



**PES University, Bengaluru**

(Established under Karnataka Act 16 of 2013)

**Department of Computer Science & Engineering**

**Session: Jan - May 2022**

**UE19CS353 – Object Oriented Analysis and Design with Java**  
**Theory ISA (Mini Project)**

Report on

**UNIVERSITY MANAGEMENT SYSTEM**

**By:**

**Sneha Adhikary – PES2UG19CS392**

**Shreesh Devi – PES2UG19CS382**

**Shishir Menon – PES2UG19CS380**

**6<sup>th</sup> Semester F**

## **Table of Contents**

No.	Chapter Name	Page number
1.	<b>Project Description</b> a. Link to Github repository	2.
2.	<b>Analysis and Design Models</b>	2.
3.	<b>Tools and Frameworks Used</b>	3.
4.	<b>Design Principles and Design Patterns Applied</b>	4.
5.	<b>Application Screenshots</b>	7.
6.	<b>Team member contributions</b>	9.
7.	<b>Conclusion</b>	10.
8.	<b>References</b>	10.

## **List of tables and figures**

Name	Number
<b>Use Case diagram</b>	<b>Fig.2.1</b>
<b>Class diagram</b>	<b>Fig.2.2</b>
<b>Facade design pattern</b>	<b>Fig.4.1</b>
<b>Factory design pattern</b>	<b>Fig.4.2</b>
<b>Login page</b>	<b>Fig. 5.1</b>
<b>Main page</b>	<b>Fig. 5.2</b>
<b>About us page</b>	<b>Fig. 5.3</b>
<b>Student fee form page</b>	<b>Fig. 5.4</b>
<b>Add student page</b>	<b>Fig. 5.5</b>

# **1. Project Description**

- a. [GITHUB REPO](https://github.com/Sneha0811/University_Management_System) -[https://github.com/Sneha0811/University\\_Management\\_System](https://github.com/Sneha0811/University_Management_System)

The University Management System is a project written entirely in Java. It is a fully functional Java project which includes information about colleges, universities, and schools in general. It was created for universities and its affiliated institutions to conduct, monitor, and analyze complicated activities such as centralized admission, centralized examinations, attendance management and much more.

