CORE JAVA PROJECT

Project Title: Student Management System

Prepared by: Naveen Venigandla

Aim: To implement a Student Management System with database connectivity.

Languages:

- Java,
- MySQL

Software Used:

- Eclipse for Java
- MySQL Database

Synopsis:

There are seven main operations performed in this project.

1. Adding Student
2. Enrolling Students into a Course
3. Checking Student Balance
4. Pay Tuition Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Operations.

There is an admin login, and the admin can log in and perform various operations like:

- View course.
- Add a new course.
- View all the students enrolled.
- Add a new student.

The View Course option is used to check all the courses that are available.

Database:

The following three tables are created in the backend. The structure of the three tables is shown below.

There are three tables in the backend. The Course tables has all the courses that can be shown to the student while enrolling to a new Course. The course table has 3 rows, Course ID(cid), Course Name(course_name), Course Fee(cfee). The structure of the course table is as shown below.

Field	Type	Null	Key	Default	Extra
cid	int	NO	PRI	NULL	
course_name	varchar(30)	NO		NULL	
cfee	float	NO		NULL	

Course Table Structure

The Student table has all the student details. There are four columns, Student ID(sid), Student Name(sname), Student Email(semail), and balance. The structure of the student table is as shown below. The balance has an check constraint its value should be greater than or equal to 0(balance>=0).

Field	Type	Null	Key	Default	Extra
sid	int	NO	PRI	NULL	auto_increment
sname	varchar(30)	NO		NULL	
semail	varchar(30)	NO	UNI	NULL	
balance	float	YES		NULL	

Student Table Structure

The enroll table contains all the students enrolled for different keys. There are two rows in the enroll table Student ID(sid) and Course ID(cid). This table doesn't have a primary key and both rows present are foreign keys. The sid references the sid from student table and the cid represents cid from the course table. The structure of enroll table is shown below.

Field	Type	Null	Key	Default	Extra
sid	int	YES	MUL	NULL	

cid	int	YES	MUL	NULL	

Enroll Table Structure

The tables are created in a separate database. In this project I have created a database as “studentdb”.

Eclipse:

Create a Maven Project, in this project I have named the Project as “StudentManagement”. There are 4 classes in the project. The classes are

- DbConnection
- StudentManagement
- Operations
- Admin

Note: In “pom.xml” file we need to add dependencies for connecting mysql to eclipse java file. Here’s the dependency I’ve used in the project.

```
  <dependencies><!-- https://mvnrepository.com/artifact/mysql/mysql-
connector-java -->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.28</version>
</dependency>
  </dependencies>
```

The source code for the project:

“StudentManagement.java” class

```
package com.navi;


import java.sql.SQLException;

import java.util.Scanner;

/**

 * @author Naveen

 * @version 06/10/2023

 */

public class StudentManagement {
```

```
	/**
	 * This method displays the menu of operations user can select
	 * @param args some arguments for user input into main method
	 * @throws SQLException because database errors may occur while
connecting to database
	 */
	public static void main(String[] args) throws SQLException {
		Scanner sc = new Scanner(System.in);

		while(true) {
			System.out.println("<=========Menu=========>");
			System.out.println("1. Add Student");
			System.out.println("2. Enroll a Course");
			System.out.println("3. View Balance");
			System.out.println("4. Pay Tution Fee");
			System.out.println("5. Show Student Status");
			System.out.println("6. Forgot Student ID");
			System.out.println("7. Admin Login");
			System.out.println("<=========Menu=========>");
			System.out.println("Enter your choice");
			int choice = sc.nextInt();

			switch(choice) {
			case 1:
				// To add new students we call add students method
				Operations.addStudent();
				break;
			case 2:
				// Enroll student to a specific course
				Operations.enrollCourse();
				break;
```

```
                                        case 3:
                                                //To view Balance
                                                Operations.viewBalance();
                                                break;
                                        case 4:
                                                // To pay tution fee
                                                Operations.payTutionFee();
                                                break;
                                        case 5:
                                                Operations.showStatus();
                                                break;
                                        case 6:
                                                Operations.getStudentID();
                                                break;
                                        case 7:
                                                Admin.operations();
                                                break;
                                        default:
                                                System.out.println("Invalid Option try again.");
                                        }
                                        System.out.println("Do you want to continue (y/n)");
                                        char c = sc.next().toLowerCase().charAt(0);
                                        if(c!='y') {
                                                break;
                                        }
                                }
                        }
}
```

