Department of Information and Communication Technology

Faculty of Technology

University of Ruhuna

Object Oriented Programming Practicum

ICT2132

Mini Project

Group Name : Cyber Creepers

Group NO : 08

Submitted to: Mr.P.H.P. Nuwan Laksiri.

Group Members: -

TG/2022/1344 - Ravindu Sadaruwan

TG/2022/1345 - Akalanka Perera

TG/2022/1356 - Prabashi Nuwani

TG/2022/1363 - Heshan Dilsara

Content

1. Introduction

This mini project, developed for the ICT2132 Object-Oriented Programming Practicum, is a management system designed for the Faculty of Technology at the University of Ruhuna. The system simplifies administrative tasks by handling user profiles, course management, student attendance, medical records, notices, timetables, and marks. It provides an organized digital platform to streamline faculty operations, making it easier for staff and students to access and manage important academic information. By automating these processes, the system improves efficiency and reduces manual paperwork

This project was developed using Java for the backend logic and Java Swing for building a user-friendly Graphical User Interface (GUI), with NetBeans as the primary Integrated Development Environment (IDE). For secure and efficient data storage, MySQL was used as the database management system. The team collaborated on the project using GitHub for version control and seamless code sharing. This setup ensured a structured and organized development process while maintaining data integrity and team coordination.

This project was built using Object-Oriented Programming (OOP) concepts like inheritance, polymorphism, abstraction, and encapsulation to keep the code organized and efficient. We also added error handling to prevent crashes when users interact with the system or when the app communicates with the database.

Working on this project helped us gain real-world experience in developing a desktop application with Java, Swing, and MySQL. It improved our skills in:

2. Objectives

- To design and develop a user-friendly desktop application to manage administrative tasks for the Faculty of Technology.
- To implement a role-based access control system for Admin, Lecturer, Technical Officer, and Undergraduate users.
- To manage and maintain undergraduate attendance, medical records, course details, timetables, notices, and marks efficiently.
- To apply Object-Oriented Programming (OOP) principles such as inheritance, abstraction, polymorphism, and encapsulation in the development.
- To ensure reliable data storage and retrieval by connecting the application with a MySQL database.
- To handle errors and exceptions gracefully, ensuring a smooth user experience.
- To practice collaborative development using GitHub for source code management.

3. System Overview

Users and Functionalities:

- **Admin**:
    - Create and maintain user profiles (Admin, Lecturer, Technical Officer, Undergraduate).
    - Create and manage course details, notices, and timetables.
    - Provide CRUD operations for Course, Notice, Timetable and User Management
    - Create user Login from and clear approach for session management for each logged user.
- **Lecturer**:
    - Update their profile (excluding username and password).
    - Upload learning materials, add and update undergraduate marks.
    - View undergraduate details including attendance, medicals, eligibility, grades, and GPA.
    - View faculty notices.
- **Technical Officer**:
    - Update their profile (excluding username and password).
    - Maintain undergraduate attendance and medical records.
    - View department timetables and notices.
- **Undergraduate**:
    - Update personal contact information and profile picture.
    - View their course details, attendance, medical records, grades, GPA, timetables, and faculty notices.

4. Implementation

The system was built using Java for the backend logic and Java Swing with NetBeans GUI Builder for designing an interactive GUI. MySQL served as the database, connected to the application through JDBC for efficient data management.

Implementation Highlights:

Object-Oriented Programming (OOP) Concepts:

We applied inheritance, polymorphism, abstraction, and encapsulation throughout the system to ensure code reusability and clarity.

Database Handling:

A separate Database Connection class was to manage MySQL connections. Prepared Statements were used to prevent SQL injection and handle data securely.

GUI Development:

Each module (User Management, Course created Management, Attendance, Medicals, Marks) had a dedicated Swing form, Hard code for a clean appearance**.**

Error and Exception Handling:

All database operations and user inputs were enclosed within try-catch blocks to capture and display meaningful error messages without crashing the application.

Authentication System:

A secure login system was created where users are authenticated based on their role (Admin, Lecturer, Technical Officer, Undergraduate) and redirected to their respective dashboards.

