Project Report

On

Student Management System

(UCS-310)

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Submitted By:

Mayank Aggarwal	**102103350**
Saksham Mutneja	**102103359**
Samarth Thakur	**102103364**
Naman Goyal	**102153001**
Shivam Gupta	**102283026**

Submitted to:

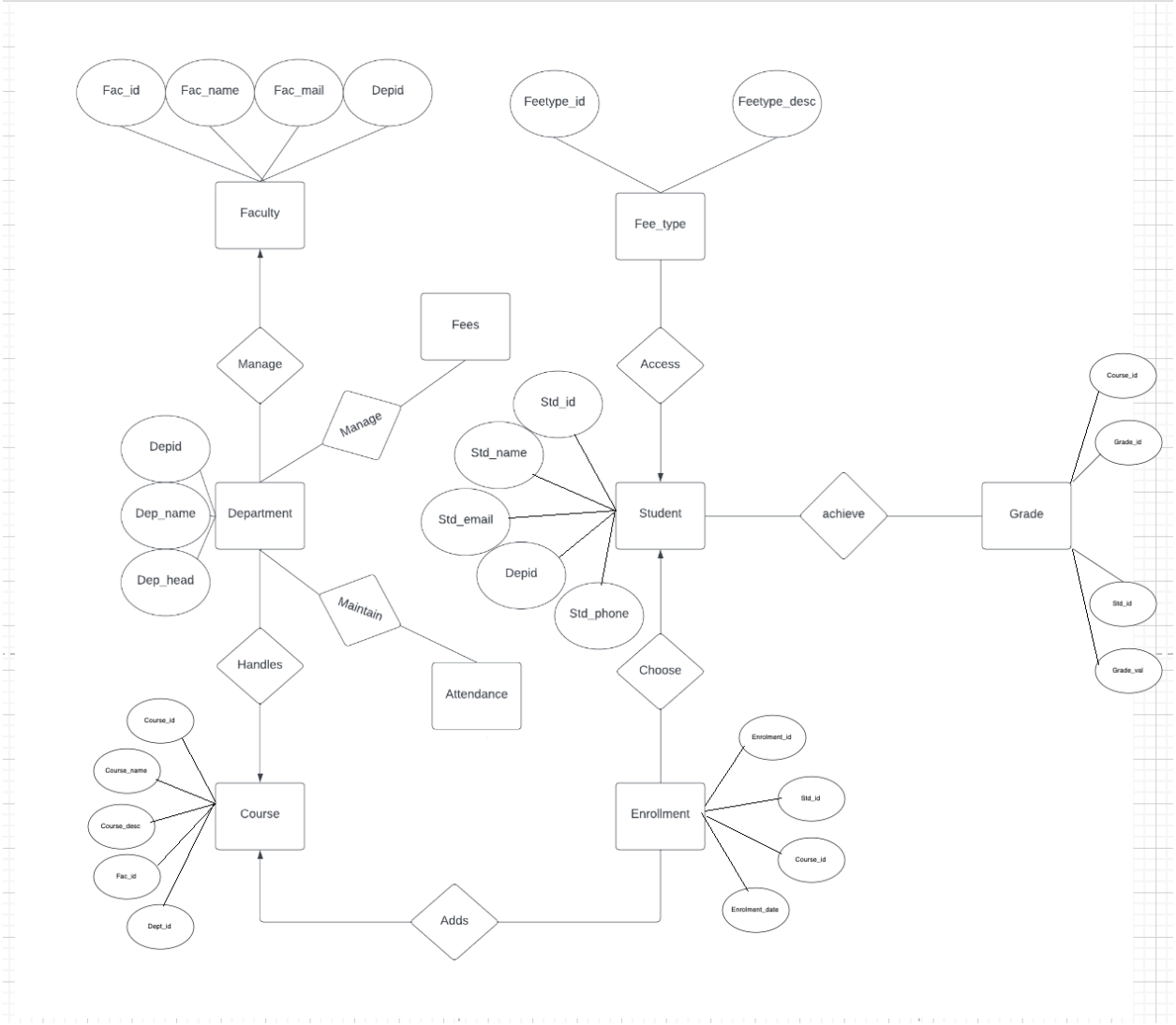
Dr Radhika Bansal

INDEX

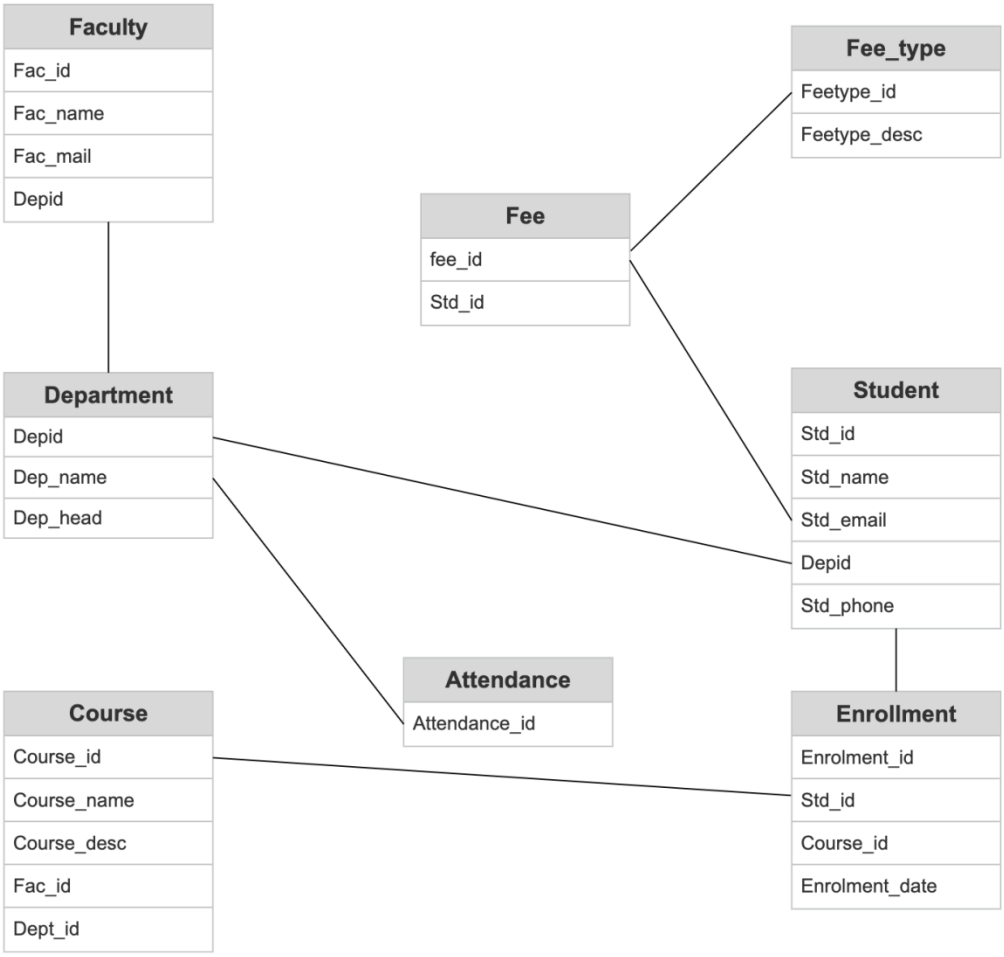
Problem Statement

To develop a system that can efficiently and accurately identify students by their facial features and retrieve their academic and personal information from a database. The system should be able to capture and store the images of students, as well as their personal and academic data, in a secure and organized manner.

Entity-Relationship Diagram



Normalized Table



Creating Table

```sql
CREATE TABLE department (
  department_id NUMBER(10) PRIMARY KEY,
  department_name VARCHAR2(50) NOT NULL,
  department_head VARCHAR2(50) NOT NULL
);

CREATE TABLE student (
  student_id NUMBER(10) PRIMARY KEY,
  student_name VARCHAR2(50) NOT NULL,
  student_email VARCHAR2(50) NOT NULL,
  student_phone VARCHAR2(20) NOT NULL,
  department_id NUMBER(10) REFERENCES department(department_id)
);
```

```sql
CREATE TABLE faculty (
  faculty_id NUMBER(10) PRIMARY KEY,
  faculty_name VARCHAR2(50) NOT NULL,
  faculty_email VARCHAR2(50) NOT NULL,