DbConnection.java class:

```
package com.navi;


import java.sql.Connection;
import java.sql.DriverManager;
/**
 * Provides Database connection to the project.
 */
public class DbConnection {
	private static String driver = "com.mysql.cj.jdbc.Driver";
	private static String url = "jdbc:mysql://localhost:3306/studentdb";
	private static String uname = "root";
	private static String pass = "root";
	private static Connection conn;


	/**
	 * Is used for providing connection to the database/
	 * @return connection data.
	 */
	public static Connection getConnection() {


		try {
			//1. Load the driver
			Class.forName(driver);
			//2. Make connection
			conn = DriverManager.getConnection(url, uname, pass);
		}catch(Exception e) {
			e.printStackTrace();
		}
		return conn;
	}
```

```
}
```

“Operations.java” class

```
package com.navi;


import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;


/**
 * This class implements all of the operations.
 */
public class Operations {
	private static Scanner sc;
	private static Connection conn;
	private static ResultSet rs;
	private static PreparedStatement ps;
	private static String s,sname,semail;
	private static int sid,cid,i;
	private static Float amount,bal ;
	private static List<Integer> courseId;



	/**
	 * This method is used to add new students.
```

```
	 * @throws SQLException because database errors may occur while connecting to database
	 */
	public static void addStudent() throws SQLException {

		System.out.println("ADD STUDENT");
		//Initialize scanner class to take inputs from user
		sc = new Scanner(System.in);

		System.out.println("Enter Student name");
		sname = sc.nextLine();

		System.out.println("Enter Student email");
		semail = sc.next();

		s= "insert into student(sname,semail,balance) values(?,?,0)";
		conn = DbConnection.getConnection();

		ps = conn.prepareStatement(s);
		ps.setString(1, sname);
		ps.setString(2, semail);

		i = ps.executeUpdate();
		if(i>0) {
			System.out.println("Records added Successfully");
			//If records are added successfully we are displaying all the inserted records
			//This is to ensure that correct records are entered and to give the student
their student Id.
			s = "select sid,sname,semail from student where semail = ?";
			ps = conn.prepareStatement(s);
			ps.setString(1, semail);
			rs = ps.executeQuery();
```

```
                        System.out.printf("%-15s%-20s%-20s%n","Student ID","Student
Name","Student Email");
                        while(rs.next()) {
                                System.out.printf("%-15d%-20s%-
20s%n",rs.getInt("sid"),rs.getString("sname"),rs.getString("semail"));
                        }
                }else {
                        System.out.println("An error occurred while entering records");
                }
        }
        /**
         * This method is to get student ID
         * @throws SQLException because database errors may occur while connecting to database
         */
        public static void getStudentID() throws SQLException {
                System.out.println("Forgot Student ID");
                conn = DbConnection.getConnection();

                sc = new Scanner(System.in);
                System.out.println("Enter Student Mail");
                semail = sc.next();
                try {
                        s = "select sid from student where semail=?";
                        ps = conn.prepareStatement(s);
                        ps.setString(1, semail);
                        rs = ps.executeQuery();
                        if(rs.next()) {
                                System.out.println(rs.getInt("sid"));
                        }
                }catch(Exception e) {
                        System.out.println("Student already exists with given mail.");
                }
```

```
}
/**
 * This method is for checking if the sid exists or not in the student table
 * @param sid represents student id
 * @return is used to return if student id exists or not
 * @throws SQLException because database errors may occur while connecting to database
 */
public static boolean studentId(int sid) throws SQLException {
		conn = DbConnection.getConnection();
		s="select sid from student where sid=?";
		ps = conn.prepareStatement(s);
		ps.setInt(1, sid);
		rs = ps.executeQuery();
		if(rs.next()) {
			return true;
		}else {
			return false;
		}

}

/**
 * This method allows student to enroll into a course
 * @throws SQLException because database errors may occur while connecting to database.
 */
public static void enrollCourse() throws SQLException {
		System.out.println("Enroll course");
		conn = DbConnection.getConnection();
		float cfee;
```

```
		//Initializing courseId array list to store all the course id's in the list.
		courseId = new ArrayList<Integer>();
		sc = new Scanner(System.in);

		//Printing course List
		s ="select * from course";
		ps = conn.prepareStatement(s);
		rs = ps.executeQuery();
