

**Department of Information and Communication Technology Faculty of Technology**

**University of Ruhuna**

**Database Management Systems Practicum ICT 1222**

**Assignment 02 – Mini Project**

Group 19

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

Submitted by: <Index no> - <Name>

**Content**

1. Introduction to the problem/group project
2. Introduction to the solution we have found to the related problem
3. DRD [Data Requirement Document]

* Introduction
* Data requirements
* Detailed data descriptions
* Relationships between Entities
* Data Constraints and Assumptions
* ER/EER Diagram
* Relational Mapping Diagram
* Data Volume & Sample data

1. Table Structures
2. Architecture
3. Tools & Technologies
4. Security measures to protect the Database
5. Users in the Database
6. Code Snippets
7. Challenges we have faced during the development
8. How we faced the above challenges
9. Backend Hosting
10. Cloud Hosting
11. Individual Contribution of team members
12. References

**Introduction to the Group Project**

The Faculty of Technology of University of Ruhuna requires an efficient and centralized system to manage student information, including personal details, attendance, marks, and results. This Student Management System aims to automate and streamline these operations using a robust Database Management System. This group project is to build a useful, manageable system to handle, store and output accurate, consistent and timely information. Also, it provides authorized users with access to the information they need.

**Introduction to the Solution**

The proposed solution is a Database Management System developed using MySQL to manage all student-related data. We have built a Database to store students’ relevant data and output the required information by performing calculations.

The system provides functionalities for:

* Managing student and staff details
* Recording and analyzing attendance
* Managing assessment and exam marks
* Calculating grades, eligibility, SGPA, and CGPA
* Handling user-specific access through role-based privileges

Objectives of the proposed Student Management System:

* Centralized Data Storage
* Role-based Access Control
* Enhanced Reporting and Decision support
* Better student performance monitoring
* Smooth result Management
* Scalability and Reliability
* Reduced Workload
* Data Accuracy

**Data Requirement Document [DRD]**

**Introduction**

The Data Requirement Document defines all necessary data items, their structures, relationships and constraints, ER diagram and the relational mapping used for building the Student Management System database. This defines the data requirements needed to support functionalities such as student enrollment, attendance tracking, mark recording, eligibility checking, and result generation.

**Data Requirements**

Main entities to be included in the database:

* User 🡪 Student, Lecturer, Technical Officer
* Course
* Enrollment
* Department
* Attendance\_session
* Attendance\_Records
* Medical
* Mark
* Assessment\_Type
* Result\_summary

**Detailed Data Description**

**User** → Represents all system users including Students, Lecturers, and Technical Officers, each assigned with specific access privileges.

**Course** → Stores details of each course such as course code, title, credits, and lecturer in charge.

**Enrollment** → Links students to the courses they register for in a semester.

**Department** → Contains information about departments within the faculty offering the courses.

**Attendance\_Session** → Records details of each class session (date, time, type: theory/practical) conducted for a course.

**Attendance\_Records** → Keeps individual attendance data of students for each session, indicating presence, absence, or medical status.

