN ✓ DKT	Provide solutions to the challenges faced by implementing several modules. For content, a recommendation strategy is built.

Shilowaras and Jusoh, 2022	Implement Chatbot	Dialog Flow baseline Used, RUP approach	Chatbot helps students find information about their courses that they do not ask from educators or search from other sites.

Chapter 3: Research Methodology

Introduction

The AI-LMS (Artificial Intelligence Learning Management System) project plans to create a broad and robust learning management system that uses artificial intelligence to upgrade the learning experience. In the present hi-tech time, the requirement for compelling web-based learning platforms has become progressively significant. The computer-based intelligence LMS project tries to address this need by giving an easy-to-use and skilled system for overseeing courses, working with joint effort, and supporting customised learning.

Rationale for Research Purpose

The decision to make an AI-LMS is driven by a couple of persuading reasons that address the lack of existing learning management systems (LMS) and aim to give a more useful and convincing informative experience. The thinking for encouraging an AI-LMS integration:

Addressing Limitations of Traditional LMS: Traditional LMS platforms often need advanced features and intelligence that can redesign the learning system. By coordinating artificial intelligence, the AI-LMS can overcome these limitations and suggest imaginative plans**.**

Personalised Learning Experience: One of the fundamental motivations for making an AI-LMS is to give a personalised learning experience. Traditional LMS platforms habitually present a one-size-fits-all technique, ignoring individual learning tendencies and necessities. By using AI estimations, the AI-LMS can acclimate to each student's learning style, speed, and data level, giving tailored content and recommendations.

Intelligent Content Recommendations: AI can look at huge proportions of data, including student execution, tendencies, and learning plans, to give assigned content recommendations. By proposing significant resources, invaluable materials, and personalised learning ways, the AI-LMS can work on students' responsibility and data obtainment.

Automating Administrative Tasks: Regular LMS platforms habitually require immense manual effort for administrative tasks like assessing tasks, making reports, and supervising course materials. The AI-LMS can automate tasks by allowing teachers to focus more on teaching, giving ideal analysis, and working with student-teacher collaboration.

Data-Driven Insights: The AI-LMS can utilise data examination and AI estimations to deliver significant insights about student execution, course reasonability, and districts for improvement. This data-driven approach engages instructors and heads to make informed decisions, work on instructive systems, and perceive mediations for combating students.

Enhanced Collaboration and Communication: Collaboration and communication are crucial elements of practical learning. The AI-LMS can merge intelligent chatbots, discussion conversations, and agreeable gadgets to energise reliable associations among students and educators. Continuous info, shared help, and dynamic responsibility can basically additionally foster the learning experience.

Continuous Improvement and Adaptation: An AI-controlled LMS can continuously gain from client participation and change its functionalities as required. By inspecting client direct, analysis, and execution data, the AI-LMS can create and chip away at long term, ensuring that it keeps alert to date with the progressing educational landscape. By addressing the limitations of traditional LMS platforms and using the capacity of AI, the AI-LMS project aims to give a more personalised, intelligent, and attractive learning experience. The benefits consolidate better student results, expanded instructor efficiency, data-driven courses, and enhanced collaboration among understudies. The AI-LMS project attempts to empower enlightening associations and work with the gathering of cutting-edge propels in the field of training.

State of the Art

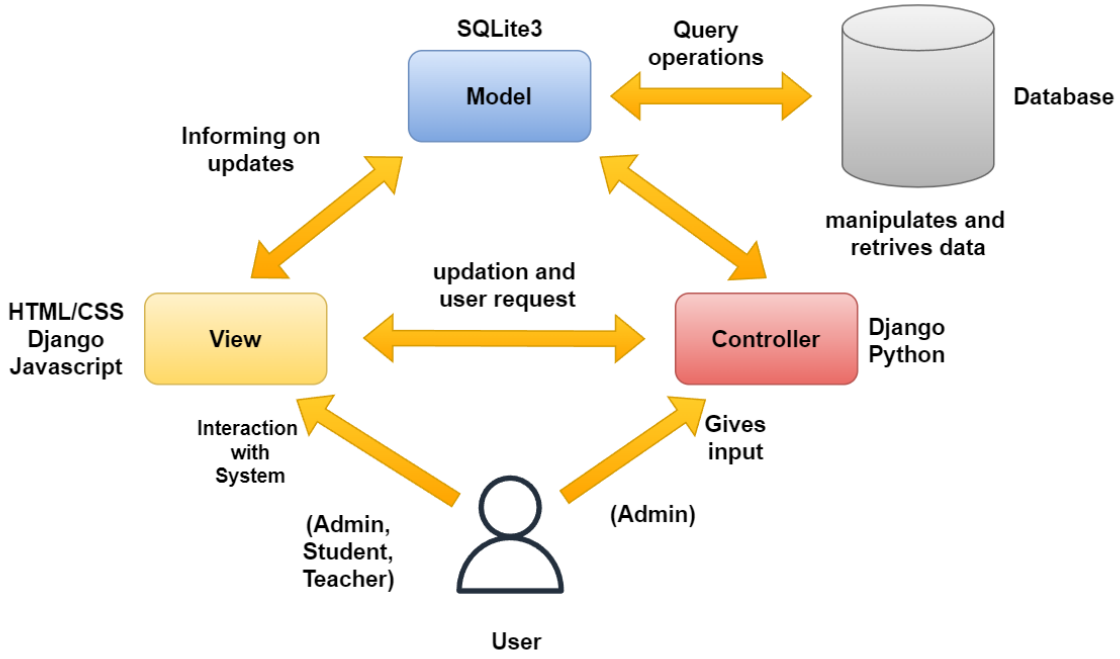


Figure 1 State-of-the-art figure for the AI-LMS

This figure depicts a system that includes the utilisation of SQLite3 as the database, a model-view-controller (MVC) design with Django as the controller and different web innovations like HTML/CSS, JavaScript, and Django for the user interface.

Here is a breakdown of the various parts referenced and their jobs:

SQLite3: It's a lightweight, record-based relational database management system (RDBMS) that is commonly used for small to medium-sized applications. It can control and recovering information productively. Some of SQLite3's benefits include SQLite is free to use, and no licencing is required. SQLite allows you to work on several databases in the same session.

It may be accessed using a broad range of third-party programs. The database's contents may be viewed and updated with simple SQL queries. SQLite has a few restrictions, including the ability to handle only low to medium-traffic HTTP queries and a database storage limit of 2GB.

Model: It is a backend that contains the data logic. The model is responsible for data representation, hence working as an interface for the data in the database. It allows one to interact with the data without getting perturbed by the complexities of the Underlying database. About Django, the model addresses the information construction and business

rationale of my application. It characterises the database pattern and furnishes a connection point to collaborate with the database, including performing CRUD (Create, Read, Update, Delete) activities.

View: The view in Django handles the show rationale and renders the suitable HTML formats to show the information to the user. It collaborates with the model to recover information and passes it to the layout for delivery.

HTML/CSS: HTML is the markup language used to structure the substance of website pages, while CSS is utilised for styling and design. They characterise how the user interface components are shown in the program.

JavaScript: a programming language that empowers dynamic ways of behaving and perception on the client side of a web application. It tends to be utilised to deal with user connections, perform AJAX requests to the server, and control the HTML/CSS.

Controller: In the MVC design, the controller is liable for taking care of user requests, handling input, and planning the connections between the model and the view. For your situation, Django, written in Python, fills in as the controller, taking care of HTTP requests, steering, and dealing with the general application stream.

Admin, Student, Teacher: These are different user jobs in your system. Every job might have various consents and access levels to play out specific activities or view explicit information. Because of these duties, the controller (Django) would handle user validation and approval.

Updation and user requests: Users (Admin, Student, Teacher) can submit requests through the user interface, like refreshing information or making explicit questions. These requests would be taken care of by the proper Django view, which might interface with the model to refresh the database or recover the mentioned data.

By and large, this system follows MVC engineering with SQLite3 as the basic database, Django as the controller, and a mix of HTML/CSS, JavaScript, and Django layouts for the user interface.

Research Setting

Educational Organisations: The AI-LMS can be tailored to different sorts of instructive establishments, including public or private schools, advanced education foundations, professional training centres, or corporate training programs. Every establishment might have

unique demands, educational plan structures, and administrative cycles that should be considered during the plan and execution of the AI-LMS.

Target Clients: The essential objective clients of the AI-LMS are teachers, employees, and understudies. Teachers and employees will use the system to make and oversee courses, transfer learning materials, track understudy progress, and participate in communication with understudies. Understudies will get to the AI-LMS to sign up for courses, access course materials, partake in conversations, submit tasks, and view their grades.

Courses and Subjects: The AI-LMS ought to intend to oblige many courses and subjects. This incorporates scholarly disciplines like math, science, writing, and history, as well as expert courses in fields like business, design, and medication; from there, the sky is the limit. The system ought to be sufficiently adaptable to help various sorts of content, evaluations, and intuitive learning exercises well defined for each branch of knowledge.

Technological Framework: The availability and dependability of the innovative framework inside the instructive setting ought to be considered. This incorporates factors like web network, server limit, similarity with various devices (desktops, PCs, tablets, cell phones), and coordination with existing IT systems or platforms utilised by the instructive establishment.

Regulatory and Protection Considerations: Compliance with applicable data assurance and protection guidelines is pivotal while planning and executing the AI-LMS. Adherence to nearby, territorial, or worldwide guidelines like the General Data Protection Regulation (GDPR) or Family Educational Rights and Privacy Act (FERPA) ought to be guaranteed to safeguard the protection and security of understudy data and individual data.

Scalability and Development: The AI-LMS ought to oblige adaptability and likely future development. As educational establishments extend their contributions, increment the number of courses, or take special care of a developing understudy populace, the AI-LMS ought to have the option to deal with the rising requests without compromising execution or client experience.

Project Workflow

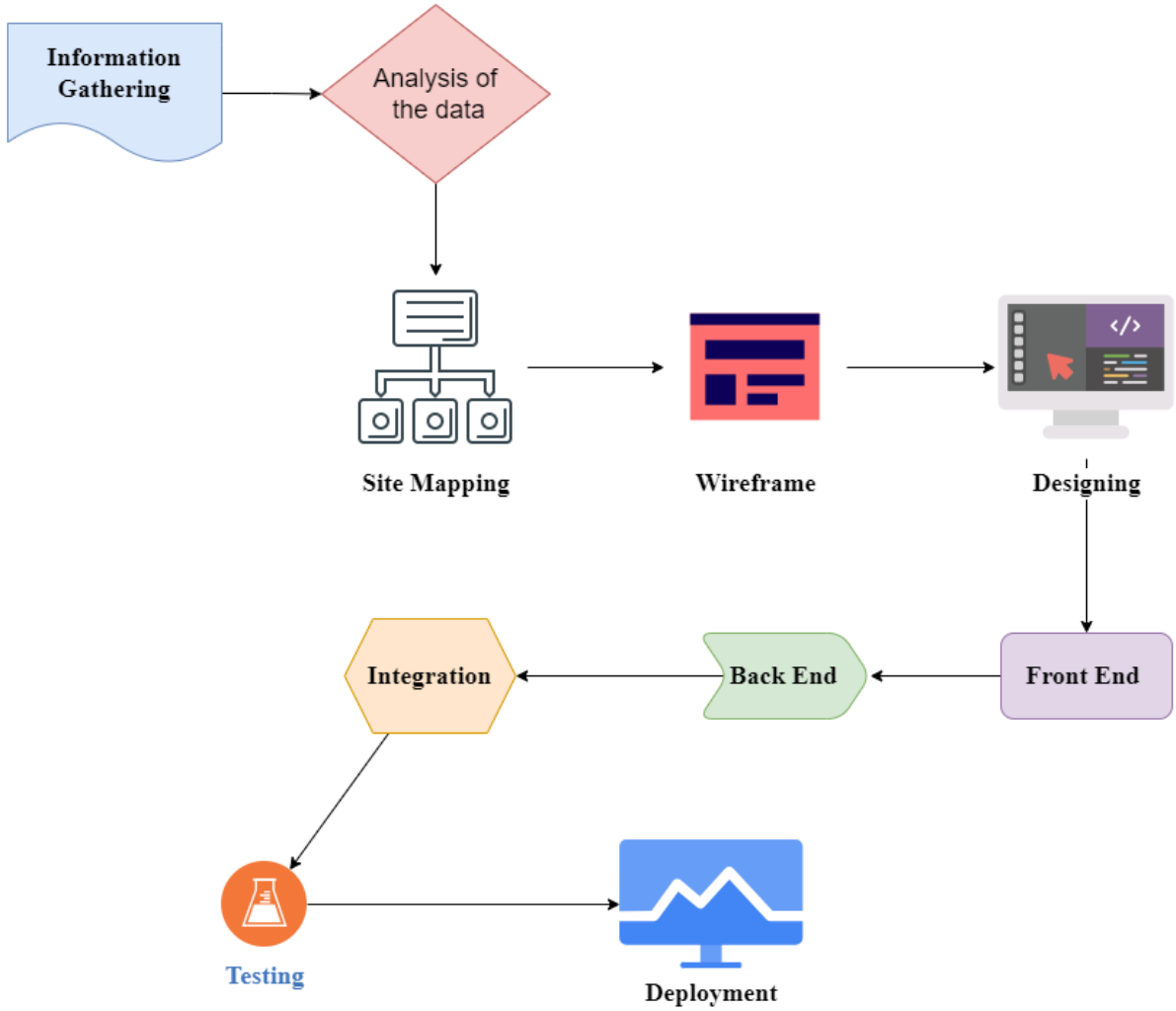


Figure 2 Project workflow

The diagram in the picture above depicts the project workflow for the proposed Django-based LMS system. First, relevant data regarding the needs and features are obtained, and then the collected data is examined. The practical implementation of mapping the site for the Django website follows. The wireframe will be built after mapping the site, and it will be finalised in this step after good design. During the design phase, the wireframes will be connected, and the front-end work will be done properly, while the back-end tasks and operations will also be examined and accomplished. After the front-end and back-end processes and activities are completed, the process of completing effective project integration will begin. The suggested LMS website will next be tested to see if it is responsible and meets the features and processes that have been examined appropriately. Then, following the correct completion of the feature and successful testing, only the project deployment for operations will be completed, and this is the project workflow.

Ethical Considerations:

Ethical considerations are the most important thing that needs to be managed in any project because if there is no maintenance of ethical considerations, the user's trust cannot be gained. So, with that aspect, the purpose of this project of LMS is to maintain and follow the effective legal procedure so this LMS project is fully legal and prepared with considerations of the ethical way. Also, there will be storage of the user data with the best security feature, and there will be no data leakage as the aim is not to make the data public and get leaked. There will be better authentication features to make better effects, better data ownership for the users and maintain good security features. Further ethical considerations followed in this project do not harm, managing confidentiality, fairness, trustworthiness, accessibility and better privacy and security.

Issues of Trustworthiness:

Data Security and Client Assurance:

Encryption: Execute strong encryption to shield data transmission and limit. This integrates secure HTTPS communication and mixed databases.

Access Control: serious areas of strength for the use of control systems to restrict unapproved permission to fragile data. This consolidates job-based admittance control (RBAC) and client affirmation measures.

Client Consent: Obtain explicit client agreement concerning data collection, storage, and use. Convey the insurance technique and obtain assent during the selection collaboration.

Secrecy: Anonymise client data in any spot conceivable to restrict the gamble of data breaks and assurance security affirmation.

Customary Security Surveys: Direct occasional security audits to recognise shortcomings, perform penetration testing, and address any security provisos quickly.

System Reliability:

Powerful Establishment: Construct the AI-LMS on a dependable foundation that can deal with the normal client load and give sufficient overt repetitiveness to limit free time.

Computerised Fortifications: Do motorised support parts to ensure data decency and quick recovery, assuming there ought to emerge an event of system failures or data hardship.

Debacle Recovery Plan: Cultivate an extensive fiasco recuperation and expect to address potential system failures, disastrous occasions, or other unexpected events that could impact system availability.

Customary Maintenance and Updates: Perform ordinary maintenance practices and apply security fixes and updates to direct shortcomings and overhaul system steadfastness.

Accuracy and Suitability Endorsement:

Client Testing: Direct broad client testing to collect information, perceive convenience issues and assess client satisfaction. Consolidate client analysis in iterative progression cycles to chip away at the system's accuracy and sufficiency.

Execution Estimations: Describe key execution estimations, for instance, response time, system availability, and throughput, and continuously screen and redesign them to ensure strong and capable system execution.

Client Analysis and Surveys: Urge clients to give info and partake in examinations to really look at their experience with the AI-LMS. Take apart the contribution to perceive areas for improvement and address any concerns quickly.

Continuous Improvement: Reliably update and redesign the AI-LMS considering client analysis, emerging advancements, and types of progress in the field of learning management systems to ensure its precision, suitability, and importance over an extended time.

Limitations and Delimitations

Technical Limits:

Compatibility: The AI-LMS might confront similarity challenges with certain gadgets, working frameworks, or programs, restricting its openness to clients.