		System.out.printf("%-10s%-25s%-10s%n%n","Course ID","Course Name","Course
Fee");
		while(rs.next()) {
			System.out.printf("%-10d%-25s%-
10.2f%n",rs.getInt(1),rs.getString(2),rs.getFloat(3));
			courseId.add(rs.getInt("cid"));
		}



		while(true) {
			System.out.println("Enter your student id");
			sid = sc.nextInt();
			if(studentId(sid)==true) {
				break;
			}else {
				System.out.println("Student ID is incorrect. Please enter correct
student ID.");
			}
		}
		System.out.println("Enter a course ID which you want to enroll.....(from the abouve
list)");
		cid = sc.nextInt();
```

```
                if(courseId.contains(cid)) {

                        /*Checking if the student is already registered for the same course
                         * If yes, then we will print "You have already registered for this course
                         * If no, then we will allow the student to register for the course
                         */
                        List<Integer> lst = new ArrayList<Integer>();
                        s="select cid from enroll where sid=?";
                        ps = conn.prepareStatement(s);
                        ps.setInt(1, sid);
                        rs = ps.executeQuery();
                        while(rs.next()) {
                                lst.add(rs.getInt("cid"));
                        }
                        if(lst.contains(cid)) {
                                System.out.println("You have already registered for this course.");
                        }else {
                                s = "select cfee from course where cid=?";
                                ps = conn.prepareStatement(s);
                                ps.setInt(1, cid);
                                rs = ps.executeQuery();
                                if(rs.next()) {
                                        cfee = rs.getFloat("cfee");

                                        while(true) {
                                                System.out.println("Enter the amount you want to
pay.");
                                                amount = sc.nextFloat();
                                                if(amount>=0) {
                                                        break;
```

```
                                        }else {
                                                System.out.println("Invalid amount. Enter
amount not less than 0");
                                        }
                                }
                                amount = cfee-amount;

                                // Get existing balance
                                s= "select balance from student where sid=?";
                                ps = conn.prepareStatement(s);
                                ps.setInt(1, sid);
                                rs = ps.executeQuery();
                                if(rs.next()) {
                                        bal = rs.getFloat("balance");
                                }
                                amount +=bal;

                                //Updating Balance
                                s = "update student set balance =? where sid =?";
                                ps = conn.prepareStatement(s);
                                ps.setFloat(1, amount);
                                ps.setInt(2, sid);
                                i = ps.executeUpdate();
                                if(i<=0) {
                                        System.out.println("Error while updating course
fee");
                                }


                        // After Updating the balance inserting student and course details in
enroll table
                        s = "insert into enroll(sid,cid) values(?,?)";
```

```
                                ps = conn.prepareStatement(s);
                                ps.setInt(1, sid);
                                ps.setInt(2, cid);
                                i =ps.executeUpdate();
                                if(i>0) {
                                        System.out.println("Enrollment successfull");
                                }else {
                                        System.out.println("Error occured while enrolling course");
                                }
                                }
                        }

                }else {
                        System.out.println("Course ID doesn't match.");
                }

        }

        /**
         * This method is to check the balance, so the student knows the pending balance
         * @throws SQLException because database errors may occur while connecting to database
         */
        public static void viewBalance() throws SQLException {
                System.out.println("View Balance");
                conn = DbConnection.getConnection();

                sc = new Scanner(System.in);
                // Validating whether the student ID exists or not.