## **2. Analysis and Design Models**

Use Case diagram

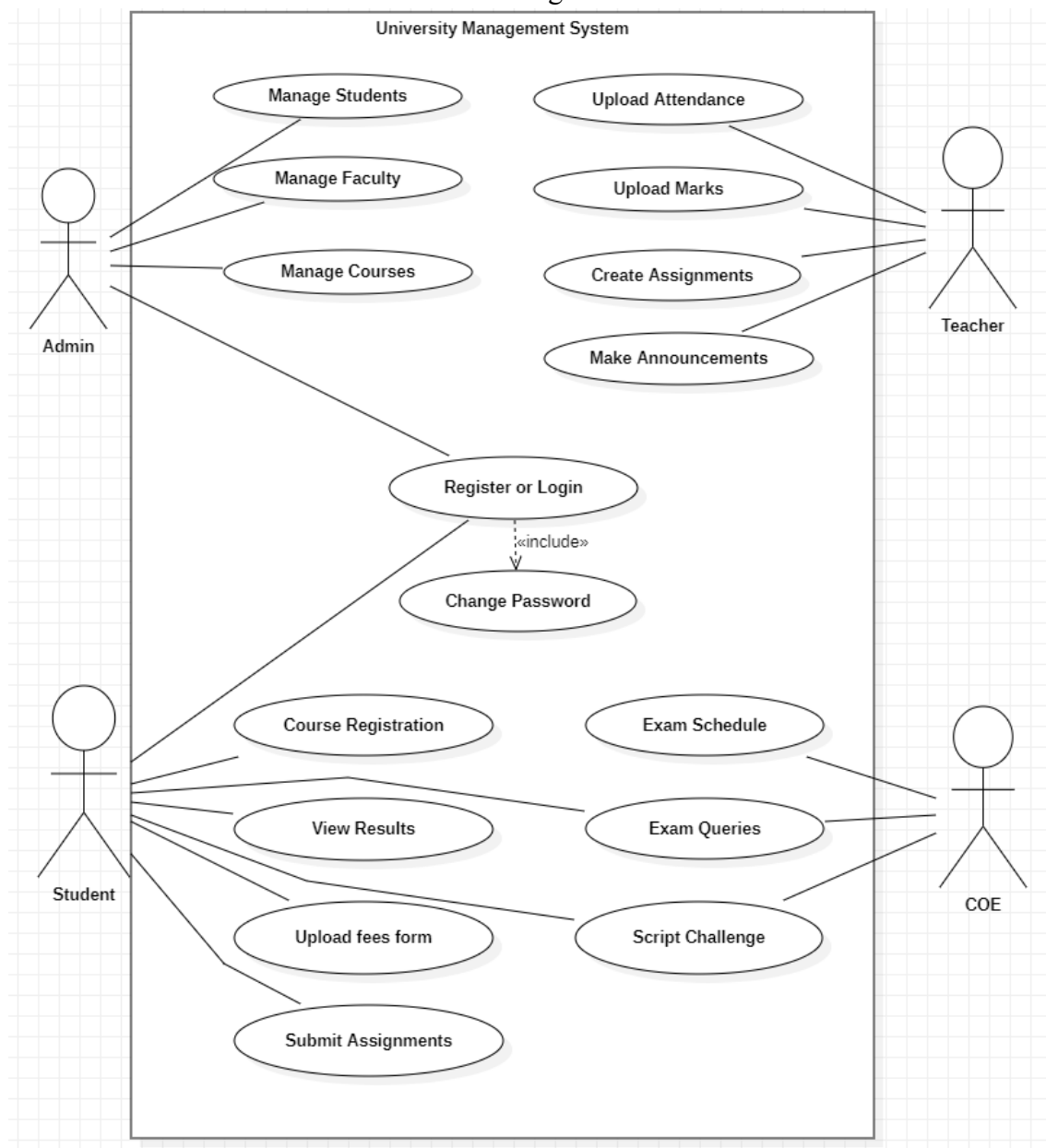


Fig. 2.1

Class diagram

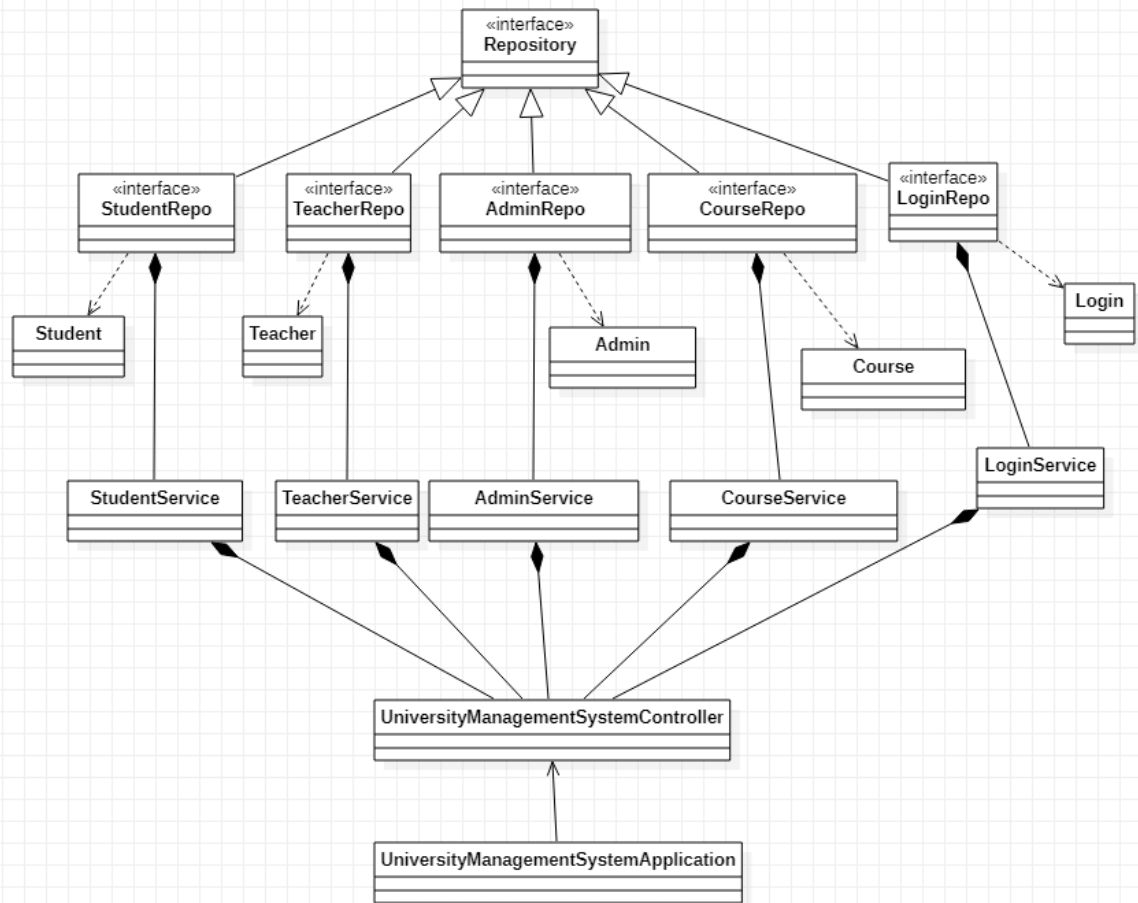


Fig. 2.2

### **3. Tools and frameworks**

- MySQL
- Eclipse IDE
- JDBC
- Swing

## **4.Design Principles and Design Patterns**

### a. MVC

- i. Model: Handles data logic.
- ii. View: It displays the information from the model to the user.
- iii. Controller: It controls the data flow into a model object and updates the view whenever data changes.

In our project-

Data Access Objects(Model) – The Instructor DAO, Course DAO and Student DAO are responsible for interfacing with the Instructor Table, Course Table and the Student Table respectively.

Services(View) – This is the intermediary layer between the controller and the database. The main job of services is to integrate data from multiple repositories/DAOs. For example, the Instructor Service uses both Instructor DAO and Course DAO to get its data. Moreover, the Student Service uses both Student DAO and Course DAO to get its data.

Controllers – The controllers handle requests. Store data in the model and send it to the appropriate view template.