Technical Ability: The task's prosperity relies upon the availability of a gifted, specialised workforce who can create, convey, and maintain the AI-LMS framework.

Integration Difficulties: Incorporating the AI-LMS with existing institutional frameworks, databases, or outsider applications might introduce specialised intricacies or require extra assets.

Resource Limits:

Financial Constraints: The undertaking's degree might be restricted by monetary constraints, influencing the availability of assets for improvement, foundation, and continuous maintenance.

Infrastructure Prerequisites: The AI-LMS might require a hearty mechanical foundation, including servers, stockpiling, and transfer speed, which might present asset challenges for more modest establishments or associations.

Human Assets: The availability of qualified staff and their ability to devote time and work to the task can influence its encouragement and execution.

Time-Related Limits:

Project Timetable: The AI-LMS advancement, testing, and sending might have to stick to explicit courses of events, which can restrict the degree or profundity of certain highlights because of time constraints.

Stakeholder Availability: Teaming up with different partners, like teachers, heads, and understudies, requires coordination and availability, which can affect the venture's course of events and expectations.

Project Limits and Limitations:

Included Functionalities: Obviously, characterise the centre functionalities and elements that the AI-LMS venture will incorporate, like course management, content sharing, understudy educator communications, and evaluations.

Excluded Functionalities: Determine any functionalities or elements that are unequivocally avoided from the task scope because of asset restrictions, specialised constraints, or explicit venture goals. For the model, the AI-LMS task might prohibit progressed highlights, for example, augmented reality reproductions, mechanised article evaluation, or complex data investigation because of asset or specialised restrictions. It is fundamental to deal with partners' assumptions by plainly conveying the venture limits and guaranteeing arrangement with the available assets, course of events, and goals.

By recognising and addressing these constraints and delimitations forthrightly, the AI-LMS venture can lay out sensible assumptions, focus on fundamental highlights, allot assets, and guarantee fruitful execution inside the characterised limits.

Summary

The methodology segment gives a complete outline of the AI-LMS project, its examination reason, research setting, dependability measures, and impediments. The central issues talked about are as per the following:

Research Reason: The reasoning for fostering the AI-LMS is driven by the need to conquer the impediments of traditional learning management frameworks (LMS) and give a personalised, intelligent, and connecting learning experience. The venture aims to improve understudy results, work with data-driven independent direction, and cultivate collaboration among students.

Research Setting: The AI-LMS will be executed inside instructive establishments, focusing on different settings like schools, universities, colleges, or web-based learning platforms. The framework will take special care of teachers, employees, and understudies, obliging different courses and subjects.

Trustworthiness Measures: To guarantee the dependability and unwavering quality of the AI-LMS, measures like data security, client protection insurance, framework unwavering quality, and precision approval are carried out. Encryption, access control, and ordinary security reviews defend client data. Client testing, execution measurements, and nonstop improvement procedures approve the framework's precision and adequacy.

Limitations and Delimitations: The AI-LMS project faces impediments and constraints concerning specialised similarity, asset availability, and time-related factors. Monetary constraints, specialised skills, and partner availability influence the task scope. The limits and limitations of the undertaking are characterised to explain included and prohibited functionalities. The methodology segment makes way for the ensuing areas of the undertaking, like the turn of events, execution, and assessment. The data given in this part directs the plan and execution of the AI-LMS, considering the examination setting, reliability measures, and venture constraints. The resulting segments will dig into the particular details of the AI-LMS advancement, execution techniques, and the assessment of its viability, ease of use, and effect on the learning experience.

Chapter 4: Implementation

Introduction

Through the brainstorming of possible approaches, the best fit is evaluated and designed as state of the art presented in the previous chapter. It led to the handling as well as the organisation of the segments that needed to be worked on, planned and executed for the development of the entire LMS. The initiation of the practical implementation begins with the identification of the possible pages, or in other words, it could be presented as the functionalities each sort of user will get. This planning and segregation of roles lead to the development of the entities and the establishment of relations between them. Then, take the entire journey from the development of the functionality along with the interfaces that will, in the end presented to the users. The entire development practice is presented comprehensively and interactively.

Database schema

1. 'attendance_attendance': This table has sections like 'id', 'date', 'status', 'created_at', 'updated_at', 'course_id', and 'student_id'. It appears to store participation data for understudies.

2. 'auth_group': This table contains data about client gatherings, with sections 'id' and 'name'.

3. 'auth_group_permissions': This table relates client gatherings to explicit authorisations, with sections 'id', 'group_id', and 'permission_id'.

4. 'auth_permission': This table holds authorisation data, with segments 'id', 'content_type_id', 'codename', and 'name'.

5. 'auth_user': This table stores client data, including sections, for example, 'id', 'secret phrase', 'last_login', 'is_superuser', 'username', 'last_name', 'email', 'is_staff', 'is_active', 'date_joined', and 'first_name'.

6. 'auth_user_groups': This table shows a connection among clients and client gatherings, with sections 'id', 'user_id', and 'group_id'.

7. 'auth_user_user_permissions': This table links clients to explicit consents, with segments 'id', 'user_id', and 'permission_id'.

8. 'discussion_facultydiscussion': This table appears to store conversations started by employees. It has segments like 'id', 'content', 'sent_at', 'course_id', and 'sent_by_id'.

9. 'discussion_studentdiscussion': This table seems to store conversations started by understudies. It has segments 'id', 'content', 'sent_at', 'course_id', and 'sent_by_id'.

The “Unloading information for table” segment contains Additional explanations to populate the tables with information trails each table.

Figure 3 Entity Relationship diagram

Site-Map

For the understanding of the overall design and flow of available functionality, a sitemap is presented in a diagrammatic manner.

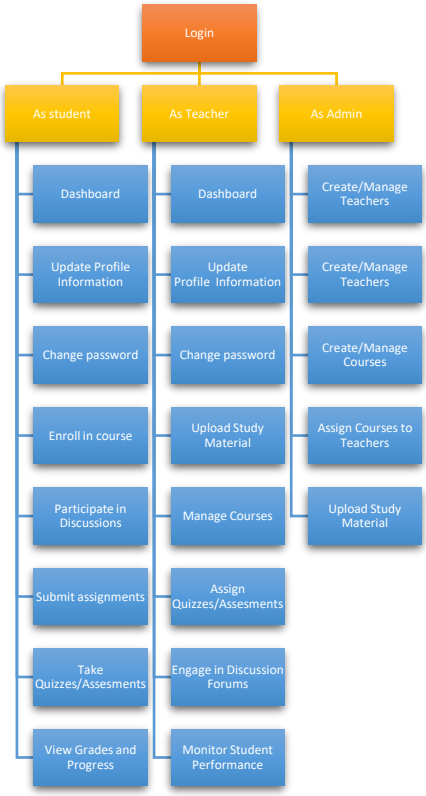


Figure 4 Sitemap of the entire website

This includes all the users, such as the student, teacher, as well as admin. The roles and responsibilities of using this system lie under their names, as shown in Figure 3.

Implementing a learning management system and online assessment system for scholarly training, AI-LMS includes a few key features and functionalities. In this clarification, I'll give a detailed outline of each element, its execution, and its capabilities.

Admin Management

The admin is answerable for dealing with the overall system. They have the following capabilities:

Adding courses, teachers, and students: The admin can make new courses, add teachers to those courses, and enrol students in the system.

Course task: The admin assigns teachers to explicit courses, permitting them to deal with the content and assessments for those courses.

```
@login_required
@admin_required
def staff_add_view(request):
    if request.method == 'POST':
        form = StaffAddForm(request.POST)
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        if form.is_valid():
            form.save()
            messages.success(request, "Account for lecturer " + first_name + ' ' + last_name + " has been created.")
            return redirect("lecturer_list")
    else:
        form = StaffAddForm()

    context = {
        'title': 'Lecturer Add | DjangoSMS',
        'form': form,
    }

    return render(request, 'accounts/add_staff.html', context)
```

Figure 5 Add Teacher by Admin

```python
# Student views
# ####################################################
@login_required
@admin_required
def student_add_view(request):
    if request.method == 'POST':
        form = StudentAddForm(request.POST)
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        if form.is_valid():
            form.save()
            messages.success(request, 'Account for ' + first_name + ' ' + last_name + ' has been created.')
            return redirect('student_list')
        else:
            messages.error(request, 'Correct the error(s) below.')
    else:
        form = StudentAddForm()

    return render(request, 'accounts/add_student.html', {
        'title': "Add Student | DjangoSMS",
        'form': form
    })
```

Figure 6 Add Student by Admin

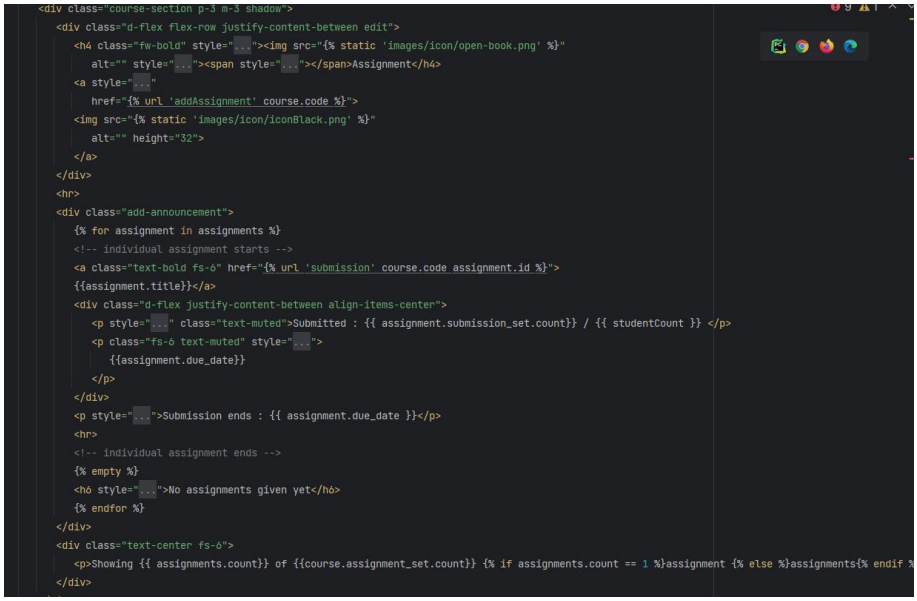
Teacher Management

Teachers include different obligations inside the system. They can play out the accompanying tasks:

Course content creation: Teachers can make and transfer course materials, for example, address slides, readings, and sight and sound content for their allocated courses.

Announcements: Teachers can post significant reports or updates connected with the course, guaranteeing successful student correspondence.

Assignments: Teachers can make assignments, characterise accommodation cutoff times, and give detailed directions. Students can present their work electronically, and teachers can assess and provide feedback in Fig 6.



```
<div class="course-section p-3 m-3 shadow">
    <div class="d-flex flex-row justify-content-between edit">
        <h4 class="fw-bold" style="..."><img src="{% static 'images/icon/open-book.png' %}"
            alt="" style="..."><span style="..."></span>Assignment</h4>
        <a style="..."
            href="{% url 'addAssignment' course.code %}">
        <img src="{% static 'images/icon/iconBlack.png' %}"
            alt="" height="32">
        </a>
    </div>
    <hr>
    <div class="add-announcement">
        {% for assignment in assignments %}
        <!-- individual assignment starts -->
        <a class="text-bold fs-6" href="{% url 'submission' course.code assignment.id %}">
        {{assignment.title}}</a>
        <div class="d-flex justify-content-between align-items-center">
            <p style="..." class="text-muted">Submitted : {{ assignment.submission_set.count}} / {{ studentCount }} </p>
            <p class="fs-6 text-muted" style="...">
                {{assignment.due_date}}
            </p>
        </div>
        <p style="...">Submission ends : {{ assignment.due_date }}</p>
        <hr>
        <!-- individual assignment ends -->
        {% empty %}
        <h6 style="...">No assignments given yet</h6>
        {% endfor %}
    </div>
    <div class="text-center fs-6">
        <p>Showing {{ assignments.count}} of {{course.assignment_set.count}} {% if assignments.count == 1 %}assignment {% else %}assignments{% endif %
    </div>
```

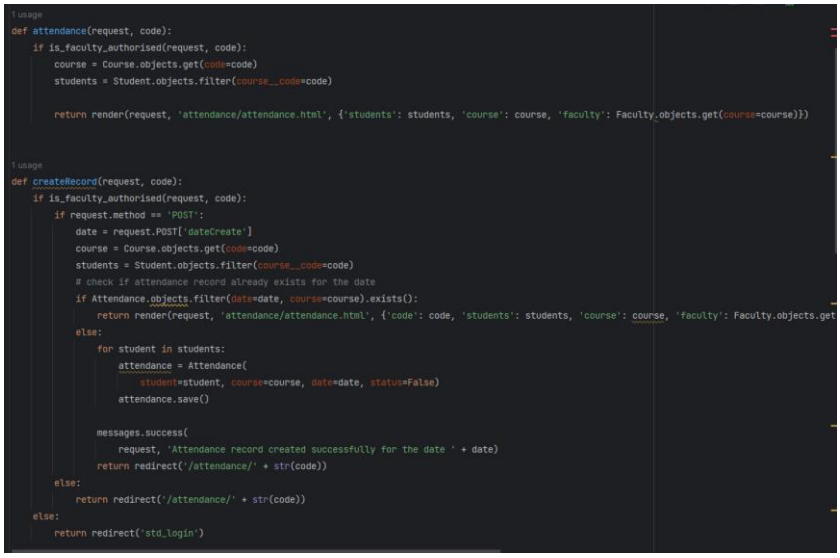
Figure 7 Add assignment.

Quizzes: Teachers can make quizzes with various decisions, valid/bogus, or short-response questions. They can set time restrictions and determine the number of endeavours permitted. Students can take quizzes online, and teachers can assess their reactions in Fig 7.

```
def quiz(request, code):
    try:
        course = Course.objects.get(code=code)
        if is_faculty_authorised(request, code):
            if request.method == 'POST':
                title = request.POST.get('title')
                description = request.POST.get('description')
                start = request.POST.get('start')
                end = request.POST.get('end')
                publish_status = request.POST.get('checkbox')
                quiz = Quiz(title=title, description=description, start=start,
                            end=end, publish_status=publish_status, course=course)
                quiz.save()
                return redirect('addQuestion', code=code, quiz_id=quiz.id)
            else:
                return render(request, 'quiz/quiz.html', {'course': course, 'faculty': Faculty.objects.get(faculty_id=request.session['faculty_id'])})

        else:
            return redirect('std_login')
    except:
        return render(request, 'error.html')
```

Figure 8 Quiz

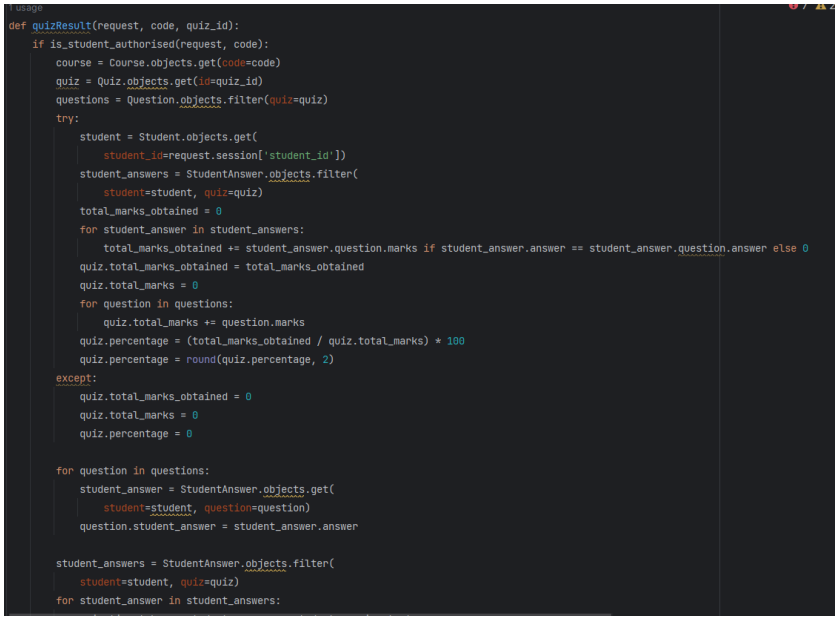
Attendance: Teachers can mark attendance for each class meeting, keeping track of student interest and attendance records in Fig 8.



```
1 usage
def attendance(request, code):
    if is_faculty_authorised(request, code):
        course = Course.objects.get(code=code)
        students = Student.objects.filter(course__code=code)

        return render(request, 'attendance/attendance.html', {'students': students, 'course': course, 'faculty': Faculty.objects.get(course=course)})


1 usage
def createRecord(request, code):
    if is_faculty_authorised(request, code):
        if request.method == 'POST':
            date = request.POST['dateCreate']
            course = Course.objects.get(code=code)
            students = Student.objects.filter(course__code=code)
            # check if attendance record already exists for the date
            if Attendance.objects.filter(date=date, course=course).exists():
                return render(request, 'attendance/attendance.html', {'code': code, 'students': students, 'course': course, 'faculty': Faculty.objects.get
            else:
                for student in students:
                    attendance = Attendance(
                        student=student, course=course, date=date, status=False)
                    attendance.save()

                messages.success(
                    request, 'Attendance record created successfully for the date ' + date)
                return redirect('/attendance/' + str(code))
        else:
            return redirect('/attendance/' + str(code))
    else:
        return redirect('std_login')
```

Figure 9 Student Attendance

Assessment analysis: Teachers can see the detailed analysis of assessments, including student execution, normal scores, and question-level insights. This aids in assessing the adequacy of showing strategies and distinguishing regions for development in Fig 9.



```
1 usage
def quizResult(request, code, quiz_id):
    if is_student_authorised(request, code):
        course = Course.objects.get(code=code)
        quiz = Quiz.objects.get(id=quiz_id)
        questions = Question.objects.filter(quiz=quiz)
        try:
            student = Student.objects.get(
                student_id=request.session['student_id'])
            student_answers = StudentAnswer.objects.filter(
                student=student, quiz=quiz)
            total_marks_obtained = 0
            for student_answer in student_answers:
                total_marks_obtained += student_answer.question.marks if student_answer.answer == student_answer.question.answer else 0
            quiz.total_marks_obtained = total_marks_obtained
            quiz.total_marks = 0
            for question in questions:
                quiz.total_marks += question.marks
            quiz.percentage = (total_marks_obtained / quiz.total_marks) * 100
            quiz.percentage = round(quiz.percentage, 2)
        except:
            quiz.total_marks_obtained = 0
            quiz.total_marks = 0
            quiz.percentage = 0

        for question in questions:
            student_answer = StudentAnswer.objects.get(
                student=student, question=question)
            question.student_answer = student_answer.answer

        student_answers = StudentAnswer.objects.filter(
            student=student, quiz=quiz)
        for student_answer in student_answers:
```

Figure 10 Student results

Student Management

Students approach different features and work inside the system, including Course enrolment: Students can look for and enrol in courses using access keys given by the admin or teachers.

