

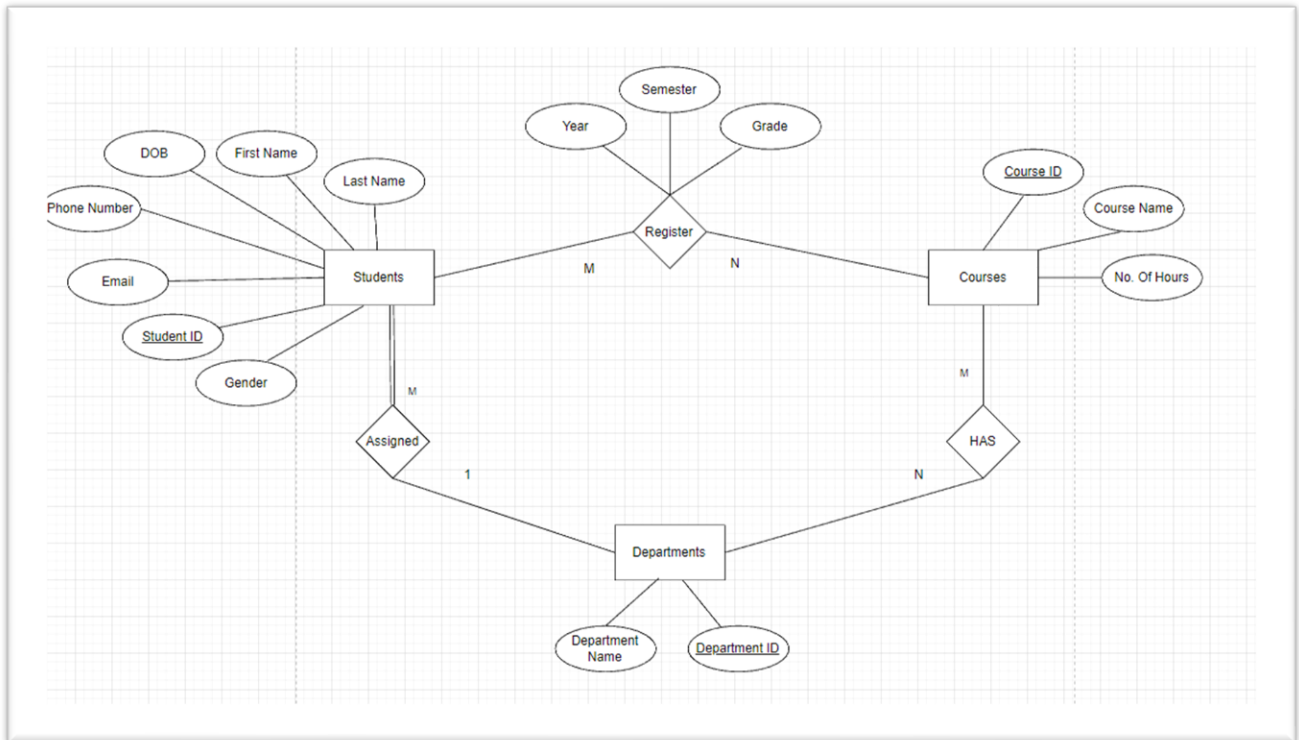


# University Management System

Omar Ashraf - Data Management Case Study

# Database Design

ERD:



- Students data stored in table students.
- Departments data stored in table departments.
- Courses data stored in table courses.
- Many students must be assigned in only one department and one department may have many students.
- Many courses may be assigned in many departments and many departments may have many courses.
- Many students may register for many courses and many courses may be registered by many students.
- Students can register the course one if he didn't succeeded yet he can't register the course again.
- If student fail the course he can register the course again in different year or semester
- If student pass the course he can not register the course again in different year

## Database Schema:

### 1. Students Table:

- StudentID (Primary Key)
- FirstName (Not Null)
- LastName (Not Null)
- Email (Not Null, Unique)
- DOB (Not Null)
- Gender (Not Null)
- PhoneNumber (Not Null)
- DepartmentID (Foreign Key: Departments (DepartmentID))

### 2. Departments Table:

- DepartmentID (Primary Key)
- DepartmentName (Unique, Not Null)

### 3. Courses Table:

- CourseID (Primary Key)
- CourseName (Not Null, Unique)
- NoOfHours (Not Null)

### 4. DepartmentCourses Table:

- DepartmentID (Foreign Key: Departments (DepartmentID))
- CourseID (Foreign Key: Courses (CourseID))
- Primary Key (DepartmentID, CourseID)

### 5. StudentRegisterCourse Table:

- StudentID (Foreign Key: Students (StudentID))
- CourseID (Foreign Key: Courses (CourseID))
- Year (Not Null)
- Semester (Not Null)
- Grade (Not Null)
- Primary Key (StudentID, CourseID, Year, Semester)