**Medical** → Stores information about medical certificates submitted by students to justify absences.

**Mark** → Maintains the marks obtained by students for different assessments in each course.

**Assessment\_Type** → Defines types of evaluations such as quizzes, assignments, mid-semester, and final exams.

**Result\_Summary** → Summarizes results including grades, eligibility, and GPA calculations for each student.

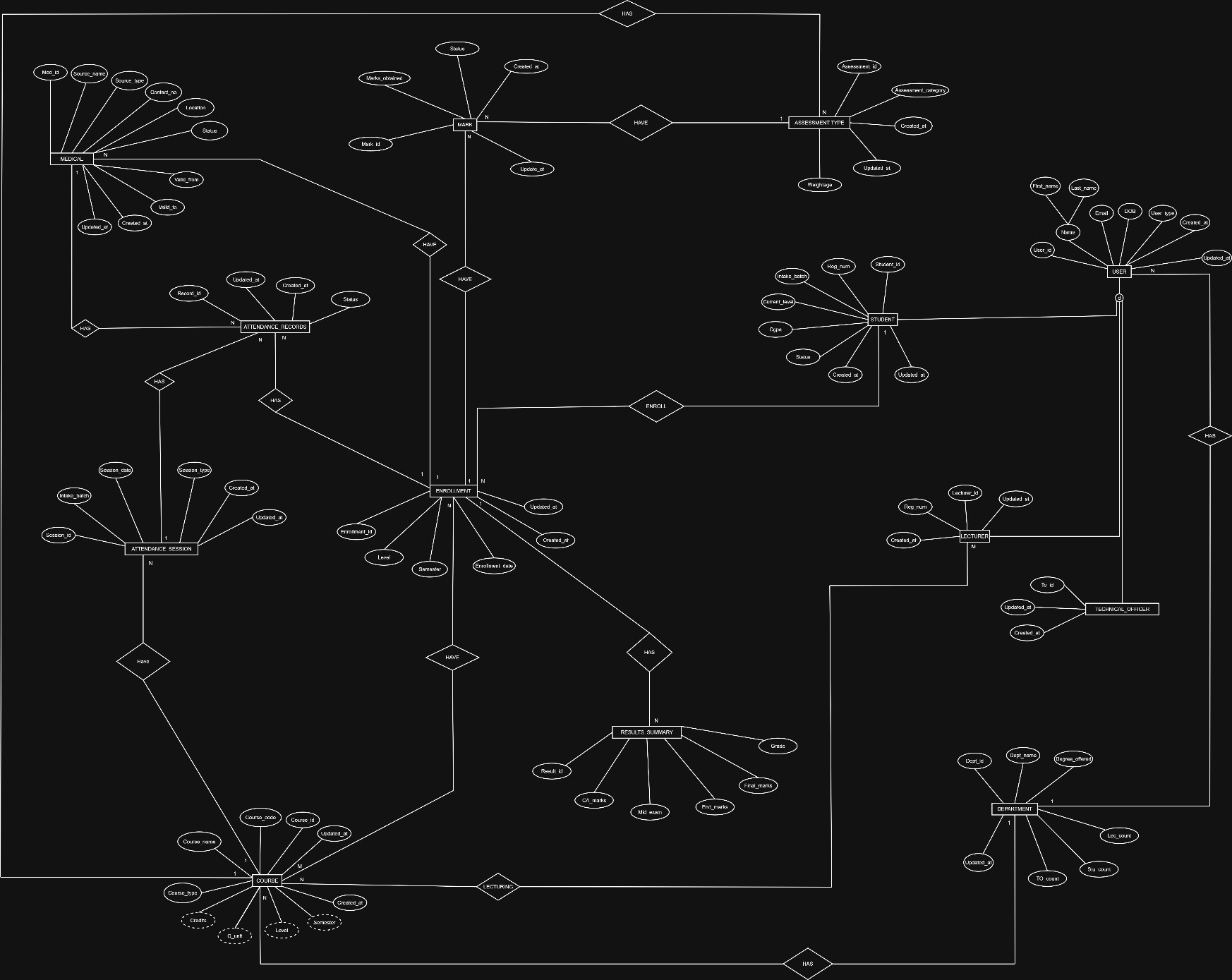
**Relationships Between Entities**

* A Department has many Courses and many Lecturers
* A Lecturer teaches many courses
* A Student can enroll in many Courses, and each Course can have many Students.
* Each Course consists of multiple Attendance\_sessions
* Each Attendance\_session has multiple Attendance\_records, each linled to Student
* A Student may submit multiple Medical records related to Attendance\_records
* Each Course has multiple Assessment\_types
* Marks are recorded per Student, Course, and Assessment\_type
* Result\_summary aggregates the final marks and grades for each student per course

**Data Constraints and Assumptions**

* Attendance eligibility must be > 80%
* All marks are stored as percentages
* Students with medical approval will have “MC” displayed in their results
* Repeat students have a maximum achievable grade of “C”
* Suspended students have results displayed as “WH”

**EER Diagram**

****

**Relational Mapping**

A black screen with white lines

AI-generated content may be incorrect.

**Table Structures**

User

**A screen shot of a computer

AI-generated content may be incorrect.**

Student

**A black screen with white text

AI-generated content may be incorrect.**

Lecturer

**A screen shot of a computer

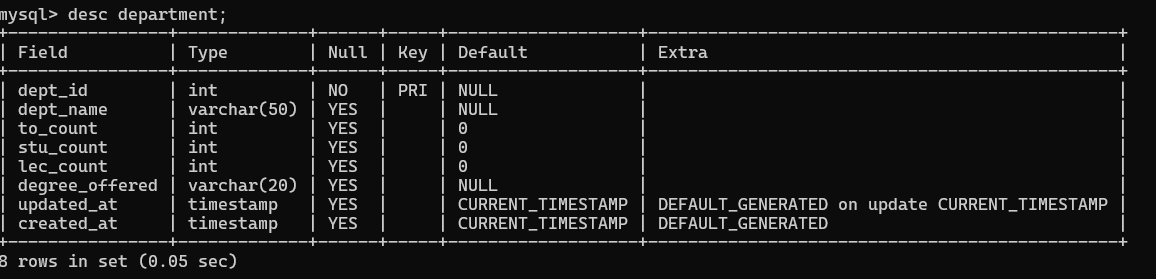
AI-generated content may be incorrect.**