                while(true) {
                        System.out.println("Enter your student id");
                        sid = sc.nextInt();
```

```
                        if(studentId(sid)==true) {
                                break;
                        }else {
                                System.out.println("Student ID is incorrect. Please enter correct
student ID.");
                        }
                }
                //For getting balance
                s = "select balance from student where sid=?";
                ps = conn.prepareStatement(s);
                ps.setInt(1, sid);
                rs = ps.executeQuery();
                if(rs.next()) {
                        System.out.println(rs.getFloat("balance"));
                }

        }

        /**
         * This method allows student to pay tution fee
         * @throws SQLException because database errors may occur while connecting to database
         */
        public static void payTutionFee() throws SQLException {
                sc = new Scanner(System.in);

                conn = DbConnection.getConnection();

                // Validating whether the student ID exists or not.
                while(true) {
                        System.out.println("Enter your student id");
                        sid = sc.nextInt();
```

```
                    if(studentId(sid)==true) {
                            break;
                    }else {
                            System.out.println("Student ID is incorrect. Please enter correct
student ID.");
                    }
            }

            s = "select balance from student where sid=?";
            ps = conn.prepareStatement(s);
            ps.setInt(1, sid);
            rs = ps.executeQuery();
            if(rs.next()) {
                    bal = rs.getFloat("balance");
            }
            System.out.println("Your Balance is "+bal);
            if(bal==0) {
                    System.out.println("You have no pending fee.");
            }else {
                    while(true) {
                            System.out.println("Enter the amount to pay");
                            amount = sc.nextFloat();
                            if(amount>=0) {
                                    break;
                            }else {
                                    System.out.println("Invalid amount. Enter amount not less
than 0");
                            }
                    }
                    amount = bal-amount;
                    s = "update student set balance=? where sid=?";
                    ps = conn.prepareStatement(s);
```

```
                    ps.setFloat(1, amount);
                    ps.setInt(2, sid);
                    i = ps.executeUpdate();
                    if(i>0) {
                        System.out.println("Tution fee paid.");
                    }
            }

}

/**
 * This method shows the  student status(like student details,courses enrolled,balance....)
 * @throws SQLException because database errors may occur while connecting to database
 */
public static void showStatus() throws SQLException {
            sc = new Scanner(System.in);
            courseId = new ArrayList<Integer>();
            System.out.println("Enter Student ID");
            sid = sc.nextInt();
            //Get connection
            conn = DbConnection.getConnection();

            //Get courses enrolled by student
            s= "select cid from enroll where sid = ?";
            ps = conn.prepareStatement(s);
            ps.setInt(1, sid);
            rs = ps.executeQuery();
            while(rs.next()) {
                courseId.add(rs.getInt("cid"));
            }
```

```
                    // Preparing statement
                    s = "select * from student where sid =?";
                    ps = conn.prepareStatement(s);
                    ps.setInt(1, sid);
                    rs = ps.executeQuery();
                    while(rs.next()) {
                            System.out.printf("%-15s : %-10d%n%-15s : %-20s%n%-15s : %-30s%n%-15s :
%-10.2f%n"
                                    ,"Student ID",rs.getInt(1),"Student
Name",rs.getString(2),"Student Email",rs.getString(3),"Balance",rs.getFloat(4));
                            System.out.printf("%s :","Courses Enrolled");
                            for(int course: courseId) {
                                    s = "select course_name from course where cid=?";
                                    ps =conn.prepareStatement(s);
                                    ps.setInt(1, course);
                                    rs = ps.executeQuery();
                                    while(rs.next()) {
                                            System.out.printf("%-20s%n",rs.getString("course_name"));
                                    }

                            }
                    }
            }



}
```