Course content access: Enrolled students can access course materials, including address notes, introductions, and different assets transferred by the teachers.

```
def addQuestion(request, code, quiz_id):
    try:
        course = Course.objects.get(code=code)
        if is_faculty_authorised(request, code):
            quiz = Quiz.objects.get(id=quiz_id)
            if request.method == 'POST':
                question = request.POST.get('question')
                option1 = request.POST.get('option1')
                option2 = request.POST.get('option2')
                option3 = request.POST.get('option3')
                option4 = request.POST.get('option4')
                answer = request.POST.get('answer')
                marks = request.POST.get('marks')
                explanation = request.POST.get('explanation')
                question = Question(question=question, option1=option1, option2=option2,
                                    option3=option3, option4=option4, answer=answer, quiz=quiz, marks=marks, explanation=explanation)
                question.save()
                messages.success(request, 'Question added successfully')
            else:
                return render(request, 'quiz/addQuestion.html', {'course': course, 'quiz': quiz, 'faculty': Faculty.objects.get(faculty_id=request.sessio
            if 'saveOnly' in request.POST:
                return redirect('allQuizzes', code=code)
            return render(request, 'quiz/addQuestion.html', {'course': course, 'quiz': quiz, 'faculty': Faculty.objects.get(faculty_id=request.session['
        else:
            return redirect('std_login')
    except:
        return render(request, 'error.html')
```

Figure 11 Quiz questions

Assessments: Students can participate in assignments, quizzes, and tests. They can present their work on the web and get grades and teacher feedback.

Results and execution tracking: Students can see their assessment brings about detail, including scores, feedback, and any significant insights. This permits them to screen their advancement and distinguish regions for development.

Discussion section: Students can participate in course-explicit discussion sections, where they can ask questions, seek explanations, and participate in cooperative learning. Teachers can likewise communicate with students in these discussion gatherings, working in a supportive learning Environment.

Figure 12 Discussion

Overall System Functionality

User confirmation and access control: The system ought to have secure user validation instruments to guarantee that the main approved users (admin, teachers, and students) can access the system and its functionalities.

User-friendly interface: The system ought to have a natural and simple to-utilise interface that permits users to easily explore different features.

Information management: The system ought to store and oversee user information, course data, assessment details, and other important information safely. This incorporates a data set plan and execution to guarantee proficient capacity and recovery of data.

Notices and alarms: The system ought to give warnings and cautions to users for significant updates, like new assignments, impending cut-off times, or new discussion posts.

Revealing and investigation: The system ought to produce reports and give examinations to administrators and teachers, permitting them to break down overall system execution, course viability, and student progress.

System maintenance and support: The system ought to have arrangements for standard maintenance, bug fixes, and specialised support to guarantee continuous activity and user fulfilment. Implementing the AI-LMS system requires a mix of programming dialects (like Python), web improvement frameworks (like Django), and data set innovations (like MySQL or SQLite). Moreover, secure coding rehearses information encryption and adherence to protection guidelines ought to be considered during the advancement cycle. Important to implement a whole AI-LMS system with every one of its features and functionalities commonly includes a group of engineers, fashioners, and analysers working cooperatively to guarantee a hearty and solid arrangement. The above clarification gives a far-reaching outline, yet the genuine execution details might fluctuate in light of explicit necessities, innovations, and structural decisions.

Analysis of Implemented Features

The AI-LMS project incorporates various features that are planned to satisfy the needs of administrators, educators, and students. Here is an intensive breakdown of the features offered to every job:

1. Administrator:

Administrators are accountable for regulating and administering the AI-LMS system. They are furnished areas of strength for with for performing administrative tasks and guaranteeing the stage's perfect activity. Among the essential features available to admins are in Fig 12; administrators can add, eliminate, and change data about teachers and students.

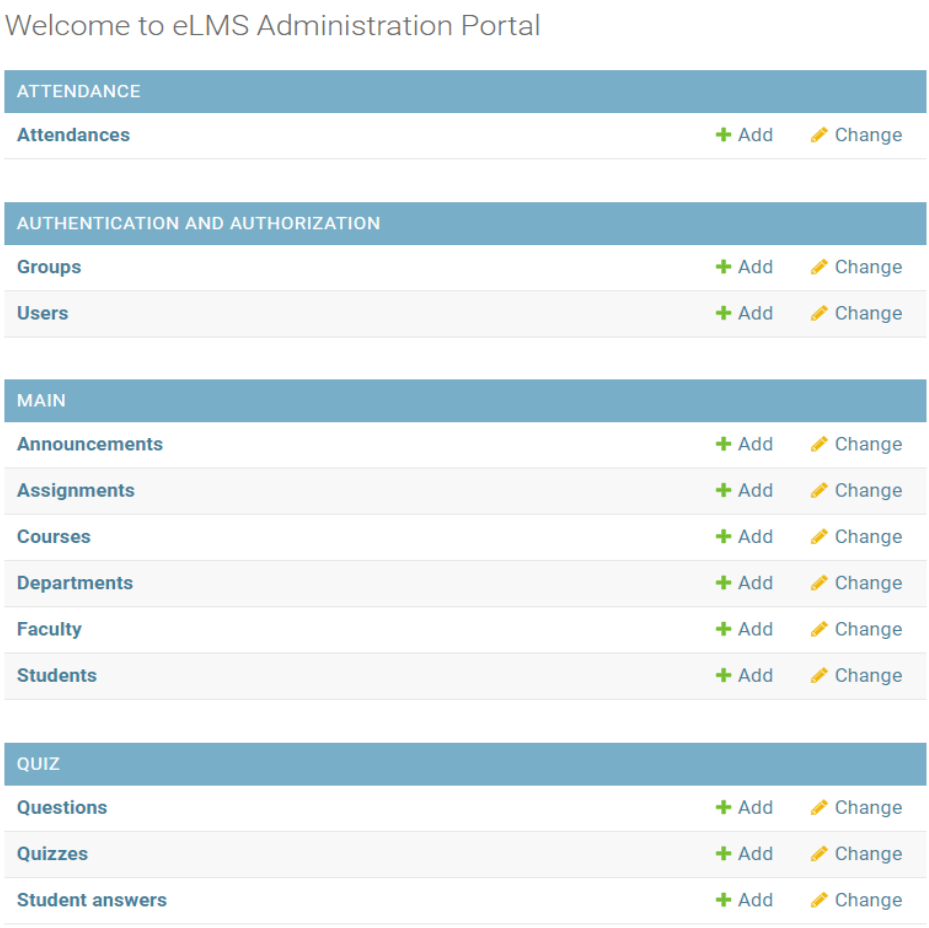


Figure 13 Admin management

They can create user accounts, assign jobs and authorizations, and manage user verification and access control. Administrators may develop, update, and manage courses and departments in the LMS. They may design course frameworks, establish criteria, and assign instructors to specific courses. Administrators can control instructional content inside the LMS. They can upload and coordinate study materials, multimedia resources, and assessments.

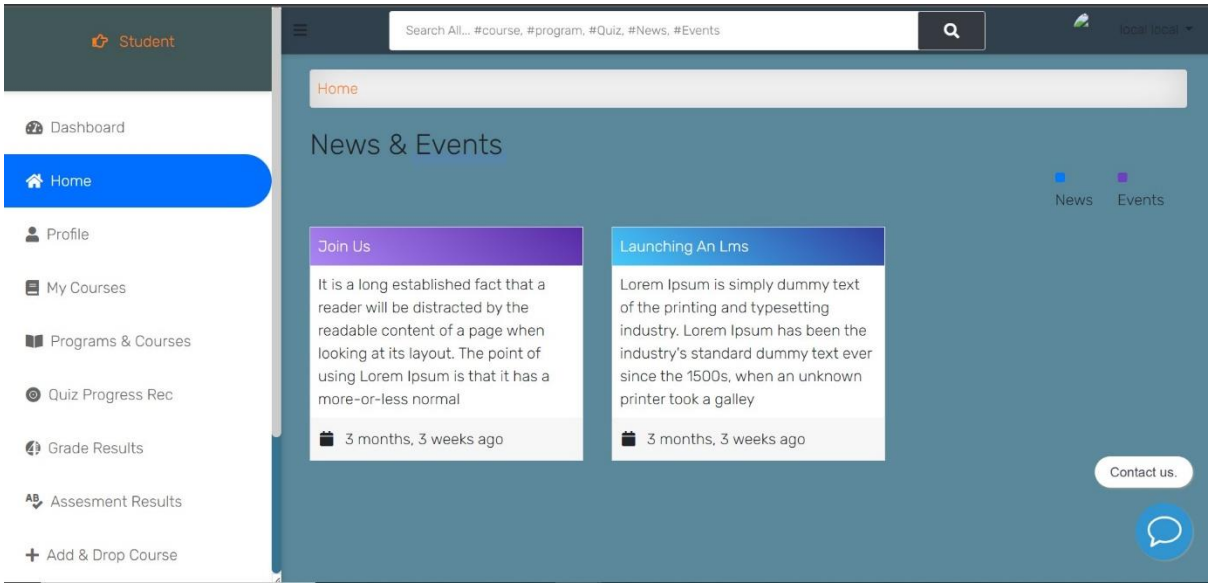


Figure 14 Home page of Admin management

They can also update content, archive or delete obsolete information, and guarantee the quality and usefulness of the content. Administrators may receive insights into overall system performance, student growth, and instructor effectiveness by accessing analytics and generating reports. They may monitor important metrics like course enrolments, completion rates, and evaluation scores and utilise this data to make data-driven choices and changes.

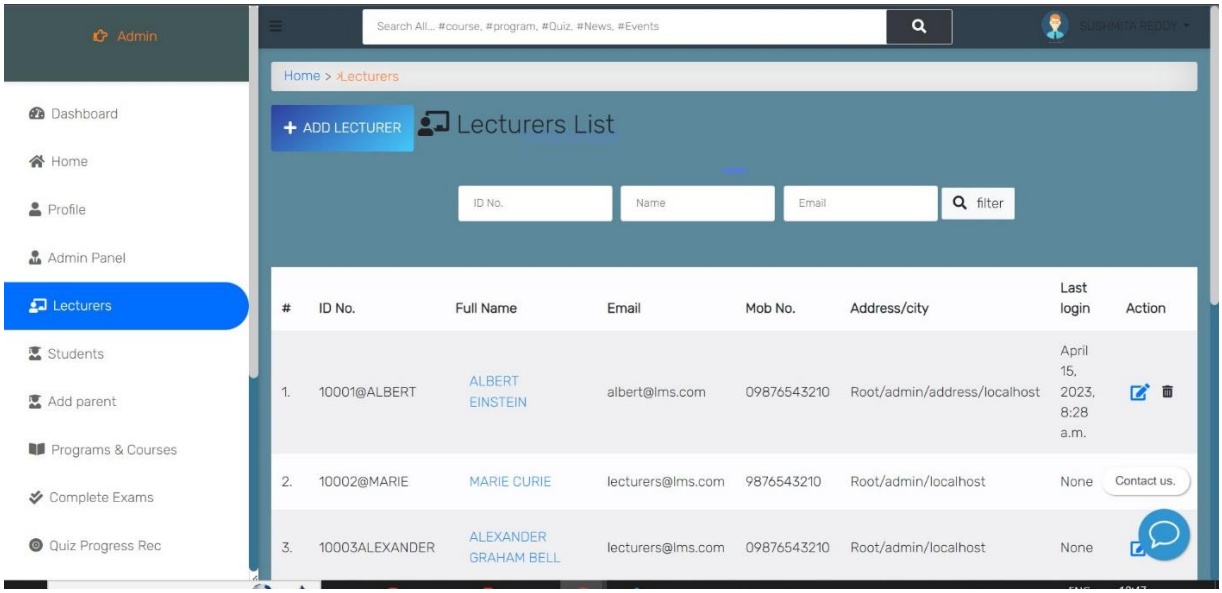


Figure 15 Admin manages Teachers.

Administrators may customise a variety of system parameters, including LMS look, user interface customisation, and interaction with external tools or services. They may also handle system upgrades, backups, and security procedures to maintain the stability and data security of the platform.

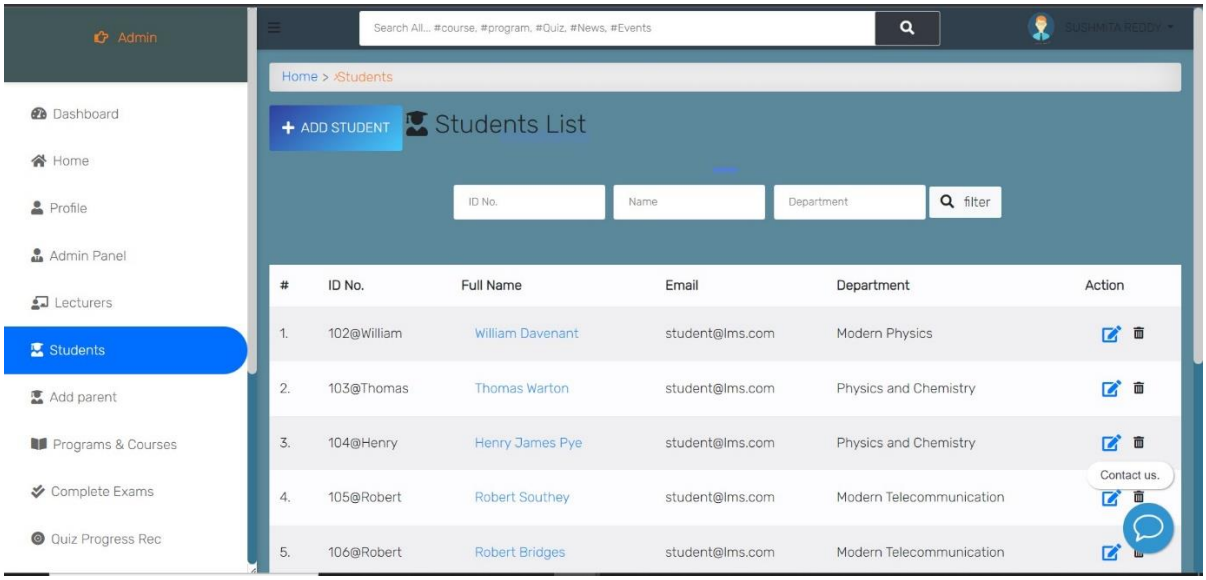


Figure 16 Admin manages Students

2. Teacher:

Teachers are critical in providing instructional information, monitoring student performance, and offering direction. The AI-LMS project offers teachers a variety of functions to help them with their educational work. Among the primary features available to teachers are the following: Teachers may use the LMS to design and manage their courses. They can define course objectives, syllabi, and learning outcomes.

