

EduBridge



A project Report on “SCHOOL MANAGEMENT SYSTEM”

Amruta Bondre

Batch-EON (2021-5736)

Under the Guidance of,

Amruta Deore

(Technical Trainer)

EduBridge India Pvt. Ltd.

INDEX

1.INTRODUCTION

2.REQUIREMENT GATHERING

3 IMPLEMENTED SYSTEM

3.1 Coding

3.2 Results (Screen Shots)

4.CONCLUSION

5.REFERENCES

Introduction

- This Project aims to All manually work related to expanse and venue by done by this system. A School management System consist of students and teachers.
- The school management system is a web-based system which will use as a platform for interaction between student, teachers
- The main objective of the School Management System is to manage the details of School, Students, Teachers.
- It manages all the information related to fees and expenses.
- The project is beneficial for school.
- The software system allows the students total fees , paidFees , teacher salary, school revenue and school expenses.

REQUIREMENT GATHERING

SOFTWARE REQUIREMENT

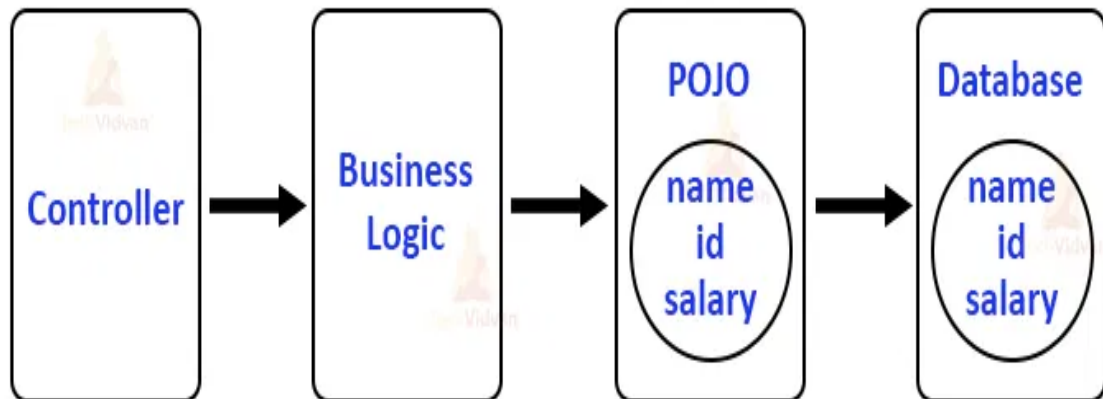
Operating System	:	Windows10
Programming Language	:	Java
IDE's	:	Eclipse

HARDWARE REQUIREMENT

Ram	:	4.00GB
Processor	:	Intel core i3
Hard Disk	:	Minimum 10 GB

JAVA BEANS

Java Beans



1. All JavaBeans are POJOs but not all POJOs are JavaBeans.
 2. Serializable i.e. they should implement Serializable interface. Still, some POJOs who don't implement Serializable interface are called POJOs because Serializable is a marker interface and therefore not of much burden.
 3. Fields should be private. This is to provide the complete control on fields.
 4. Fields should have getters or setters or both.
 5. A no-arg constructor should be there in a bean.
 6. Fields are accessed only by constructor or getter setters.
- Getters and Setters have some special names depending on field name.

What are JavaBean Properties?

A JavaBean property can be accessed by the user of the object. The feature can be of any Java data type, containing the classes that you define. It may be of the following mode: *read*, *write*, *read-only*, or *write-only*. JavaBean features are accessed through two method-

1. `getStudentName ()`

For example, if the Student name is `firstName`, the method name would be `getFirstName()` to read that employee name. This method is known as an **accessor**. Properties of getter methods are as follows:

1. Must be public in nature
2. Return-type should not be void
3. The getter method should be prefixed with the word *get*
4. It should not take any argument

2. `setStudentName ()`

For example, if the Student name is `firstName`, the method name would be `setFirstName()` to write that employee name. This method is known as a **mutator**. Properties of setter methods:

1. Must be public in nature
2. Return-type should be void
3. The setter method has to be prefixed with the word set
4. It should take some argument