Technical Officer

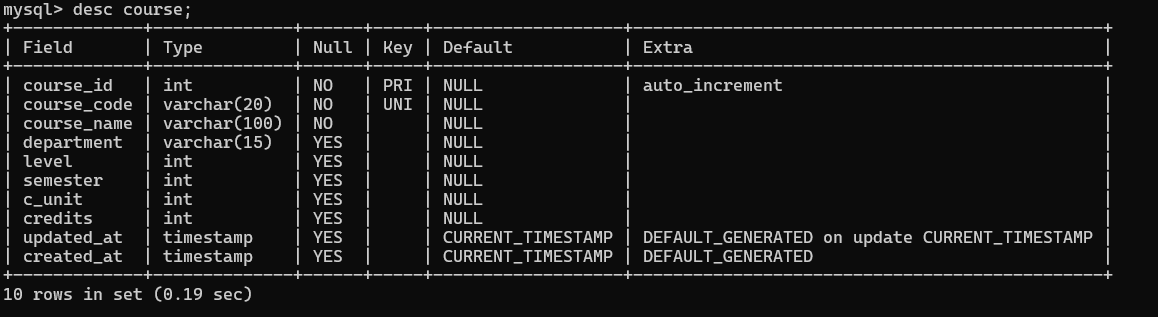
**A screen shot of a computer screen

AI-generated content may be incorrect.**

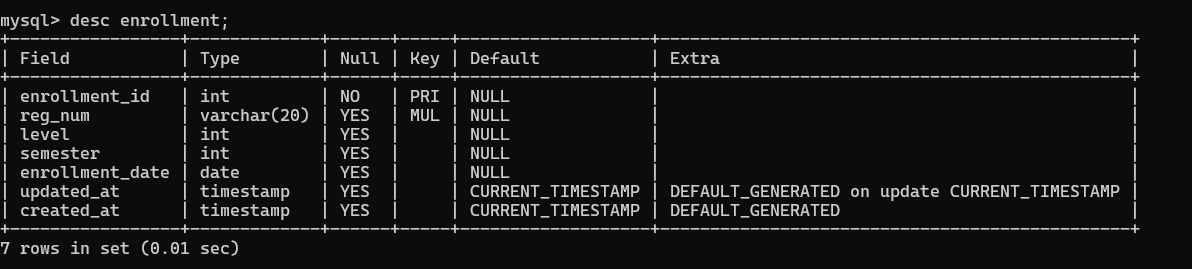
Department

****

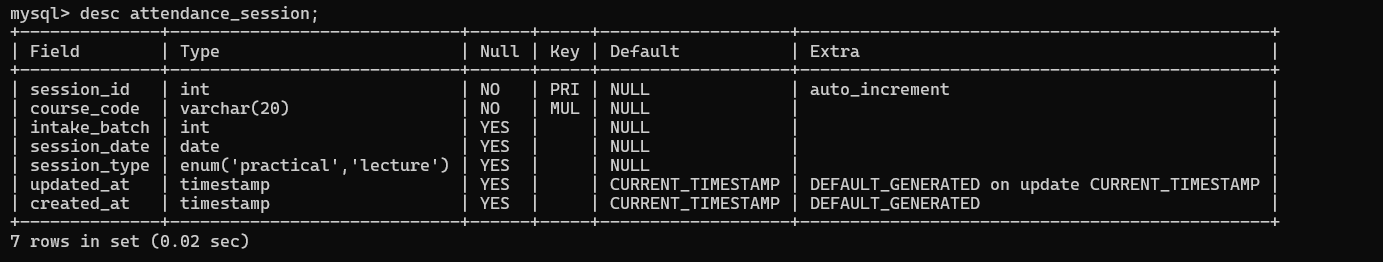
Course

****

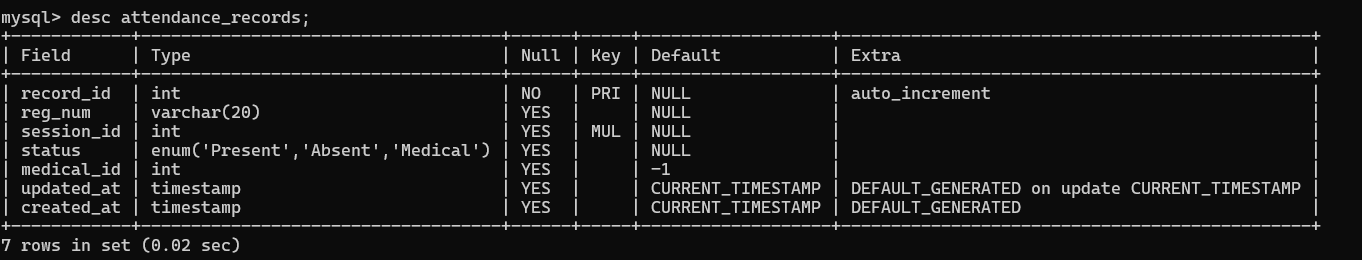
Enrollment

****

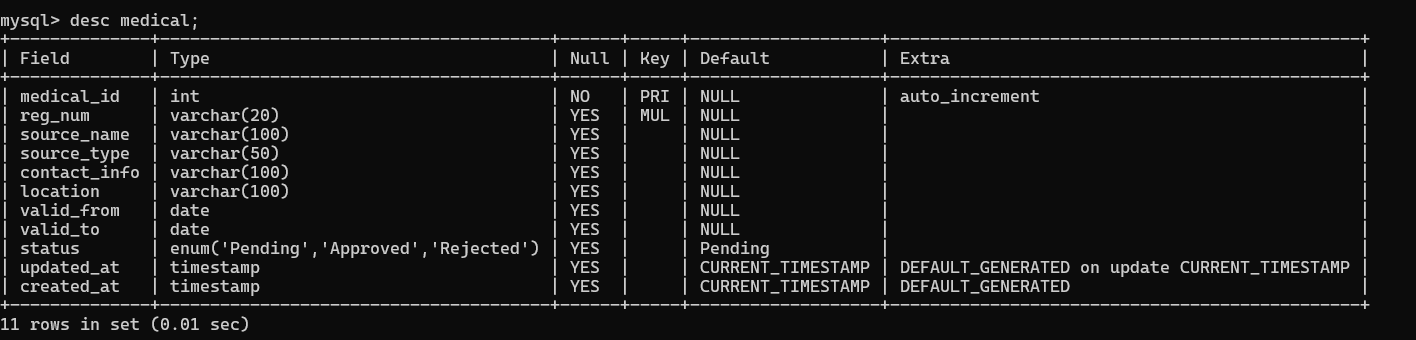
Attendance\_session

****

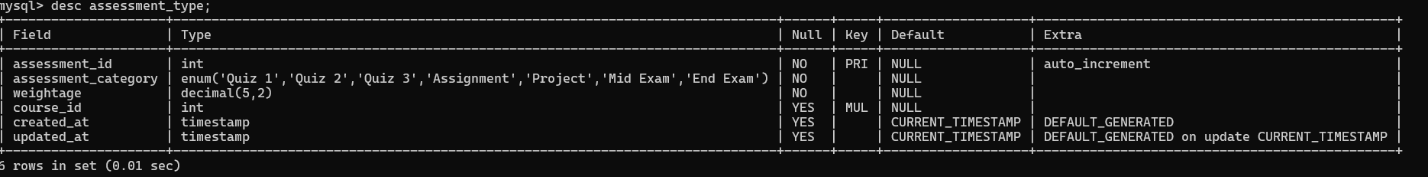
Attendance\_record

****

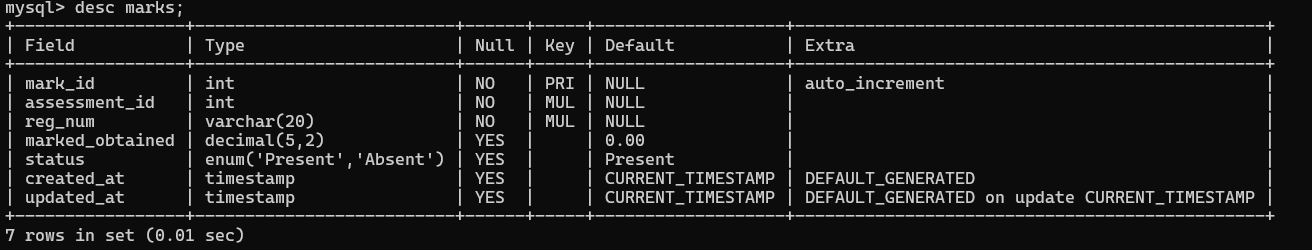
Medical

****

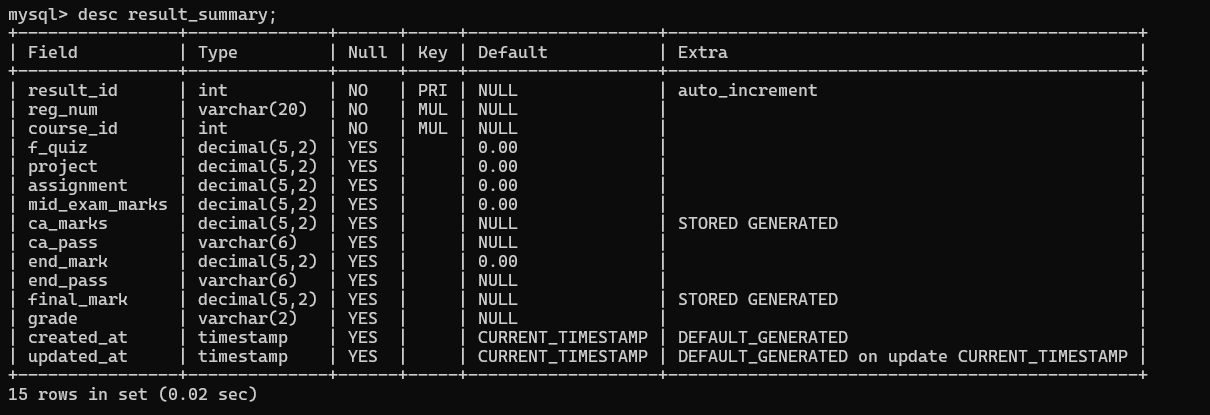
Assessment\_type

****

Marks

****

Result\_summary

****

**Architecture of the Database**

User Interface

Application / Business Logic Layer

MySQL Database

* The user (Studentt/Lecturer/Technical\_officer/Admin/Dean) logs into the system.
* The Application Layer handles data validation, processing and communication with the MySQL backend.
* The MySQL database layer stores all students, lecturers, marks and attendance data and results.