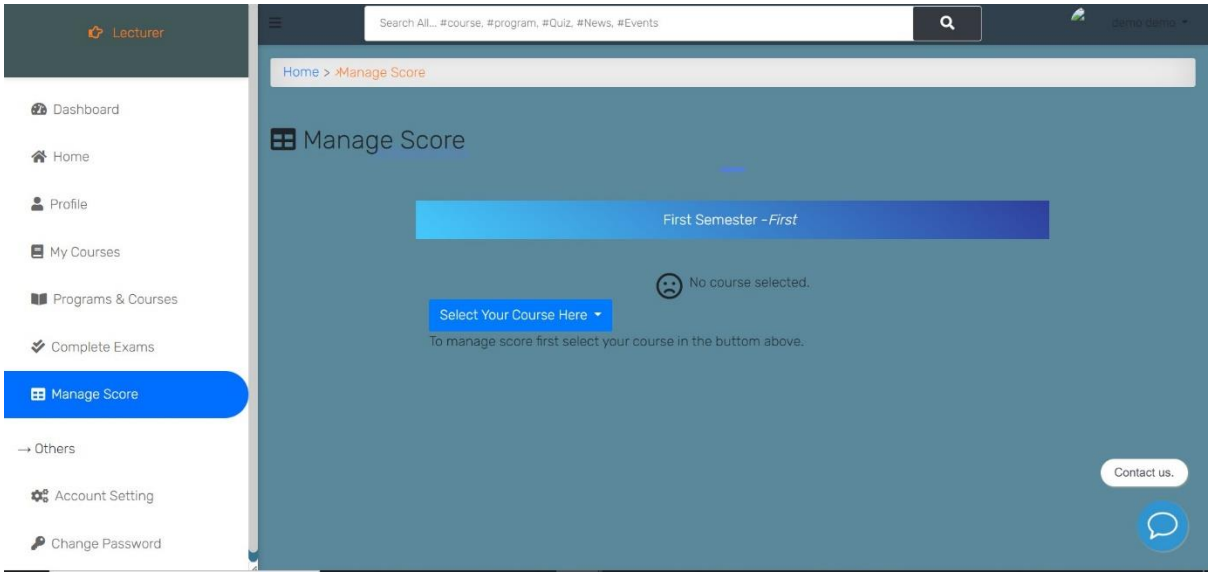


Figure 17 Teacher Dashboard

They can also construct lesson plans, modules, and units, as well as systematically organise information. Teachers may use the LMS to upload study materials, lecture notes, presentations, and other educational resources. They can arrange and categorise information based on subjects, weeks, or modules to make it easier for students to navigate and access.

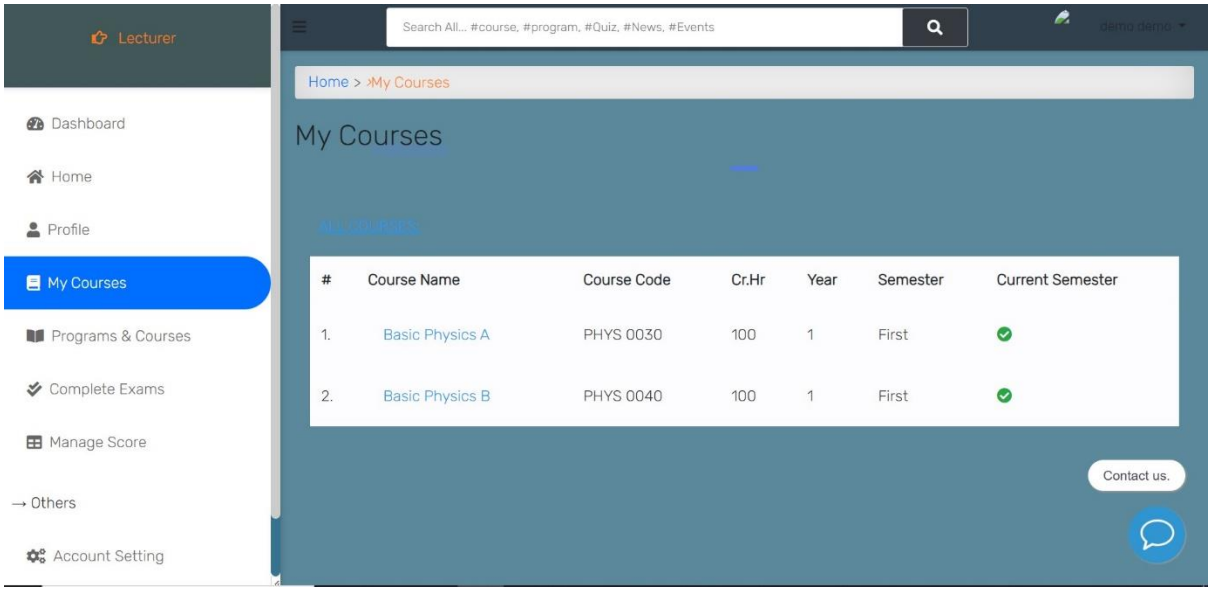


Figure 18 Courses name of the Teacher dashboard

Within the LMS, teachers may design quizzes, assignments, and assessments. They can specify time limitations, grading rubrics, and submission criteria. They can evaluate student work, offer feedback, and monitor student development. Teachers may monitor and evaluate enrolment and performance statistics for their students. Individual student progress may be monitored, areas of improvement or difficulty identified, and tailored instruction or support provided.

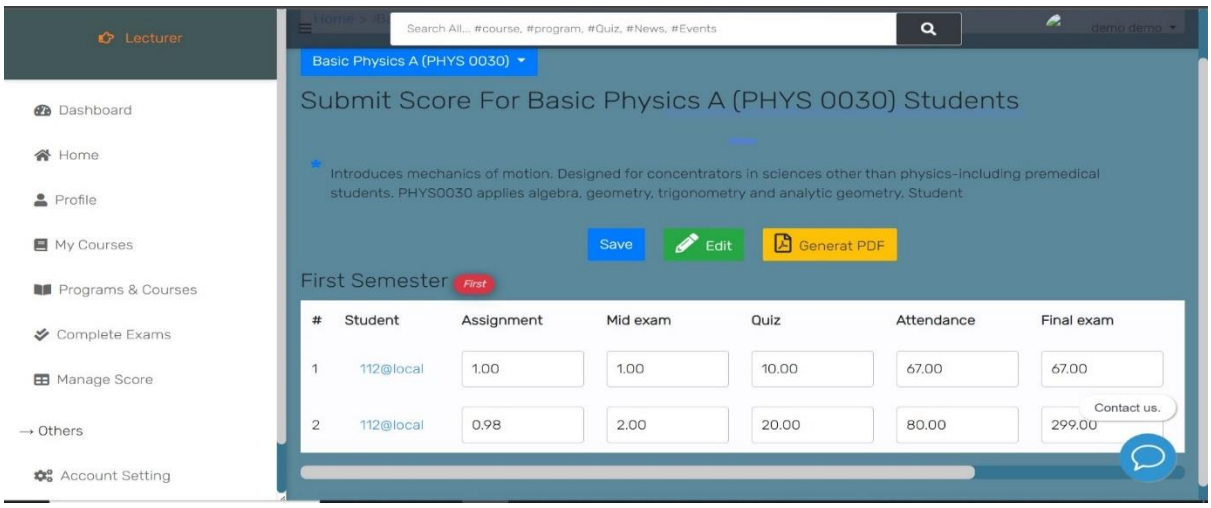


Figure 19 Teacher discussion

Teachers can interact with students via discussion boards, chat rooms, or message systems built into the LMS. They can answer questions, give explanations, enable peer-to-peer exchanges, and build collaborative learning settings. Teachers may get analytics and reports tailored to their unique courses and students. Attendance, assignment submissions, quiz scores, and other pertinent indicators may be tracked. This data supports teachers in analysing their teaching tactics and adjusting course content to fit the requirements of their students in Fig 19.

```python
class Faculty(models.Model):
    faculty_id = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=100, null=False)
    email = models.EmailField(max_length=100, null=True, blank=True)
    password = models.CharField(max_length=255, null=False)
    department = models.ForeignKey(
        'Department', on_delete=models.CASCADE, null=False, related_name='faculty')
    role = models.CharField(
        default="Faculty", max_length=100, null=False, blank=True)
    photo = models.ImageField(upload_to='profile_pics', blank=True,
                              null=False, default='profile_pics/default_faculty.png')

    2 usages (2 dynamic)
    def delete(self, *args, **kwargs):
        if self.photo != 'profile_pics/default_faculty.png':
            self.photo.delete()
        super().delete(*args, **kwargs)

    class Meta:
        verbose_name_plural = 'Faculty'

    def __str__(self):
        return self.name
```

Figure 20 Faculty model

3. Student

The AI-LMS project's major benefactors are students since it intends to create tailored learning experiences and help their academic path. The following are the primary features provided to students: Inside the LMS, students can investigate and enrol in offered courses. They will want to see course details, prerequisites, and availability.

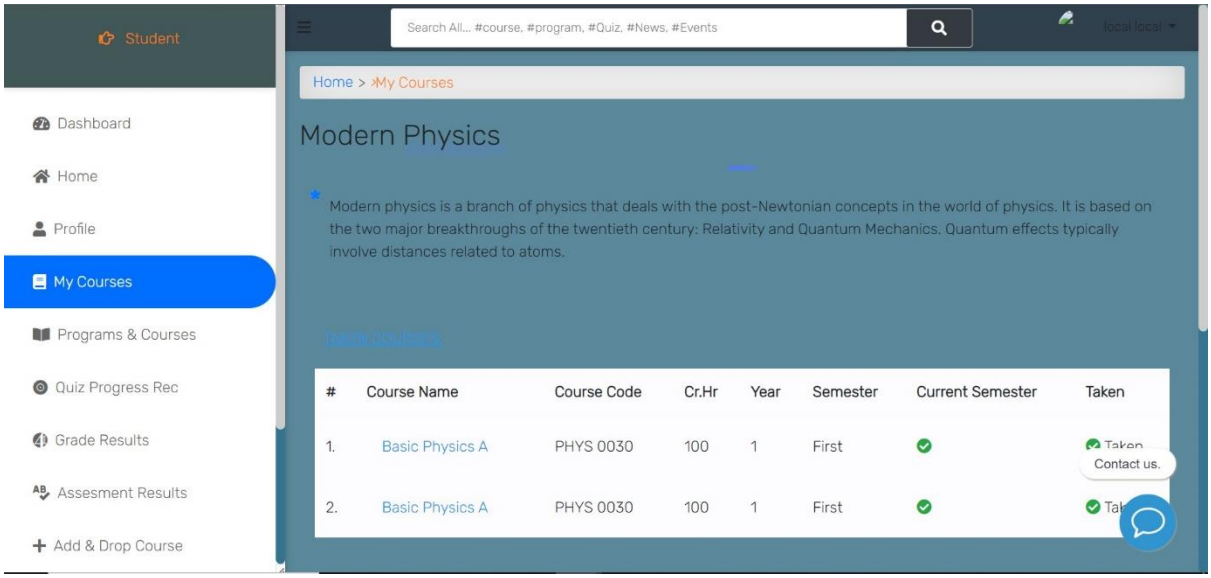


Figure 21 Student Dashboard

They approach enrolled courses and can peruse the content. Students gain admittance to concentrate on materials, address notes, introductions, and media instruments that professors have submitted. They can learn at their speed by reviewing or downloading the materials during their relaxation.

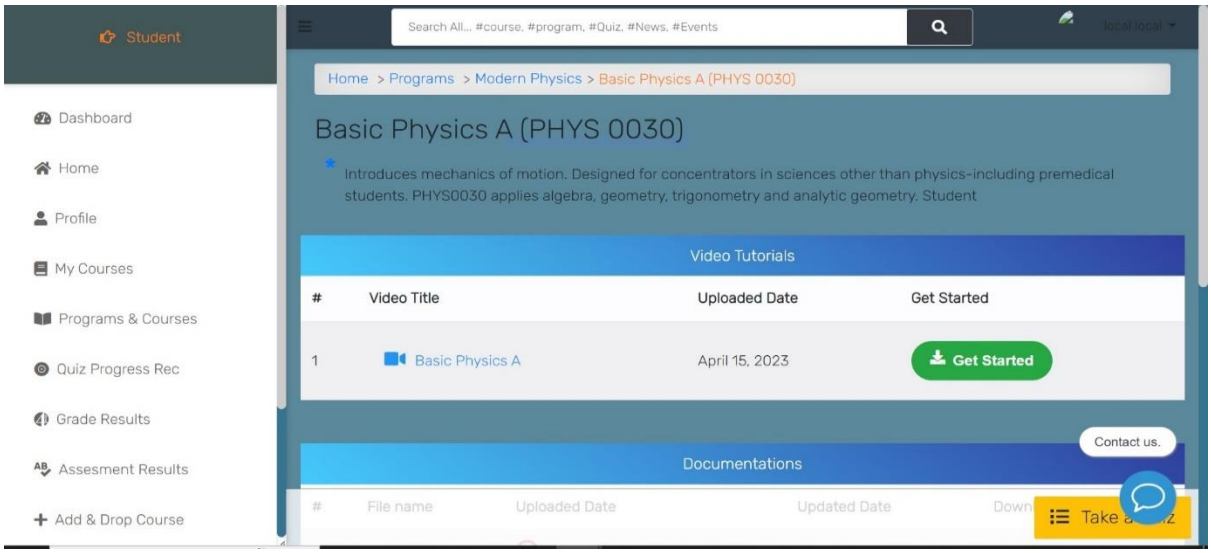


Figure 22 Subject details on the Student dashboard

Students can take quizzes, assignments, and assessments offered by teachers. They can turn in their reactions or complete tasks by the dates. They might inspect their grades as well as teacher feedback. Students can screen their advancement during a course. They can see their completed module or units, screen their test results, and track their overall exhibition. This permits students to evaluate their perception and focus on the areas that need more consideration.

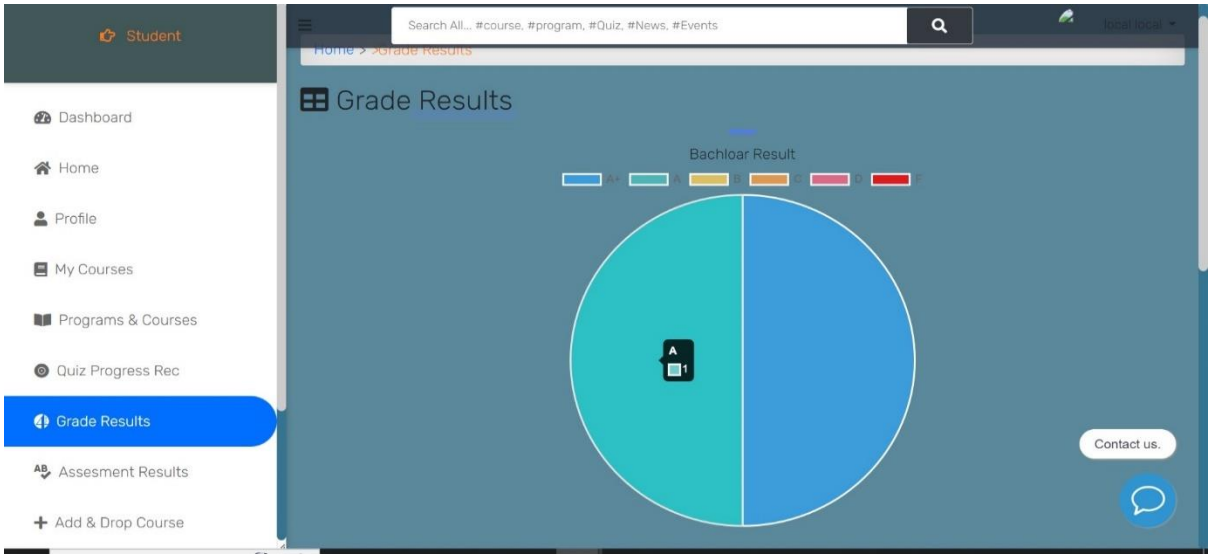


Figure 23 Student Quiz

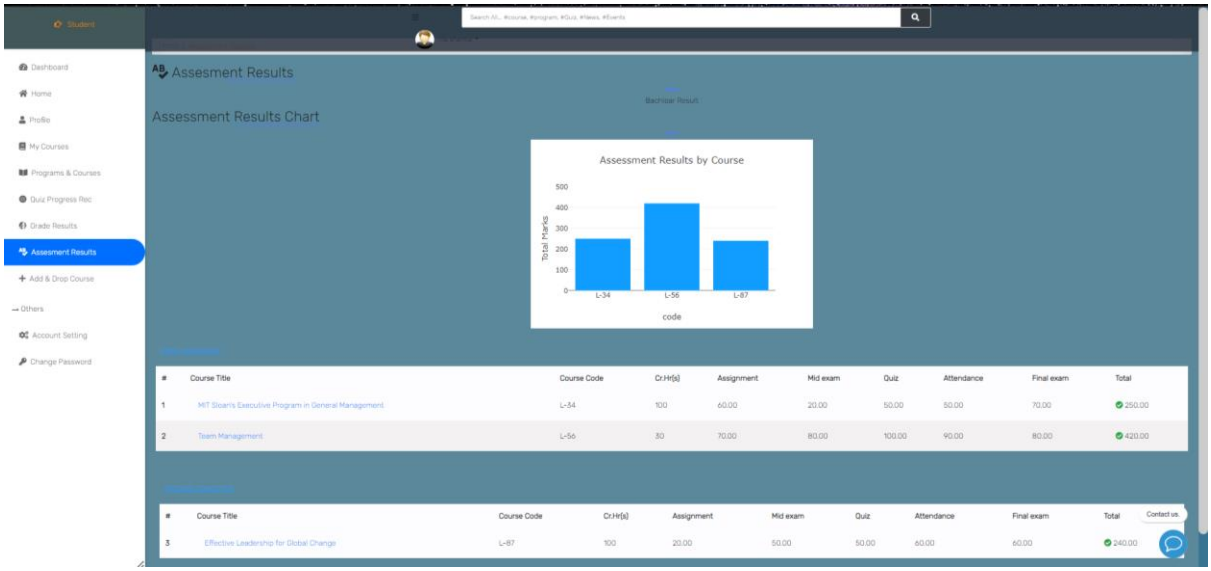


Figure 24 Student Course Assessment

Students can participate in the LMS's discussion visits, gatherings, or informing systems. They might ask questions, take part in discussions, and work with their companions to foster a feeling of local area and support. Students approach their scores, feedback, and execution information. They can recognise regions for development and, if important, seek additional help or explanation from teachers. Overall, the AI-LMS project gives a complete assortment of features and works designated to administrators, educators, and students with specific obligations. These features are planned to improve administrative cycles, upgrade compelling education and learning, and increment student commitment and accomplishment inside the AI-Learning Management System in Fig 23.

```python
class Student(models.Model):
    student_id = models.IntegerField(primary_key=True)
    name = models.CharField(max_length=100, null=False)
    email = models.EmailField(max_length=100, null=True, blank=True)
    password = models.CharField(max_length=255, null=False)
    role = models.CharField(
        default="Student", max_length=100, null=False, blank=True)
    course = models.ManyToManyField(
        'Course', related_name='students', blank=True)
    photo = models.ImageField(upload_to='profile_pics', blank=True,
                              null=False, default='profile_pics/default_student.png')
    department = models.ForeignKey(
        'Department', on_delete=models.CASCADE, null=False, blank=False, related_name='students')

    2 usages (2 dynamic)
    def delete(self, *args, **kwargs):
        if self.photo != 'profile_pics/default_student.png':
            self.photo.delete()
        super().delete(*args, **kwargs)

    class Meta:
        verbose_name_plural = 'Students'

    def __str__(self):
        return self.name
```