CODING

1) Create sms.bean.com package

->Student.java

->Teacher.java

1.1) Student.java

package com.sms.beans;

```
public class Student {  
    private int id;  
    private String name;  
    private String grade;  
    private double feesPaid;  
    private double totalFees;  
    public int getId() {  
        return id;  
    }  
    public void setId(int id) {  
        this.id = id;  
    }  
    public String getName() {  
        return name;  
    }  
    public void setName(String name) {  
        this.name = name;  
    }  
    public String getGrade() {  
        return grade;  
    }  
    public void setGrade(String grade) {  
        this.grade = grade;  
    }  
    public double getFeesPaid() {  
        return feesPaid;  
    }  
    public void setFeesPaid(double feesPaid) {  
        this.feesPaid = feesPaid;  
    }  
}
```

```

    public double getTotalFees() {
        return totalFees;
    }
    public void setTotalFees(double totalFees) {
        this.totalFees = totalFees;
    }

    @Override
    public String toString() {
        return "[id=" + id + ", name=" + name + ", grade=" + grade + ",
        feesPaid=" + feesPaid + ", totalFees="
            + totalFees + "]";
    }
}

```

1.2)Teacher.java

```

package com.sms.beans;

public class Teacher
{
    private String name;
    private int id;
    private double salary;

    //using getter setter method
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public double getSalary() {
        return salary;
    }
}

```



```
    }  
    public void setSalary(double salary) {  
        this.salary = salary;  
    }  
    @Override  
    public String toString() {  
        return " [name=" + name + ", id=" + id + ", salary=" + salary +  
        " ]";  
    }  
}
```

2) com.sms.services package

-> SchoolServiceImpl.java

-> SchoolServiceInt

2.1) SchoolServiceImpl.java

```
package com.sms.services;
```

```
import java.util.List;
```

```
import com.sms.beans.Student;  
import com.sms.beans.Teacher;  
import com.sms.dao.StudentDaoImpl;  
import com.sms.dao.StudentDaoInt;  
import com.sms.dao.TeacherDaoImpl;  
import com.sms.dao.TeacherDaoInt;
```

```
public class SchoolServiceImpl implements SchoolServiceInt {  
    StudentDaoInt studentDao = new StudentDaoImpl();  
    TeacherDaoInt teacherDao = new TeacherDaoImpl();
```

```
    /** Student Details */  
    @Override  
    public void addStudent(Student student) {  
        studentDao.addStudent(student);  
    }
```

```
    @Override  
    public void displayStudentByID(int id) {  
        Student std = studentDao.getStudentByID(id);  
        if (std != null) {  
            System.out.println(std);  
        } else {  
            System.out.println("Wrong Student ID");  
        }  
    }
```

```
    @Override  
    public void displayAllStudent() {  
        List<Student> studentList = studentDao.getAllStudent();
```

```

        for (Student std : studentList) {
            System.out.println(std);
        }
    }

    /** Student Details */

    @Override
    public void addTeacher(Teacher teacher) {
        teacherDao.addTeacher(teacher);
    }

    @Override
    public void displayAllTeacher() {
        List<Teacher> teacherList = teacherDao.getAllTeacher();
        for (Teacher teacherObj : teacherList) {
            System.out.println(teacherObj);
        }
    }

    @Override
    public void displayTeacherByID(int id) {
        Teacher obj = teacherDao.getTeacherByID(id);
        if (obj != null) {
            System.out.println(obj);
        } else {
            System.out.println("Wrong Teacher ID");
        }
    }

    @Override
    public double calculateRevenue() {
        List<Student> studentList = studentDao.getAllStudent();
        double revenue = 0;
        for (Student student : studentList) {
            revenue = revenue + student.getFeesPaid();
        }
    }

```

```

        return revenue;
    }

    @Override
    public double calculateExpenses() {

        List<Teacher> teacherList = teacherDao.getAllTeacher();
        double expenses = 0;
        for (Teacher teacher : teacherList) {
            expenses = expenses + teacher.getSalary();
        }

        return expenses;
    }

```

```

}

```

2.2) SchoolServiceInt

```

package com.sms.services;

import com.sms.beans.Student;
import com.sms.beans.Teacher;

public interface SchoolServiceInt
{
    public double calculateRevenue();
    public double calculateExpenses();

    public void addStudent(Student student);
    public void displayAllStudent();
    public void displayStudentByID(int id);

    public void addTeacher(Teacher teacher);
    public void displayAllTeacher();
    public void displayTeacherByID(int id);
}

```

3)com.sms.dao

- > StudentDaoImpl.java
- > StudentDaoInt.java
- > TeacherDaoImpl.java
- > TeacherDaoInt.java