**Tools & Technologies**

* MySQL Server 🡪 Create, manage and query the relational database.
* Draw.io 🡪 Design the EER diagram and the Relational Mapping Diagram
* Notepad++ 🡪 Utilizing the writing and editing SQL queries, scripts, and code snippets efficiently.
* GitHub 🡪 Version Control and Collaboration, allowing the team to manage code updates, track changes, and share progress seamlessly.

**Security Measures**

Database Security is essential to protect sensitive academic information such as student marks, grades and attendance records from unauthorized access or modification. We have taken few steps to solve the security issues that can be raised related to the database.

1. Role-based Access Control
2. Privilege Limitation
3. Use of Views for secure data access
4. Authentication and password protection
5. Data validation and consistency

**Users in the Database**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

|  |  |  |
| --- | --- | --- |
| User Role | Privileges | Description |
| Admin | ALL PRIVILEGES on all tables with GRANT OPTION | Responsible for database maintenance, user creation, backups, and overall system management. |
| Dean | ALL PRIVILEGES (without GRANT OPTION) | Oversees academic operations, manages course details, and reviews marks and attendance reports. |
| Lecturer | SELECT, INSERT, UPDATE on marks and attendance tables; SELECT on student and course tables | Enters and updates student marks and attendance; reviews reports. |
| Technical Officer (TO) | SELECT, INSERT, UPDATE on attendance-related tables and views | Manages attendance for practical sessions and verifies records. |
| Student | SELECT on attendance and results views only | Can view personal attendance percentage, eligibility, and final grades. |

**Code Snippets Used**

* Stored Procedures (for calculating eligibility or grades)

A screenshot of a computer program

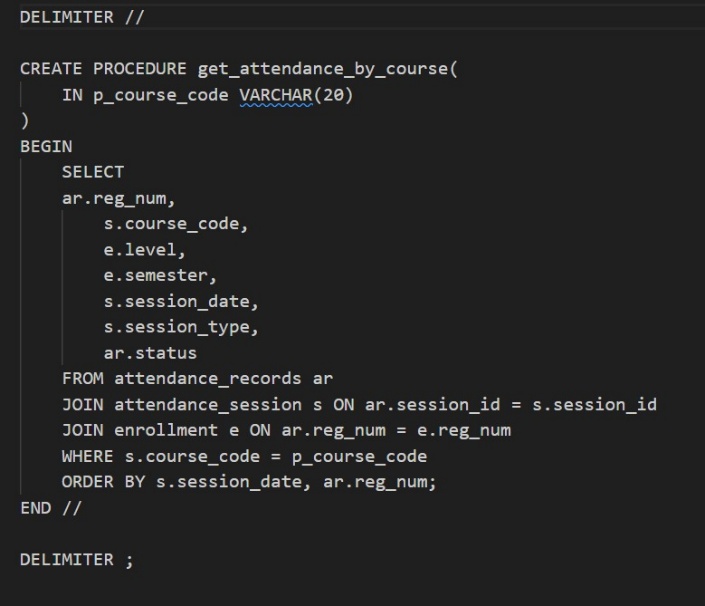
AI-generated content may be incorrect.

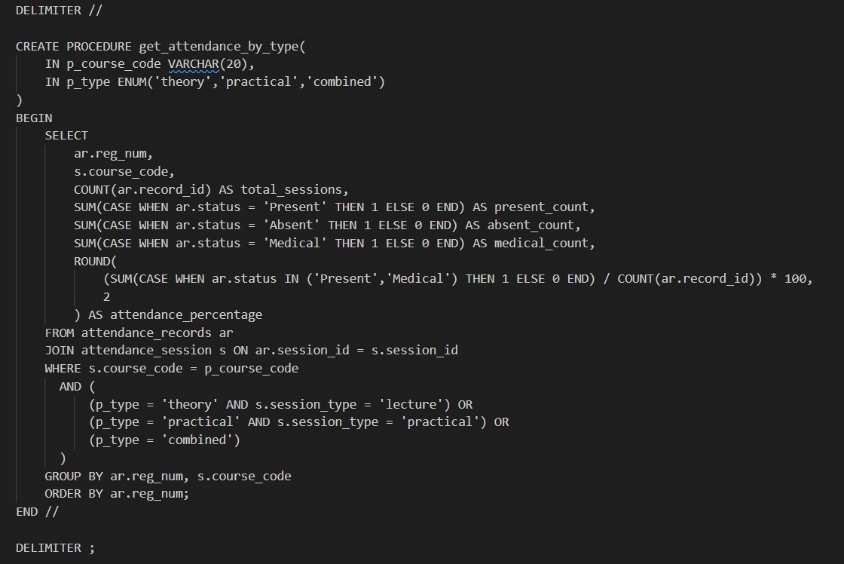
A screen shot of a computer

AI-generated content may be incorrect.