Figure 25 Student model

A powerful Chabot created for the Learning Management System (LMS) project is included to improve interactions with students. This system, which was created using the Python and Django frameworks, combines machine learning with HTML, CSS, and JavaScript to provide a simple and effective learning environment. The database makes use of SQLite, while Jinja2 makes sure that dynamic material is shown. The Chabot's primary characteristics include several crucial ones. It skilfully displays the names of the instructors, the courses that are offered, and offers technical help. Students may easily access details concerning their courses and teachers and get help as required, thanks to an intuitive user design.

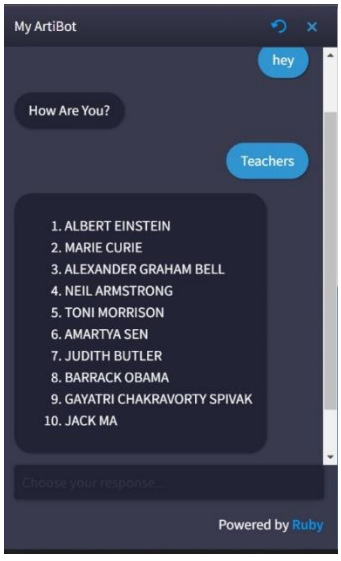


Figure 26 Chatbot application: Teachers Name displaying

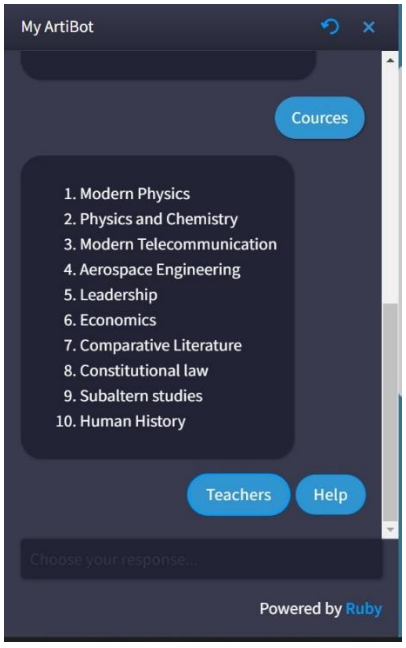


Figure 27 Chatbot application: Courses displaying

The Chabot can comprehend and intelligently answer student inquiries thanks to Python and machine learning. The backend architecture of Django allows for efficient data management and module interaction. Together, HTML, CSS, and JavaScript create a visually beautiful and approachable front-end experience. Additionally, the use of SQLite guarantees trustworthy data storage and retrieval. The involvement of users is increased by Jinja2's dynamically generated tailored content.

Summary

The implementation part of this report outlines a painstakingly planned design for the thorough creation and introduction of an AI-powered Learning Management System (AI-LMS), with a focus on administrators, teachers, and students. An extensive brainstorming stage precedes the creation of an innovative LMS, as was discussed in the preceding chapter. This methodical technique emphasises the tactical arrangement and operation of system parts, concentrating intently on meeting the all-inclusive LMS ecosystem's developing criteria. The database schema was meticulously constructed and includes necessary entities. The foundation of the system's functional framework is made up of these entities, which guarantees efficient data management. Operational liberty, which makes it easy to add, handle, and distribute courses, instructors, students, and user access, is a key component of AI-LMS capability. Teachers have the power to manage attendance, assignments, quizzes, and other aspects of complete teaching. To enrol, access resources, participate in exams, and

work with peers and instructors, participants use the AI-LMS. The paper also provides a ground-breaking Chatbot integration that is completely woven into the AI-LMS fabric.

Chapter 5: Discussion and Results

Introduction

The AI-LMS effort is a large-scale project that synergistically combines learning management systems (LMS) with artificial intelligence (AI) techniques to bring about improvements and paradigm-shifting changes in the field of education. The main objective of this project is to create an intelligent and adaptable educational environment that is capable of customising learning experiences to meet the distinctive needs of various students, simplifying didactic materials, and offering crucial insights to instructors.

Analysis of Implemented Features

The distinctiveness and efficacy of the created system are determined by comparing the properties of the artificial intelligence (AI) e-learning management system built with Django to the literature sources offered. Let's talk about this literary analysis:

The implemented system uses AI technologies to improve the e-learning experience, which is similar to what has been done in the literature. However, the system that has been put into place focuses exclusively on using the Django framework as a base for development. Since the implemented system is also Django-based, it is consistent with the literature source. The constructed system differs from a conventional Django-based e-learning website only in that it incorporates AI. As opposed to this literature source, the implemented system places more emphasis on AI-based features for individualised learning and suggestions. Along with material distribution, the installed system also emphasises course management and examinations. Because the implemented system integrates artificial intelligence (AI) features with the e-learning management system, it is consistent with the literature source. However, the precise features and specifics of implementation may vary.

The implemented system's general objective is to address the difficulties students confront with online learning, but its specific characteristics and AI integration may vary. As stated in this literature source, the established system takes into account user experience by putting a strong emphasis on usability, intuitiveness, and user-friendliness. However, the AI-based elements built into the implemented system give the user experience a new depth. This body of literature offers advice on overseeing Django web development, which is pertinent given that the implemented system uses Django as its framework. As mentioned in this literature origin, the implemented system gains from the security practices and features connected with Django. While this literature source focuses especially on incorporating a chatbot with the

Moodle LMS, the built system differs from that source even though it also uses AI. This literature resource emphasises student records management that differs from the actual system's major focus on e-learning management systems. The actual system's AI integration also includes personalised recommendations and assessments. But in terms of how it uses the framework, the Django-based implementation might be similar. The system that has been implemented uses AI. However, it varies in that it also includes chatbot capability. Instead, the system in place places a strong emphasis on individualised recommendations and evaluations.

This literature origin, which addresses the creation of chatbots for educational and instructional aids, is different from the implemented system. Instead of focusing on chatbot capabilities, the deployed system's AI integration centres on customised recommendations and assessments.

The system stands out due to its utilisation of Django as the framework or the implementation of AI-based features, such as tailored suggestions, assessments, and content delivery, when the features of the implemented system are contrasted with those of the provided literature. These distinctive characteristics help to improve the educational experience and set the system apart from other e-learning management systems (Sankar S et al., 2022).

Limitations and Challenges

Several limits and obstacles may have been encountered during the creation of the AI-LMS project. Here are some frequent places that may present difficulties:

Technical Constraints and Scalability

Creating an AI-powered LMS necessitates sophisticated technological implementations. Integrating artificial intelligence and models for machine learning into the system necessitates AI development knowledge and may be hampered by technological restrictions. Data availability and quality for training AI models might sometimes be a problem. Scaling the system to support a high number of concurrent users while maintaining smooth performance might be difficult. Managing huge workloads while balancing computing needs and response times can be a technological issue (Jia et al., 2022).

User Experience and Usability

Developing an easy-to-use and simple interface is critical for an LMS's success. It can be difficult to design an interface that appeals to users with varied degrees of technical ability

while still assuring easy navigation through course content. It can be difficult to balance the display of sophisticated instructional information with simplicity and ease of usage. Another critical factor that must be carefully considered is providing an interesting and dynamic user experience that stimulates student involvement and motivation.

Data Privacy and Security

An AI-LMS handles sensitive data from students and teachers, such as personal data and academic records. It is vital to have strong data privacy and security safeguards that secure user details from illegal access, data breaches, or abuse. Implementing robust authentication procedures, encryption techniques, and compliance with data protection requirements may be difficult and time-consuming, necessitating continual monitoring and changes to keep ahead of any risks.

Integrating External Tools and Services

Integration with additional services and tools such as video calling, content creation tools, or analytics for learning systems is frequently advantageous for LMS platforms. Because of variations in technology, data formats, or APIs, ensuring smooth integration and interoperability with multiple third-party products and services can be difficult. Maintaining compatibility and providing a seamless user experience while accessing linked features may need careful preparation and creation.

Feature Implementation and Functionality Alignment

Implementing specialised features inside the AI-LMS project might be difficult, especially since they must be aligned with the unique requirements of the educational institution or business. Customising the LMS to fit different course formats, grading systems, or instructional methodologies can take a significant amount of time and may necessitate changes to the underlying system architecture. It can be difficult to strike a balance between the demand for flexibility and customisation and the necessity for a cohesive and efficient system (Rumu, 2021).

Teacher and Student Adoption

The implementation of an AI-powered LMS may need a considerable shift in teaching and learning techniques, which may pose obstacles in terms of teacher and student uptake. Educating and educating instructors on how to use AI-LMS features successfully and

incorporate them into their educational techniques can be a time-consuming and continuing task. Similarly, students may require assistance in navigating and exploiting AI-enhanced functions to optimise their learning outcomes (NG, 2015). To overcome these constraints and problems, a combination of technical competence, competent project management, user-centred design principles, and continual improvement based on user feedback and assessment is required. To guarantee the success and sustainability of the AI-LMS project, the development team should be prepared to handle these difficulties iteratively throughout the development phase and beyond.

Future Enhancements and Directions

There are various possible additions and extra features for the AI-LMS project in the future that might be explored to further improve the user experience, increase functionality, and incorporate new technologies. Here are some suggestions: Extending the LMS's AI capabilities to give students even more individualised learning paths can be a huge benefit (Damiani et al., 2021). By incorporating natural language processing skills into the AI-LMS project, students and the system may engage more effectively by maintaining adaptive assessments and gamification. Adding adaptive assessments to the AI-LMS project's assessment features can give more targeted evaluations of student knowledge and abilities. By incorporating virtual and augmented reality (VR/AR) technology into the AI-LMS project, immersive learning experiences may be created. Extending the AI-LMS project's learning analytics capabilities can give significant insights to administrators, teachers, and students. Strengthening the AI-LMS project's social learning and collaborative elements might encourage more engagement and knowledge sharing among students. Students can access course materials and engage in activities on their smartphones or tablets if the AI-LMS project is made more accessible through mobile applications.

Summary

The AI-LMS project sought to transform education by incorporating artificial intelligence (AI) technology into a learning management system (LMS). The project's goals were to customise learning, improve instructional material, and deliver important information to administrators, instructors, and students. The AI-LMS project's success is determined by its potential influence on the target consumers. It simplifies administrative processes, increases data-driven decision-making, and promotes overall system administration for administrators. Teachers can offer tailored learning experiences, organise and distribute course content

effectively, assess student performance, and promote participation and cooperation. Personalised learning paths, access to educational materials, interactive exams, progress monitoring, and increased communication and support all benefit students.

Chapter 6: Conclusion and Recommendation

Conclusion

The AI-LMS project aims to transform the educational environment by providing educational institutions with a comprehensive learning management system. This includes several features and capabilities that are designed to meet the diverse needs of those who use it, including administrators, professors, other lecturers, and students. The project is organised around three key responsibilities, each of which is crucial to the system's efficient operation. The administrator's primary responsibility is to oversee and regulate the overall functioning of the AI-LMS. To fulfil the needs of their educational institutions, administrators may set up and organise courses, administer user accounts, and change other system settings. Additionally, they have utilisation of a huge selection of tools & and control. A second function for an AI-LMS project is played by the instructor or teacher member. These folks are provided access to a unique collection of tools that make it simple for them to create and disseminate educational content. They can administer courses, set assignment due dates, construct and manage courses, as well as upload course resources like presentations, papers, and multimedia files. Teachers can also keep tabs on students' development, assess their progress through interactive exams and assessments, and provide customised feedback to enhance the learning experience.

Lastly, a student's involvement in the AI-LMS project gives students the ability to use course materials, participation in discussions submission of tasks, and collaborative learning activities. Students get access to their dashboards, which provide an overview of the programs they are registered for, as well as information on deadlines and achievement indicators. Students can connect with teachers and peers via discussion boards, creating a lively learning environment. The AI-LMS project offers a wide variety of features that further improve the educational experience for all interested parties. Administrators can simply enrol and manage many users within the system thanks to user management capabilities, resulting in a positive user experience. With the use of course management tools, instructors may effectively organise and arrange their classes, facilitating student access to pertinent information. The ability to upload files makes it simple to distribute course materials, guaranteeing that all relevant materials are readily available to students. Teachers and students may both measure progress and pinpoint areas for growth thanks to performance-tracking methods. Students may contribute, have meaningful discussions, and ask questions about subjects on interactive discussion boards.

Recommendation

Several suggestions can be put into practice to improve the AI-LMS project. First and foremost, concentrate on designing a simple and user-friendly interface with responsive design to provide accessibility across devices. User testing and feedback collecting may then be done to continuously enhance UI/UX. Secondly, to secure user data, use strong security measures such as strong authentication protocols, encryption methods, and frequent updates. Thirdly, improve role-based access management by giving each role the proper permissions and limitations. This will provide administrators with the ability to govern teacher and student access with finer granularity. Additionally, enhance course management tools, such as simple enrolment and course creation procedures. Improved message and discussion chat features will boost communication and teamwork. Improve performance analytics and tracking to deliver in-depth insights and produce reports for data-driven decision-making. Utilise database optimisations and caching techniques to increase performance and scalability. Finally, to effectively aid consumers, offer thorough documentation and responsive customer assistance.

References:

Aldahwan, N.S. and Alsaeed, N.I. (2020). Use of Artificial Intelligent in Learning Management System (LMS): A Systematic Literature Review. International Journal of Computer Applications, 175(13), pp.16–26. doi:https://doi.org/10.5120/ijca2020920611.

Alenezi, A. (2020). The Role of e-Learning Materials in Enhancing Teaching and Learning Behaviors. *International Journal of Information and Education Technology*, 10(1), pp.48–56. doi:https://doi.org/10.18178/ijiet.2020.10.1.1338.

Chandiramani, A. (2021). Management of Django Web Development in Python. Journal of Management and Service Science (JMSS), 1(2), pp.1–17. doi:https://doi.org/10.54060/jmss/001.02.005.

Duisebekova, K.S., Khabirov, R. and Zholzhan, A. (2021). DJANGO AS SECURE WEB-FRAMEWORK IN PRACTICE. Вестник КазАТК, 116(1), pp.275–281. doi:https://doi.org/10.52167/1609-1817-2021-116-1-275-281.

Haghshenas, M. (2019). A Model for Utilising Social Softwares in Learning Management System of E-Learning. 1(4). doi:https://doi.org/10.30473/idej.2019.6124.

Hassen, Q.K. and Aliakbari, M. (2022). The Expectations and Reality of E-Learning. Mediterranean Journal of Social & Behavioral Research, [online] 6(2), pp.61–66. doi:https://doi.org/10.30935/mjosbr/11926.