# SQL Implementation

```
create table students(  
  StudentID number primary key,  
  FirstName varchar2(50) not null,  
  LastName varchar2(50) not null,  
  Email varchar2(100) not null unique,  
  DOB date not null,  
  Gender char(1) not null,  
  PhoneNumber number not null,  
  DepartmentID number references Departments(DepartmentID) on delete set null  
);
```

```
create table departments(  
  DepartmentID number primary key,  
  DepartmentName varchar2(50) unique not null  
);
```

```
create table courses(  
  CourseID number primary key,  
  CourseName varchar2(50) unique not null,  
  NoOfHours number not null  
);
```

```
create table DepartmentCourses(  
  DepartmentID number references Departments(DepartmentID) on delete cascade,  
  CourseID number references Courses(CourseID) on delete cascade,  
  primary key (DepartmentID,CourseID)  
);
```

```
create table StudentRegisterCourse(  
  StudentID number references Students(StudentID) on delete cascade,  
  CourseID number references Courses(CourseID) on delete cascade,  
  Year number not null,  
  Semester number not null,  
  Grade number default null,  
  primary key(StudentID,CourseID,Year,Semester)  
);
```

# PLSQL Implementation

## Procedures:

### 1. updateStudent

- **Purpose:** Updates information for a specific student based on the provided parameters.
- **Parameters:**
  - V\_StudentID **number**
  - V\_FirstName **varchar2**
  - V\_LastName **varchar2**
  - V\_Email **varchar2**
  - V\_DOB **date**
  - V\_Gender **char**
  - V\_PhoneNumber **number**
  - V\_DepartmentID **number**

## Triggers:

### 1. checkCourseInsert

- **Purpose:** it prevents that the student cannot register the same course again if he passed it and if he failed the course, he can register it again but in different year or semester

### 2. Courses\_TRG

- **Purpose:** an identity column along with **COURSES\_SEQ** for table courses

### 3. Students\_TRG

- **Purpose:** an identity column along with **STUDENTS\_SEQ** for table students

### 4. Departments\_TRG

- **Purpose:** an identity column along with **DEPARTMENTS\_SEQ** for table Departments

## Functions:

### 1. calculateStudentGPA:

- **Purpose:** Calculate GPA for specific student



- **Parameters:** V\_StudentID **number**

## 2. calculateStudentLevel:

- **Purpose:** Calculate Level for specific student
- **Parameters:** V\_StudentID **number**

## 3. calculateCourseGrade:

- **Purpose:** Calculate Grade in terms of (A, B, C, D, F) for a specific course done
- **Parameters:** v\_gradeNum **number**

## 4. calculateCourseGPA:

- **Purpose:** Calculate GPA for specific course
- **Parameters:** v\_courseID **number**

## 5. calculateCoursePassedNumber:

- **Purpose:** Calculate Number of passed students in specific course
- **Parameters:** v\_courseID **number**

## 6. calculateCourseFailedNumber:

- **Purpose:** Calculate Number of failed students in specific course
- **Parameters:** v\_courseID **number**

# Automation Script

## 1. Disk Space Monitoring:

- **Purpose:** Monitors disk space usage and sends an alert if it exceeds a specified threshold.
- **Functionality:**
  - **Threshold:**  
Defines a threshold (e.g., 50%) for disk space.
  - **Check Disk Space:**  
Checks current disk space usage.
  - **Check Threshold:**  
If the current disk space above the threshold, logs a warning; otherwise, logs an informational message.
  - **Log File:**  
Appends status messages to a log file for future reference.

```
#!/bin/bash

log_file="D:/ITI/Case Study/Bash Scripts/disk_log.txt"

threshold=50

disk_space=`df -h | awk 'NR==2 {print $6}' | sed 's/%//`
echo $disk_space

if [[ $disk_space -gt $threshold ]]; then
    echo "Warning Disk Space Is Above $threshold%" >> /D/ITI/Case\ Study/Bash\ Scripts/disk_log.txt
else
    echo "Disk Space Is Under $threshold%" >> /D/ITI/Case\ Study/Bash\ Scripts/disk_log.txt
fi
```

## 2. Database Backup:

- **Purpose:** Perform a full backup of the database.
- **Functionality:**
  - **Database Connection:**

Specifies database connection details using variables like DB\_USER, DB\_PASSWORD, and DB\_SID.
  - **Date Formatting:**

Uses the date command to generate a timestamp in the format "YYYYMMDD\_HHMMSS" and assigns it to DATE\_FORMAT.
  - **Export File Naming:**

Constructs the export file name with the format "backup\_YYYYMMDD\_HHMMSS.dmp" using the previously generated timestamp.
  - **Database Export:**

Utilizes the expdp command to perform a full database export. Connects to the specified database using the provided credentials. Specifies the data pump directory and the dump file to store the exported data.
  - **Status Check:**

Checks the exit status of the expdp command using \$?.

```
#!/bin/bash

DB_USER=University
DB_PASSWORD=123
DB_SID=XE

DATE_FORMAT=$(date +"%Y%m%d_%H%M%S")

EXPORT_FILE="backup_${DATE_FORMAT}.dmp"

expdp ${DB_USER}/${DB_PASSWORD}@${DB_SID} DIRECTORY=DATA_PUMP_DIR DUMPFILE=${EXPORT_FILE} FULL=Y

if [ $? -eq 0 ]; then
    echo "Database backup successful. File: ${EXPORT_FILE}"
else
    echo "Error: Database backup failed."
fi
```



# Java Application Development

## Overview:

The University Management System allows admins to perform operations such as adding, updating, and selecting information related to students, courses, grades, and departments. It also supports the assignment of courses to students.

