INTRODUCTION

Welcome to the Student Registration System project! This system is designed to efficiently manage student information, course enrollment, and registration processes for a university. The project aims to streamline the academic operations, ensuring a smooth and hassle-free experience for both students and faculty members.

The Student Registration System offers various functionalities to handle student management, course management, course enrollment, transcript generation, and progress reporting. With this system, students can easily enroll in courses each semester, and administrators can track students' academic progress seamlessly.

This presentation/documentation will take you through the database design, SQL queries used to implement different functionalities, and an overview of the system's features. We will also discuss the challenges faced during development and the testing process to ensure the system's accuracy and reliability.

Let's dive into the details of the Student Registration System and explore how it can enhance the university's academic management.

Database Design:

The database design is a crucial aspect of the Student Registration System, as it forms the foundation for storing and managing student information, course details, and enrollment data. A well-structured database ensures data integrity, efficient data retrieval, and seamless operations within the system.

- Students Table:

This table stores information about the students enrolled in the university.

Columns: StudentID (Primary Key), Name, Email, ContactDetails, AcademicRecords.

- Courses Table:

This table maintains a catalog of available courses offered by the university.

Columns: CourseID (Primary Key), Title, Department, CreditHours, Prerequisites.

- Enrollment Table:

This table records the course enrollments made by students each semester.

Columns: EnrollmentID (Primary Key), StudentID (Foreign Key), CourseID (Foreign Key), Semester EnrollmentDate.

➢ Relationships:

One-to-Many Relationship (Students to Enrollment):

The "Students" table has a one-to-many relationship with the "Enrollment" table.

Each student can have multiple enrollments, but each enrollment belongs to only one student.

This relationship is established through the "StudentID" column in the "Students" table, which acts as a foreign key in the "Enrollment" table.

One-to-Many Relationship (Courses to Enrollment):

The "Courses" table has a one-to-many relationship with the "Enrollment" table.

Each course can have multiple enrollments, but each enrollment corresponds to only one course.

This relationship is established through the "CourseID" column in the "Courses" table, which acts as a foreign key in the "Enrollment" table.

➢Primary and Foreign Keys:

In the "Students" table, the "StudentID" column serves as the primary key, uniquely identifying each student record.

In the "Courses" table, the "CourseID" column serves as the primary key, uniquely identifying each course record.

In the "Enrollment" table, the "EnrollmentID" column serves as the primary key, uniquely identifying each enrollment record.

The "StudentID" and "CourseID" columns in the "Enrollment" table act as foreign keys, establishing the relationships with the "Students" and "Courses" tables, respectively.

SQL Queries:

STUDENT TABLE QUERY

```
CREATE TABLE StudentsTBL (
    StudentID INT PRIMARY KEY,
    Name VARCHAR(100),
    Email VARCHAR(100),
    ContactDetails VARCHAR(100),
    AcademicRecords VARCHAR(255)
);
```

COURSE TABLE QUERY

```
CREATE TABLE CoursesTBL (
    CourseID INT PRIMARY KEY,
    Title VARCHAR(100),
    Department VARCHAR(100),
    CreditHours INT,
    Prerequisites VARCHAR(100)
);
```

ENRONMENT TABLE QUERY

```
CREATE TABLE EnrollmentTBL (
   EnrollmentID INT PRIMARY KEY,
   StudentID INT,
   CourseID INT,
   EnrollmentDate Date,
   Semester VARCHAR(50),
   FOREIGN KEY (StudentID) REFERENCES StudentsTBL(StudentID),
   FOREIGN KEY (CourseID) REFERENCES CoursesTBL(CourseID)
);
```

INSERT INTO STUDENT TABLE QUERY

```
INSERT INTO StudentsTBL ( StudentID, Name,
Email, ContactDetails, AcademicRecords )

VALUES (1, 'Thomas Antwi',
'thomlee123@gmail.com', '0597537575',
'Excellent');
```

INSERT INTO COURSE TABLE QUERY

```
INSERT INTO CoursesTBL ( CourseID, Title,
Department, CreditHours, Prerequisites )

VALUES (101, 'Introduction DBMS', 'Computer
Science And Engineering', 3, 'None');
```

ENROLL STUDENT IN A COURSE QUERY

```
INSERT INTO EnrollmentTBL ( EnrollmentID,
StudentID, CourseID, Semester, EnrollmentDate )
VALUES (113, 1, 101, 'Sem2', DATE());
```

DELETE STUDENT DETAILS QUERY

```
DELETE *
FROM StudentsTBL
WHERE StudentID = 1;
```

DELETE COURSE DATA QUERY

```
DELETE *
FROM CoursesTBL
WHERE CourseID = 101;
```

UDATE STUDENT DETAIL QUERY

```
UPDATE StudentsTBL SET Email =
'thomas12@gmail.com'
WHERE StudentID = 1;
```

UDATE COURSE DATA QUERY

```
UPDATE CoursesTBL SET Title = 'Web
Programming'
WHERE CourseID = 102;
```

READ STUDENT INFORMATION QUERY

```
SELECT *
FROM StudentsTBL;
```

READ COURSE DETAILS QUERY

```
SELECT *
FROM CoursesTBL;
```

STUDENT TRANSCRIPT QUERY

```
SELECT StudentsTBL.StudentID, StudentsTBL.Name,
CoursesTBL.Title, EnrollmentTBL.EnrollmentDate,
CoursesTBL.CreditHours

FROM (StudentsTBL INNER JOIN EnrollmentTBL ON
StudentsTBL.StudentID = EnrollmentTBL.StudentID)
INNER JOIN CoursesTBL ON EnrollmentTBL.CourseID =
CoursesTBL.CourseID

WHERE StudentsTBL.StudentID = 1;
```

STUDENT PROGRESS REPORT QUERY

```
SELECT StudentsTBL.Name, CoursesTBL.Title,
EnrollmentTBL.EnrollmentDate,
CoursesTBL.CreditHours

FROM (StudentsTBL INNER JOIN EnrollmentTBL ON
StudentsTBL.StudentID = EnrollmentTBL.StudentID)
INNER JOIN CoursesTBL ON EnrollmentTBL.CourseID =
CoursesTBL.CourseID

WHERE StudentsTBL.StudentID = 1;
```

COURSE ENRONMENT REPORT QUERY

```
SELECT CoursesTBL.Title,
COUNT(EnrollmentTBL.StudentID) AS
NumOfStudentsEnrolled

FROM CoursesTBL LEFT JOIN EnrollmentTBL ON
CoursesTBL.CourseID = EnrollmentTBL.CourseID

GROUP BY CoursesTBL.Title;
```

CONCLUSION:

The Student Registration System project has been successfully developed, providing an efficient and comprehensive solution for managing student information, course enrollment, and academic records at our university. Throughout the development process, we focused on creating a well-structured database, writing SQL queries for various functionalities, and ensuring seamless system functionality.

BY THOMAS ANTWI
SRI.41.008.038.22
CE(B)