Hirankerd, K. and Kittisunthonphisarn, N. (2020). E-Learning Management System Based on Reality Technology with AI. International Journal of Information and Education Technology, 10(4), pp.259–264. doi:https://doi.org/10.18178/ijiet.2020.10.4.1373.

Huang, J., Saleh, S. and Liu, Y. (2021). A Review on Artificial Intelligence in Education. Academic Journal of Interdisciplinary Studies, [online] 10(3), p.206. doi:https://doi.org/10.36941/ajis-2021-0077.

Karkar, A.J.M., Fatlawi, H.K. and Al-Jobouri, A.A. (2020). Highlighting E-learning Adoption Challenges using data Analysis Techniques: University of Kufa as a Case Study. *Electronic Journal of e-Learning*, 18(2). doi:https://doi.org/10.34190/ejel.20.18.2.003.

Kovan Mzwri and Márta Turcsányi-Szabó (2023). Chatbot Development using APIs and Integration into the MOOC. Chatbot Development using APIs and Integration into the MOOC, 5(1), pp.18–30. doi:https://doi.org/10.36427/cejntrep.5.1.5041.

Luque, D. and Francisco José García-Peñalvo (2020). Overview of European educational projects on eLearning and related methodologies. doi:https://doi.org/10.1145/3434780.3436550.

Malik, S., Al-Emran, M., Mathew, R., Tawafak, R. and Alfarsi, G. (2020). Comparison of E-Learning, M-Learning and Game-based Learning in Programming Education – A Gendered Analysis. *International Journal of Emerging Technologies in Learning (iJET)*, [online] 15(15), pp.133–146. Available at: https://www.learntechlib.org/p/217975 [Accessed 12 Jun. 2023].

Maslov, I., Nikou, S. and Hansen, P. (2021). Exploring user experience of learning management system. The International Journal of Information and Learning Technology, 38(4), pp.344–363. doi:https://doi.org/10.1108/ijilt-03-2021-0046.

Mendoza, S., Sánchez-Adame, L.M., Urquiza-Yllescas, J.F., González-Beltrán, B.A. and Decouchant, D. (2022). A Model to Develop Chatbots for Assisting the Teaching and Learning Process. Sensors, 22(15), p.5532. doi:https://doi.org/10.3390/s22155532.

Murtaza, M., Ahmed, Y., Shamsi, J.A., Sherwani, F. and Usman, M. (2022). AI-Based Personalised E-Learning Systems: Issues, Challenges, and Solutions. IEEE Access, [online] 10, pp.81323–81342. doi:https://doi.org/10.1109/ACCESS.2022.3193938.

Patil, V.K., Patil, D.S., Satpute, H.B. and Tayade, S.N. (2023). Student Record Management System using Django. International Journal for Research in Applied Science and Engineering Technology, 11(5), pp.18–23. doi:https://doi.org/10.22214/ijraset.2023.51366.

Sameena, Ms. (2021). Django Based E-Learning Website. International Journal for Research in Applied Science and Engineering Technology, 9(12), pp.2266–2272. doi:https://doi.org/10.22214/ijraset.2021.39737.

Sarker, M.F.H., Mahmud, R.A., Islam, M.S. and Islam, M.K. (2019). Use of e-learning at higher educational institutions in Bangladesh. *Journal of Applied Research in Higher Education*, 11(2), pp.210–223. doi:https://doi.org/10.1108/jarhe-06-2018-0099.

Shaiba, H., John, M. and Meshoul, S. (2023). Female Saudi College students' e-learning experience amidst COVID-19 pandemic: An investigation and analysis. *Heliyon*, 9(1), p.e12768. doi:https://doi.org/10.1016/j.heliyon.2022.e12768.

Shilowaras, M. and Jusoh, N.A. (2022). Implementing Artificial Intelligence Chatbot in Moodle Learning Management System. Engineering, Agriculture, Science and Technology Journal (EAST-J), 1(1), pp.70–75. doi:https://doi.org/10.37698/eastj.v1i1.122.

Shyam, A. and Mukesh, N. (2020). A Django Based Educational Resource Sharing Website: Shreic. Journal of scientific research, 64(01), pp.138–152. doi:https://doi.org/10.37398/jsr.2020.640134.

Thakker, S.V., Parab, J. and Kaisare, S. (2020). Systematic research of e-learning platforms for solving challenges faced by Indian engineering students. Asian Association of Open Universities Journal, 16(1), pp.1–19. doi:https://doi.org/10.1108/aaouj-09-2020-0078.

Wang, X.-Y., Li, G., Malik, S. and Anwar, A. (2021). Impact of COVID-19 on achieving the goal of sustainable development: E-learning and educational productivity. *Economic Research-Ekonomska Istraživanja*, pp.1–17. doi:https://doi.org/10.1080/1331677x.2021.1927789.

Wang, Y. and Wang, Q. (2022). A Student Grouping Method for Massive Online Collaborative Learning. *International Journal of Emerging Technologies in Learning (iJET)*, [online] 17(3), pp.18–33. Available at: https://www.learntechlib.org/p/220520 [Accessed 12 Jun. 2023].

Zuhairi, A., Karthikeyan, N. and Priyadarshana, S.T. (2019). Supporting students to succeed in open and distance learning in the Open University of Sri Lanka and Universitas Terbuka Indonesia. *Asian Association of Open Universities Journal*, 15(1), pp.13–35. doi:https://doi.org/10.1108/aaouj-09-2019-0038.

APPENDIX:

Admin Management:

validate the user:

```
def validate_username(request):
    username = request.GET.get("username", None)
    data = {
        "is_taken": User.objects.filter(username__iexact = username).exists()
    }
    return JsonResponse (data)
def register(request):
    if request.method == 'POST':
        form = StudentAddForm(request.POST)
        if form.is_valid():
            form.save()
            messages.success(request, f'Account created successfuly.')
        else:
            messages.error(request, f'Somthing is not correct, please fill all fields correctly.')
    else:
        form = StudentAddForm(request.POST)
    return render(request, "registration/register.html", {'form': form})
@login_required
def profile(request):
    """ Show profile of any user that fire out the request """
    try:
        current_session = get_object_or_404(Session, is_current_session=True)
```

```
        current_semester = get_object_or_404(Semester, is_current_semester=True, session=current_session)
    except Semester.MultipleObjectsReturned and Semester.DoesNotExist and Session.DoesNotExist:
        raise Http404
    if request.user.is_lecturer:
        courses = Course.objects.filter(allocated_course__lecturer__pk=request.user.id).filter(
            semester=current_semester)
        return render(request, 'accounts/profile.html', {
            'title': request.user.get_full_name,
            "courses": courses,
            'current_session': current_session,
            'current_semester': current_semester,
        })
    elif request.user.is_student:
        level = Student.objects.get(student__pk=request.user.id)
        try:
            parent = Parent.objects.get(student=level)
        except:
            parent = "no parent set"
        courses = TakenCourse.objects.filter(student__student__id=request.user.id, course__level=level.level)
        context = {
            'title': request.user.get_full_name,
            'parent': parent,
```

```
            'courses': courses,
            'level': level,
            'current_session': current_session,
            'current_semester': current_semester,
        }
        return render(request, 'accounts/profile.html', context)
    else:
        staff = User.objects.filter(is_lecturer=True)
        return render(request, 'accounts/profile.html', {
            'title': request.user.get_full_name,
            "staff": staff,
            'current_session': current_session,
            'current_semester': current_semester,
        })
@login_required
@admin_required
def profile_single(request, id):
    """ Show profile of any selected user """
    if request.user.id == id:
        return redirect("/profile/")
    current_session = get_object_or_404(Session, is_current_session=True)
    current_semester = get_object_or_404(Semester, is_current_semester=True,
session=current_session)
    user = User.objects.get(pk=id)
    if user.is_lecturer:
```

```
        courses = Course.objects.filter(allocated_course__lecturer__pk=id).filter(semester=current_semester)
        context = {
            'title': user.get_full_name,
            "user": user,
            "user_type": "Lecturer",
            "courses": courses,
            'current_session': current_session,
            'current_semester': current_semester,
        }
        return render(request, 'accounts/profile_single.html', context)
    elif user.is_student:
        student = Student.objects.get(student__pk=id)
        courses = TakenCourse.objects.filter(student__student__id=id, course__level=student.level)
        context = {
            'title': user.get_full_name,
            'user': user,
            "user_type": "student",
            'courses': courses,
            'student': student,
            'current_session': current_session,
            'current_semester': current_semester,
        }
        return render(request, 'accounts/profile_single.html', context)
```

```
    else:
        context = {
            'title': user.get_full_name,
            "user": user,
            "user_type": "superuser",
            'current_session': current_session,
            'current_semester': current_semester,
        }
        return render(request, 'accounts/profile_single.html', context)
@login_required
@admin_required
def admin_panel(request):
    return render(request, 'setting/admin_panel.html', {})
# ############################################################
# ############################################################
# Setting views
# ############################################################
@login_required
def profile_update(request):
    if request.method == 'POST':
        form = ProfileUpdateForm(request.POST, request.FILES, instance=request.user)
        if form.is_valid():
            form.save()
            messages.success(request, 'Your profile has been updated successfully.')
```

```
            return redirect('profile')
        else:
            messages.error(request, 'Please correct the error(s) below.')
    else:
        form = ProfileUpdateForm(instance=request.user)
    return render(request, 'setting/profile_info_change.html', {
        'title': 'Setting | DjangoSMS',
        'form': form,
    })
@login_required
def change_password(request):
    if request.method == 'POST':
        form = PasswordChangeForm(request.user, request.POST)
        if form.is_valid():
            user = form.save()
            update_session_auth_hash(request, user)
            messages.success(request, 'Your password was successfully updated!')
            return redirect('profile')
        else:
            messages.error(request, 'Please correct the error(s) below. ')
    else:
        form = PasswordChangeForm(request.user)
    return render(request, 'setting/password_change.html', {
        'form': form,
```

```
    })
# ############################################################
```

Manage Teachers by Admin:

```
@login_required
@admin_required
def staff_add_view(request):
    if request.method == 'POST':
        form = StaffAddForm(request.POST)
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        if form.is_valid():
            form.save()
            messages.success(request, "Account for lecturer " + first_name + ' ' + last_name + "
has been created.")
            return redirect("lecturer_list")
    else:
        form = StaffAddForm()
    context = {
        'title': 'Lecturer Add | DjangoSMS',
        'form': form,
    }
    return render(request, 'accounts/add_staff.html', context)
@login_required
@admin_required
def edit_staff(request, pk):
```

```
    instance = get_object_or_404(User, is_lecturer=True, pk=pk)
    if request.method == 'POST':
        form = ProfileUpdateForm(request.POST, request.FILES, instance=instance)
        full_name = instance.get_full_name
        if form.is_valid():
            form.save()
            messages.success(request, 'Lecturer ' + full_name + ' has been updated.')
            return redirect('lecturer_list')
        else:
            messages.error(request, 'Please correct the error below.')
    else:
        form = ProfileUpdateForm(instance=instance)
    return render(request, 'accounts/edit_lecturer.html', {
        'title': 'Edit Lecturer | DjangoSMS',
        'form': form,
    })
@method_decorator([login_required, admin_required], name='dispatch')
class LecturerListView(ListView):
    queryset = User.objects.filter(is_lecturer=True)
    template_name = "accounts/lecturer_list.html"
    paginate_by = 10  # if pagination is desired
    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = "Lecturers | DjangoSMS"
```

```
        return context
# @login_required
# @lecturer_required
# def delete_staff(request, pk):
#     staff = get_object_or_404(User, pk=pk)
#     staff.delete()
#     return redirect('lecturer_list')
@login_required
@admin_required
def delete_staff(request, pk):
    lecturer = get_object_or_404(User, pk=pk)
    full_name = lecturer.get_full_name
    lecturer.delete()
    messages.success(request, 'Lecturer ' + full_name + ' has been deleted.')
    return redirect('lecturer_list')
# ########################################################
```

Manage student by Admin

```
# ########################################################
# Student views
# ########################################################
@login_required
@admin_required
def student_add_view(request):
    if request.method == 'POST':
        form = StudentAddForm(request.POST)
```

```
        first_name = request.POST.get('first_name')
        last_name = request.POST.get('last_name')
        if form.is_valid():
            form.save()
            messages.success(request, 'Account for ' + first_name + ' ' + last_name + ' has been created.')
            return redirect('student_list')
        else:
            messages.error(request, 'Correct the error(s) below.')
    else:
        form = StudentAddForm()

    return render(request, 'accounts/add_student.html', {
        'title': "Add Student | DjangoSMS",
        'form': form
    })
@login_required
@admin_required
def edit_student(request, pk):
    # instance = User.objects.get(pk=pk)
    instance = get_object_or_404(User, is_student=True, pk=pk)
    if request.method == 'POST':
        form = ProfileUpdateForm(request.POST, request.FILES, instance=instance)
        full_name = instance.get_full_name
        if form.is_valid():
```

```
            form.save()
            messages.success(request, ('Student ' + full_name + ' has been updated.'))
            return redirect('student_list')
        else:
            messages.error(request, 'Please correct the error below.')
    else:
        form = ProfileUpdateForm(instance=instance)
    return render(request, 'accounts/edit_student.html', {
        'title': 'Edit-profile | DjangoSMS',
        'form': form,
    })
@method_decorator([login_required, admin_required], name='dispatch')
class StudentListView(ListView):
    template_name = "accounts/student_list.html"
    paginate_by = 10  # if pagination is desired
    def get_queryset(self):
        queryset = Student.objects.all()
        query = self.request.GET.get('student_id')
        if query is not None:
            queryset = queryset.filter(Q(department=query))
        return queryset
    def get_context_data(self, **kwargs):
        context = super().get_context_data(**kwargs)
        context['title'] = "Students | DjangoSMS"
```

```
        return context
@login_required
@admin_required
def delete_student(request, pk):
    student = get_object_or_404(Student, pk=pk)
    # full_name = student.user.get_full_name
    student.delete()
    messages.success(request, 'Student has been deleted.')
    return redirect('student_list')
# ############################################################
class ParentAdd(CreateView):
    model = Parent
    form_class = ParentAddForm
    template_name = 'accounts/parent_form.html'
def parent_add(request):
    if request.method == 'POST':
        form = ParentAddForm(request.POST)
        if form.is_valid():
            form.save()
            return redirect('student_list')
    else:
        form = ParentAddForm(request.POST)
```