  department_id NUMBER(10) REFERENCES department(department_id)
);

CREATE TABLE course (
  course_id NUMBER(10) PRIMARY KEY,
  course_name VARCHAR2(50) NOT NULL,
  course_description VARCHAR2(200),
  faculty_id NUMBER(10) REFERENCES faculty(faculty_id),
  department_id NUMBER(10) REFERENCES department(department_id)
);
```

```sql
CREATE TABLE enrollment (
  enrollment_id NUMBER(10) PRIMARY KEY,
  student_id NUMBER(10) REFERENCES student(student_id),
  course_id NUMBER(10) REFERENCES course(course_id),
  enrollment_date DATE NOT NULL
);

CREATE TABLE attendance (
  attendance_id NUMBER(10) PRIMARY KEY,
  student_id NUMBER(10) REFERENCES student(student_id),
  course_id NUMBER(10) REFERENCES course(course_id),
  attendance_date DATE NOT NULL,
  attendance_status VARCHAR2(10) NOT NULL
);
```

```sql
CREATE TABLE grade (
  grade_id NUMBER(10) PRIMARY KEY,
  student_id NUMBER(10) REFERENCES student(student_id),
  course_id NUMBER(10) REFERENCES course(course_id),
  grade_value NUMBER(3,2) NOT NULL
);

CREATE TABLE fee_type (
  fee_type_id NUMBER(10) PRIMARY KEY,
  fee_type_description VARCHAR2(50) NOT NULL
);
```

```sql
CREATE TABLE fees (
  fee_id NUMBER(10) PRIMARY KEY,
  student_id NUMBER(10) REFERENCES student(student_id),
  fee_type_id NUMBER(10) REFERENCES fee_type(fee_type_id),
  fee_amount NUMBER(10,2) NOT NULL,
  fee_date DATE NOT NULL
);
```