### b. Design principles

- i. Single-responsibility Principle-

A class should have one and only one reason to change, meaning that a class should have only one job.

In our project - we have made the classes as single responsibility classes so that each of them only have one job.

### c. Design pattern

- i. Facade -

Facade is a part of the Gang of Four design pattern and it is categorized under Structural design patterns. It acts as an interface

that hides the complexities.

In our project - the JDBC interface can be called a facade because, we as users or clients create a connection using the “java.sql.Connection” interface, the implementation of which is dealt by the provider of the driver.

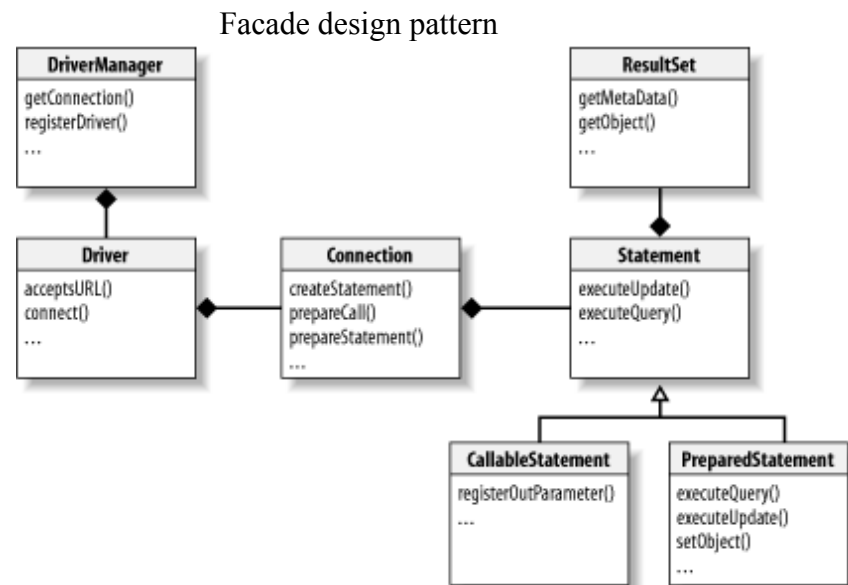


Fig. 4.1

ii. Factory-

Factory design pattern -We defined a factory method inside an interface. The subclass implements the above factory method and decide which object to create. Since this design patterns talk about instantiation of an object and so it comes under the category of creational design pattern.

In our project - The adding Teacher and the adding Student classes is making a use of factory design pattern.