Manage All Course Work

```
import random

from django.contrib.auth.decorators import login_required, permission_required

from django.core.exceptions import PermissionDenied

from django.shortcuts import get_object_or_404, render, redirect

from django.utils.decorators import method_decorator

from django.views.generic import DetailView, ListView, TemplateView, FormView, CreateView, FormView, DeleteView, UpdateView

from django.contrib import messages

from django.urls import reverse_lazy

from django.db import transaction

from django.forms import inlineformset_factory

from django.http import HttpResponseRedirect

from accounts.decorators import student_required, lecturer_required

from .models import *

from .forms import *

@method_decorator([login_required, lecturer_required], name='dispatch')

class QuizCreateView(CreateView):

    model = Quiz

    form_class = QuizAddForm

    def get_context_data(self, *args, **kwargs):

        context = super(QuizCreateView, self).get_context_data(**kwargs)

        context['course'] = Course.objects.get(slug=self.kwargs['slug'])

        if self.request.POST:

            context['form'] = QuizAddForm(self.request.POST)
```

```
            # context['quiz'] = self.request.POST.get('quiz')
        else:
            context['form'] = QuizAddForm(initial={'course': Course.objects.get(slug=self.kwargs['slug'])})
        return context
    def form_valid(self, form, **kwargs):
        context = self.get_context_data()
        form = context['form']
        with transaction.atomic():
            self.object = form.save()
            if form.is_valid():
                form.instance = self.object
                form.save()
                return redirect('mc_create', slug=self.kwargs['slug'], quiz_id=form.instance.id)
        return super(QuizCreateView, self).form_invalid(form)
@method_decorator([login_required, lecturer_required], name='dispatch')
class QuizUpdateView(UpdateView):
    model = Quiz
    form_class = QuizAddForm
    def get_context_data(self, *args, **kwargs):
        context = super(QuizUpdateView, self).get_context_data(**kwargs)
        context['course'] = Course.objects.get(slug=self.kwargs['slug'])
        quiz = Quiz.objects.get(pk=self.kwargs['pk'])
        if self.request.POST:
            context['form'] = QuizAddForm(self.request.POST, instance=quiz)
```

```
        else:
            context['form'] = QuizAddForm(instance=quiz)
        return context
    def form_valid(self, form, **kwargs):
        context = self.get_context_data()
        course = context['course']
        form = context['form']
        with transaction.atomic():
            self.object = form.save()
            if form.is_valid():
                form.instance = self.object
                form.save()
                return redirect('quiz_index', course.slug)
        return super(QuizUpdateView, self).form_invalid(form)
@login_required
@lecturer_required
def quiz_delete(request, slug, pk):
    quiz = Quiz.objects.get(pk=pk)
    course = Course.objects.get(slug=slug)
    quiz.delete()
    messages.success(request, f'successfuly deleted.')
    return redirect('quiz_index', quiz.course.slug)
@method_decorator([login_required, lecturer_required], name='dispatch')
class MCQuestionCreate(CreateView):
```

```
    model = MCQuestion
    form_class = MCQuestionForm
    def get_context_data(self, **kwargs):
        context = super(MCQuestionCreate, self).get_context_data(**kwargs)
        context['course'] = Course.objects.get(slug=self.kwargs['slug'])
        context['quiz_obj'] = Quiz.objects.get(id=self.kwargs['quiz_id'])
        context['quizQuestions'] = Question.objects.filter(quiz=self.kwargs['quiz_id']).count()
        if self.request.POST:
            context['form'] = MCQuestionForm(self.request.POST)
            context['formset'] = MCQuestionFormSet(self.request.POST)
        else:
            context['form'] = MCQuestionForm(initial={'quiz': self.kwargs['quiz_id']})
            context['formset'] = MCQuestionFormSet()
        return context
    def form_valid(self, form):
        context = self.get_context_data()
        formset = context['formset']
        course = context['course']
        with transaction.atomic():
            form.instance.question = self.request.POST.get('content')
            self.object = form.save()
            if formset.is_valid():
                formset.instance = self.object
                formset.save()
```

```
            if "another" in self.request.POST:
                return redirect('mc_create', slug=self.kwargs['slug'],
quiz_id=self.kwargs['quiz_id'])
            return redirect('quiz_index', course.slug)
        return super(MCQuestionCreate, self).form_invalid(form)
@login_required
def quiz_list(request, slug):
    quizzes = Quiz.objects.filter(course__slug = slug).order_by('-timestamp')
    course = Course.objects.get(slug = slug)
    return render(request, 'quiz/quiz_list.html', {'quizzes': quizzes, 'course': course})
    # return render(request, 'quiz/quiz_list.html', {'quizzes': quizzes})
@method_decorator([login_required, lecturer_required], name='dispatch')
class QuizMarkerMixin(object):
    @method_decorator(login_required)
    # @method_decorator(permission_required('quiz.view_sittings'))
    def dispatch(self, *args, **kwargs):
        return super(QuizMarkerMixin, self).dispatch(*args, **kwargs)
# @method_decorator([login_required, lecturer_required], name='get_queryset')
class SittingFilterTitleMixin(object):
    def get_queryset(self):
        queryset = super(SittingFilterTitleMixin, self).get_queryset()
        quiz_filter = self.request.GET.get('quiz_filter')
        if quiz_filter:
            queryset = queryset.filter(quiz__title__icontains=quiz_filter)
        return queryset
```

```
@method_decorator([login_required], name='dispatch')
class QuizUserProgressView(TemplateView):
    template_name = 'progress.html'
    def dispatch(self, request, *args, **kwargs):
        return super(QuizUserProgressView, self).dispatch(request, *args, **kwargs)
    def get_context_data(self, **kwargs):
        context = super(QuizUserProgressView, self).get_context_data(**kwargs)
        progress, c = Progress.objects.get_or_create(user=self.request.user)
        context['cat_scores'] = progress.list_all_cat_scores
        context['exams'] = progress.show_exams()
        context['exams_counter'] = progress.show_exams().count()
        return context
from result.models import TakenCourse
@method_decorator([login_required, lecturer_required], name='dispatch')
class QuizMarkingList(QuizMarkerMixin, SittingFilterTitleMixin, ListView):
    model = Sitting
    # def get_context_data(self, **kwargs):
    #     context = super(QuizMarkingList, self).get_context_data(**kwargs)
    #         context['queryset_counter'] = super(QuizMarkingList, self).get_queryset().filter(complete=True).filter(course__allocated_course__lecturer__pk=self.request.user.id).count()
    #         context['marking_list'] = super(QuizMarkingList, self).get_queryset().filter(complete=True).filter(course__allocated_course__lecturer__pk=self.request.user.id)
    #     return context
```

```
    def get_queryset(self):
        if self.request.user.is_superuser:
            queryset = super(QuizMarkingList, self).get_queryset().filter(complete=True)
        else:
            queryset = super(QuizMarkingList, self).get_queryset().filter(quiz__course__allocated_course__lecturer__pk=self.request.user.id).filter(complete=True)
        # search by user
        user_filter = self.request.GET.get('user_filter')
        if user_filter:
            queryset = queryset.filter(user__username__icontains=user_filter)
            return queryset
@method_decorator([login_required, lecturer_required], name='dispatch')
class QuizMarkingDetail(QuizMarkerMixin, DetailView):
    model = Sitting
    def post(self, request, *args, **kwargs):
        sitting = self.get_object()
        q_to_toggle = request.POST.get('qid', None)
        if q_to_toggle:
            q = Question.objects.get_subclass(id=int(q_to_toggle))
            if int(q_to_toggle) in sitting.get_incorrect_questions:
                sitting.remove_incorrect_question(q)
            else:
                sitting.add_incorrect_question(q)
        return self.get(request)
```

```
    def get_context_data(self, **kwargs):
        context = super(QuizMarkingDetail, self).get_context_data(**kwargs)
        context['questions'] = context['sitting'].get_questions(with_answers=True)
        return context
# @method_decorator([login_required, student_required], name='dispatch')
@method_decorator([login_required], name='dispatch')
class QuizTake(FormView):
    form_class = QuestionForm
    template_name = 'question.html'
    result_template_name = 'result.html'
    # single_complete_template_name = 'single_complete.html'
    def dispatch(self, request, *args, **kwargs):
        self.quiz = get_object_or_404(Quiz, slug=self.kwargs['slug'])
        self.course = get_object_or_404(Course, pk=self.kwargs['pk'])
        quizQuestions = Question.objects.filter(quiz=self.quiz).count()
        course = get_object_or_404(Course, pk=self.kwargs['pk'])
        if quizQuestions <= 0:
            messages.warning(request, f'Question set of the quiz is empty. try later!')
            return redirect('quiz_index', self.course.slug)
        if self.quiz.draft and not request.user.has_perm('quiz.change_quiz'):
            raise PermissionDenied
        self.sitting = Sitting.objects.user_sitting(request.user, self.quiz, self.course)
        if self.sitting is False:
            # return render(request, self.single_complete_template_name)
```

```
            messages.info(request, f'You have already sat this exam and only one sitting is permitted')
            return redirect('quiz_index', self.course.slug)
        return super(QuizTake, self).dispatch(request, *args, **kwargs)
    def get_form(self, *args, **kwargs):
        self.question = self.sitting.get_first_question()
        self.progress = self.sitting.progress()


        if self.question.__class__ is Essay_Question:
            form_class = EssayForm
        else:
            form_class = self.form_class
        return form_class(**self.get_form_kwargs())
    def get_form_kwargs(self):
        kwargs = super(QuizTake, self).get_form_kwargs()
        return dict(kwargs, question=self.question)
    def form_valid(self, form):
        self.form_valid_user(form)
        if self.sitting.get_first_question() is False:
            return self.final_result_user()
        self.request.POST = {}
        return super(QuizTake, self).get(self, self.request)
    def get_context_data(self, **kwargs):
        context = super(QuizTake, self).get_context_data(**kwargs)
        context['question'] = self.question
```

```
        context['quiz'] = self.quiz
        context['course'] = get_object_or_404(Course, pk=self.kwargs['pk'])
        if hasattr(self, 'previous'):
            context['previous'] = self.previous
        if hasattr(self, 'progress'):
            context['progress'] = self.progress
        return context
    def form_valid_user(self, form):
        progress, c = Progress.objects.get_or_create(user=self.request.user)
        guess = form.cleaned_data['answers']
        is_correct = self.question.check_if_correct(guess)
        if is_correct is True:
            self.sitting.add_to_score(1)
            progress.update_score(self.question, 1, 1)
        else:
            self.sitting.add_incorrect_question(self.question)
            progress.update_score(self.question, 0, 1)
        if self.quiz.answers_at_end is not True:
            self.previous = {
                'previous_answer': guess,
                'previous_outcome': is_correct,
                'previous_question': self.question,
                'answers': self.question.get_choices(),
                'question_type': {self.question.__class__.__name__: True}
```

```
            }
        else:
            self.previous = {}
        self.sitting.add_user_answer(self.question, guess)
        self.sitting.remove_first_question()
    def final_result_user(self):
        results = {
            'course': get_object_or_404(Course, pk=self.kwargs['pk']),
            'quiz': self.quiz,
            'score': self.sitting.get_current_score,
            'max_score': self.sitting.get_max_score,
            'percent': self.sitting.get_percent_correct,
            'sitting': self.sitting,
            'previous': self.previous,
            'course': get_object_or_404(Course, pk=self.kwargs['pk'])
        }
        self.sitting.mark_quiz_complete()
        if self.quiz.answers_at_end:
            results['questions'] = self.sitting.get_questions(with_answers=True)
            results['incorrect_questions'] = self.sitting.get_incorrect_questions
        if self.quiz.exam_paper is False or self.request.user.is_superuser or
self.request.user.is_lecturer :
            self.sitting.delete()
        return render(self.request, self.result_template_name, results)
```

Chat Bot

```
{% load static %}

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <meta http-equiv="X-UA-Compatible" content="ie=edge">

    <title>{% block title %}Django LMS{% endblock title %}</title>

    <link rel="stylesheet" type="text/css" href="{% static 'css/bootstrap.min.css' %}">

    <link rel="stylesheet" type="text/css" href="{% static 'css/all.css' %}">

    <link rel="stylesheet" type="text/css" href="{% static 'css/style.css' %}">

    {% block header %}{% endblock %}

</head>

<body>

    {% include 'aside.html' %}

    <div id="main">

        {% include 'navbar.html' %}

        <div class="container-fluid" id="main-content">

            {% block content %}{% endblock content %}

        </div>  </div>

    <script type="text/javascript" src="{% static 'js/jquery-3.3.1.min.js' %}"></script>

    <script type="text/javascript" src="{% static 'js/popper.min.js' %}"></script>

    <script type="text/javascript" src="{% static 'js/bootstrap.min.js' %}"></script>
```

```
    <script type="text/javascript" src="{% static 'js/main.js' %}"></script>
    <script>
        $(document).ready(function () {
            $('.au-input').focus(function () {
                $('#top-navbar').attr('class', 'dim');
                $('#side-nav').css('pointer-events', 'none');
                $('#main-content').css('pointer-events', 'none');
            });
            $('.au-input').focusout(function () {
                $('#top-navbar').removeAttr('class');
                $('#side-nav').css('pointer-events', 'auto');
                $('#main-content').css('pointer-events', 'auto');
            });   })</script>
<script  type="text/javascript">
!function(t,e){t.artibotApi={l:[],t:[],on:function(){this.l.push(arguments)},trigger:function()
{this.t.push(arguments)}};
Var a=!1,i=e.createElement("script");
i.async=!0,i.type="text/javascript",i.src="https://app.artibot.ai/loader.js",e.getElementsByTag
Name("head").item(0).appendChild(i),i.onreadystatechange=i.onload=function()
{if(!(a||this.readyState&&"loaded"!=this.readyState&&"complete"!=this.readyState))
{new window.ArtiBot({i:"a9dfe1ea-a68c-4d77-973e-5e645854b12c"});
a=!0}  }   }(window,document);
</script>
    {% block js %}{% endblock js %}
</body></html>
```

