College Admission Management System

Project Report

Submitted by: MD FAIZAN HUSSAIN

Internship – JAVA DEVERLOPER

ELEVATE LABS

September 2025

Acknowledgement

I hereby express my deepest gratitude to my internship guide, mentors, and all who have supported and guided me through the successful completion of this College Admission Management System project. Their invaluable knowledge, suggestions, and encouragement helped me accomplish this task.

Abstract

This project implements a College Admission Management System using Java and MySQL, providing an automated solution for handling student registrations, course allocations, merit calculations, and administrative approvals. The system streamlines and simplifies the admission process through a command-line interface, ensuring accuracy, efficiency, and easy management of application data.

Table of Contents

---

1. Introduction

Admissions are a crucial process in any educational institution. Manually handling admissions is error-prone and time-consuming. The College Admission Management System automates these processes, improving data accuracy and optimizing workload. This project provides a menu-driven system to manage student and course records, calculate merit lists, obtain admin approvals, and export admission lists for easy record keeping.

---

2. Objectives

- Automate student registration and course application processes.
- Manage student, course, and application data efficiently.
- Calculate merit lists based on cutoff and marks.
- Allow admin approval or rejection of applications.
- Export admission results in CSV format.
- Provide an intuitive command line interface.

---

3. Tools and Technologies

- Programming Language: Java (JDK 21 LTS)
- Database: MySQL
- Connectivity: JDBC with MySQL Connector/J
- IDE: Visual Studio Code
- Platform: Windows

---

4. System Design

The system consists of modular Java classes handling different functionalities:

- **StudentRegistration:** Manages student data input and database insertion.
- **CoursesManager:** Manages adding and displaying courses.
- **ApplicationManager:** Handles student course applications.
- **MeritCalculator:** Calculates merit list based on marks and course cutoff with seat limits.
- **AdminPanel:** Admin approval or rejection of student applications.
- **ListExporter:** Exports admission lists as CSV files.
- **Main:** Provides a user menu to access all functionalities.

---

5. Database Design

The project uses a relational database with three main tables:

- **Students:** Stores student details (StudentID (PK), Name, DOB, Email, Address, Marks, Status).
- **Courses:** Stores course info (CourseID (PK), CourseName, Cutoff, SeatsAvailable).
- **Applications:** Links students and courses (AppID (PK), StudentID (FK), CourseID (FK), Marks, Status).

Primary and foreign keys enforce data integrity and relationships.

---

6. Implementation

- Established JDBC connection to MySQL with proper driver loading and URL.
- Created console menus for each module using Java Scanner.
- Implemented core methods for insertion, retrieval, update, and export functions.
- Merit calculation consists of querying valid applicants filtered by cutoff and ordering by marks, limiting by available seats.
- Admin approval changes application status dynamically.
- Export module generates a CSV file with the admission list for offline use.

(Code snippets, UML or ER diagrams, and screenshots added in annexure.)

---

7. Testing and Results

- Thorough manual testing was performed on all functionalities including registrations, course addition, application submission, merit calculation, and admin approval processes.
- All inputs were validated for correctness (e.g., valid dates, marks range).
- SQL queries were validated and optimized to avoid syntax errors and foreign key violations.
- Exported CSV files were successfully opened and verified in spreadsheet applications.
- Sample outputs and input scenarios are appended.

---

8. Challenges Faced

- Configuring JDBC connectivity and driver classpath in VS Code environment.
- Handling SQL syntax restrictions on dynamic LIMIT clauses requiring separate queries.
- Managing foreign key constraints between tables during the application process.

- Designing user-friendly command line interface with proper input validation and error handling.
- Exporting data to CSV while handling file IO exceptions.
- Learning curve of integrating Java with MySQL using JDBC API.

---

9. Conclusion

The College Admission Management System successfully digitizes the admission process, ensuring better data management, speed, and integrity. The modular design allows easy maintenance and future extensions. The internship project enhanced practical knowledge of database connectivity, SQL, and Java programming.

---

10. Future Enhancements

- Implement graphical user interface (GUI) with JavaFX or Swing.
- Add authentication and role-based access control for students and admins.
- Integrate email notification system for admission status updates.
- Develop a web-based interface with REST API backend.
- Add reporting dashboards and analytics features.

---

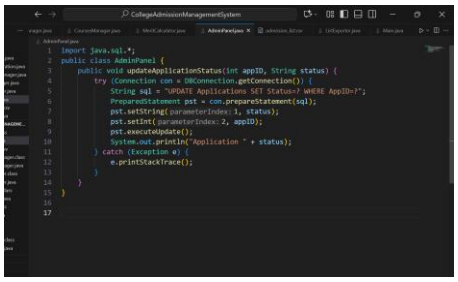
11. References

- Oracle JDBC Documentation
- MySQL 8.x Documentation
- Java Tutorials for JDBC and Database Connectivity
- Online forums and tutorials for Java and MySQL Integration

---

12. Annexures

- Screenshots of application workflow and SQL query results.
- 
- Key code snippets for main modules.

- 
- 
- SQL script (schema.sql) for database creation.

```
    Marks INT,
    Status VARCHAR(20),
    FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
    FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);

select * from students;
select * from courses;
select * from Applications;
```

AppID	StudentID	CourseID	Marks	Status
2	1	1	95	Approved
3	2	1	87	Approved

-

13. Workflow Diagram

- Insert a flowchart or diagram illustrating the sequence of operations in your system:
    - Student registration → Course addition → Application submission → Merit list calculation → Admin approval → Export results
- You can create this with tools like draw.io, Microsoft Visio, or even hand-drawn and scanned.
- This visually explains the process logic and data flow in your system.

---

14. User Manual

- Provide step-by-step instructions for each feature or menu option, including sample inputs and expected outputs.
- For example:
    - How to register a student (inputs required, sample data)
    - How to add a course
    - How to apply for a course
    - How admin approves/rejects
    - How to export the admission list

- Include screenshots of the console menus and sample data entry if possible.
- This helps users operate the system easily without external help.