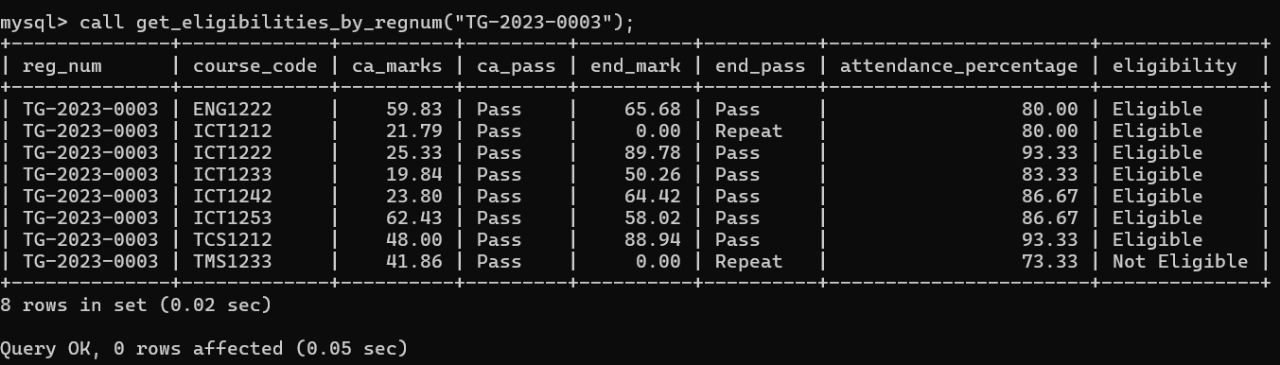
**A screen shot of a computer

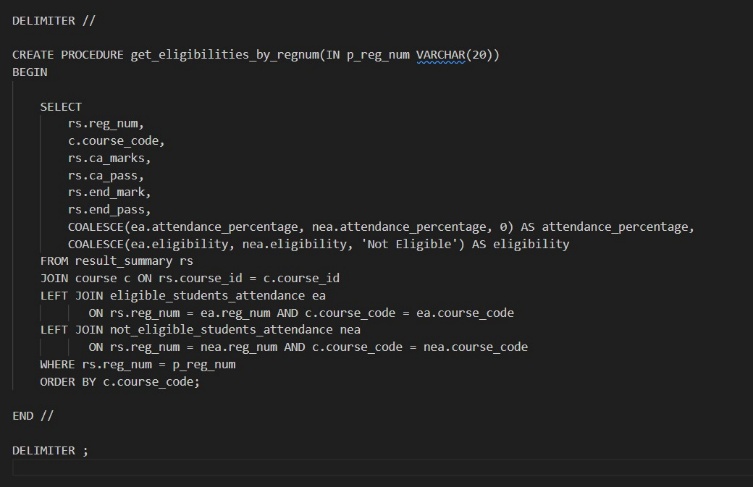
AI-generated content may be incorrect.A screen shot of a computer

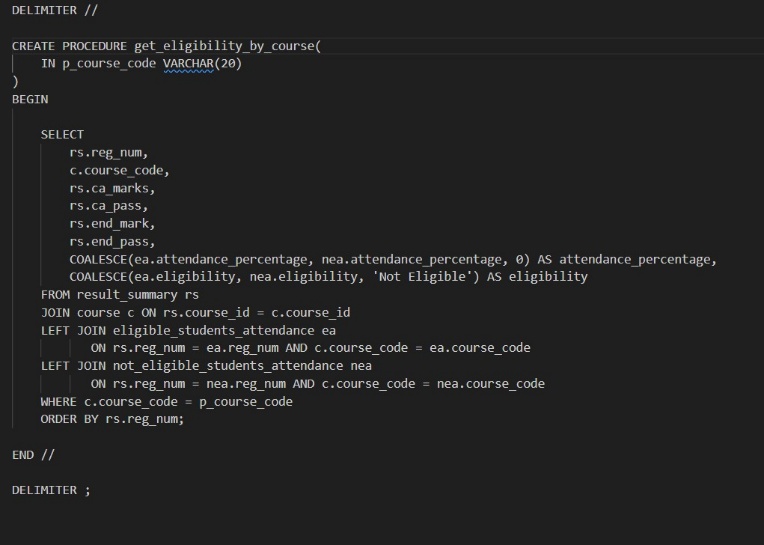
AI-generated content may be incorrect.**