SQL Output Screenshots

Table-1
COURSE

COURSE_ID	COURSE_NAME	COURSE_DESCRIPTION	FACULTY_ID	DEPARTMENT_ID
1	Introduction to Computer Science	An overview of the fundamentals of computer science	1	101
2	Marketing Principles	An introduction to basic marketing concepts and strategies	2	102
3	Abnormal Psychology	An examination of psychological disorders and their treatments	3	103
4	Evolutionary Biology	An exploration of the mechanisms and patterns of evolution	4	104

Table-2
DEPARTMENT

DEPARTMENT_ID	DEPARTMENT_NAME	DEPARTMENT_HEAD
101	Computer Science	Dr. John Smith
102	Business Administration	Dr. Sarah Johnson
103	Psychology	Dr. Michael Brown
104	Biology	Dr. Maria Garcia
105	English	Dr. Emily Thomas

Table-3
STUDENT

STUDENT_ID	STUDENT_NAME	STUDENT_EMAIL	STUDENT_PHONE	DEPARTMENT_ID
1006	Alice Smith	alice.smith@email.com	555-1234	101
1007	Bob Johnson	bob.johnson@email.com	555-5678	102
1008	Carla Ramirez	carla.ramirez@email.com	555-9012	103
1009	David Lee	david.lee@email.com	555-3456	101
1010	Emily Chen	emily.chen@email.com	555-7890	102
1011	Frank Martin	frank.martin@email.com	555-2345	104
1012	Gina Rodriguez	gina.rodriguez@email.com	555-6789	103
1013	Henry Wong	henry.wong@email.com	555-0123	101
1014	Isabel Martinez	isabel.martinez@email.com	555-4567	102
1015	Jack Thompson	jack.thompson@email.com	555-8901	104
1016	Katie Brown	katie.brown@email.com	555-1234	101

TABLE-4

FACULTY

FACULTY_ID	FACULTY_NAME	FACULTY_EMAIL	DEPARTMENT_ID
1	John Doe	jdoe@university.edu	101
2	Jane Smith	jsmith@university.edu	102
3	Michael Johnson	mjohnson@university.edu	102
4	Maria Garcia	mgarcia@university.edu	104

TABLE-5
FEE

FEE_ID	STUDENT_ID	FEE_TYPE_ID	FEE_AMOUNT	FEE_DATE
1	1006	1	100	01-MAR-23
2	1006	2	50	01-MAR-23
3	1007	1	100	15-MAR-23
4	1007	2	50	15-MAR-23
5	1008	1	100	01-APR-23
6	1008	2	50	01-APR-23
7	1009	1	100	15-APR-23
8	1009	2	50	15-APR-23
9	1010	1	100	01-MAR-23
10	1010	2	50	01-MAR-23

TABLE-6
FEE TYPE

FEE_TYPE_ID	FEE_TYPE_DESCRIPTION
1	Tuition Fee
2	Library Fee
3	Exam Fee

TABLE-7
ENROLLMENT

ENROLLMENT_ID	STUDENT_ID	COURSE_ID	ENROLLMENT_DATE
1	1006	1	01-SEP-22
2	1007	2	01-SEP-22
3	1008	3	01-SEP-22
4	1009	1	01-SEP-22
5	1010	2	01-SEP-22
6	1011	4	01-SEP-22
7	1012	3	01-SEP-22
8	1013	1	01-SEP-22
9	1014	4	01-SEP-22
10	1015	4	01-SEP-22
11	1016	1	01-SEP-22

TABLE-8

GRADE

GRADE_ID	STUDENT_ID	COURSE_ID	GRADE_VALUE
1	1006	1	3.5
2	1007	2	4
3	1008	3	3.7
4	1009	1	3.2
5	1010	2	4
6	1011	4	3.9
7	1012	3	3.5

13.