Attendance and Marks Management:

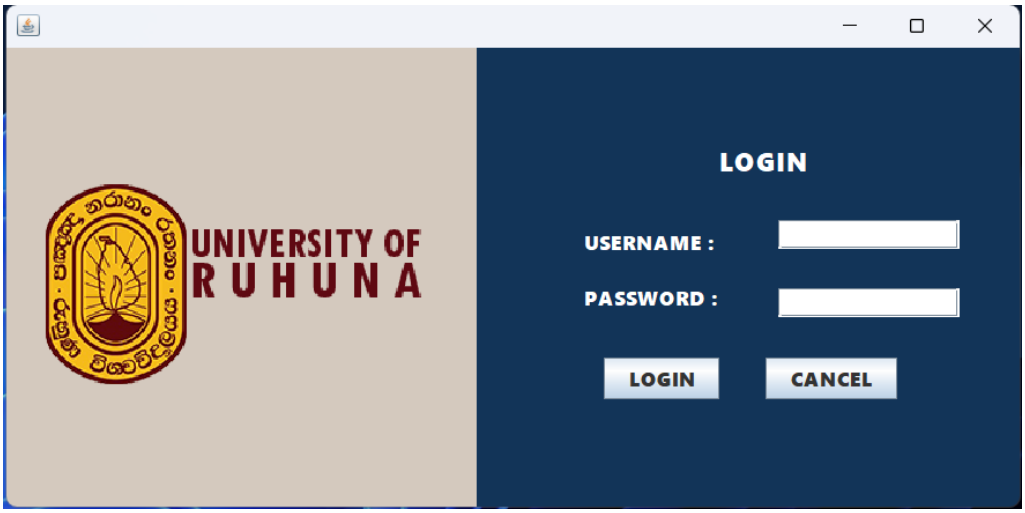
Attendance was recorded separately for Theory and Practical sessions. Medical records could be attached to justify missed sessions. Marks were dynamically calculated to determine undergraduate eligibility.

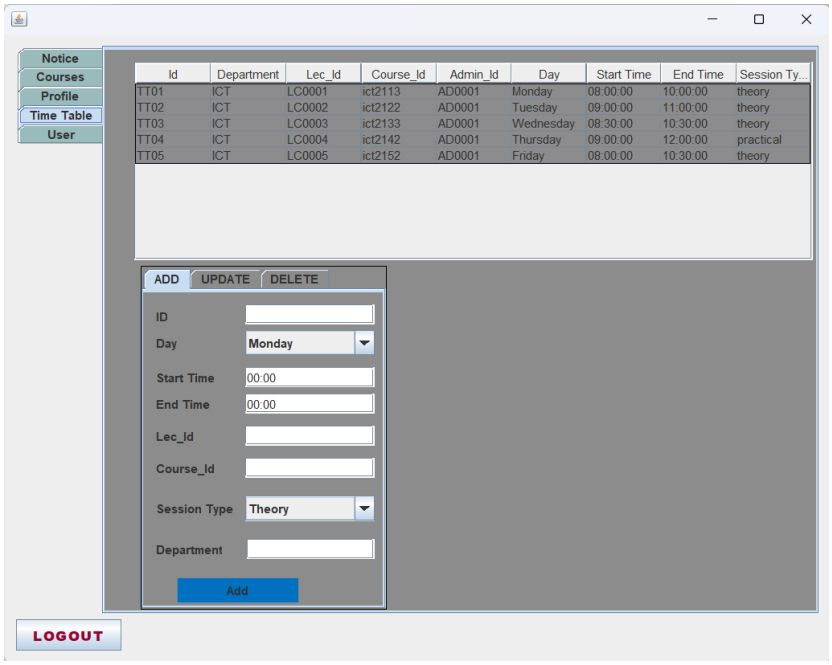
Eligibility and GPA Calculation:

The system automatically calculated CA marks, determined final eligibility, assigned grades, and computed SGPA and CGPA based on predefined rules.

