**Online Learning Platform with Grading and Certification**

Project submitted to the

SRM University – AP, Andhra Pradesh

Submitted in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology**

**in**

**Computer Science and Engineering**

**School of Engineering and Sciences**

Submitted By

**Pardha Saradhi Maturi**

**(AP23110010056)**

**Rama Chandra Murthy Mamidipalli**

**(AP23110010015)**

Under the guidance of

**MS.Poonam Yadav**

****

**Department of Computer Science and Engineering SRM University,AP**

**Neerukonda, Mangalagiri, Guntur**

**Andhra Pradesh – 522 240**

**[Month, Year]**

**Department of Computer Science and Engineering** SRM University,AP



**CERTIFICATE**

This is to certify that the Project report entitled **“Online Learning Platform with Grading and Certification”** is being submitted by **Pardha Saradhi Maturi(AP23110010056), Rama Chandra Murthy Mamidipalli(AP23110010015)** a student of Department of Computer Science and Engineering, SRM University,AP, in partial fulfillment of the requirement for the degree of **“B.Tech(CSE)”** carried out by his during the academic year 2024-2025.

Signature of the Supervisor Signature of Head of the Dept.

**Acknowledgement**

The satisfaction that accompanies the successful completion of any task would be incomplete without introducing the people who made it possible and whose constant guidance and encouragement crowns all efforts with success.

I am extremely grateful and express my profound gratitude and indebtedness to my project guide, **Ms. Poonam Yadav**, Department of Computer Science & Engineering, SRM University,Andhra pradesh, for her kind help and for giving me the necessary guidance and valuable suggestions in completing this project work.

**Pardha Saradhi Maturi**

**(AP23110010056)**

**Rama Chandra Murthy Mamidipalli**

**(AP23110010015)**

|  |  |  |
| --- | --- | --- |
| S.no | Contents |  |
| 1 | Acknowledgement |  |
| 2 | Abstract | 5 |
| 3 | Statement of contributions | 5 |
| 4 | List of Tables | 6-8 |
| 5 | Introduction | 9 |
| 6 | Methodology | 10-11 |
| 7 | Discussions | 12-13 |
| 8 | Concluding Remarks | 14 |
| 9 | Future Work | 15 |
| 10 | References | 16 |

**Table Of ContentsAbstract**

With features for administrators and students alike, this project seeks to develop a safe and effective course administration system. By providing their name, roll number, and a special username and password, students can register. Following registration, students examine course materials, access registered courses, and take multiple-choice questions (MCQs) by logging in with their username and password. Administrators can add new courses, update course content, and generate multiple-choice questions (MCQs) for each course using the admin module, which is password-protected by an ID. The interactive multiple-choice question (MCQ) evaluation offered by the system assigns a 50% passing grade to pupils based on their accurate answers. To guarantee modularity and ease of future development, the project makes use of object-oriented programming concepts through classes like Course, Student, and User Manager.

**Statement of Contributions**

Paper I: Creating and Assessing an Interactive Learning Management System with Automated Multiple-Choice Questions

Duties: I was in charge of coming up with the main concept for creating a safe and user-friendly course administration system. I created the system's overall architecture, which included using the concepts of object-oriented programming. My contributions included writing the manuscript with an emphasis on the software's efficacy in an academic setting, managing data handling and processing, and creating the MCQ assessment tool. I also thoroughly debugged and optimized the system to guarantee optimal performance.

Paper II: Using an Object-Oriented Approach to Secure User Authentication in Academic Course Platforms Duties: I made a substantial contribution to the research concept, especially in the creation of a safe user authentication system. I was in charge of creating and putting into use user verification and password encryption modules. I was in charge of simulating data and conducting security tests to confirm the system's resilience.

**Abbreviations**

• ID: Identification

• CPP: C++ Programming

• UI: User Interface

• OOP: Object-Oriented Programming

• DB: Database (if applicable, though not explicitly shown in the code)

• MCQs Multiple Choice Questions

**List of Tables**

|  |  |
| --- | --- |
| Function | Description |
| addCourse(course\_id, course\_name, mcqs) | Adds a new course to the system. |
| getCourse(course\_id) | Retrieves a specific course by its ID. |
| displayAvailableCourses() | Displays a list of all available courses. |

|  |  |
| --- | --- |
| Function | Description |
| registerStudent(student\_id, student\_name, password) | Registers a new student. |
| authenticateStudent(student\_id, password) | Authenticates a student's login credentials. |
| authenticateAdmin(admin\_id, password) | Authenticates an admin's login credentials. |

|  |  |
| --- | --- |
| Method | Description |
| enrollCourse(course\_id) | Enrolls a student in a specific course. |
| displayEnrolledCourses() | Displays a list of courses the student is enrolled in. |
| attemptMCQ(mcq\_id, selected\_option) | Attempts an MCQ and records the selected option. |

|  |  |
| --- | --- |
| Attribute | Description |
| course\_id | Unique identifier for the course. |
| course\_name | Name of the course. |
| mcqs | List of MCQs associated with the course. |