## Factory design pattern

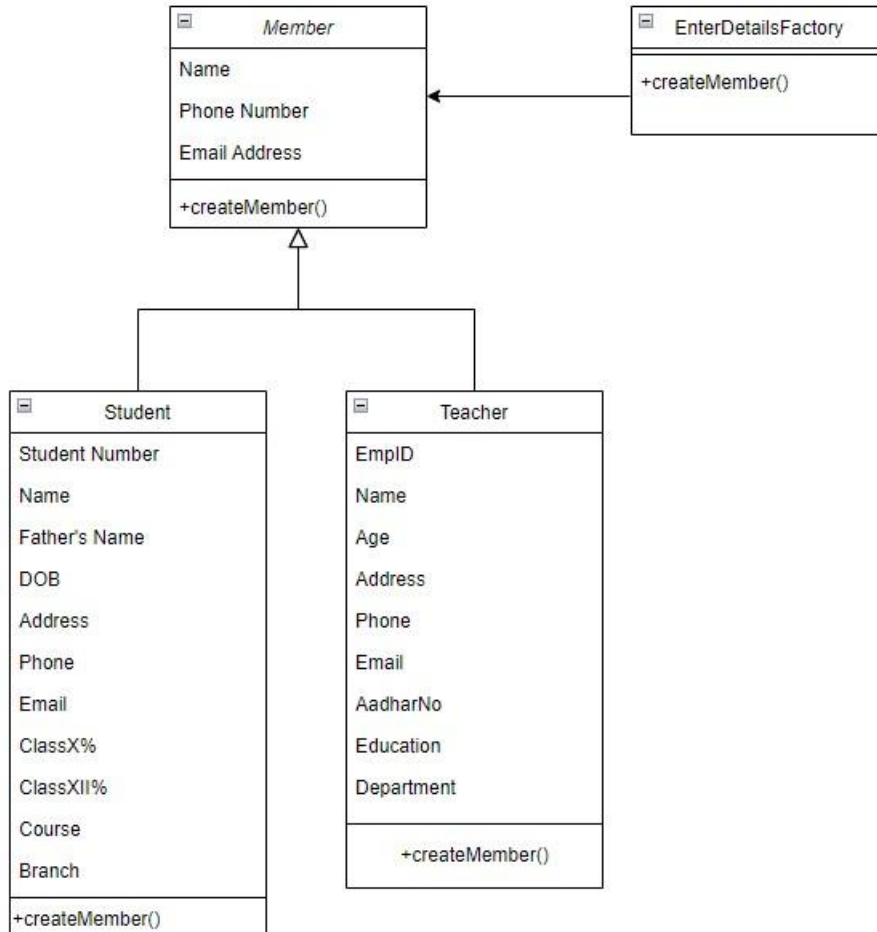


Fig. 4.2

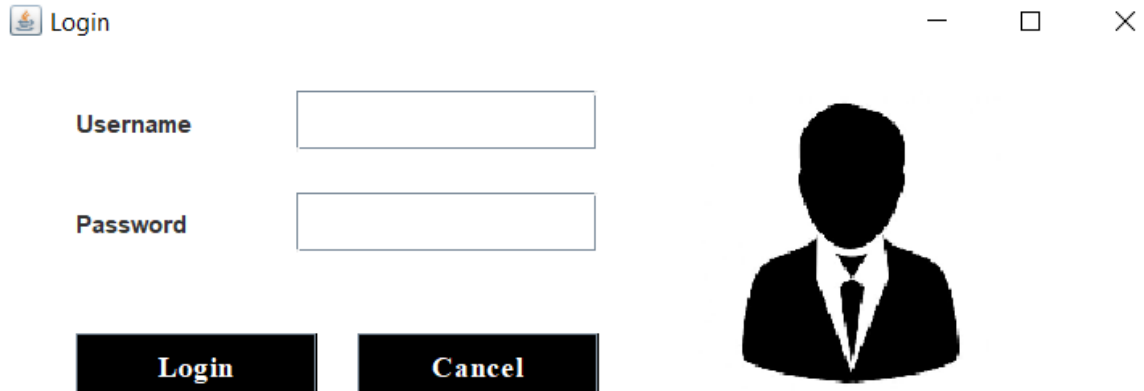
### iii. Singleton-

The singleton pattern is a software design pattern that restricts the instantiation of a class to one "single" instance. This is useful when exactly one object is needed to coordinate actions across the system.

In our system - each class creates only a single instance hence singleton design pattern is used.

