

A PRELIMINARY REPORT ON

“Student Management System”

SUBMITTED TO THE EDUBRIDGE INDIA PRIVATE LIMITED

SUBMITTED BY

K.VISHALI

BATCH NO: EON-5755

Under The Guidance Of

Amruta Deore

ACKNOWLEDGEMENT

It gives all of us great pleasure in presenting the preliminary project report on “**Student Management System**”. With due respect and gratitude we would like to take this opportunity to thank internal guide of project **Mrs.Amruta Deore** for giving us all help and guidance we needed. We are really grateful for her kind support. She always encouraged us and given us the motivation to move ahead. She has put in a lot of time and effort in this project along with us and given us a lot of confidence. Also we wish to thank all the other people who had helped us in the successful completion of this project.

K.Vishali

ABSTRACT

Student forms a main part in any institutions. But the institutions find it difficult to keep the details of many students of the organization just in one stretch. The Student Management System application will help in managing the student's report will be easier with the one system. It will also help in saving time and effort. If you want to get any information regarding the particular student then it can be obtained by just entering the roll number of the student to be searched. This Student Management System will make the work of storing the data in an organized way.

Chapter 1

INTRODUCTION

1.1 INTRODUCTION

The objective of Student Management System is to allow the administrator of any organization to edit and find out the personal details of a student and allows the student to keep up to date his/her profile. It will also facilitate keeping all the records of the students such as their roll number, name, grade level, courses enrolled and tuition balance of each course. So all the information about an student will be available within a few seconds. Overall it will make the Student Management System an easier job for the administrator and the student of any organization. The main purpose of Student Management System is to illustrate the requirements of the student to help for the organization to maintain and manage the student's personal data.

1.2 SCOPE

Without a Student Management System, managing and maintaining the details of the student is a tedious job for any organization. Student Management System will store all the details of the student such as roll number, name, grade level, courses enrolled and the tuition balance for each course.

1.3 SYSTEM REQUIREMENTS:

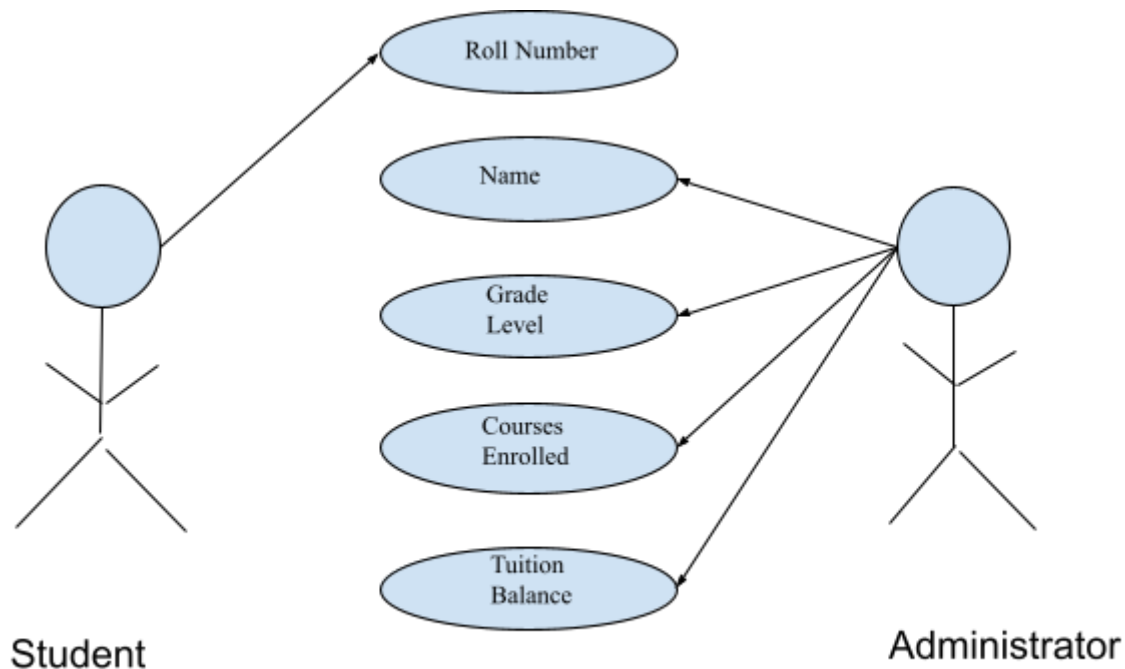
1.3.1 Software Requirements:

1. Operating System - Windows 10.
2. Platform - Eclipse, Command Prompt.
3. Language - Core Java, My SQL.
4. Files - Chrome.