Manage Results of the coursework

```
from django.shortcuts import render, redirect, get_object_or_404

from django.contrib import messages

from django.http import HttpResponseRedirect

from django.urls import reverse_lazy

from django.conf import settings

from django.contrib.auth.decorators import login_required

from django.core.paginator import Paginator

from accounts.models import User, Student

from app.models import Session, Semester

from course.models import Course

from accounts.decorators import lecturer_required, student_required

from .models import TakenCourse, Result

import json

User = settings.AUTH_USER_MODEL

from django.core.files.storage import FileSystemStorage

from django.http import HttpResponse, JsonResponse

from reportlab.platypus import SimpleDocTemplate, Paragraph, Spacer, Table, TableStyle, Image, LongTable

from reportlab.lib.styles import getSampleStyleSheet, black, ParagraphStyle

from reportlab.lib.enums import TA_JUSTIFY,TA_LEFT,TA_CENTER,TA_RIGHT

from reportlab.platypus.tables import Table

from reportlab.lib.units import inch

from reportlab.lib import colors

cm = 2.54
```

```
from LMS.settings import MEDIA_ROOT, BASE_DIR, STATIC_URL
import os
from .models import *
# ##############################################################
# Score Add & Add for
# ##############################################################
@login_required
@lecturer_required
def add_score(request):
    """
    Shows a page where a lecturer will select a course allocated to him for score entry.
    in a specific semester and session
    """
    current_session = Session.objects.get(is_current_session=True)
    current_semester = get_object_or_404(Semester, is_current_semester=True, session=current_session)
    # semester = Course.objects.filter(allocated_course__lecturer__pk=request.user.id, semester=current_semester)
    courses = Course.objects.filter(allocated_course__lecturer__pk=request.user.id).filter(semester=current_semester)
    context = {
        "current_session": current_session,
        "current_semester": current_semester,
        "courses": courses,
```

```
    }
    return render(request, 'result/add_score.html', context)
@login_required
@lecturer_required
def add_score_for(request, id):
    """
    Shows a page where a lecturer will add score for students that are taking courses allocated to him
    in a specific semester and session
    """
    current_session = Session.objects.get(is_current_session=True)
    current_semester = get_object_or_404(Semester, is_current_semester=True, session=current_session)
    if request.method == 'GET':
        courses = Course.objects.filter(allocated_course__lecturer__pk=request.user.id).filter(
            semester=current_semester)
        course = Course.objects.get(pk=id)
        students = TakenCourse.objects.filter(course__allocated_course__lecturer__pk=request.user.id).filter(
            course__id=id).filter(course__semester=current_semester)
        context = {
            "title": "Submit Score | DjangoSMS",
            "courses": courses,
            "course": course,
            # "myclass": myclass,
```

```
            "students": students,
            "current_session": current_session,
            "current_semester": current_semester,
        }
        return render(request, 'result/add_score_for.html', context)
    if request.method == 'POST':
        ids = ()
        data = request.POST.copy()
        data.pop('csrfmiddlewaretoken', None)  # remove csrf_token
        for key in data.keys():
            ids = ids + (str(key),)  # gather all the all students id (i.e the keys) in a tuple
        for s in range(0, len(ids)):  # iterate over the list of student ids gathered above
            student = TakenCourse.objects.get(id=ids[s])
            # print(student)
            # print(student.student)
            # print(student.student.department.id)
            courses = Course.objects.filter(level=student.student.level).filter(program__pk=student.student.department.id).filter(
                semester=current_semester)  # all courses of a specific level in current semester
            total_credit_in_semester = 0
            for i in courses:
                if i == courses.count():
                    break
                else:
```

```
            total_credit_in_semester += int(i.credit)
        score = data.getlist(ids[s])  # get list of score for current student in the loop
        assignment = score[0]  # subscript the list to get the fisrt value > ca score
        mid_exam = score[1]  # do the same for exam score
        quiz = score[2]
        attendance = score[3]
        final_exam = score[4]
        obj = TakenCourse.objects.get(pk=ids[s])  # get the current student data
        obj.assignment = assignment  # set current student assignment score
        obj.mid_exam = mid_exam  # set current student mid_exam score
        obj.quiz = quiz  # set current student quiz score
        obj.attendance = attendance  # set current student attendance score
        obj.final_exam = final_exam  # set current student final_exam score
        obj.total = obj.get_total(assignment=assignment, mid_exam=mid_exam, quiz=quiz,
attendance=attendance, final_exam=final_exam)
        obj.grade = obj.get_grade(total=obj.total)
        obj.point = obj.get_point(grade=obj.grade)
        obj.comment = obj.get_comment(grade=obj.grade)
        obj.save()
        gpa = obj.calculate_gpa(total_credit_in_semester)
        cgpa = obj.calculate_cgpa()
        try:
            a = Result.objects.get(student=student.student, semester=current_semester,
session=current_session, level=student.student.level)
            a.gpa = gpa
```

```
            a.cgpa = cgpa
            a.save()
        except:
            Result.objects.get_or_create(student=student.student, gpa=gpa,
semester=current_semester,
                                         session=current_session, level=student.student.level)
        messages.success(request, 'Successfully Recorded! ')
        return HttpResponseRedirect(reverse_lazy('add_score_for', kwargs={'id': id}))
    return HttpResponseRedirect(reverse_lazy('add_score_for', kwargs={'id': id}))
# ########################################################
@login_required
@student_required
def grade_result(request):
    student = Student.objects.get(student__pk=request.user.id)
    courses =
TakenCourse.objects.filter(student__student__pk=request.user.id).filter(course__level=studen
t.level)
    results = Result.objects.filter(student__student__pk=request.user.id)
    result_set = set()
    for result in results:
        result_set.add(result.session)
    sorted_result = sorted(result_set)
    total_first_semester_credit = 0
    total_sec_semester_credit = 0
    for i in courses:
```

```
        if i.course.semester == "First":
            total_first_semester_credit += int(i.course.credit)
        if i.course.semester == "Second":
            total_sec_semester_credit += int(i.course.credit)
    previousCGPA = 0
    for i in results:
        previousLEVEL = i.level
        try:
            a = Result.objects.get(student__student__pk=request.user.id, level=previousLEVEL,
semester="Second")
            previousCGPA = a.cgpa
            break
        except:
            previousCGPA = 0
    # Prepare data for the pie chart
    grade_distribution = {
        'A+': 0,
        'A': 0,
        'B': 0,
        'C': 0,
        'D': 0,
        'F': 0,
    }
    for course in courses:
        grade_distribution[course.grade] += 1
```

```
    grade_labels = json.dumps(list(grade_distribution.keys()))
    grade_data = json.dumps(list(grade_distribution.values()))
    context = {
        "courses": courses,
        "results": results,
        "sorted_result": sorted_result,
        "student": student,
        'total_first_semester_credit': total_first_semester_credit,
        'total_sec_semester_credit': total_sec_semester_credit,
        'total_first_and_second_semester_credit': total_first_semester_credit + total_sec_semester_credit,
        "previousCGPA": previousCGPA,
        "grade_labels": grade_labels,
        "grade_data": grade_data,
    }
    return render(request, 'result/grade_results.html', context)
@login_required
@student_required
def assessment_result(request):
    student = Student.objects.get(student__pk=request.user.id)
    courses = TakenCourse.objects.filter(student__student__pk=request.user.id, course__level=student.level)
    result = Result.objects.filter(student__student__pk=request.user.id)
    context = {
        "courses": courses,
```

```
        "result": result,
        "student": student,
    }
        return render(request, 'result/assessment_results.html', context)
@login_required
@lecturer_required
def result_sheet_pdf_view(request, id):
    current_semester = Semester.objects.get(is_current_semester=True)
    current_session = Session.objects.get(is_current_session=True)
    result = TakenCourse.objects.filter(course__pk=id)
    course = get_object_or_404(Course, id=id)
    no_of_pass = TakenCourse.objects.filter(course__pk=id, comment="PASS").count()
    no_of_fail = TakenCourse.objects.filter(course__pk=id, comment="FAIL").count()
    fname = str(current_semester) + '_semester_' + str(current_session) + '_' + str(course) +
'_resultSheet.pdf'
    fname = fname.replace("/", "-")
    flocation = settings.MEDIA_ROOT + "/result_sheet/" + fname
    doc = SimpleDocTemplate(flocation, rightMargin=0, leftMargin=6.5 * cm, topMargin=0.3
* cm, bottomMargin=0)
    styles = getSampleStyleSheet()
    styles.add(ParagraphStyle( name="ParagraphTitle", fontSize=11,
fontName="FreeSansBold"))
    Story = [Spacer(1,.2)]
    style = styles["Normal"]
    logo = settings.MEDIA_ROOT + "/logo/you-logo-here.png"
```

```
    im = Image(logo, 1*inch, 1*inch)
    im.__setattr__("_offs_x", -200)
    im.__setattr__("_offs_y", -45)
    Story.append(im)
    style = getSampleStyleSheet()
    normal = style["Normal"]
    normal.alignment = TA_CENTER
    normal.fontName = "Helvetica"
    normal.fontSize = 12
    normal.leading = 15
    title = "<b> "+str(current_semester) + " Semester " + str(current_session) + " Result Sheet</b>"
    title = Paragraph(title.upper(), normal)
    Story.append(title)
    Story.append(Spacer(1,0.1*inch))
    style = getSampleStyleSheet()
    normal = style["Normal"]
    normal.alignment = TA_CENTER
    normal.fontName = "Helvetica"
    normal.fontSize = 10
    normal.leading = 15
    title = "<b>Course lecturer: " + request.user.get_full_name + "</b>"
    title = Paragraph(title.upper(), normal)
    Story.append(title)
    Story.append(Spacer(1,0.1*inch))
```

```
normal = style["Normal"]
normal.alignment = TA_CENTER
normal.fontName = "Helvetica"
normal.fontSize = 10
normal.leading = 15
level = result.filter(course_id=id).first()
title = "<b>Level: </b>" + str(level.course.level)
title = Paragraph(title.upper(), normal)
Story.append(title)
Story.append(Spacer(1,.6*inch))
elements = []
count = 0
header = [('S/N', 'ID NO.', 'FULL NAME', 'TOTAL', 'GRADE', 'POINT', 'COMMENT')]
table_header = Table(header, [inch], [0.5*inch])
table_header.setStyle(
   TableStyle([
      ('BACKGROUND',(0,0),(-1,-1),colors.black),
      ('TEXTCOLOR',(1,0),(-1,-1),colors.white),
      ('TEXTCOLOR',(0,0),(0,0),colors.cyan),
      ('ALIGN',(0,0),(-1,-1),'CENTER'),
      ('VALIGN',(0,0),(-1,-1),'MIDDLE'),
      ('BOX',(0,0),(-1,-1),1,colors.black),
      ]))
Story.append(table_header)
```

```
    for student in result:
        data=[(count+1,student.student.student.username.upper(),
Paragraph(student.student.student.get_full_name.capitalize(), styles['Normal']),
        student.total, student.grade, student.point, student.comment)]
        color = colors.black
        if student.grade == 'F':
            color = colors.red
        count += 1
        t_body = Table(data, colWidths=[inch])
        t_body.setStyle(
            TableStyle([
                ('INNERGRID', (0,0), (-1,-1), 0.05, colors.black),
                ('BOX', (0,0), (-1,-1), 0.1, colors.black),
                ]))
        Story.append(t_body)
    Story.append(Spacer(1,1*inch))
    style_right = ParagraphStyle(name='right', parent=styles['Normal'],
alignment=TA_RIGHT)
    tbl_data = [
    [Paragraph("<b>Date:</b>_____", styles["Normal"]),
Paragraph("<b>No. of PASS:</b> " + str(no_of_pass), style_right)],
    [Paragraph("<b>Siganture / Stamp:</b> _____",
styles["Normal"]), Paragraph("<b>No. of FAIL: </b>" + str(no_of_fail), style_right)]]
    tbl = Table(tbl_data)
    Story.append(tbl)
```

```
    doc.build(Story)
    fs = FileSystemStorage(settings.MEDIA_ROOT + "/result_sheet")
    with fs.open(fname) as pdf:
        response = HttpResponse(pdf, content_type='application/pdf')
        response['Content-Disposition'] = 'inline; filename=' + fname + "
        return response
    return response
@login_required
@student_required
def course_registration_form(request):
    current_semester = Semester.objects.get(is_current_semester=True)
    current_session = Session.objects.get(is_current_session=True)
    courses = TakenCourse.objects.filter(student__student__id=request.user.id)
    fname = request.user.username + '.pdf'
    fname = fname.replace("/", "-")
    # flocation = '/tmp/' + fname
    # print(MEDIA_ROOT + "\\" + fname)
    flocation = settings.MEDIA_ROOT + "/registration_form/" + fname
    doc = SimpleDocTemplate(flocation, rightMargin=15, leftMargin=15, topMargin=0,
bottomMargin=0)
    styles = getSampleStyleSheet()
    Story = [Spacer(1,0.5)]
    Story.append(Spacer(1,0.4*inch))
    style = styles["Normal"]
    style = getSampleStyleSheet()
```

```
normal = style["Normal"]
normal.alignment = TA_CENTER
normal.fontName = "Helvetica"
normal.fontSize = 12
normal.leading = 18
title = "<b>EZOD UNIVERSITY OF TECHNOLOGY, ADAMA</b>"
title = Paragraph(title.upper(), normal)
Story.append(title)
style = getSampleStyleSheet()
    school = style["Normal"]
school.alignment = TA_CENTER
school.fontName = "Helvetica"
school.fontSize = 10
school.leading = 18
school_title = "<b>SCHOOL OF ELECTRICAL ENGINEERING & COMPUTING</b>"
school_title = Paragraph(school_title.upper(), school)
Story.append(school_title)
style = getSampleStyleSheet()
Story.append(Spacer(1,0.1*inch))
department = style["Normal"]
department.alignment = TA_CENTER
department.fontName = "Helvetica"
department.fontSize = 9
department.leading = 18
```

```
    department_title = "<b>DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING</b>"
    department_title = Paragraph(department_title, department)
    Story.append(department_title)
    Story.append(Spacer(1,.3*inch))
        title = "<b><u>STUDENT COURSE REGISTRATION FORM</u></b>"
    title = Paragraph(title.upper(), normal)
    Story.append(title)
    student = Student.objects.get(student__pk=request.user.id)
    style_right = ParagraphStyle(name='right', parent=styles['Normal'])
    tbl_data = [
        [Paragraph("<b>Registration Number : " + request.user.username.upper() + "</b>", styles["Normal"])],
        [Paragraph("<b>Name : " + request.user.get_full_name.upper() + "</b>", styles["Normal"])],
        [Paragraph("<b>Session : " + current_session.session.upper() + "</b>", styles["Normal"]), Paragraph("<b>Level: " + student.level + "</b>", styles["Normal"])
        ]]
    tbl = Table(tbl_data)
    Story.append(tbl)
    Story.append(Spacer(1, 0.6*inch))
    style = getSampleStyleSheet()
    semester = style["Normal"]
    semester.alignment = TA_LEFT
    semester.fontName = "Helvetica"
```

```
    semester.fontSize = 9
    semester.leading = 18
    semester_title = "<b>FIRST SEMESTER</b>"
    semester_title = Paragraph(semester_title, semester)
    Story.append(semester_title)
    elements = []
    # FIRST SEMESTER
    count = 0
    header = [('S/No', 'Course Code', 'Course Title', 'Unit', Paragraph('Name, Siganture of course lecturer & Date', style['Normal']))]
    table_header = Table(header,1*[1.4*inch], 1*[0.5*inch])
    table_header.setStyle(
      TableStyle([
            ('ALIGN',(-2,-2), (-2,-2),'CENTER'),
            ('VALIGN',(-2,-2), (-2,-2),'MIDDLE'),
            ('ALIGN',(1,0), (1,0),'CENTER'),
            ('VALIGN',(1,0), (1,0),'MIDDLE'),
            ('ALIGN',(0,0), (0,0),'CENTER'),
            ('VALIGN',(0,0), (0,0),'MIDDLE'),
            ('ALIGN',(-4,0), (-4,0),'LEFT'),
            ('VALIGN',(-4,0), (-4,0),'MIDDLE'),
            ('ALIGN',(-3,0), (-3,0),'LEFT'),
            ('VALIGN',(-3,0), (-3,0),'MIDDLE'),
            ('TEXTCOLOR',(0,-1),(-1,-1),colors.black),
            ('INNERGRID', (0,0), (-1,-1), 0.25, colors.black),
```

```
            ('BOX', (0,0), (-1,-1), 0.25, colors.black),
        ]))
    Story.append(table_header)
    first_semester_unit = 0
    for course in courses:
        if course.course.semester == FIRST:
            first_semester_unit += int(course.course.credit)
            data = [(count+1, course.course.code.upper(), Paragraph(course.course.title, style['Normal']), course.course.credit, '')]
            color = colors.black
            count += 1
            table_body=Table(data,1*[1.4*inch], 1*[0.3*inch])
            table_body.setStyle(
                TableStyle([
                    ('ALIGN',(-2,-2), (-2,-2),'CENTER'),
                    ('ALIGN',(1,0), (1,0),'CENTER'),
                    ('ALIGN',(0,0), (0,0),'CENTER'),
                    ('ALIGN',(-4,0), (-4,0),'LEFT'),
                    ('TEXTCOLOR',(0,-1),(-1,-1),colors.black),
                    ('INNERGRID', (0,0), (-1,-1), 0.25, colors.black),
                    ('BOX', (0,0), (-1,-1), 0.25, colors.black),
                    ]))
            Story.append(table_body)

    style = getSampleStyleSheet()
```

```
    semester = style["Normal"]
    semester.alignment = TA_LEFT
    semester.fontName = "Helvetica"
    semester.fontSize = 8
    semester.leading = 18
    semester_title = "<b>Total Second First Credit : " + str(first_semester_unit) + "</b>"
    semester_title = Paragraph(semester_title, semester)
    Story.append(semester_title)
    # FIRST SEMESTER ENDS HERE
    Story.append(Spacer(1, 0.6*inch))
    style = getSampleStyleSheet()
    semester = style["Normal"]
    semester.alignment = TA_LEFT
    semester.fontName = "Helvetica"
    semester.fontSize = 9
    semester.leading = 18
    semester_title = "<b>SECOND SEMESTER</b>"
    semester_title = Paragraph(semester_title, semester)
    Story.append(semester_title)
    # SECOND SEMESTER
    count = 0
    header = [('S/No', 'Course Code', 'Course Title', 'Unit', Paragraph('<b>Name, Signature of course lecturer & Date</b>', style['Normal']))]
    table_header = Table(header,1*[1.4*inch], 1*[0.5*inch])
    table_header.setStyle(
```

```
        TableStyle([
            ('ALIGN',(-2,-2), (-2,-2),'CENTER'),
            ('VALIGN',(-2,-2), (-2,-2),'MIDDLE'),
            ('ALIGN',(1,0), (1,0),'CENTER'),
            ('VALIGN',(1,0), (1,0),'MIDDLE'),
            ('ALIGN',(0,0), (0,0),'CENTER'),
            ('VALIGN',(0,0), (0,0),'MIDDLE'),
            ('ALIGN',(-4,0), (-4,0),'LEFT'),
            ('VALIGN',(-4,0), (-4,0),'MIDDLE'),
            ('ALIGN',(-3,0), (-3,0),'LEFT'),
            ('VALIGN',(-3,0), (-3,0),'MIDDLE'),
            ('TEXTCOLOR',(0,-1),(-1,-1),colors.black),
            ('INNERGRID', (0,0), (-1,-1), 0.25, colors.black),
            ('BOX', (0,0), (-1,-1), 0.25, colors.black),
        ]))
    Story.append(table_header)


    second_semester_unit = 0
    for course in courses:
        if course.course.semester == SECOND:
            second_semester_unit += int(course.course.credit)
            data = [(count+1, course.course.code.upper(), Paragraph(course.course.title,
style['Normal']), course.course.credit, '')]
            color = colors.black
            count += 1
```

```
            table_body=Table(data,1*[1.4*inch], 1*[0.3*inch])
            table_body.setStyle(
               TableStyle([
                  ('ALIGN',(-2,-2), (-2,-2),'CENTER'),
                  ('ALIGN',(1,0), (1,0),'CENTER'),
                  ('ALIGN',(0,0), (0,0),'CENTER'),
                  ('ALIGN',(-4,0), (-4,0),'LEFT'),
                  ('TEXTCOLOR',(0,-1),(-1,-1),colors.black),
                  ('INNERGRID', (0,0), (-1,-1), 0.25, colors.black),
                  ('BOX', (0,0), (-1,-1), 0.25, colors.black),
                  ]))
            Story.append(table_body)
    style = getSampleStyleSheet()
    semester = style["Normal"]
    semester.alignment = TA_LEFT
    semester.fontName = "Helvetica"
    semester.fontSize = 8
    semester.leading = 18
    semester_title = "<b>Total Second Semester Credit : " + str(second_semester_unit) +
"</b>"
    semester_title = Paragraph(semester_title, semester)
    Story.append(semester_title)
    Story.append(Spacer(1, 2))
    style = getSampleStyleSheet()
    certification = style["Normal"]
```

```
certification.alignment = TA_JUSTIFY

certification.fontName = "Helvetica"

certification.fontSize = 8

certification.leading = 18

student = Student.objects.get(student__pk=request.user.id)

certification_text = "CERTIFICATION OF REGISTRATION: I certify that <b>" + str(request.user.get_full_name.upper()) + "</b>\

has been duly registered for the <b>" + student.level + " level </b> of study in the department\

of COMPUTER SICENCE & ENGINEERING and that the courses and credits registered are as approved by the senate of the University"

certification_text = Paragraph(certification_text, certification)

Story.append(certification_text)

# FIRST SEMESTER ENDS HERE

logo = MEDIA_ROOT + "/logo/you-logo-here.png"

im_logo = Image(logo, 1*inch, 1*inch)

im_logo.__setattr__("_offs_x", -218)

im_logo.__setattr__("_offs_y", 480)

Story.append(im_logo)

picture =  BASE_DIR + request.user.get_picture()

im = Image(picture, 1.0*inch, 1.0*inch)

im.__setattr__("_offs_x", 218)

im.__setattr__("_offs_y", 550)

Story.append(im)

doc.build(Story)
```

```
    fs = FileSystemStorage(settings.MEDIA_ROOT + "/registration_form")
    with fs.open(fname) as pdf:
        response = HttpResponse(pdf, content_type='application/pdf')
        response['Content-Disposition'] = 'inline; filename='+fname+"
        return response
    return response
```