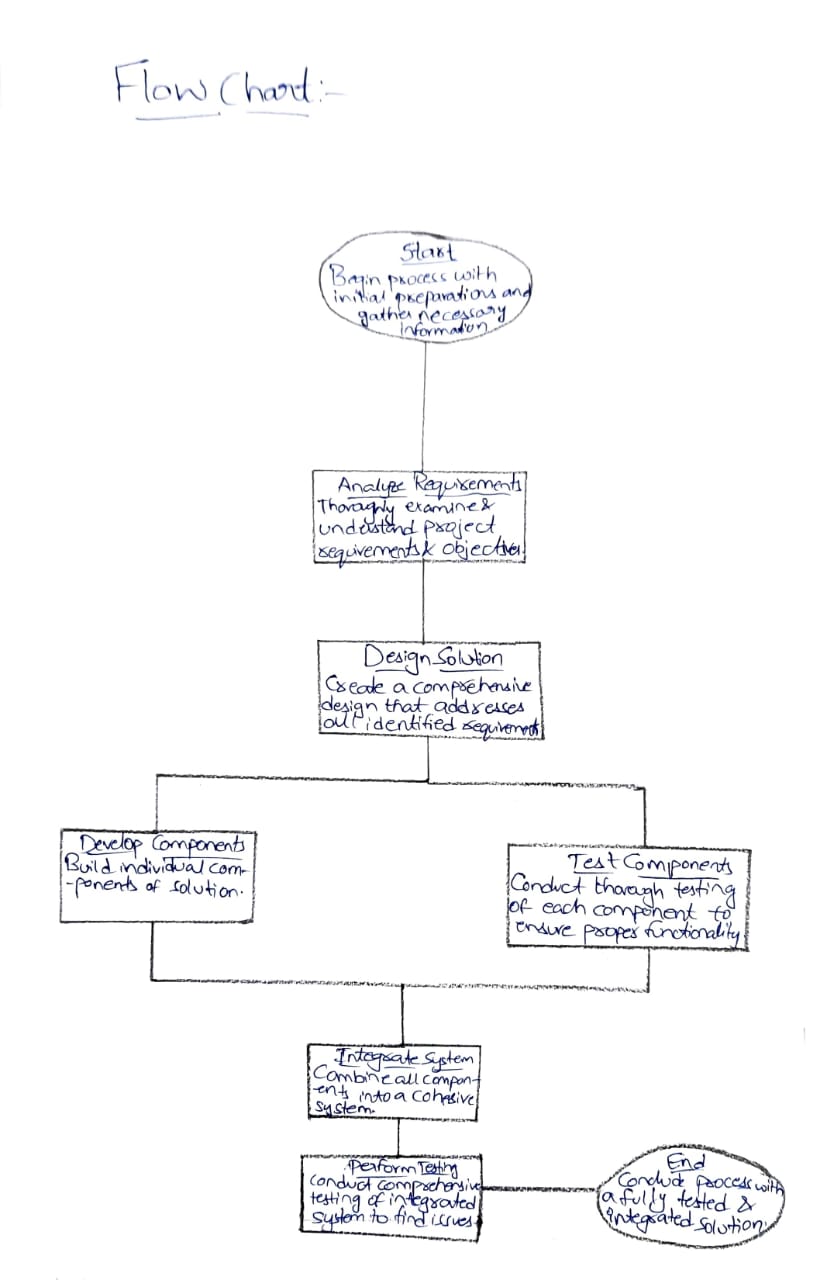
|  |  |
| --- | --- |
| Method | Description |
| calculateScore(attempted\_mcqs) | Calculates the student's score based on their attempted MCQs. |
| attemptAssessment() | Initiates the assessment process for a student. |

|  |  |
| --- | --- |
| Main Menu | Submenu |
| Student Login | - |
| Admin Login | - |
| Exit | - |
|  | Student Menu |
|  | - Enroll in a course |
|  | - Attempt MCQs |
|  | - View results |
|  | - Logout |
|  | Admin Menu |
|  | - Add a course |
|  | - Add MCQs to a course |
|  | - View student performance |
|  | - Logout |

**List of Equations**

|  |  |  |
| --- | --- | --- |
| Equation Number | Equation | Description |
| 1 | Score = (Correct Answers × 100) / Total Questions | Calculation of the student's score based on correct answers and total questions in MCQs. |
| 2 | toupper(userAnswer) == mcqs[i].second | Condition to check if the user's answer matches the correct answer for a question. |
| 3 | username ≠ adminID | Expression used in verifying if the current user is an admin or a student. |
| 4 | studentChoice ≠ 4 | Loop exit condition for student menu options, allowing continuation unless "Logout" is selected. |