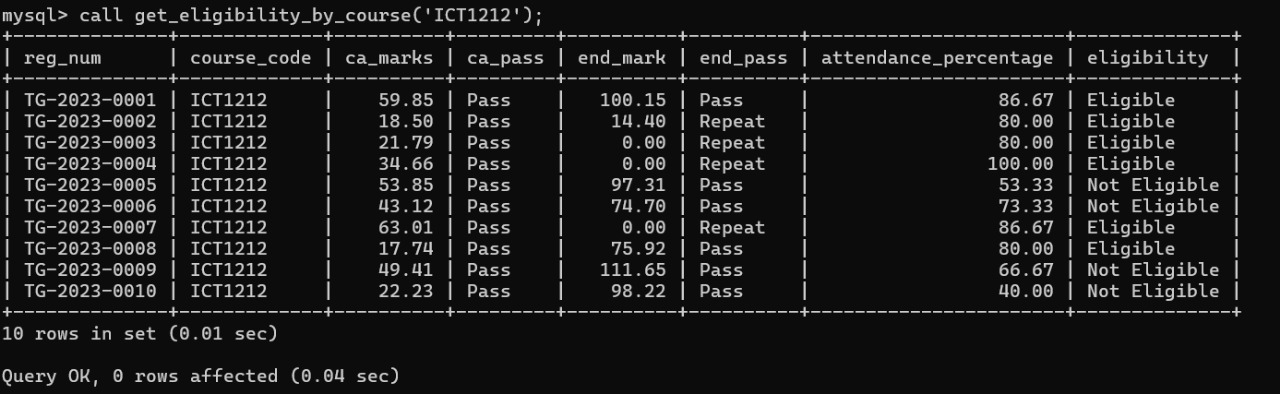
****

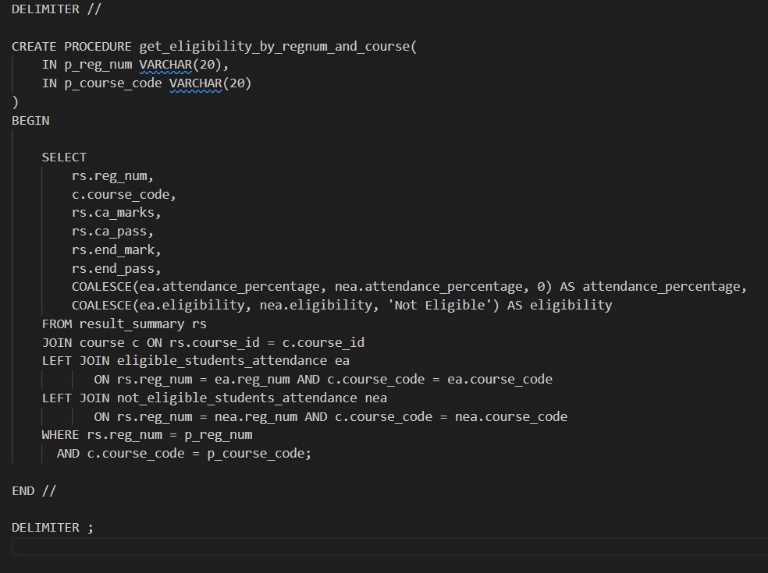
**A screen shot of a computer

AI-generated content may be incorrect.**

****

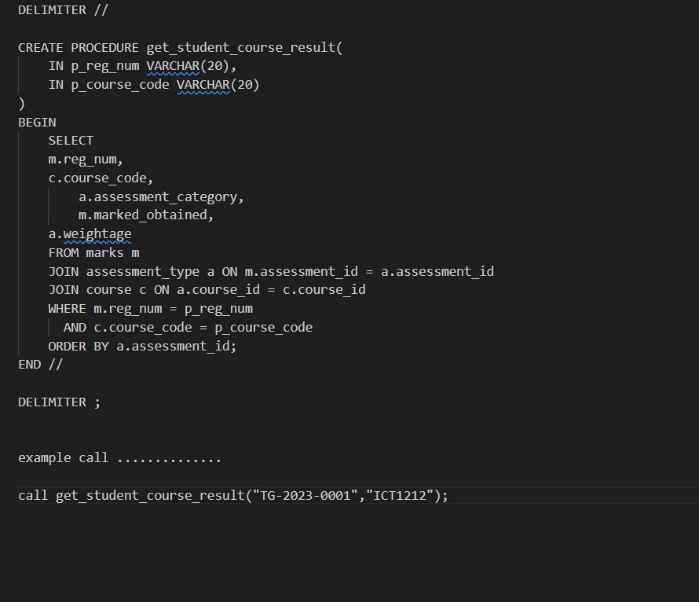
****

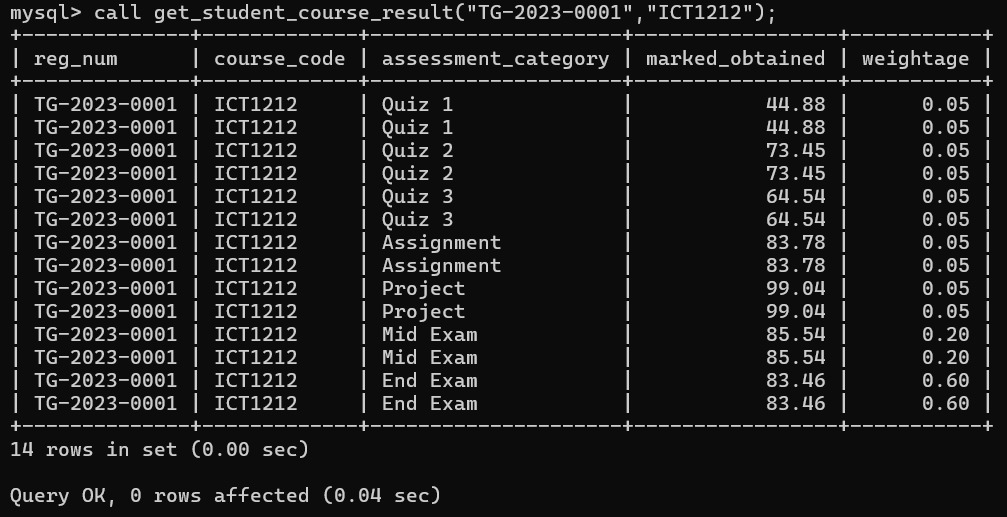
****

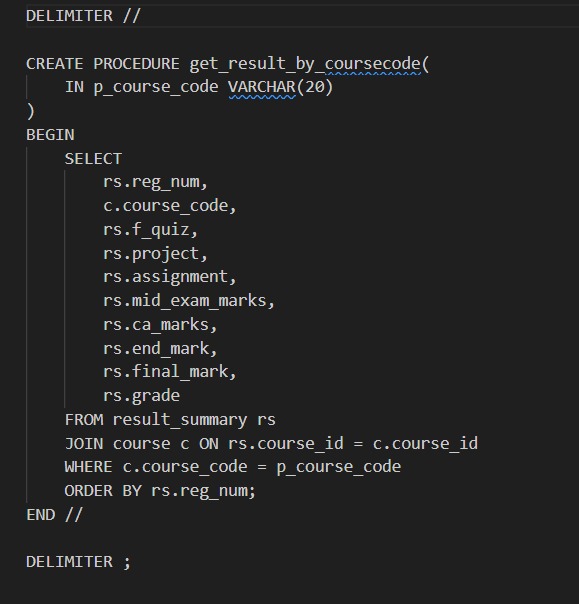
****

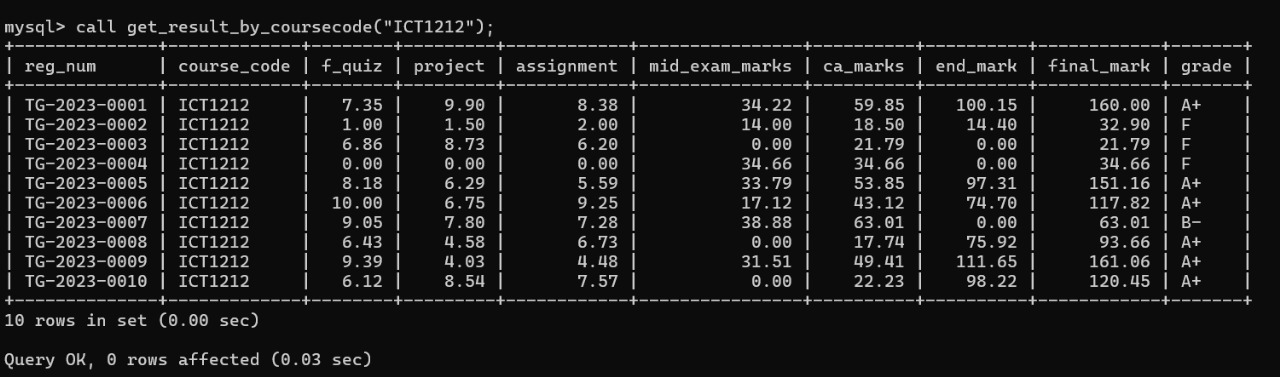
**A black screen with white text

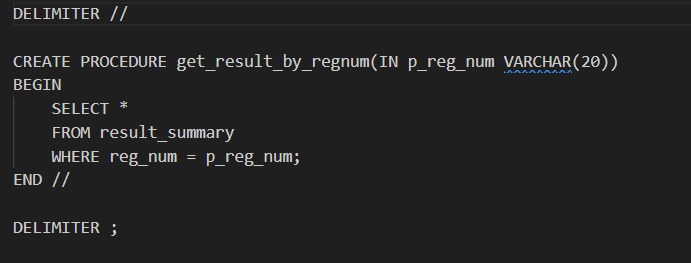
AI-generated content may be incorrect.**