5. User Interfaces

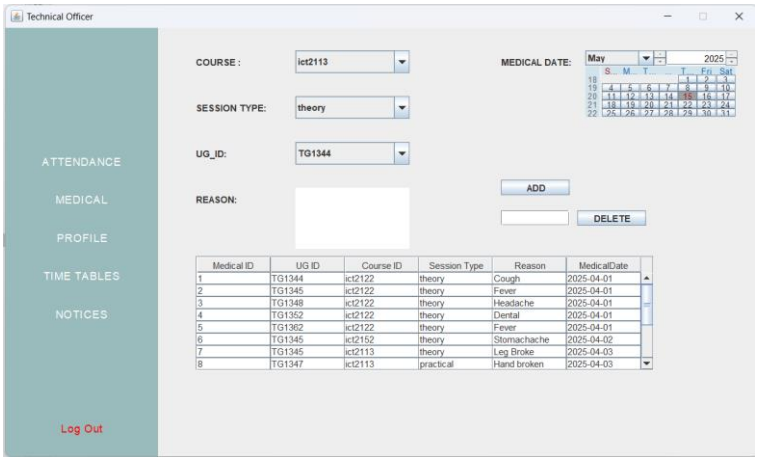
Admin





- Login Page
- Admin Dashboard
- Attendance Summary
- Marks Management
- Notices Management
- Timetable View

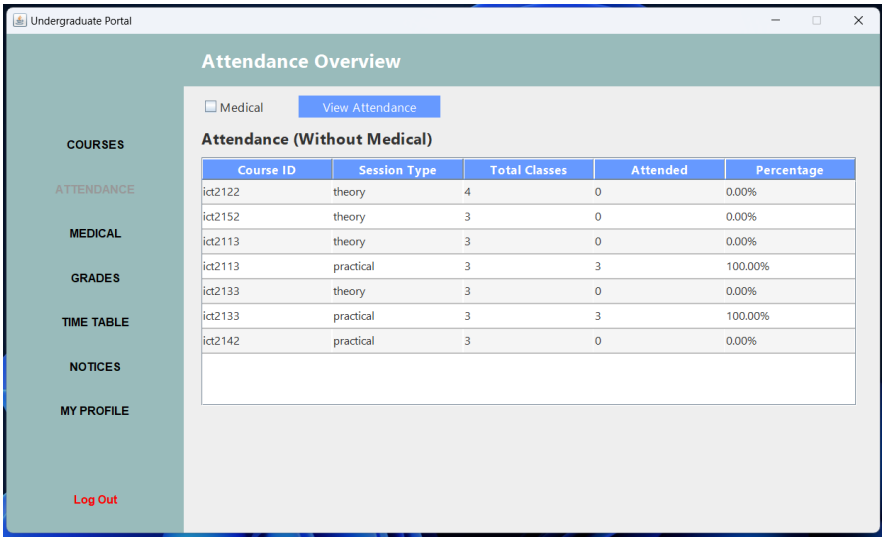
Technical Officer



- Maintain Attendance
- Maintain Medical
- View and Update Profile

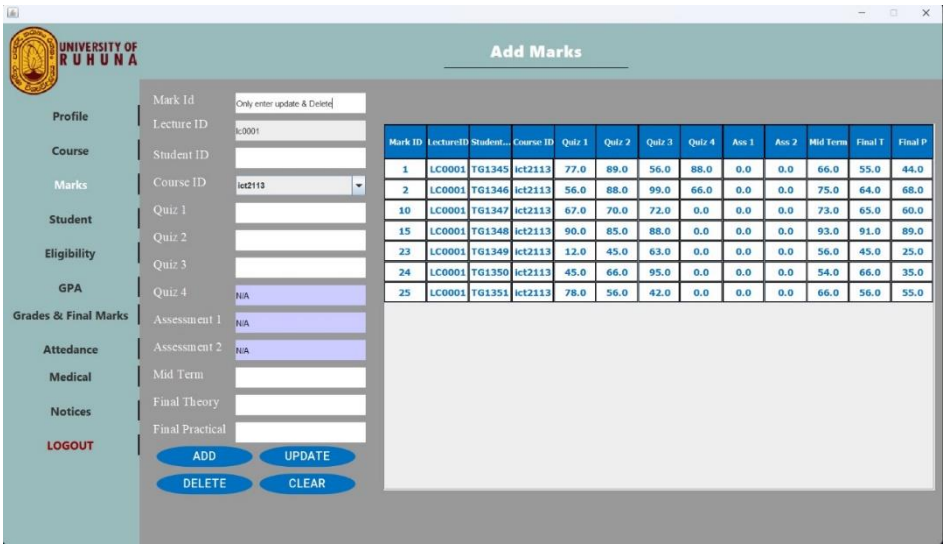
- View Timetables
- View Notices

Undergraduate



- View Courses
- View Attendance
- View Medical
- View Grades
- View Timetables
- View Notices
- View & Update Profile