“Admin.java” class

```
package com.navi;


import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
/**
 * Provides admin controls.
 */
public class Admin {
                        private static Connection conn;
                        private static ResultSet rs;
                        private static PreparedStatement ps;


                        private static Scanner sc;
                        private static String s;


                        private static String adm_uname="admin";
                        private static String adm_pass="admin123";



                        /**
                         * Provides menu for admin and lists all the operations admin can perform
after login.
                         * Admin details are directly stored in the variables instead of database.
                         * @throws SQLException because database errors may occur while
connecting to database.
                         */
                        public static void operations() throws SQLException {
```

```
sc = new Scanner(System.in);


System.out.println("Enter user name");
String uname = sc.next();
System.out.println("Enter password");
String pass = sc.next();
if(uname.equalsIgnoreCase(adm_uname) && pass.equals(adm_pass)) {
        while(true) {
                System.out.println("<=========Menu=========>");
                System.out.println("1. View Courses");
                System.out.println("2. Add Course");
                System.out.println("3. Show Students Enrolled");
                System.out.println("4. Add Student");
                System.out.println("<=========Menu=========>");
                System.out.println("Enter your choice");
                int choice = sc.nextInt();


                switch(choice) {
                case 1:
                        existingcourses();
                        break;
                case 2:
                        // Adding new course
                        addCourse();
                        break;
                case 3:
                        //Show Students enrolled and their details
                        studentDetails();
                        break;
                case 4:
                        // To pay tution fee
```

```
                                        Operations.addStudent();
                                        break;
                                default:
                                        System.out.println("Invalid Option try again.");
                                }
                                System.out.println("Do you want to continue or do you want
to logout enter 'L' if you want to logout");
                                char ch = sc.next().toLowerCase().charAt(0);
                                if(ch=='l') {
                                        System.out.println("Logout successful.");
                                        break;
                                }
                        }//Closing menu loop(while)
                        }else {
                                System.out.println("Username or password is wrong");
                        }

                }



                /**
                 * This method lists all the existing courses available in the course table.
                 * @throws SQLException because database errors may occur while
connecting to database
                 */
                public static void existingcourses() throws SQLException {
                System.out.println("Existing Course List");
                conn = DbConnection.getConnection();
                s = "select * from course";
                ps = conn.prepareStatement(s);
                rs = ps.executeQuery();
```

```
                        System.out.printf("%-10s%-25s%-10s%n","Course ID","Course
Name","Course Fee");
                        while(rs.next()) {
                                System.out.printf("%-10d%-25s%-
10.2f%n",rs.getInt(1),rs.getString(2),rs.getFloat(3));
                        }

                        }

                        /**
                         * This method is used to add new courses to the course table.
                         * @throws SQLException because database errors may occur while
connecting to database
                         */
                        public static void addCourse() throws SQLException {
                        System.out.println("Add course");
                        sc = new Scanner(System.in);
                        conn = DbConnection.getConnection();
                        System.out.println("Enter new course ID");
                        int cid = sc.nextInt();
                        System.out.println("Enter the course name");
                        sc.nextLine();
                        String cname = sc.nextLine();
                        System.out.println("Enter the course fee");
                        float cfee = sc.nextFloat();

                        s = "insert into course values(?,?,?)";
                        ps = conn.prepareStatement(s);
                        ps.setInt(1, cid);
                        ps.setString(2, cname);
                        ps.setFloat(3, cfee);
                        int i = ps.executeUpdate();
```

```
                        if(i>0) {
                                System.out.println("Course added successfully");
                        }else {
                                System.out.println("Error occurred while adding new course");
                        }
                        }


                        /**
                         * This method is used to show details of all the students enrolled for the
course.
                         * @throws SQLException because database errors may occur while
connecting to database
                         */
                        private static void studentDetails() throws SQLException {
                        String s2;
                        PreparedStatement ps2;
                        ResultSet rs2;
                        List<Integer> lst = new ArrayList<Integer>();
                        s = "select * from student";
                        conn = DbConnection.getConnection();
                        ps = conn.prepareStatement(s);
                        rs = ps.executeQuery();
                        System.out.printf("%-15s%-20s%-20s%-10s%-20s%n","Student ID","Student
Name","Student Email","Balance","Course(s) Enrolled");
                        while(rs.next()) {
                                int sid = rs.getInt(1);
                                System.out.printf("%-15d%-20s%-20s%-
10.2f",rs.getInt(1),rs.getString(2),rs.getString(3),rs.getFloat(4));
                                s2 = "select cid from enroll where sid=?";
                                ps2 = conn.prepareStatement(s2);
                                ps2.setInt(1, sid);
                                rs2=ps2.executeQuery();
```

```
                                while(rs2.next()) {
                                        lst.add(rs2.getInt("cid"));
                                }
                                if(lst.isEmpty()) {
                                        System.out.printf("%-10s%n","None");
                                }else {
                                        for(int c:lst) {
                                                s2 = "select course_name from course where cid
=?";
                                                ps2 = conn.prepareStatement(s2);
                                                ps2.setInt(1, c);
                                                rs2 = ps2.executeQuery();
                                                while(rs2.next()) {
                                                        System.out.printf("%-
20s",rs2.getString("course_name"));
                                                }System.out.println();
                                        }lst.clear();
                                }
                        }
                        }


}
```