3.1) StudentDaoImpl.java

```
package com.sms.dao;
```

```
import java.util.ArrayList;  
import java.util.List;
```

```
import com.sms.beans.Student;
```

```
public class StudentDaoImpl implements StudentDaoInt {  
    List<Student> studentsDB = new ArrayList();  
  
    @Override  
    public void addStudent(Student student) {  
        studentsDB.add(student);  
        System.out.println("Student added successfully");  
    }  
  
    @Override  
    public List<Student> getAllStudent() {  
        return studentsDB;  
    }  
  
    @Override  
    public Student getStudentByID(int id) {  
        Student student = null;  
  
        for(Student std:studentsDB) {  
            if(std.getId() == id) {  
                student = std;  
            }  
        }  
        return student;  
    }  
}
```

3.2) StudentDaoInt.java

```
package com.sms.dao;

import java.util.List;
import com.sms.beans.Student;

public interface StudentDaoInt {

    public void addStudent(Student student);
    public List<Student> getAllStudent();
    public Student getStudentByID(int id);
}
```

3.3) TeacherDaoImpl.java

```
package com.sms.dao;

import java.util.ArrayList;
import java.util.List;

import com.sms.beans.Student;
import com.sms.beans.Teacher;

public class TeacherDaoImpl implements TeacherDaoInt {

    List<Teacher> teacherDB = new ArrayList();

    @Override
    public void addTeacher(Teacher teacher) {
        teacherDB.add(teacher);
        System.out.println("Teacher added successfully");
    }

    @Override
    public List<Teacher> getAllTeacher() {
        return teacherDB;
    }

    @Override
    public Teacher getTeacherByID(int id) {
```

```
        Teacher teacher = null;

        for (Teacher obj : teacherDB) {
            if (obj.getId() == id) {
                teacher = obj;
            }
        }
        return teacher;
    }
}
```

3.4) TeacherDaoInt.java

```
package com.sms.dao;
import java.util.List;

import com.sms.beans.Teacher;

public interface TeacherDaoInt {
    public void addTeacher(Teacher teacher);
    public List<Teacher> getAllTeacher();
    public Teacher getTeacherByID(int id);

}
```

4)com.sms.client

->sms client .java

4.1) sms client .java

```
package com.sms.client;
```

```
import java.util.Scanner;
```

```
import com.sms.beans.Student;
```

```
import com.sms.beans.Teacher;
```

```
import com.sms.services.SchoolServiceImpl;
```

```
import com.sms.services.SchoolServiceInt;
```

```
public class SmsClient {
```

```
    public static void main(String[] args) {
```

```
        SchoolServiceInt schoolService = new SchoolServiceImpl();
```

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

```
        do {
```

```
            System.out.println("\n=====");
```

```
            System.out.println("School Management System");
```

```
            System.out.println("=====");
```

```
            System.out.println("1. Register Student ");
```

```
            System.out.println("2. Display All Student ");
```

```
            System.out.println("3. Display Student By ID ");
```

```
            System.out.println("-----");
```

```
            System.out.println("4. Register Teacher ");
```

```
            System.out.println("5. Display All Teacher ");
```

```
            System.out.println("6. Display Teacher By ID ");
```

```
            System.out.println("-----");
```

```
            System.out.println("7. Total Revenue ");
```

```
            System.out.println("8. Total Expences ");
```

```
            System.out.println("-----");
```

```
            System.out.println("9. Exit ");
```

```
            System.out.println("-----\n");
```

```
            System.out.print("Enter your Choice :: ");
```

```
            int choice = sc.nextInt();
```



```
switch (choice) {
case 1:
    Student student = new Student();

    System.out.print("Enter ID = ");
    student.setId(sc.nextInt());

    System.out.print("Enter Name = ");
    student.setName(sc.next());

    System.out.print("Enter Grade = ");
    student.setGrade(sc.next());

    System.out.print("Enter feesPaid = ");
    student.setFeesPaid(sc.nextDouble());

    System.out.print("Enter totalFees = ");
    student.setTotalFees(sc.nextDouble());

    schoolService.addStudent(student);
    break;

case 2:
    schoolService.displayAllStudent();
    break;

case 3:
    System.out.print("Enter Student ID = ");
    int id = sc.nextInt();
    schoolService.displayStudentByID(id);
    break;

case 4:
    Teacher teacher = new Teacher();

    System.out.print("Enter ID = ");
    teacher.setId(sc.nextInt());
    ;

    System.out.print("Enter Name = ");
    teacher.setName(sc.next());
```

```

        System.out.print("Enter Salary = ");
        teacher.setSalary(sc.nextDouble());

        schoolService.addTeacher(teacher);
        break;

    case 5:
        schoolService.displayAllTeacher();
        break;

    case 6:
        System.out.print("Enter Teacher ID = ");
        int tID = sc.nextInt();
        schoolService.displayTeacherByID(tID);
        break;

    case 7:
        double revenue =
schoolService.calculateRevenue();
        System.out.println("School Revenue = "+revenue);
        break;

    case 8:
        double expenses =
schoolService.calculateExpenses();
        System.out.println("School expenses =
"+expenses);
        break;

    case 9:
        System.out.println("Exit");
        System.exit(0);
    default:
        System.out.println("Wrong Choice.\n Enter 1 or 2
or 3 .....");
    }
    } while (true);

    }

}