## Code Structure:

- **University Folder:** have the scenes, controllers and images used in the application.
- **DTO Folder:** have classes to use in the application each class use OOP concepts.
- **CSS Folder:** have css files that are used for the gui.
- **Database Folder:** have the DataAccessLayer gets the data from database, also it has class called ConnectionSingleton that used to handle database connection with database that it works as singleton design pattern.

## We Have 7 Scenes:

- **Login As Administrator**
- **Home Scene**
- **Students Scene**
- **Departments Scene**
- **Courses Scene**
- **Register Course Scene**
- **Grades Scene**

```

public class ConnectionSingleton {
    public static Connection connectDB() {
        try {
            DriverManager.registerDriver(new OracleDriver());
            Connection conn = DriverManager.getConnection(
                "jdbc:oracle:thin:@localhost:1521:XE",
                "University", "123");
            return conn;
        } catch (SQLException e) {
            e.printStackTrace(); // Handle the exception appropriately
        }
        return null;
    }
}

```

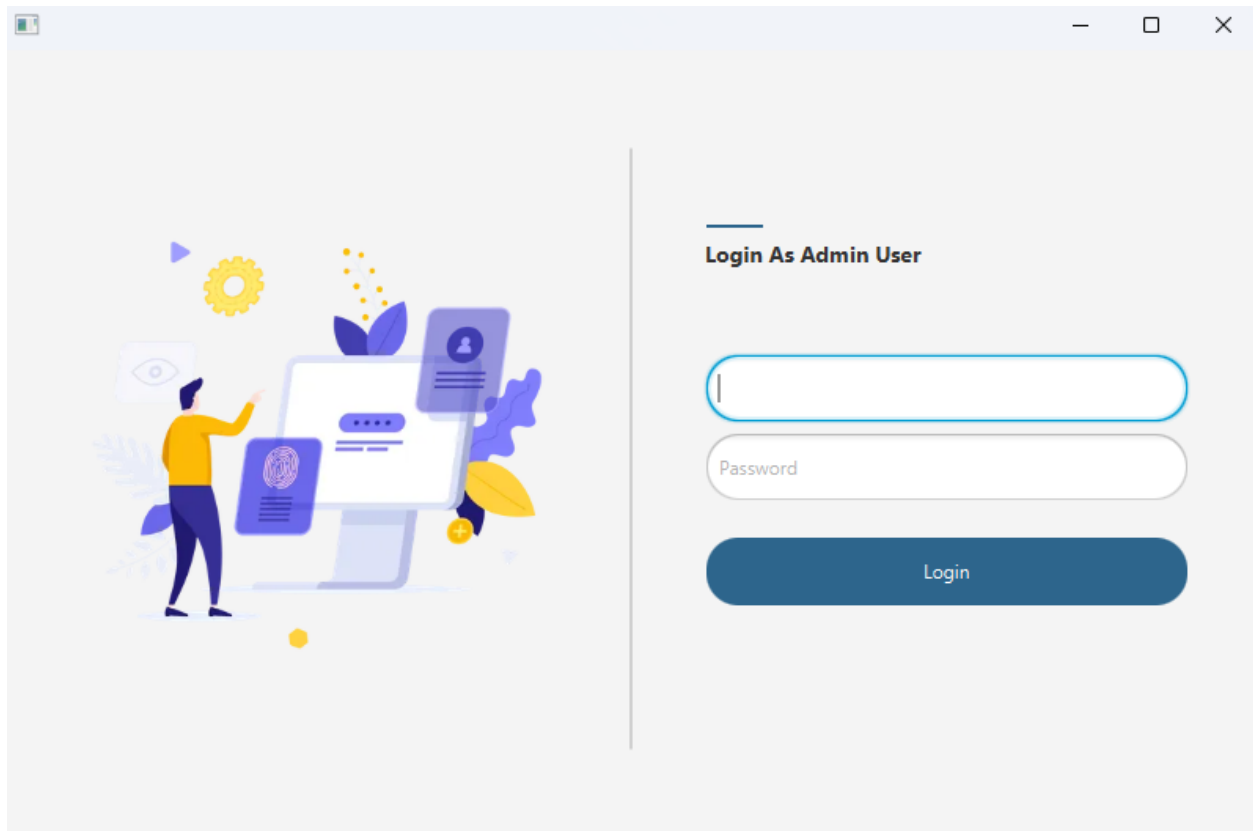
This class employs **Singleton design pattern** to make only one connection and it has method **connection** as a **static** method so it can be called in the **DataAccessLayer**.