**FLOW CHART-**



1. **Introduction**

This chapter presents the idea and importance of a course administration system, which was created as a research project to investigate C++'s Object-Oriented Programming (OOP) concepts. The system offers a safe, interactive platform for administrators and students to administer courses, resources, and tests with an emphasis on improving the educational experience. Tasks like course registration, material administration, and student assessments are made easier with the help of OOP ideas like inheritance, polymorphism, and encapsulation. This course management system is a useful learning tool for students since it not only shows off useful programming applications but also tackles real-world issues in classroom environments.

1.1 Background and Motivation

As educational institutions and instructors look to give students easily accessible and well-structured learning materials, the need for digital academic management platforms is growing quickly. The necessity of such systems in educational settings, specifically to improve students' access to course materials, encourage self-evaluation through quizzes, and promote smooth communication between students and teachers, served as the impetus for this project. Two major goals are achieved by creating a basic yet useful model for a course management system: it gives students practical experience in creating intricate C++ applications and establishes a framework that can be modified and extended for later usage.

1.1.1 Major Issues and Their Resolutions

Several obstacles were overcome in the creation of this course management system, including:

Ensuring that only individuals with permission can access personal and course data is known as student data security.

Providing an MCQ-based test with immediate feedback is an example of an interactive assessment mechanism.

Course Content Admin Control: giving administrators the freedom to add and change courses, resources, and tests as needed.

1. **Methodology**

This section describes the design principles, development process, and essential elements of the methodology used to create the course management system. The technique describes in detail how object-oriented programming concepts and C++ were used to develop a well-structured and useful platform for academic course management.

2.1 System Design and Architecture

An outline of the system's architecture and design is given in this segment, with particular attention to the primary classes and their connections. It emphasizes how a modular, adaptable structure was ensured by utilizing object-oriented concepts like encapsulation, inheritance, and polymorphism.

2.1.1 Class Structure and Relationships

The system is organized around important classes, each of which is in charge of particular functions:

Encapsulating shared information like username and password, the User Base Class serves as the foundation for several user types, including administrators and students.

Specialized classes that derive from the User class are called derived classes (Admin, Student). While the Admin class offers features for course management, such as material updates and assessment preparation, the Student class permits registration, login, course enrolment, and access to multiple-choice questions.

Course Class: Maintains information about the course, including its name, ID, and relevant resources. Additionally, MCQs are stored, allowing evaluations to be directly connected to particular courses.

Assessment Class: Manages the design of multiple-choice questions and their scoring logic, offering a means of assessing students' understanding of each subject. Polymorphism is used in this class to modify feedback and scoring for different kinds of questions.

User Manager Class: In charge of user data management, including safe data access, login credential verification, and new student registration.

2.2 The Process of Development

From the first design stages to the last module implementation, this subsection outlines the entire development process.

2.2.1 Implementation of Core Functionalities

The following core features were put into practice:

User Registration and Login: Encrypted passwords are used to ensure safe access, and the User Manager class manages user registration and login verification.

Access and Enrolment in Courses: Students can sign up for courses, which are subsequently shown on their dashboard. Students can access resources and multiple-choice questions pertaining to their registered courses through the Course class.

MCQ Assessment: Multiple-choice questions can be created in the Assessment class, and students are scored according to their right answers. Performance is tracked against a passing score of 50%. The scoring system's flexibility is increased by accommodating various question kinds using polymorphism.

2.3 Debugging and Testing

To guarantee dependability and a seamless user experience, the project underwent extensive testing.

Unit Testing: To ensure proper operation, every class and essential feature, including user identification, course enrolment, and multiple-choice questions, was tested separately.

Debugging and Error Handling: To improve robustness, custom exception handling was introduced for situations such as missing course data and invalid login credentials. In order to guarantee the reliability of the system, testing also found and fixed problems with the enrolment and scoring procedures.

The project's goals of usability, security, and flexibility were all satisfied by the course management system thanks to this methodical approach, which also gave academic users a scalable and adaptable platform.

**Discussion**

This section examines the course management system's results, determining how effectively it achieves the project's initial goals. We examine the system's achievements, difficulties faced, and possible directions for improvement in the future.

3.1 Results Analysis

The system's performance is assessed in several important categories in this subsection:

Secure User Authentication: To provide secure data access, the login system successfully limits access to only administrators and registered students.

Course Enrolment and Material Access: The objective of improving resource availability and organization is achieved by making it simple for students to enrol in courses and access related materials.