```

SCREENSHOT

Add Student

SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)

```
=====
School Management System
=====
1. Register Student
2. Display All Student
3. Display Student By ID
-----
4. Register Teacher
5. Display All Teacher
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expences
-----
9. Exit
-----

Enter your Choice ::
```

Added Student

```
Enter your Choice :: 1
Enter ID = 101
Enter Name = Amruta
Enter Grade = a
Enter feesPaid = 888
Enter totalFees = 5000
Student added successfully
```

```
=====
School Management System
=====
1. Register Student
2. Display All Student
3. Display Student By ID
-----
4. Register Teacher
5. Display All Teacher
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expences
```

Add teacher

```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)

9. Exit
-----

Enter your Choice :: 4
Enter ID = 12
Enter Name = T1
Enter Salary = 234
Teacher added successfully

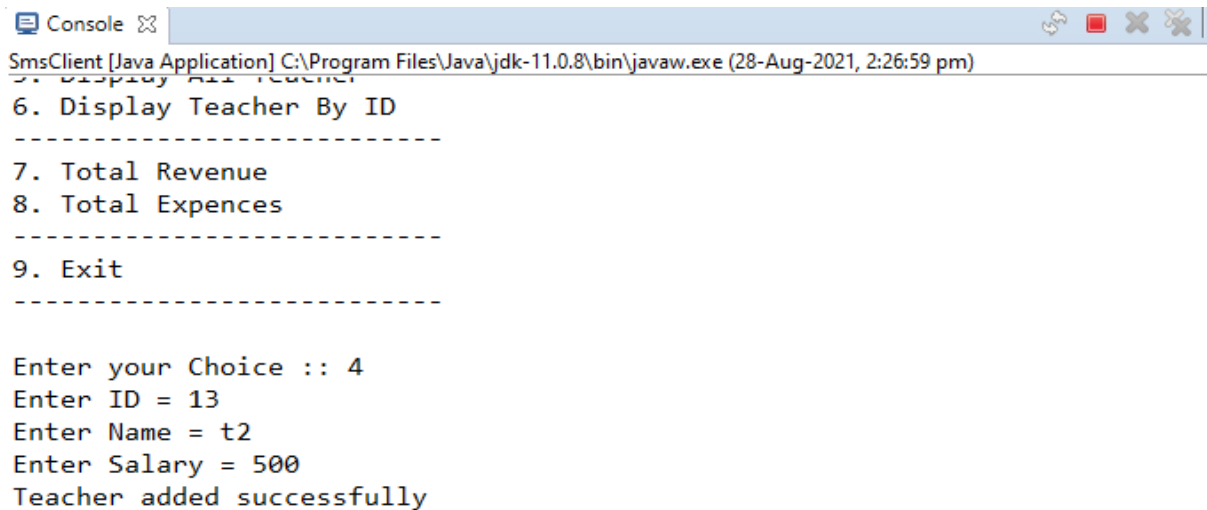
=====
School Management System
=====
1. Register Student
2. Display All Student
3. Display Student By ID
-----
4. Register Teacher
5. Display All Teacher
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expenses
```

Display Student List

```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)

Enter your Choice :: 2
[id=101, name=Amruta, grade=a, feesPaid=888.0, totalFees=5000.0]
[id=102, name=amu, grade=a, feesPaid=245.0, totalFees=5000.0]
```