For the **DataAccessLayer** it has all the methods that used with the database like(Read, Create, Delete, Update)

```
33 public class DataAccessLayer {
34
35     public static boolean checkLogin(String username , String password , Connection conn){...
60
61
62     public static ObservableList<StudentDTO> getStudentData(Connection conn){...
95
96
97     public static ObservableList<StudentDTO> getStudentCoursesData(Connection conn){...
132
133
134     public static ObservableList<String> getDepartmentNamesData(Connection conn){...
152
153
154     public static ObservableList<DepartmentDTO> getDepartmentData(Connection conn){...
193
194
195     public static ObservableList<CourseDTO> getCourseData(Connection conn){...
213
214
215     /*public static ObservableList<String> getStudentCoursesData(int studentID,Connection conn){...
244
245
246     public static ObservableList<RegisterCourseDTO> getRegisterCourseData(Connection conn){...
264
265
266     public static HomeReportDTO getHomeReportData(Connection conn){...
321
322
323
324     public static int addStudent(String firstName, String lastName , String email , Date DOB , String gender , long phoneNumber , String departmentName , Connection conn){...
369
370
371     public static int updateStudent(int studentID, String firstName, String lastName , String email , Date DOB , String gender , long phoneNumber , String departmentName , Connection conn){...
415
416
417     public static int deleteStudent(int studentID, Connection conn){...
433
434
435     public static int addDepartmentCourse(int departmentID, int courseID, Connection conn){...
463
464
465     public static int deleteDepartmentCourse(int departmentID, int courseID, Connection conn){...
492
493
494     public static int addDepartment(String departmentName, Connection conn){...
516
517
518     public static int deleteDepartment(int departmentID, Connection conn){...
534
535
536     public static int updateDepartment(int departmentID, String departmentName, Connection conn){...
557
558
559     public static int addCourse(String courseName, int noOfHours, Connection conn){...
582
583
584     public static int deleteCourse(int courseID, Connection conn){...
600
601
602     public static int updateCourse(int courseID, String courseName, int noOfHours, Connection conn){...
624
625
626     public static int addRegisterCourse(int studentID, int courseID, int year, int semester, String grade, Connection conn){...
661
662
663     public static int deleteRegisterCourse(int studentID,int courseID,int year,int semester , Connection conn){...
683
684
685     public static int updateRegisterCourse(int studentID, int courseID, int year, int semester, String grade, Connection conn){...
710
711
712 }
```

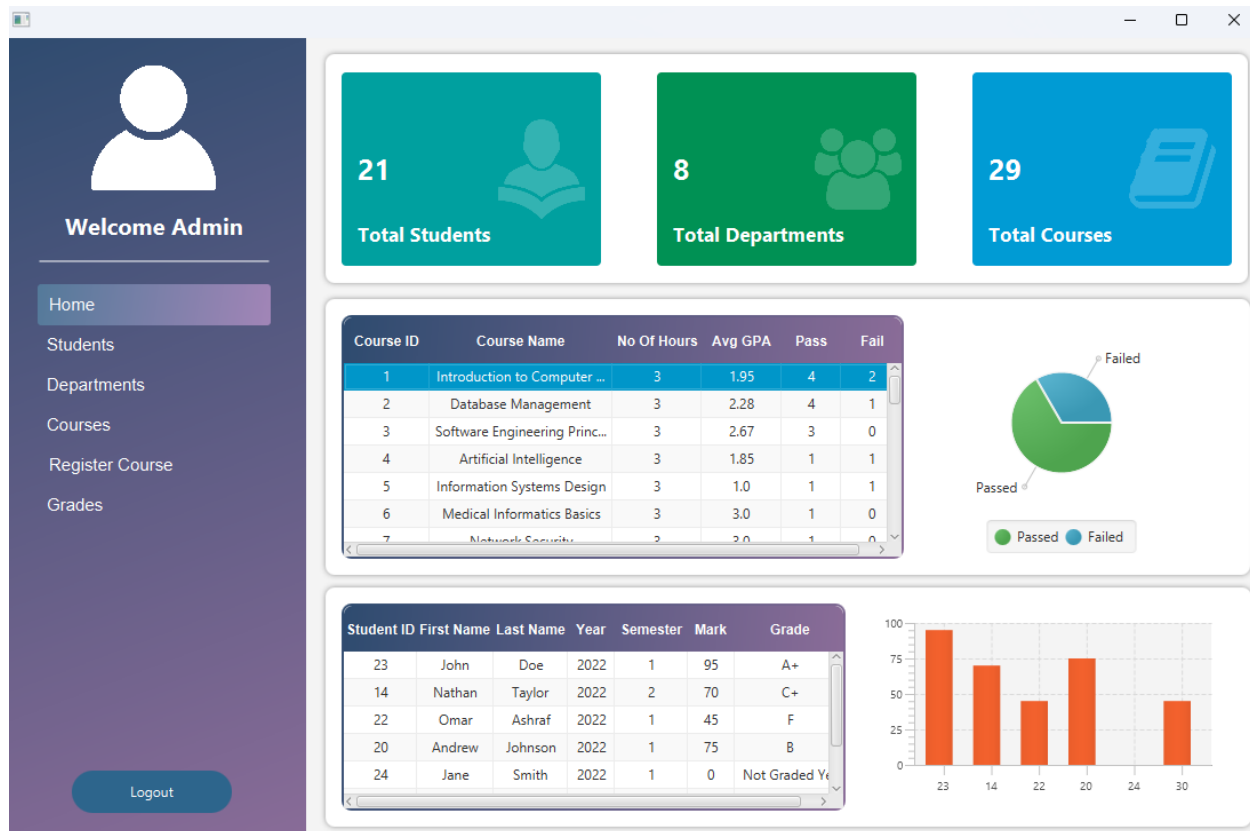
## Application Scenes:

### 1. Login Scene



This is for admin users only that can login to the system for the staff only.

## 2. Home Scene



The home scene of our system provides comprehensive reports on the overall academic landscape. Users can view key statistics, including the total number of students, departments, and courses. Additionally, the system presents a detailed breakdown of each course, showcasing the average GPA, the count of students who passed and failed the course.

Upon selecting a specific course, the system dynamically generates:

- A pie chart illustrates the distribution of students who passed and failed the course.
- A bar chart visually represents the distribution of students' grades in that course.
- A table view displays detailed information on all students who registered for the selected course, offering a holistic overview of recent registrations.

### 3. Students Scene

**Welcome Admin**

Home  
Students  
Departments  
Courses  
Register Course  
Grades  
Logout

Student ID	First Name	Last Name	Email	DOB	Gender	Phone Number	Department ID	Department Name
11	Grace	Anderson	grace.a...	1994-11-11	F	5555555555	6	Medical Informatics
12	Jake	Wilson	jake.w@e...	1993-12-12	M	6666666666	7	Bio informatics
14	Nathan	Taylor	nathan.t...	1991-02-02	M	8888888888	2	Information System
16	Caleb	Smith	caleb.s@...	1989-04-04	M	1111111111	4	Software Engineering
17	Lily	Miller	lily.m@ex...	1988-05-05	F	2222222222	5	Information Technology
19	Zoe	Wilson	zoe.w@e...	1986-07-07	F	4444444444	7	Bio informatics
20	Andrew	Johnson	andrew.j...	0085-08-08	M	5555555555	1	Computer Science
22	Omar	Ashraf	omaras@...	0001-09-25	M	123123	2	Information System
23	John	Doe	john.doe...	1995-03-15	M	1234567890	1	Computer Science
24	Jane	Smith	jane.smit...	1998-07-22	F	9876543210	2	Information System
25	Robert	Johnson	robert.joh...	1996-11-10	M	5555555555	3	Artificial Intelligence

Student ID: 14 Email: nathan.t@example.c Phone Number: 8888888888 Add Update

First Name: Nathan DOB: 2/2/1991 Department: Information ... Delete Clear

Last Name: Taylor Gender: M

**Current GPA: 2.0**  
**Level: 2**

Course ID	Course Name	Year	Semester	Mark	Grade
1	Introduction to ...	2022	2	70	C+
3	Software Engine...	2022	1	67	C
5	Information Syst...	2022	1	65	C
9	Human-Comput...	2022	1	80	B+
10	Mobile App Dev...	2022	1	35	F

The students scene offers a comprehensive view of student data, allowing users to seamlessly interact with individual student records. Users can perform the following actions:

- **Add New Student:**
  - Fill in all the required fields for a new student.
  - Press the 'Add' button to incorporate the new student into the system.
- **Select Student:**
  - Choose a student from the displayed list to view and manage their details.



- The student's information, including their current GPA, academic level, and a list of registered courses with respective marks and grades, is dynamically presented.
- **Update Student:**
  - Modify the student's information by changing the data in the fields.
  - Press the 'Update' button to save the changes.
- **Delete Student:**
  - When viewing a selected student, press the 'Delete' button to remove the student's record from the system.
- **The 'Clear' button to reset the fields and remove any changes.**

## 4. Departments Scene

The screenshot shows a web application interface for managing departments. On the left is a sidebar with a user profile icon, the text "Welcome Admin", and a list of navigation links: Home, Students, Departments (highlighted), Courses, Register Course, and Grades. At the bottom of the sidebar is a "Logout" button. The main content area is divided into three sections. The top section is a table with two columns: "Department ID" and "Department Name". The table contains nine rows of data. Below the table are input fields for "Department ID" (containing the value 2) and "Department Name" (containing the value "Information System"). Below these fields are four buttons: "Add" (blue), "Update" (green), "Delete" (red), and "Clear" (purple). The bottom section is divided into two panels. The left panel is titled "Courses Assigned In Department:" and contains a table with three columns: "Course ID", "Course Name", and "No. Of Hours". The table contains six rows of data, with the row for "Information Systems ..." (Course ID 5) highlighted in blue. The right panel is titled "Courses Not Assigned In Department:" and contains a table with three columns: "Course ID", "Course Name", and "No. Of Hours". The table contains four rows of data. Below the tables are input fields for "Course ID" (containing the value 5) and four buttons: "Add" (blue), "Delete" (red), and "Clear" (purple).