1.4 UML DIAGRAMS

1.4.1 Use Case Diagram

The following UML use case diagram shows the working of the Student Management System. Moreover it has five cases that show the particular functionality of the student. The seven cases are student roll number, student name, grade level, courses enrolled and the tuition balance for each course. The student can view the details of them by entering the roll number and can view the grade level, courses enrolled and the tuition balance for each course which they have paid recently. The administrator can check the name of a particular student, grade level, courses enrolled and the tuition balance whether the student entered the information is correct or not. The interactions of the student and the administrator are what the student management system use case diagram example.



1.5 MODULE

1.5.1 Insert student record:

The insert method is used to inserting the student roll number, name, grade level, courses enrolled and the tuition balance of each course.

Program:

```
private void insertRecord() throws SQLException {
    String sql = "insert into student(name, grade_level, courses_enrolled,
tuition_balance) values (?, ?, ?, ?)";

    PreparedStatement preparedStatement = connection.prepareStatement(sql);
    System.out.println("Enter name");
    scanner.nextLine();
    preparedStatement.setString(1, scanner.nextLine());
    System.out.println("Enter grade_level");
    preparedStatement.setInt(2, scanner.nextInt());
    System.out.println("Enter courses_enrolled");
    scanner.nextLine();
    preparedStatement.setString(3, scanner.nextLine());
    System.out.println("Enter tuition_balance");
    preparedStatement.setInt(4, scanner.nextInt());
    int rows = preparedStatement.executeUpdate();

    if (rows > 0) {
        System.out.println("*****Record inserte
successfully*****");
    }
}
```

1.5.2 Select Student Record:

The select method is used to select the information of a particular student by entering student roll number. By inserting the roll number the student can view his/her details of grade level, courses enrolled and tuition balance.

Program:

```
public void selectRecord() throws SQLException {
    System.out.println("Enter roll number to find result");
    int number = scanner.nextInt();

    String sql = "select * from student where roll_number = "+number;
    Statement statement = connection.createStatement();
    ResultSet result = statement.executeQuery(sql);

    if(result.next()) {
        int rollNumber = result.getInt("roll_number");
        String name = result.getString("name");
        int gradeLevel = result.getInt("grade_level");
        String coursesEnrolled = result.getString("courses_enrolled");
        int tuitionBalance = result.getInt("tuition_balance");

        System.out.println("Roll number is " +rollNumber);
        System.out.println("Name is " +name);
        System.out.println("Grade Level is " +gradeLevel);
        System.out.println("Courses Enrolled is " +coursesEnrolled);
        System.out.println("Tuition Balance is " +tuitionBalance);

    } else {
        System.out.println("*****No record found...*****");
    }
}
```


1.5.3 Update Student Record:

The Update Record is used for updating the student's name, grade level, courses enrolled and tuition balance if we want to update.