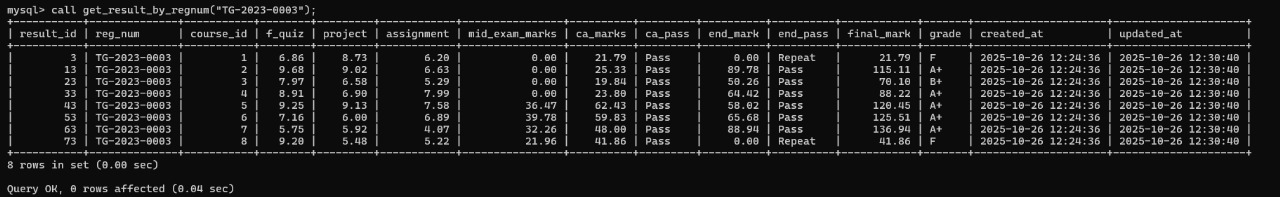
****

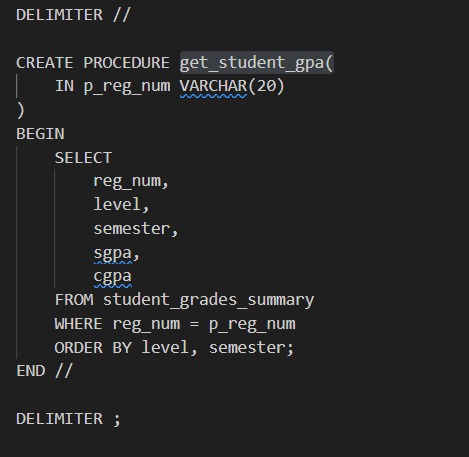
****

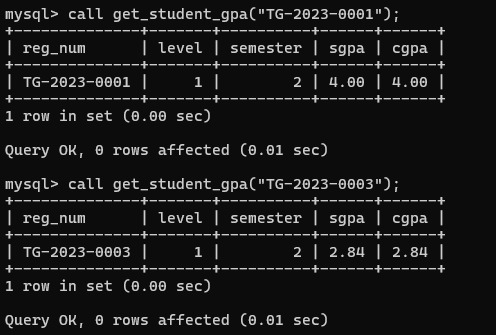
****

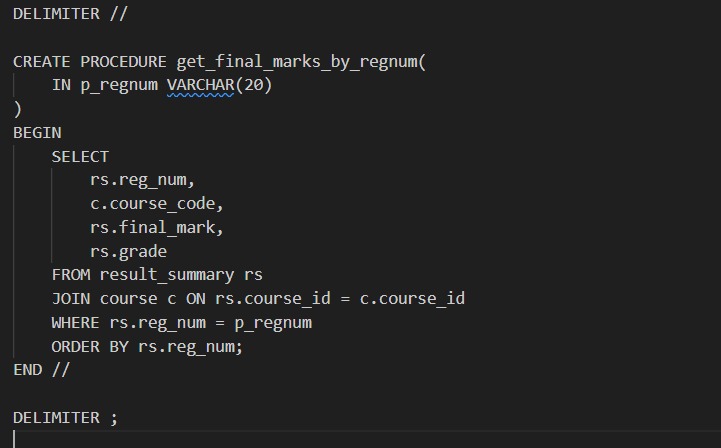
****

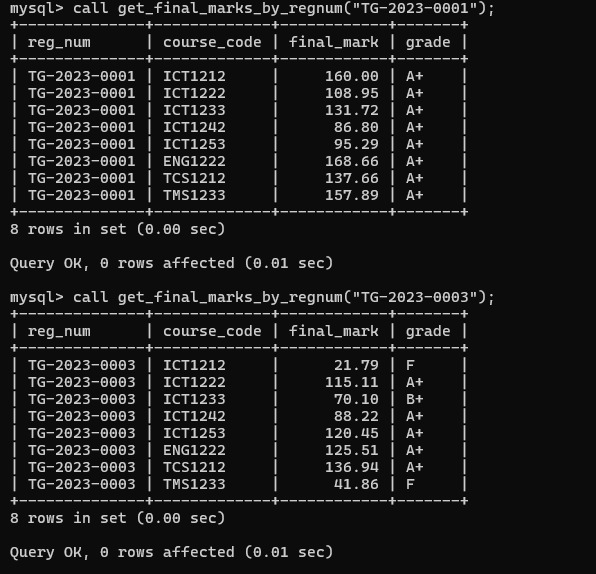
****

****

****

****

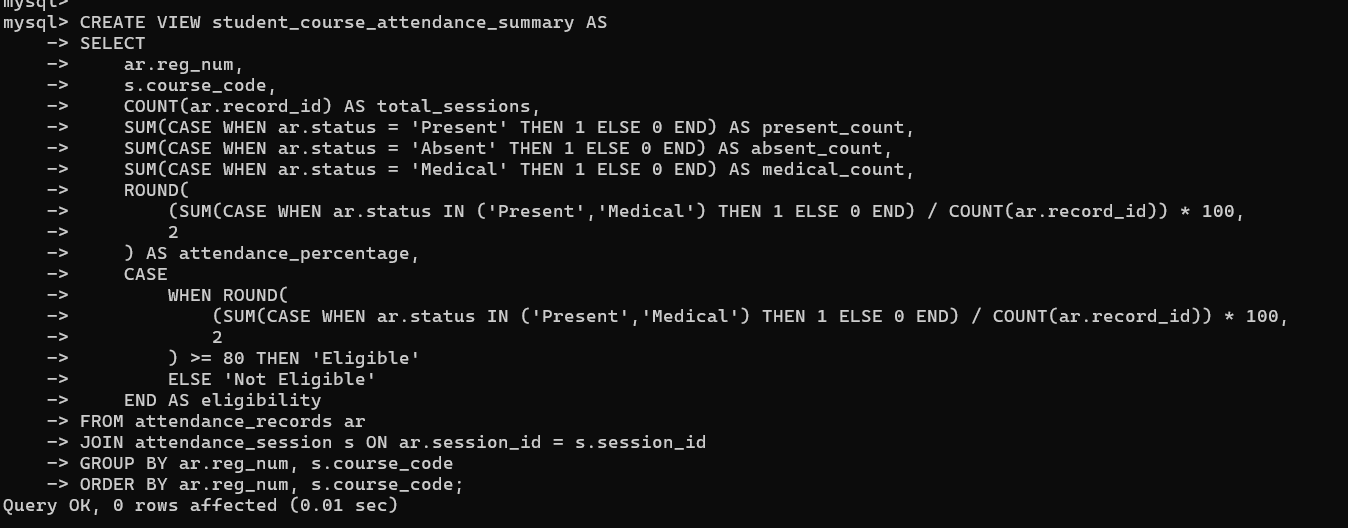
****

****

* Views (for displaying marks/attendance summaries)

A computer screen with white text

AI-generated content may be incorrect.A computer screen with white text

AI-generated content may be incorrect.