Department ID	Department Name
1	Computer Science
2	Information System
3	Artificial Intelligence
4	Software Engineering
5	Information Technology
6	Medical Informatics
7	Bio informatics
9	Data Management

Department ID:

Department Name:

Add Update  
Delete Clear

**Courses Assigned In Department:**

Course ID	Course Name	No. Of Hours
1	Introduction to Comp...	3
3	Software Engineering...	3
5	Information Systems ...	3
9	Human-Computer Int...	3
10	Mobile App Develop...	3

**Courses Not Assigned In Department:**

Course ID	Course Name	No. Of Hours
2	Database Management	3
4	Artificial Intelligence	3
6	Medical Informatics B...	3
7	Network Security	3

Course ID:

Add Delete  
Clear

The departments scene provides a centralized view of all departments with courses of that department:

- **Add New Department:**
  - Enter a department name in the department name field.
  - Press the 'Add' button to create a new department in the system.
- **Select Department:**
  - Choose a department from the displayed list to access detailed information.
  - View the courses associated with the selected department.

- **Update Department:**
  - Change the department name in the field.
  - Press the 'Update' button to save the modifications.
- **Delete Department:**
  - When viewing a selected department, press the 'Delete' button to remove the department from the system.
- **Manage Courses within Department:**
  - View the courses associated with the selected department.
  - To delete a course, select it from the 'Courses Assigned' table and press the 'Delete' button.
  - To add a course from the 'Courses Not Assigned' table, select it and press the 'Add' button.
- **Clear Button within department it clears all the data in the scene.**
- **Clear Button within courses it clears all data for the courses and the selection of courses.**

## 5. Courses Scene

The screenshot displays the 'Courses Scene' interface. On the left is a dark blue sidebar with a user profile icon and the text 'Welcome Admin'. Below this are navigation links: Home, Students, Departments, Courses (highlighted), Register Course, and Grades. At the bottom of the sidebar is a 'Logout' button. The main area features a table with three columns: 'Course ID', 'Course Name', and 'No. Of Hours'. The table contains 16 rows of course data. Below the table are three input fields labeled 'Course ID:', 'Course Name:', and 'No. Of Hours:'. At the bottom are four buttons: 'Add' (blue), 'Update' (green), 'Delete' (red), and 'Clear' (purple).

Course ID	Course Name	No. Of Hours
1	Introduction to Computer Science	3
2	Database Management	3
3	Software Engineering Principles	3
4	Artificial Intelligence	3
5	Information Systems Design	3
6	Medical Informatics Basics	3
7	Network Security	3
8	Data Structures and Algorithms	3
9	Human-Computer Interaction	3
10	Mobile App Development	3
11	Database Design	3
12	Machine Learning Fundamentals	3
13	Computer Graphics	3
14	Web Development	3
15	Operating Systems	3
16	Software Testing	3

Course ID:  Course Name:  No. Of Hours:

Add Update Delete Clear

The courses scene offers a straightforward interface for managing course information:

- **Add New Course:**
  - Fill in the necessary fields for a new course.
  - Press the 'Add' button to incorporate the new course into the system.
- **Select Course:**
  - Choose a course from the displayed list to view and manage its details.
  - The course's information is dynamically presented in the input fields.

- **Update Course:**

- Select a course to modify its information.
- Update the relevant fields.
- Press the 'Update' button to save the changes.

- **Delete Course:**

- Choose a course and press the 'Delete' button to remove it from the system.

- **Clear Fields:**

- Press the 'Clear' button to reset all fields, providing a convenient way to start anew or cancel changes.

## 6. Register Course Scene

The screenshot shows a web application interface for an administrator. On the left is a dark blue sidebar with a user profile icon and the text "Welcome Admin". Below this are navigation links: Home, Students, Departments, Courses, Register Course (highlighted), and Grades. At the bottom of the sidebar is a "Logout" button.

The main content area contains two tables. The top table lists students with columns: Student ID, First Name, Last Name, Email, DOB, Gender, Phone Number, Department ID, and Department Name. The bottom table lists courses with columns: Course ID, Course Name, and No Of Hours.

Below the tables is a form for registering a course. It includes input fields for Student ID (11), Course ID (6), Year (2024), and Semester (1). At the bottom of the form are two buttons: "Add" and "Clear".