Program: (for update record)

```
switch (choice) {
case 1:
    System.out.println("Enter new name");
    scanner.nextLine();
    String newName = scanner.nextLine();
    sqlQuery = sqlQuery + "name = ? where roll_number = "+rollNumber;
    PreparedStatement preparedStatement = connection.prepareStatement(sqlQuery);
    preparedStatement.setString(1, newName);
    int rows = preparedStatement.executeUpdate();
    if(rows > 0) {
        System.out.println("***Record updated successfully...***");
    }
    break;
case 2:
    System.out.println("Enter new grade_level");
    int newgradeLevel = scanner.nextInt();
    sqlQuery = sqlQuery + "grade_level = ? where roll_number = "+rollNumber;
    PreparedStatement preparedStatement1 = connection.prepareStatement(sqlQuery);
    preparedStatement1.setInt(1, newgradeLevel);
    int rows1 = preparedStatement1.executeUpdate();
    if(rows1 > 0) {
        System.out.println("***Record updated successfully...***");
    }
    break;
case 3:
    System.out.println("Enter new courses_enrolled");
    scanner.nextLine();
    String newcoursesEnrolled = scanner.nextLine();
    sqlQuery = sqlQuery + "courses_enrolled = ? where roll_number = "+rollNumber;
    PreparedStatement preparedStatement2 = connection.prepareStatement(sqlQuery);
```

```

preparedStatement2.setString(1, newcoursesEnrolled);
int rows2 = preparedStatement2.executeUpdate();
if(rows2 > 0) {
    System.out.println("***Record updated successfully...***");
}
break;
case 4:
    System.out.println("Enter new tuition_balance");
    int newtuitionBalance = scanner.nextInt();
    sqlQuery = sqlQuery + "tuition_balance = ? where roll_number = "+rollNumber;
    PreparedStatement preparedStatement3 = connection.prepareStatement(sqlQuery);
    preparedStatement3.setInt(1, newtuitionBalance);
    int rows3 = preparedStatement3.executeUpdate();
    if(rows3 > 0) {
        System.out.println("***Record updated successfully...***");
    }
    break;
default:
    break;
}
} else {
    System.out.println("*****Records not found...*****");
}
}

```

1.5.4 Delete Student Record:

The Delete Record is used to delete the details of the student record. By entering the roll number we can delete the particular record of the student if needed.

Program:

```
public void deleteRecord() throws SQLException{
    System.out.println("Enter roll number to delete.");
    int rollNumber = scanner.nextInt();
    String sql = "delete from student where roll_number = "+rollNumber;
    //Statement statement = connection.createStatement();
    PreparedStatement ps = connection.prepareStatement(sql);
    int rows = ps.executeUpdate(sql);
    if(rows >0 ) {
        System.out.println("*****Record is deleted successfully...*****");
    }
}
```

1.5.5 Exit:

The exit is used to exit the page.

Program:

case 5:

```
System.out.println("*****Thank you visit Again!!!*****");  
System.exit(0);
```

1.6 Advantages

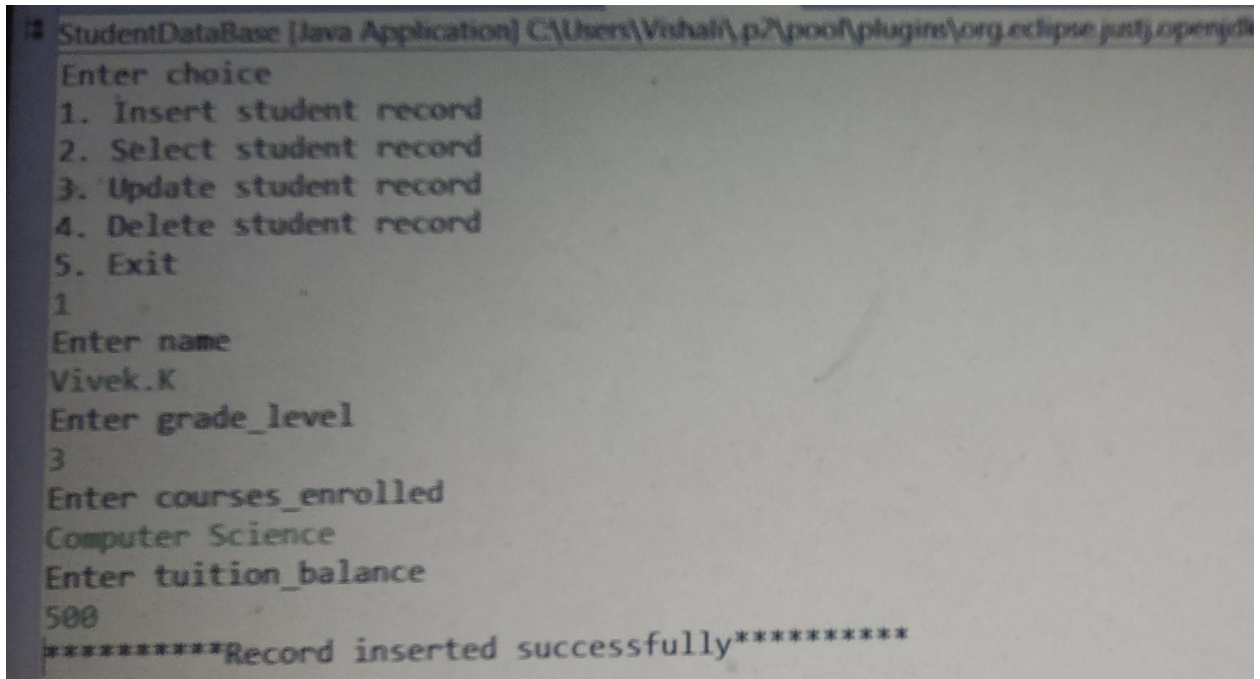
- Eco-friendly: paperwork can be avoided.
- Easier to access.
- Can get the information of the particular student within a few seconds.
- Can delete the records of the particular student.
- Single solution for the total Student management System.
- Can update the student records if necessary.
- Supervise multiple branches.
- Cost efficient and User-friendly.
- Efficient control over student data.
- Monitor student performance.