Output:

```
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
```

```
<=========Menu=========>
Enter your choice
1
ADD STUDENT
Enter Student name
Naveen
Enter Student email
naveen.@gmail.com
Records added Successfully
Student ID     Student Name        Student Email
10019          Naveen              naveen.@gmail.com
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
2
Enroll course
Course ID Course Name              Course Fee

101       AWS                      3000.00
102       Java Full Stack          3000.00
103       Cloud Computing          4000.00
104       Data Analytics           4000.00
105       DevOps                   3000.00
Enter your student id
10019
Enter a course ID which you want to enroll.....(from the abouve list)
102
Enter the amount you want to pay.
1000
Enrollment successfull
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
3
View Balance
Enter your student id
10019
2000.0
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
```

```
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
4
Enter your student id
10019
Your Balance is 2000.0
Enter the amount to pay
2000
Tution fee paid.
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
4
Enter your student id
10019
Your Balance is 0.0
You have no pending fee.
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
5
Enter Student ID
10019
Student ID      : 10019
Student Name    : Naveen
Student Email   : naveen.@gmail.com
Balance         : 0.00
Courses Enrolled :Java Full Stack
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
```

```
<=========Menu=========>
Enter your choice
2
Enroll course
Course ID Course Name              Course Fee

101       AWS                      3000.00
102       Java Full Stack          3000.00
103       Cloud Computing          4000.00
104       Data Analytics           4000.00
105       DevOps                   3000.00
Enter your student id
10019
Enter a course ID which you want to enroll.....(from the abouve list)
105
Enter the amount you want to pay.
0
Enrollment successfull
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
3
View Balance
Enter your student id
10019
3000.0
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
4
Enter your student id
10019
Your Balance is 3000.0
Enter the amount to pay
3000
Tution fee paid.
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
```

```
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
5
Enter Student ID
10019
Student ID      : 10019
Student Name    : Naveen
Student Email   : naveen.@gmail.com
Balance         : 0.00
Courses Enrolled :Java Full Stack
DevOps
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
6
Forgot Student ID
Enter Student Mail
naveen.@gmail.com
10019
Do you want to continue (y/n)
y
<=========Menu=========>
1. Add Student
2. Enroll a Course
3. View Balance
4. Pay Tution Fee
5. Show Student Status
6. Forgot Student ID
7. Admin Login
<=========Menu=========>
Enter your choice
7
Enter user name
admin
Enter password
admin123
<=========Menu=========>
1. View Courses
2. Add Course
3. Show Students Enrolled
4. Add Student
<=========Menu=========>
Enter your choice
1
Existing Course List
Course ID Course Name              Course Fee
101       AWS                      3000.00
102       Java Full Stack          3000.00
103       Cloud Computing          4000.00
```

```
104       Data Analytics           4000.00
105       DevOps                   3000.00
Do you want to continue or do you want to logout enter 'L' if you want to
logout
y
<=========Menu=========>
1. View Courses
2. Add Course
3. Show Students Enrolled
4. Add Student
<=========Menu=========>
Enter your choice
2
Add course
Enter new course ID
106
Enter the course name
Machine Learning
Enter the course fee
2500
Course added successfully
Do you want to continue or do you want to logout enter 'L' if you want to
logout
y
<=========Menu=========>
1. View Courses
2. Add Course
3. Show Students Enrolled
4. Add Student
<=========Menu=========>
Enter your choice
3
Student ID     Student Name        Student Email       Balance   Course(s)
Enrolled
10001          Naveen Venigandla   naveen@gmail.com    1000.00   Java Full
Stack
10002          Sai                 sai@gmail.com       1000.00   Java Full
Stack
AWS
10003          Manu                manu@gmail.com      1000.00   Cloud
Computing
10004          Valli               valli@gmail.com     0.00      DevOps
10005          Jayadeep            jay@gmail.com       0.00      None
10006          Suresh              suresh@gmail.com    0.00      None
10007          Ramesh              ramesh@gmail.com    0.00      None
10014          Sameer              sameer@gmail.com    0.00      None
10015          Nani                nani@gmail.com      0.00      Java Full
Stack
10016          Sai Kiran           saikiran@gmail.com  0.00      None
10017          Naveen              nave@gmail.com      2000.00   Java Full
Stack
10018          Naveen V            naveen1@gmail.com   0.00      AWS
10019          Naveen              naveen.@gmail.com   0.00      Java Full
Stack
DevOps
Do you want to continue or do you want to logout enter 'L' if you want to
logout
L
Logout successful.
Do you want to continue (y/n)
n
```

Database records:

Student Table:

sid	sname	semail	balance
10001	Naveen Venigandla	naveen@gmail.com	1000
10002	Sai	sai@gmail.com	1000
10003	Manu	manu@gmail.com	1000
10004	Valli	valli@gmail.com	0
10005	Jayadeep	jay@gmail.com	0
10006	Suresh	suresh@gmail.com	0
10007	Ramesh	ramesh@gmail.com	0
10014	Sameer	sameer@gmail.com	0
10015	Nani	nani@gmail.com	0
10016	Sai Kiran	saikiran@gmail.com	0
10017	Naveen	nave@gmail.com	2000
10018	Naveen V	naveen1@gmail.com	0
10019	Naveen	naveen.@gmail.com	0

Course Table:

cid	course_name	cfee
101	AWS	3000
102	Java Full Stack	3000
103	Cloud Computing	4000
104	Data Analytics	4000
105	DevOps	3000
106	Machine Learning	2500

Enroll Table:

sid	cid
10015	102
10001	102
10002	102
10002	101
10003	103
10004	105
10017	102
10018	101
10019	102
10019	105

Conclusion:

- ✓ Implemented a Student Management System with Database connectivity.
- ✓ It has auto-generated 5 digit Student ID.