Student ID	First Name	Last Name	Email	DOB	Gender	Phone Number	Department ID	Department Name
11	Grace	Anderson	grace.a@exa...	1994-11-11	F	5555555555	6	Medical Informatics
12	Jake	Wilson	jake.w@exampl...	1993-12-12	M	6666666666	7	Bio informatics
14	Nathan	Taylor	nathan.t@exa...	1991-02-02	M	8888888888	2	Information System
16	Caleb	Smith	caleb.s@exam...	1989-04-04	M	1111111111	4	Software Engineerin
17	Lily	Miller	lily.m@exampl...	1988-05-05	F	2222222222	5	Information Technolc
19	Zoe	Wilson	zoe.w@exampl...	1986-07-07	F	4444444444	7	Bio informatics
20	Andrew	Johnson	andrtewj@exa...	0085-08-08	M	5555555555	1	Computer Science


Course ID	Course Name	No Of Hours
11	Database Design	3
14	Web Development	3
2	Database Management	3
19	Bioinformatics Tools	3
6	Medical Informatics Basics	3
10	Mobile App Development	3
19	Bioinformatics Tools	3

Student ID:  Course ID:  Year:  Semester:

The register course scene provides an efficient way to manage student course registrations:

- **Display Students and Associated Courses:**
  - View a list of all students.
  - Upon selecting a student, the system dynamically displays the courses associated with the department that the student is enrolled in.
- **Register for a Course:**
  - Select a course from the displayed list.
  - Press the 'Add' button to register the student for the chosen course.
- **Course Registration Logic:**
  - The system prevents a student from registering for a course they have already passed successfully.
  - If a student has failed a course, they can register for it in subsequent years or terms.
  - Year and semester are calculated automatically for year it automatically with the current year and for semester for month 1 to 6 it will display semester 1, for month 7 to 12 it will display semester 2.
- **Clear Fields:**
  - Press the 'Clear' button to reset all fields and selections.





Welcome Admin

Home

Students

Departments

Courses

Register Course

Grades

Logout

Student ID	First Name	Last Name	Email	DOB	Gender	Phone Number	Department ID	Department Name
11	Grace	Anderson	grace.a@exa...	1994-11-11	F	5555555555	6	Medical Informatics
12	Jake	Wilson	jake.w@exampl...	1993-12-12	M	6666666666	7	Bio informatics
14	Nathan	Taylor	nathan.t@exa...	1991-02-02	M	8888888888	2	Information System
16	Caleb	Smith	caleb.s@exam...	1989-04-04	M	1111111111	4	Software Engineerin
17						2222222222	5	Information Technolc
19						4444444444	7	Bio informatics
20						5555555555	1	Computer Science

Error Message


Student Didn't Complete Course Yet

OK

	No Of Hours	
11	Database Design	3
14	Web Development	3
2	Database Management	3
19	Bioinformatics Tools	3
6	Medical Informatics Basics	3
10	Mobile App Development	3
19	Bioinformatics Tools	3

Student ID: 11Course ID: 11Year: 2024Semester: 1

AddClear



Welcome Admin

Home

Students

Departments

Courses

Register Course

Grades

Logout

Student ID	First Name	Last Name	Email	DOB	Gender	Phone Number	Department ID	Department Name
11	Grace	Anderson	grace.a@exa...	1994-11-11	F	5555555555	6	Medical Informatics
12	Jake	Wilson	jake.w@exampl...	1993-12-12	M	6666666666	7	Bio informatics
14	Nathan	Taylor	nathan.t@exa...	1991-02-02	M	8888888888	2	Information System
16	Caleb	Smith	caleb.s@exam...	1989-04-04	M	1111111111	4	Software Engineerin
17						2222222222	5	Information Technolc
19						4444444444	7	Bio informatics
20						5555555555	1	Computer Science