Chapter 2

PROJECT IMPLEMENTATION

2.1 SCREENS

2.1.1 Insert Student Record Page:



```
StudentDataBase [Java Application] C:\Users\Vishal\p2\poo\plugins\org.eclipse.justj.openjdk
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
1
Enter name
Vivek.K
Enter grade_level
3
Enter courses_enrolled
Computer Science
Enter tuition_balance
500
*****Record inserted successfully*****
```

2.1.2 Select Student Record:

```
Record inserted successfully
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
2
Enter roll number to find result
11001
Roll number is 11001
Name is Vishali.K
Grade Level is 3
Courses Enrolled is Computer Science
Tuition Balance is 100
```

2.1.3 Update Student Record:

2.1.3.1 Name

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
3
Enter roll number to update record.
11003
Roll number is 11003
Name is Hari.S
Grade Level is 3
Courses Enrolled is Java
Tuition Balance is 350
What do you want to update?
1. Name
2. Grade Level
3. Courses Enrolled
4. Tuition Balance
1
Enter new name
Harish.A
***Record updated successfully...***
```

2.1.3.2 Grade Level

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
3
Enter roll number to update re
11003
Roll number is 11003
Name is Harish.A
Grade Level is 3
Courses Enrolled is Java
Tuition Balance is 350
What do you want to update?
1. Name
2. Grade Level
3. Courses Enrolled
4. Tuition Balance
2
Enter new grade_level
4
***Record updated successfully
```


2.1.3.3 Course enrolled

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
3
Enter roll number to update record
11003
Roll number is 11003
Name is Harish.A
Grade Level is 4
Courses Enrolled is Java
Tuition Balance is 350
What do you want to update?
1. Name
2. Grade Level
3. Courses Enrolled
4. Tuition Balance
3
Enter new courses_enrolled
Python
Python 3.11.0 ***
```

2.1.3.4 Tuition Balance

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
3
Enter roll number to update record
11003
Roll number is 11003
Name is Harish.A
Grade Level is 4
Courses Enrolled is Python
Tuition Balance is 350
What do you want to update?
1. Name
2. Grade Level
3. Courses Enrolled
4. Tuition Balance
4
Enter new tuition_balance
400
```

2.1.4 Delete Student Record:

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
4
Enter roll number to delete.
11007
*****Record is deleted successfully...*****
```

2.1.5 Exit:

```
Enter choice
1. Insert student record
2. Select student record
3. Update student record
4. Delete student record
5. Exit
5
*****Thank you visit Again!!!*****
```

REDMI NOTE 5 PRO
MI DUAL CAMERA

Chapter 3

CONCLUSIONS

Student Management System can be used by the administrators to maintain the student records. By achieving this objective, the student can see the details by entering his/her roll number, name, grade level, courses enrolled and tuition balance. This project is useful for the administrators to maintain the student's records easily.