Lecturer



- Can View & Update Profile
- View and update Course Materials

- Upload Marks
- View Student Details
- View Student Eligibility
- View Student Marks, Grades, GPA
- View attendance
- View Medical records
- View Notices

8. Problem and solutions

i. Hard to find errors:
One of the challenges faced in software development, including the implementation of the Faculty of Technology Management System (FTMS), is the difficulty in identifying code errors. Code errors, such as syntax errors, logical errors, or runtime errors, can hinder the functionality of the system and impede its overall performance. These errors may arise during the development phase or later during system maintenance and updates.

- Solution: Use Enable Compiler Warnings and Error Messages, Unit Testing, Static Code Analysis Tools, Debugging, Version control systems, documentation, and comments.

ii. Network issues.

When encountering network issues, it typically means experiencing difficulties with connecting to the internet or communicating with other devices on a network. These problems can manifest in various ways, such as slow or unreliable internet speeds, intermittent connectivity, or complete loss of network access.

- Solution: Steps may include checking physical connections, rebooting devices, testing connectivity, adjusting router settings, updating software, and contacting the ISP for assistance. By following these steps, users can resolve network problems and restore reliable internet connectivity.

III. Technical Difficulties:

Java programming can be complex and challenging, especially for students who are new to the language. Technical difficulties can arise, including coding errors and compatibility issues.

- Solution: To address technical difficulties, teams can establish coding standards and best practices. Team members can also work collaboratively to identify and troubleshoot technical issues. Consider assigning a team member to serve as a technical lead, who can provide guidance and support as needed.

9. Conclusion

This project served as an excellent practical implementation of Object-Oriented Programming principles, resulting in a comprehensive Faculty Management System. The developed solution successfully addressed all core administrative needs, including user profile management, course administration, attendance tracking, grade management, medical record handling, notice dissemination, and timetable organization.

The system features an intuitive GUI built with Java Swing and NetBeans GUI Builder, ensuring smooth user interaction. For reliable data management, we implemented a MySQL database with secure JDBC connectivity. Throughout development, we maintained rigorous standards for error handling, data security, and followed software engineering best practices to deliver a robust academic management solution.

10. References

- Java Swing Documentation
- MySQL Documentation
- GitHub Repository: [https://github.com/Raviduofficial/JAVA_Project.git]

11. Contribution

GROUP MEMBER	**CONTRIBUTION**

TG1344 Ravindu	• Create **User** login
	• Admin GUI Interface
	• Admin Operation
	  ▪ Create and maintain all User Profiles
	  ▪ Create and maintain Courses
	  ▪ Create and maintain Notices
	  ▪ Create and maintain Timetables

TG1345 Nimesha	• Create **Undergraduate** GUI Interface
	• Create Database Query
	• Create Undergraduate Operation
	  ▪ Can update contact details and profile picture of their profile
	  ▪ Can see attendance details
	  ▪ Can see medical details
	  ▪ Can see their course details
	  ▪ Can see their grades and GPA
	  ▪ Can see their timetables
	  ▪ Can see notices Create and maintain Timetables

TG1363 Heshan	• Create **Technical Officer** GUI Interface • Create TO Login Interface • Create TO Operations ▪ Can add and maintain attendance details of students. ▪ Can add and maintain medical details of students. ▪ Can see notices. ▪ Can see the timetables of their department.
TG1356 Prabashi	• Create **Lecturer** GUI Interface • Create Lecturer Login Interface • Create Lecturer Operations ▪ Lecturer Can modify and add materials to courses. ▪ Lecturer Can upload marks for all kind of exams. ▪ Lecturer Can see student details. ▪ Lecturer Can see student eligibility. ▪ Lecturer Can see student marks, grades, and GPA. ▪ Lecturer Can see attendance and medical records of students. ▪ Lecturer Can see notices.