Error Message

Student Already passed the course

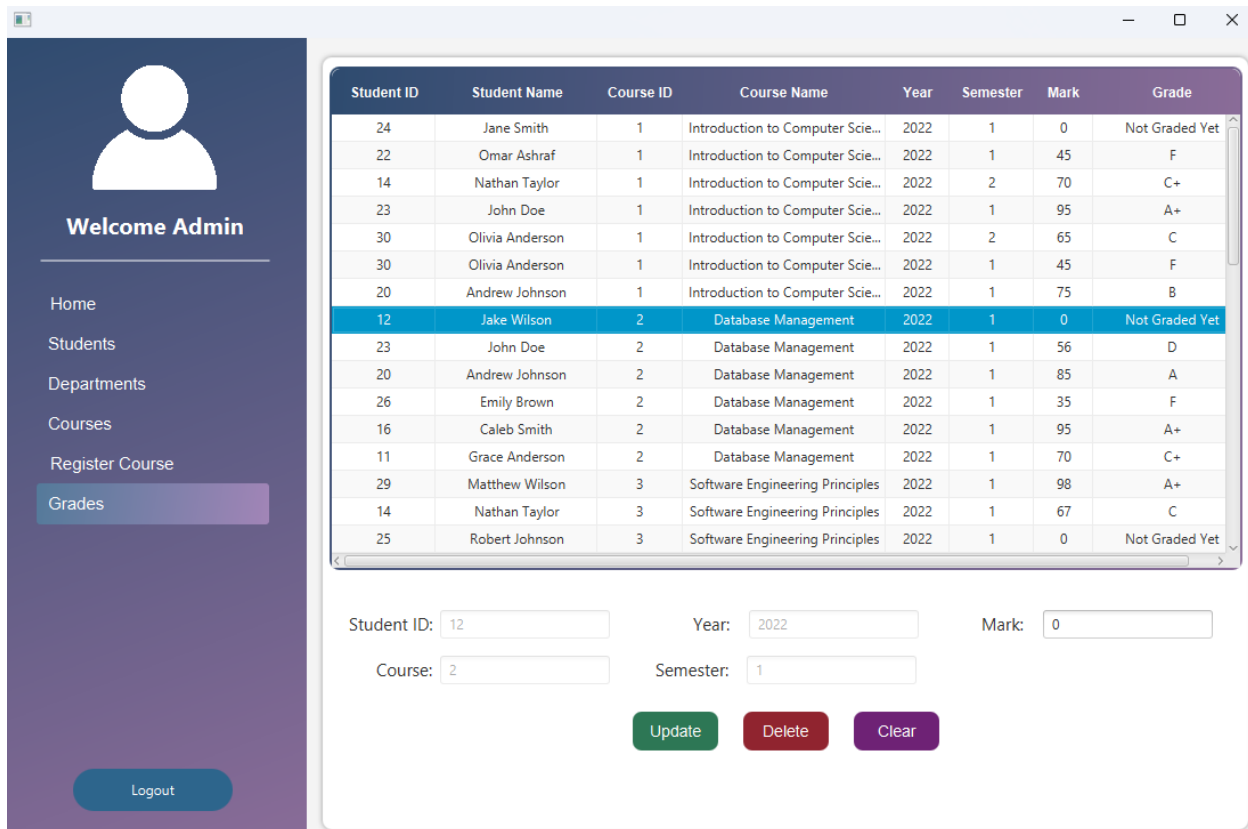
OK

	No Of Hours	
11	Database Design	3
14	Web Development	3
2	Database Management	3
19	Bioinformatics Tools	3
6	Medical Informatics Basics	3
10	Mobile App Development	3
19	Bioinformatics Tools	3

Student ID: 11Course ID: 14Year: 2024Semester: 1

AddClear

## 7. Grades Scene



Student ID	Student Name	Course ID	Course Name	Year	Semester	Mark	Grade
24	Jane Smith	1	Introduction to Computer Scie...	2022	1	0	Not Graded Yet
22	Omar Ashraf	1	Introduction to Computer Scie...	2022	1	45	F
14	Nathan Taylor	1	Introduction to Computer Scie...	2022	2	70	C+
23	John Doe	1	Introduction to Computer Scie...	2022	1	95	A+
30	Olivia Anderson	1	Introduction to Computer Scie...	2022	2	65	C
30	Olivia Anderson	1	Introduction to Computer Scie...	2022	1	45	F
20	Andrew Johnson	1	Introduction to Computer Scie...	2022	1	75	B
12	Jake Wilson	2	Database Management	2022	1	0	Not Graded Yet
23	John Doe	2	Database Management	2022	1	56	D
20	Andrew Johnson	2	Database Management	2022	1	85	A
26	Emily Brown	2	Database Management	2022	1	35	F
16	Caleb Smith	2	Database Management	2022	1	95	A+
11	Grace Anderson	2	Database Management	2022	1	70	C+
29	Matthew Wilson	3	Software Engineering Principles	2022	1	98	A+
14	Nathan Taylor	3	Software Engineering Principles	2022	1	67	C
25	Robert Johnson	3	Software Engineering Principles	2022	1	0	Not Graded Yet

Student ID:  Year:  Mark:

Course:  Semester:

The grades scene provides a comprehensive view of student grades for registered courses:

- **Display Student Grades:**
  - View a list of all students who have registered for courses along with their respective grades.
  - The grades are displayed if the students have been graded, and 'Not Graded Yet' is shown for pending evaluations.
- **Select and Manage Student Grades:**
  - Choose a student from the displayed list and data displayed in fields.

- If the student is graded, their mark is displayed. If not, the field indicates '0'.
- **Delete Grade Record:**
  - Press the 'Delete' button to remove the selected record.
- **Update Student Mark:**
  - Select a student from the list.
  - The system displays the student's data, including the current mark.
  - Modify the mark within the valid range (0 to 100).
  - Press the 'Update' button to save the changes.
- **Clear Fields:**
  - Press the 'Clear' button to reset all fields and selections, providing a convenient way to start anew or cancel changes.