PL/SQL Codes and Screenshots

1.TRIGGER-

```
CREATE TABLE student (
  student_id NUMBER(10) PRIMARY KEY,
  student_name VARCHAR2(50) NOT NULL,
  student_email VARCHAR2(50) NOT NULL,
  student_phone VARCHAR2(20) NOT NULL,
  department_id NUMBER(10) REFERENCES department(department_id)
);
CREATE OR REPLACE TRIGGER trg_student_name_capitalized
BEFORE INSERT OR UPDATE ON student
FOR EACH ROW
BEGIN
  :new.student_name := INITCAP(:new.student_name);
END;

INSERT INTO student (student_id, student_name, student_email, student_phone, department_id)
VALUES (1006, 'Alice Smith', 'alice.smith@email.com', '555-1234', 101);

INSERT INTO student (student_id, student_name, student_email, student_phone, department_id)
VALUES (1007, 'Bob Johnson', 'bob.johnson@email.com', '555-5678', 102);
```

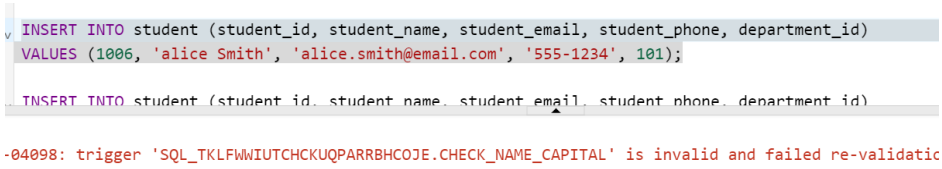
14.

INSERT INTO student (student_id, student_name, student_email, student_phone, department_id)
VALUES (1008, 'Carla Ramirez', 'carla.ramirez@email.com', '555-9012', 103);

INSERT INTO student (student_id, student_name, student_email, student_phone, department_id)
VALUES (1009, 'David Lee', 'david.lee@email.com', '555-3456', 101);

```
CREATE OR REPLACE TRIGGER trg_student_name_capitalized
BEFORE INSERT OR UPDATE ON student
FOR EACH ROW
BEGIN
  :new.student_name := INITCAP(:new.student_name);
END;
```

OUTPUT-



15.

2.PROCEDURE-

```
CREATE OR REPLACE PROCEDURE get_course_grades(
   course_id IN NUMBER,
   grades OUT SYS_REFCURSOR
)
AS
BEGIN
   OPEN grades FOR
      SELECT *
      FROM grade
      WHERE course_id = course_id;
END;
```

OUTPUT-

16.

```
CREATE OR REPLACE PROCEDURE get_course_grades(
    course_id IN NUMBER,
    grades OUT SYS_REFCURSOR
)
AS
BEGIN
    OPEN grades FOR
        SELECT *
        FROM grade
        WHERE course_id = course_id;
END;

INSERT INTO grade (grade_id, student_id, course_id, grade_value)
```

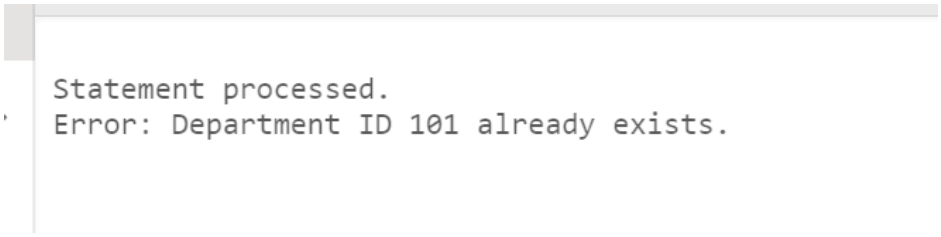
3.EXCEPTION-

```
BEGIN
  INSERT INTO department (department_id, department_name,
department_head)
  VALUES (106, 'Mathematics', 'Dr. William Brown');
EXCEPTION
  WHEN DUP_VAL_ON_INDEX THEN
```

17.

```
   DBMS_OUTPUT.PUT_LINE('Error: Department ID 106 already
exists.');
END;
```

OUTPUT-



4.CURSOR-

```
DECLARE
  CURSOR dept_faculty_cursor IS
    SELECT d.department_name, f.faculty_name
    FROM department d
    INNER JOIN faculty f ON d.department_id = f.department_id
    ORDER BY d.department_name;

  dept_name department.department_name%TYPE;
  faculty_name faculty.faculty_name%TYPE;
BEGIN
  OPEN dept_faculty_cursor;
```

18.

```
  LOOP
    FETCH dept_faculty_cursor INTO dept_name, faculty_name;
    EXIT WHEN dept_faculty_cursor%NOTFOUND;

    DBMS_OUTPUT.PUT_LINE('Department: ' || dept_name || ',
Faculty: ' || faculty_name);
  END LOOP;

  CLOSE dept_faculty_cursor;
END;
```

OUTPUT-

```
Statement processed.
Department: Biology, Faculty: Maria Garcia
Department: Business Administration, Faculty: Jane Smith
Department: Business Administration, Faculty: Michael Johnson
Department: Computer Science, Faculty: John Doe
```