FEES MANAGEMENT SYSTEM

AKASH KUMAR PES1UG20CS027

AAMIR IQBAL PES1UG21CS801

ABHISHEK DEVKATE PES1UG21CS802

Synopsis:

A fees management system is a software application designed to manage the fees and payments related to academic institutions or other organisations. The system typically includes a database that stores student or member information, fee structures, payment schedules.

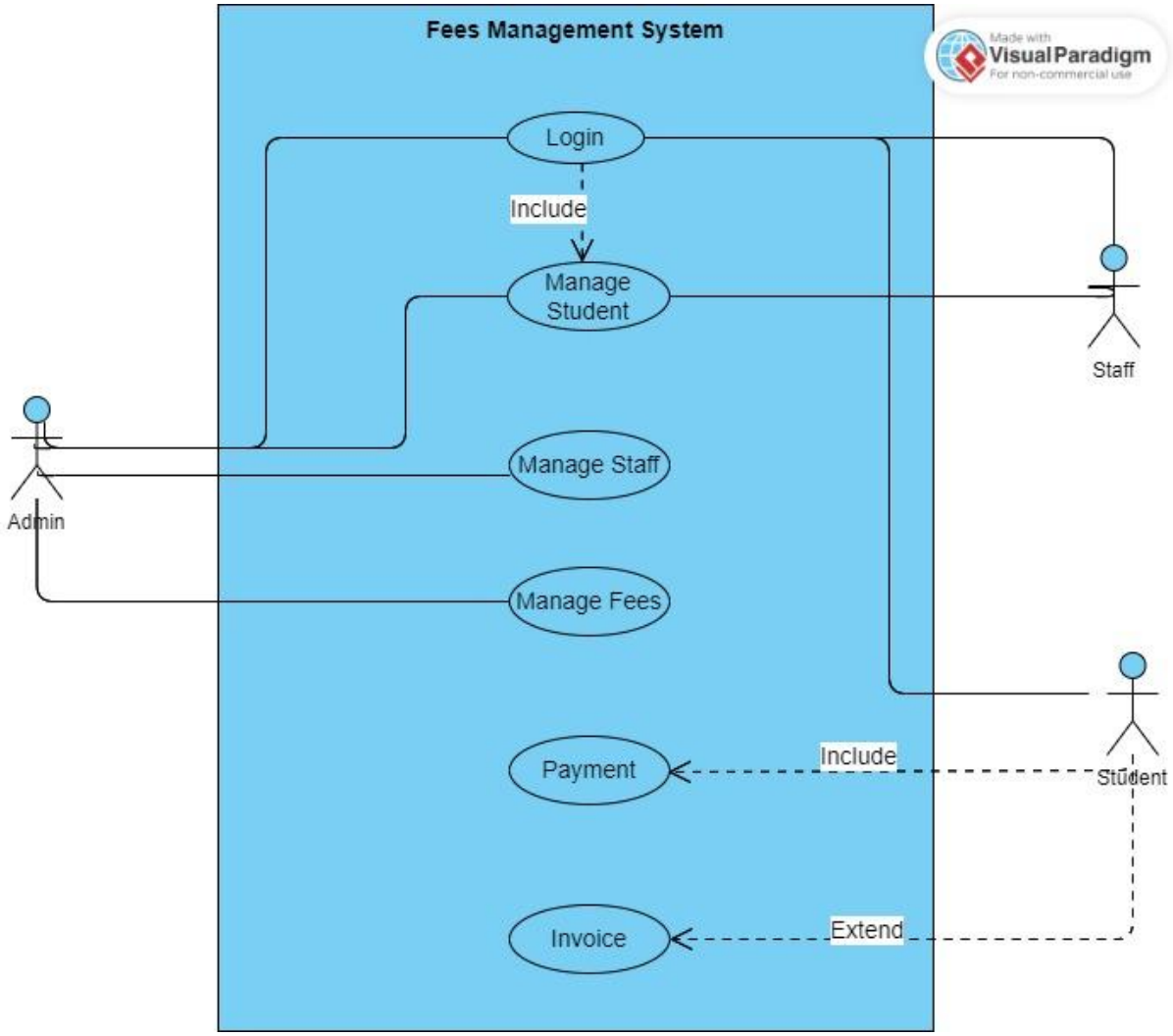
The system streamlines the process of fee management, reducing the workload of administrative staff and ensuring that payments are accurately recorded and tracked. The system can also generate invoices and receipts, automate payment reminders, and provide real-time access to financial data.

Some of the key features of a fees management system may include online payment processing, fee collection reports, fee waiver management, and customised fee structures. The system may also be integrated with other software applications such as accounting and student information systems.