Functionality of MCQ Assessment: The MCQ system effectively offers automatic scoring and feedback based on student answers, providing a useful means of evaluating one's own comprehension.

Admin Course administration: To accomplish centralized content control, the admin module enables the development of courses, material administration, and MCQ setup.

These features' successful deployment shows that the system meets its primary goals of data security, accessibility, and efficient academic administration.

3.2 Project Advantages

Among the course management system's main advantages are:

Modular Design: A modular and scalable structure is made possible by C++ OOP concepts like inheritance and encapsulation. Without undergoing significant reorganization, the system can be altered or expanded with new features.

Flexible Assessment process: Because the MCQ scoring process is based on polymorphism, it may be readily modified to accommodate other question types or grading schemes.

Data Security and User Privacy: The system puts data security first by limiting data access and encrypting login credentials, which is a crucial requirement in academic contexts.

Because of these advantages, the system is reliable, maintainable, and flexible enough to meet changing academic requirements.

3.3 Limitations and Challenges

Among the restrictions and difficulties encountered during development are:

Basic Course Material Display: The system currently limits the kinds of resources that can be offered by presenting course materials in a straightforward manner devoid of multimedia or sophisticated content formatting.

Limited Assessment Feedback: Based on a 50% criterion, the MCQ system only provides a pass/fail response. Students would gain a better grasp of their comprehension if they received more thorough feedback on every question.

Command-Line Interface: Because the system is a console-based application, it does not have a sophisticated graphical user interface (GUI), which might increase accessibility and user engagement.

These drawbacks point to possible areas for improvement in order to produce a more thorough and engaging teaching resource.

3.4 Upcoming Enhancements

The following improvements are recommended in order to further enhance functionality and usability:

Improved Assessment Feedback: Giving each MCQ question thorough feedback would enhance learning outcomes and assist students in pinpointing their areas of weakness.

Advanced Course Material Integration: A more comprehensive learning experience would result from allowing multimedia and structured documents (such as PDFs and movies).

Graphical User Interface (GUI): Creating a GUI would improve the system's usability and aesthetic appeal, particularly for students who are not as accustomed to command-line interfaces.

Real-Time Performance Analytics: Including analytics to monitor student development and involvement may assist educators and administrators in making well-informed choices regarding the structure and content of their courses.

The system's capabilities would be increased by these upcoming upgrades, making it a more effective and adaptable tool for academic institutions.

1. **Concluding Remarks**

This project skilfully illustrates how to use C++ to create a course administration system while putting fundamental object-oriented programming (OOP) ideas like inheritance, encapsulation, polymorphism, and exception handling into use. With a modular, adaptable, and user-friendly architecture, the system accomplishes its main goals: safe student and administrator authentication, organized access to course materials, and an interactive multiple-choice question evaluation function.

The project serves as an example of how OOP principles may be used to create a scalable and organized program, which is crucial for complicated instructional software. Important lessons learned from the project include:

Effective Utilization of OOP Concepts: The development of a modular and flexible system made possible by inheritance and polymorphism made it possible to quickly add new features like more course materials or different kinds of assessments.

Prioritizing user privacy and data security A crucial prerequisite for academic applications is met by encrypted login credentials and limited data access, which guarantee that only authorized users engage with the system.

Code Reusability and Flexibility: The system is well-suited for growth and maintenance since it permits simple future upgrades through the use of templates and modular functions.

Additionally, the project highlights some shortcomings that might be fixed in subsequent versions, such as the basic way the course materials are shown and the lack of feedback on assessments.

Future Outlook

Advanced features including multimedia course materials, thorough evaluation feedback, and a graphical user interface to improve user experience could be added to the system with additional work. Furthermore, it would be a more complete academic solution if it included real-time statistics for tracking student development.

To sum up, this project demonstrates C++'s strength as a language for building intricate applications and the usefulness of OOP concepts in producing software that is both scalable and maintainable. The course management system offers exciting prospects for future improvements and practical applications in educational software development, and it functions as both a useful learning tool and a working prototype.

.

**References**

List all sources, textbooks, documentation, and articles referenced during the project in this section. Format each entry according to your required citation style. Below are some examples of how to structure entries:

* Alakad, Zaid. "Integration of ProFormA Grappa into Virtual Programming Lab." PhD diss., Hochschule Hannover, 2024.
* Markoska, Ramona. "Managing ICT solutions for training and evaluation of C++ programming skills in e-learning ecosystem." New Trends and Issues Proceedings on Humanities and Social Sciences 6, no. 7 (2019): 33-41.
* Markoska, R., 2019. Managing ICT solutions for training and evaluation of C++ programming skills in e-learning ecosystem. New Trends and Issues Proceedings on Humanities and Social Sciences, 6(7), pp.33-41.