## 5. Application screenshots

Login page



The screenshot shows a login window titled "Login" with a small icon on the left and standard window controls (minimize, maximize, close) on the right. The window contains two input fields: "Username" and "Password". Below these fields are two buttons: "Login" and "Cancel". To the right of the input fields is a black silhouette of a person in a suit and tie.

Fig. 5.1

Main page

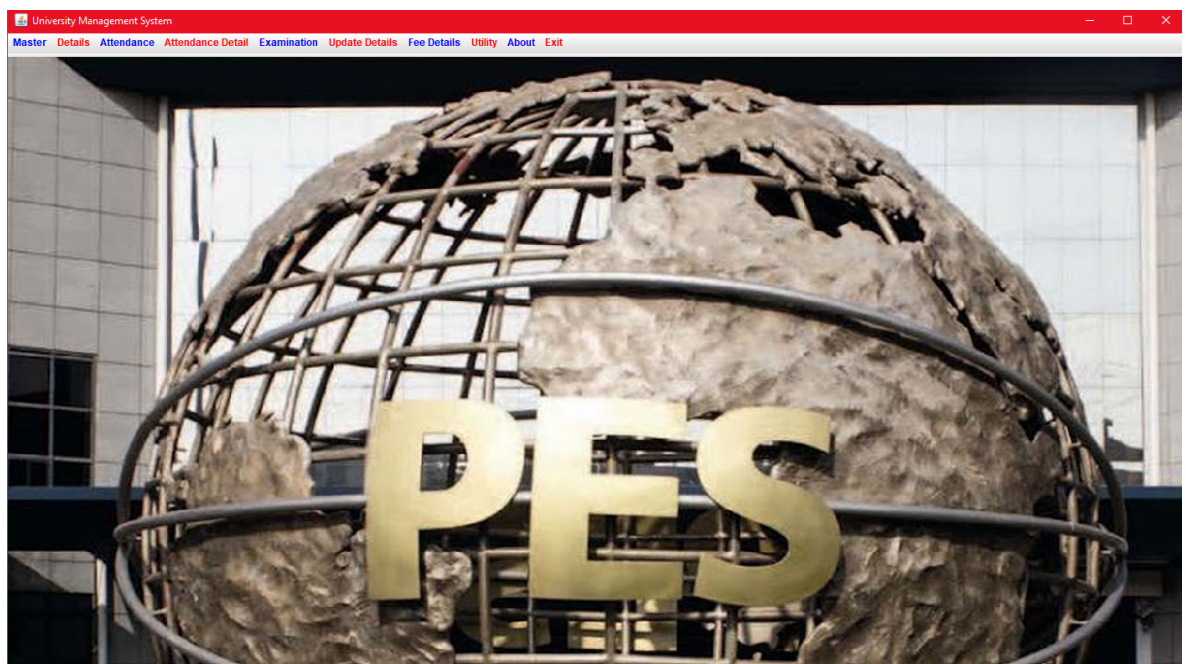


Fig. 5.2

About us page





Fig. 5.3

Student fee form page

A screenshot of a web application window titled "Student Fee Form". The window has a light gray header bar with standard window controls (minimize, maximize, close). The main content area is white and contains a form titled "Fee-Form". The form has the following fields: "Select Roll No" (dropdown menu with value "15333347"), "Name" (text input with value "shishir"), "Father's Name" (text input with value "Rajat"), "Course" (dropdown menu with value "B.Tech"), "Branch" (dropdown menu with value "CSE"), "Semester" (dropdown menu with value "6th"), and "Total Payable" (text input with value "68000"). At the bottom of the form are two buttons: "Pay" and "Back".

Fig. 5.4  
Add student page

Add Student

### New Student Details

Name

Age

Address

Email Id

Class XII(%)

Roll No

Branch

Father's Name

DOB (dd/mm/yyyy)

Phone

Class X(%)

Aadhar No

Course

Submit

Cancel

Fig. 5.5

## **6.Team member contributions**

NAME	PART CONTRIBUTED
Shishir Menon	Major Task-Faculty Minor Task-Server / Database
Shreesh Devi	Major Task-Courses Minor Task-Admin
Sneha Adhikary	Major task - Student module Minor task- Register /login

## **7. Conclusion**

A university management system using swing which was done with the help of MVC architectural pattern. We have used factory and facade design patterns. We have used the Single-responsibility Principle as a design principle.

## **8. References**

<https://www.javatpoint.com/java-swing>

<https://docs.oracle.com/javase/tutorial/jdbc/overview/architecture.html>