* Triggers (optional, for automatic updates)

**A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A computer screen shot of a program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.A screenshot of a computer program

AI-generated content may be incorrect.**

**Problems faced during the development**

During the development of Student Management System, following are the challenges we encountered.

* Database Normalization

Achieving the correct level of normalization while maintaining easy data retrieval without data redundancy

* Privilege Management

Creating users with the right access and prevent privilege escalation or loss of access.

* Handling Special Student cases

Cases like medical, repeat and suspended students were complex.

* Data consistency and relationships

Maintaining referential integrity among related tables.

* Time Management

Testing all functionalities within a limited timeframe.

**Solutions to the problems encountered**

Above challenges were addressed using different solutions:

* Query Optimization
* Stored Procedures and Views
* Normalization & Relational Mapping
* Systematic Privilege Testing
* Continuous Debugging & Validation

**Backend Hosting**

The database was hosted locally using MySQL Server through XAMPP for development and testing.

Hosting Choice:

Local hosting was selected because it provided easy access, full administrative control, and quick testing without internet dependency.  
For future scalability, the system can easily be migrated to a cloud-based MySQL environment

Benefits:

* Faster query execution and debugging during development.
* No external connectivity delays.
* Full database control for user creation and privilege testing.

**Cloud Hosting**

If the system is moved to a cloud environment such as AWS or Azure, a few changes are needed to make it secure and accessible:

* IP Access Control:  
  Only allow trusted computers or networks to connect to the cloud database.
* User Login Security:  
  Use strong passwords and create separate accounts for each user role (Admin, Lecturer, etc.).
* Automatic Backups:  
  Set up daily or weekly backups to protect against data loss.
* Scalability:  
  The system can be easily upgraded if more users or storage are needed in the future.

**Individual Contribution- I**

TG/2023/1753- U.G. Harshana Prabhath

GitHub Acc- [HarshanaPrabhath (Harshana Prabhath)](https://github.com/HarshanaPrabhath)

Gmail- harshanaprabhath147@gmail.com

Contribution Overview:  
Member 1 played a vital role in the implementation and functional development of the Student Management System. Their main responsibilities included creating database tables, inserting sample data, and developing views, stored procedures, and triggers that automated key database functions.

Detailed Contribution:  
Using MySQL Server and Notepad++, Member 1 created the complete set of database tables and ensured that all fields, constraints, and relationships matched the logical model. They entered realistic data for students, lecturers, courses, and marks.  
This member developed SQL views for simplified access to attendance and marks data, ensuring user-specific data retrieval. They also wrote stored procedures to calculate eligibility and grades automatically and implemented triggers to maintain real-time updates when data changed.

Version control and progress tracking were managed through GitHub, which helped maintain consistent development and collaboration among group members. Throughout the process, the member frequently referred to MySQL documentation, W3Schools tutorials, and ChatGPT for query optimization and syntax clarification.

Reflection:  
This member gained strong hands-on experience with SQL scripting, data manipulation, and automation in MySQL. Their contribution enhanced the system’s efficiency and reliability while ensuring that database operations adhered to normalization and consistency rules.

**Individual Contribution- II**

TG/2023/1759- U. Shonali Galpihilla

GitHub Acc- [ShonaUG (Shonali Galpihilla)](https://github.com/ShonaUG)

Gmail- shona20ug@gmail.com

Contribution Overview:  
Member 2 was responsible for the database design and documentation stages of the project. Their main tasks included designing the Enhanced Entity Relationship (EER) Diagram, assisting with table creation, inserting data, and compiling the final project report.

Detailed Contribution:  
Using Draw.io, Member 2 designed the EER diagram to represent entities, attributes, and relationships across the database. This diagram provided a clear visual reference for the database structure and guided the implementation phase.  
They also worked in MySQL Server and Notepad++ to create tables based on the EER design, insert data, and validate relationships. Member 2 led the creation of the project report, combining all sections—including introduction, DRD, ER and relational mapping diagrams, tools, security, and hosting details—into a cohesive and professional academic document.

For version tracking and document sharing, GitHub was used to maintain consistency among the group members. Member 2 also consulted AI tools (ChatGPT), W3Schools, and university-provided study materials to verify syntax, understand best practices, and ensure academic accuracy in documentation.

Reflection:  
Through this contribution, the member developed strong analytical, design, and technical writing skills. Their use of both design tools and documentation platforms ensured that the project met academic and technical standards.

A person in a suit

AI-generated content may be incorrect.**Individual Contribution- III**

TG/2023/1737- K. L. Yasiru Nimsara

GitHub Acc- [klynimsara](https://github.com/klynimsara)

Gmail- kulasinliyanageyasirunimsara@gmail.com

Contribution Overview:  
Member 3’s primary role was to design the Relational Mapping Diagram and help implement the database structure and sample data in MySQL Server. Their work focused on transforming the EER diagram into a fully normalized relational schema.

Detailed Contribution:  
Using Draw.io, Member 3 created the Relational Mapping Diagram, defining how entities from the EER model would translate into relational tables. They identified primary and foreign keys, resolved many-to-many relationships, and ensured normalization up to Third Normal Form (3NF).  
This member used MySQL Server and Notepad++ to create and populate the database tables, then tested relationships using join queries to verify data accuracy. They also helped document the relational mapping process and contributed to testing and validation.

Collaboration and version control were managed through GitHub, ensuring smooth integration of all team contributions. Member 3 used W3Schools, MySQL tutorials, and ChatGPT to refine SQL syntax, understand complex joins, and troubleshoot errors during testing.

Reflection:  
Member 3 strengthened their understanding of database normalization, relational design, and SQL query testing. Their contribution was crucial in ensuring that the database structure was logically sound, efficient, and aligned with the academic requirements of the project.

**References**

We have used different sources to gather information.

* OpenAI
* Lecture Materials
* W3Schools
* MySQL Documentations