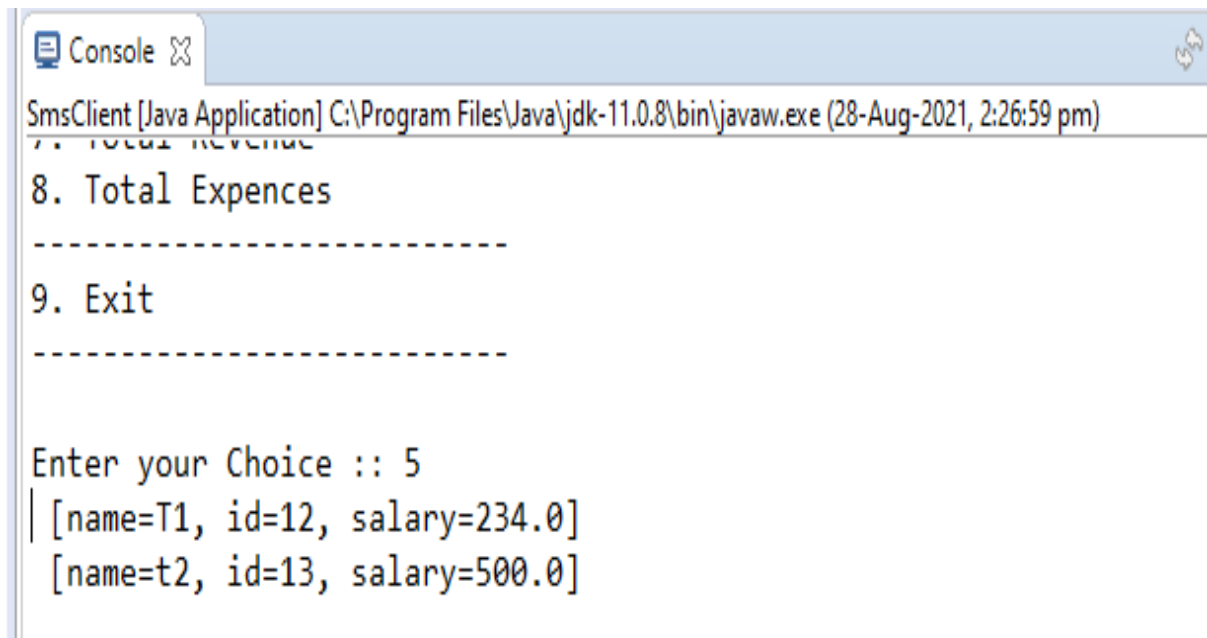
Add Teacher



```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expenses
-----
9. Exit
-----

Enter your Choice :: 4
Enter ID = 13
Enter Name = t2
Enter Salary = 500
Teacher added successfully
```

Display Teacher List



```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)
7. Total Revenue
8. Total Expenses
-----
9. Exit
-----

Enter your Choice :: 5
[name=T1, id=12, salary=234.0]
[name=t2, id=13, salary=500.0]
```

School Revenue

```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2
7. Total Revenue
8. Total Expences
-----
9. Exit
-----
|
Enter your Choice :: 7
School Revenue = 1133.0

=====
School Management System
=====
1. Register Student
2. Display All Student
3. Display Student By ID
-----
4. Register Teacher
5. Display All Teacher
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expences
```

School Expenses

```
Console
SmsClient [Java Application] C:\Program Files\Java\jdk-11.0.8\bin\javaw.exe (28-Aug-2021, 2:26:59 pm)

Enter your Choice :: 8
School expenses = 734.0

=====
School Management System
=====
1. Register Student
2. Display All Student
3. Display Student By ID
-----
4. Register Teacher
5. Display All Teacher
6. Display Teacher By ID
-----
7. Total Revenue
8. Total Expences
-----
9. Exit
-----

Enter your Choice ::
<
```

CONCLUSION

The project work titled “School Management System” has been designed using Java where in many friendly from controls have been added in order to make it a used interactive application. The system is developed in such a way that the user with common knowledge of computers can handle it easily. The system developed has proved to be user friendly and efficient in achieving basic goals.

The main idea was to develop an easy to display school related expanses and revenue . up to date all fees and salary related record are stored.

REFERENCES

- JAVA Technologies.
- JAVA Compile Reference.
- JAVA TUTORIAL.
- JAVA T POINT.
- WWW.EDUBRIDGEINDIA.COM

