Student Information System

(SYS; with id_card)

Project Report

Table of Contents

1. Introduction

In the era of digital transformation, educational institutions require efficient systems to manage their data. The Student Information System (SIS) is developed to automate and streamline the management of student records, course enrollments, and academic results. The system offers role-based access, ensuring that staff and students have the appropriate levels of visibility and control over the data.

2. Problem Statement

Manual record-keeping in educational institutions leads to inefficiencies, errors, and security issues.

Institutions face difficulties in:

- Managing a large number of student records.
- Tracking course enrollments and grades efficiently.
- Ensuring data security and restricted access.

This project aims to provide a centralized, secure, and scalable Student Information System.

3. Database Design and Implementation

The system was designed based on an Extended Entity-Relationship (EER) diagram outlining the relationships between Students, Courses, Enrollments, and Grades.

Steps:

- Created EER diagram to identify entities and relationships.
- Translated the EER model into SQL tables.
- Ensured referential integrity through primary and foreign keys.

Main tables:

- Users (for authentication, roles)
- Students
- Courses
- Enrollments
- Grades

MySQL Workbench was used to design, model, and implement the database.

4. Application Development

The application was developed using Python and MySQL.

Key aspects:

- Environment configuration using a .env file to secure database credentials.
- Database connection handling in a config.py file.
- Main logic in app.py: login system, CRUD operations, and role-based access control.

User Authentication:

- Staff/Admin users can view and modify all student information.
- Students can log in and view only their personal information and grades.

5. Results

The project successfully met its objectives:

- Staff users can create, update, view, and delete student records, courses, and grades.
- Students have restricted access to only their enrolled courses and academic records.
- Database operations are efficient and secure.
- The application is functional and user-friendly via the terminal.

Testing confirmed that access restrictions and data modifications work as intended.

6. Conclusion

The Student Information System provides a scalable, secure, and efficient way to manage student and course data. It significantly reduces administrative workload, minimizes errors, and enhances data security.

Future improvements could include:

- Developing a full web-based interface (using Flask or Django).
- Implementing password hashing for enhanced security.
- Adding reporting features like GPA calculation and detailed transcripts.

7. Appendices

Appendix A: Project Directory Structure

```
----------------------------------------
student_info_system/
|-- app.py
|-- config.py
|-- .env
|-- requirements.txt
```

Appendix B: Example of .env file

```
--------------------------------
DB_USER=root
DB_PASSWORD=your_password
DB_PORT=3306
```

Appendix C: Sample Database Tables

- students(id, first_name, last_name, email)
- courses(id, name, description)
- enrollments(id, student_id, course_id, enrollment_date)
- grades(id, enrollment_id, grade)

Appendix D: Sample Code Snippet

```
import mysql.connector
from config import db_config

connection = mysql.connector.connect(**db_config)
cursor = connection.cursor()
cursor.execute('SELECT * FROM students')
for row in cursor.fetchall():
    print(row)
connection.close()
```