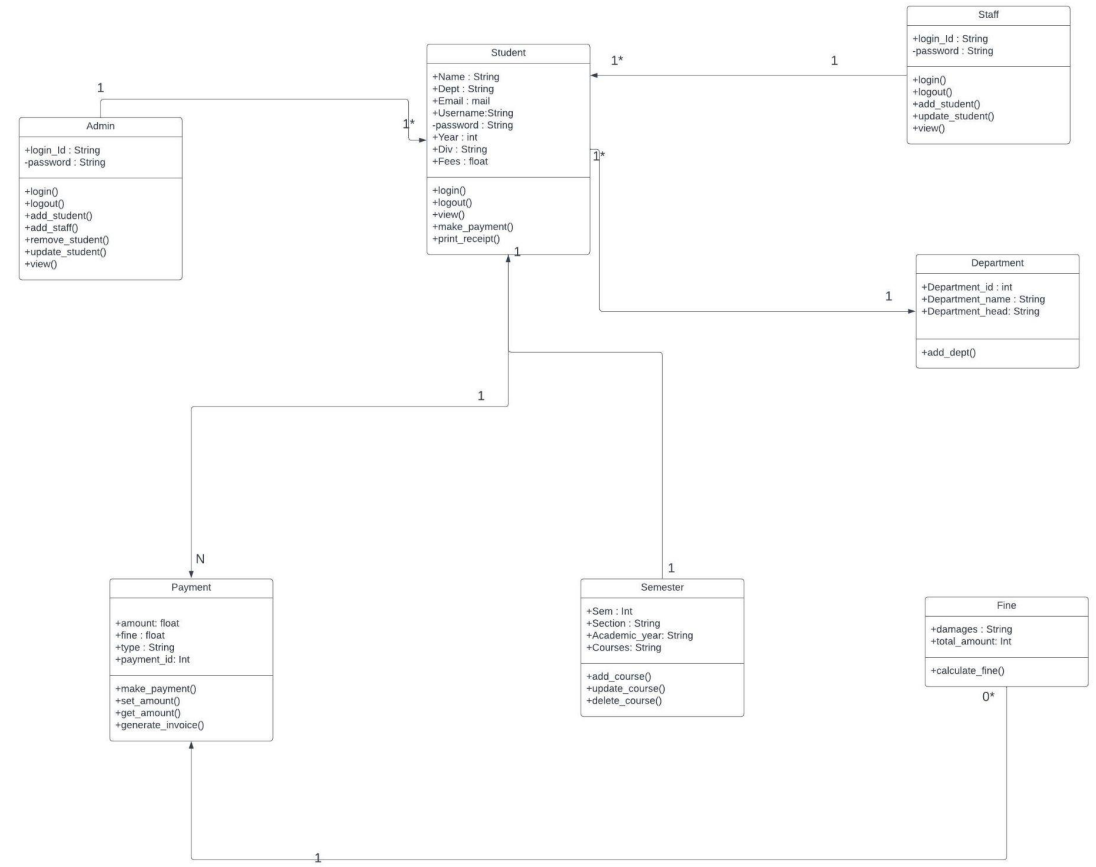
Benefits of using a fees management system include increased efficiency, reduced errors, and improved financial management. The system can also provide transparency to parents, students, and other stakeholders, enabling them to view fee information and transaction histories in real-time.

Overall, a fees management system can help educational institutions and other organisations manage their finances more effectively and efficiently, while providing a better user experience for students and members.

Use Case Diagram:

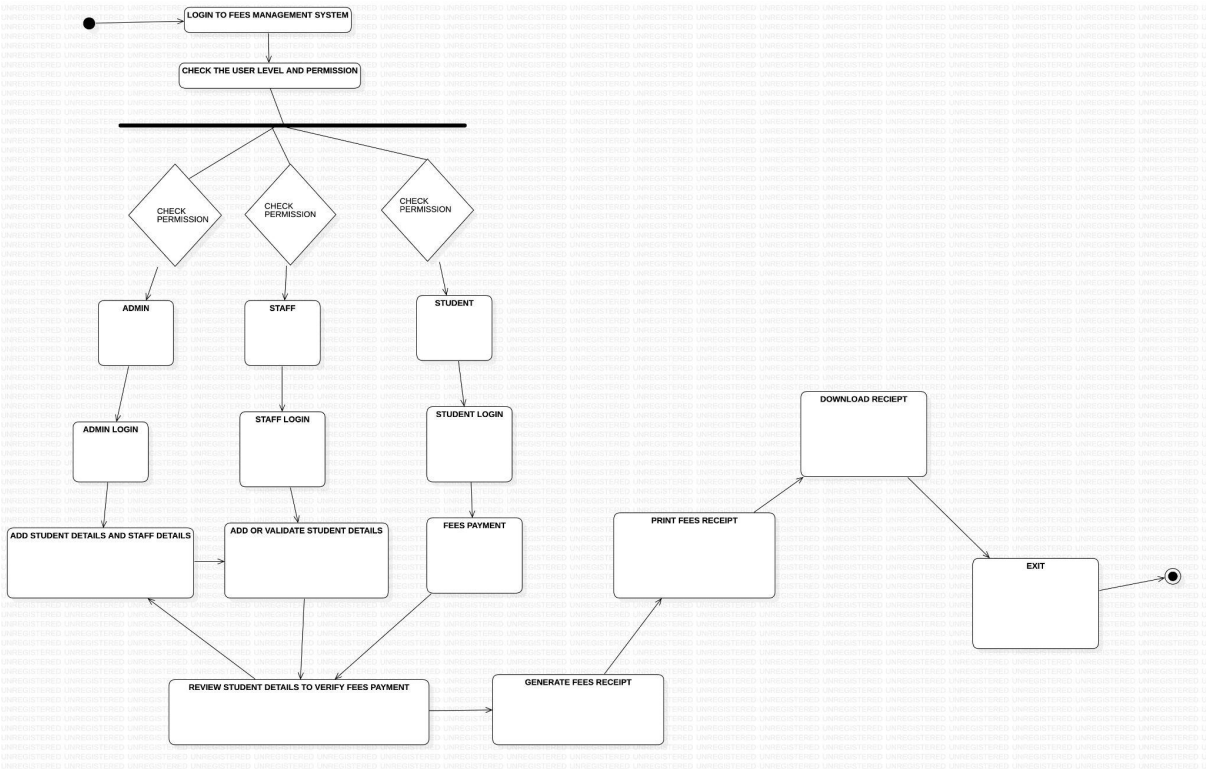


Class Diagram:

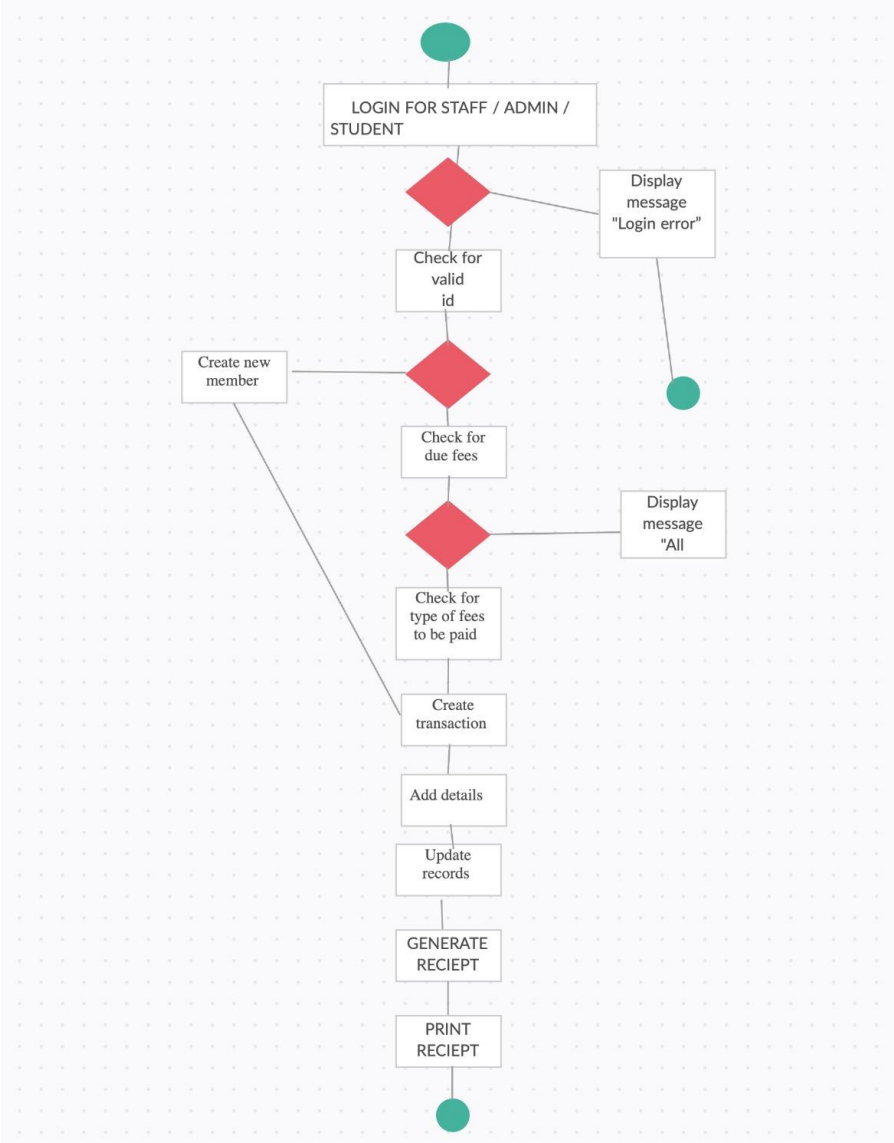


Activity And State Diagram

Activity Diagram:



State Diagram:



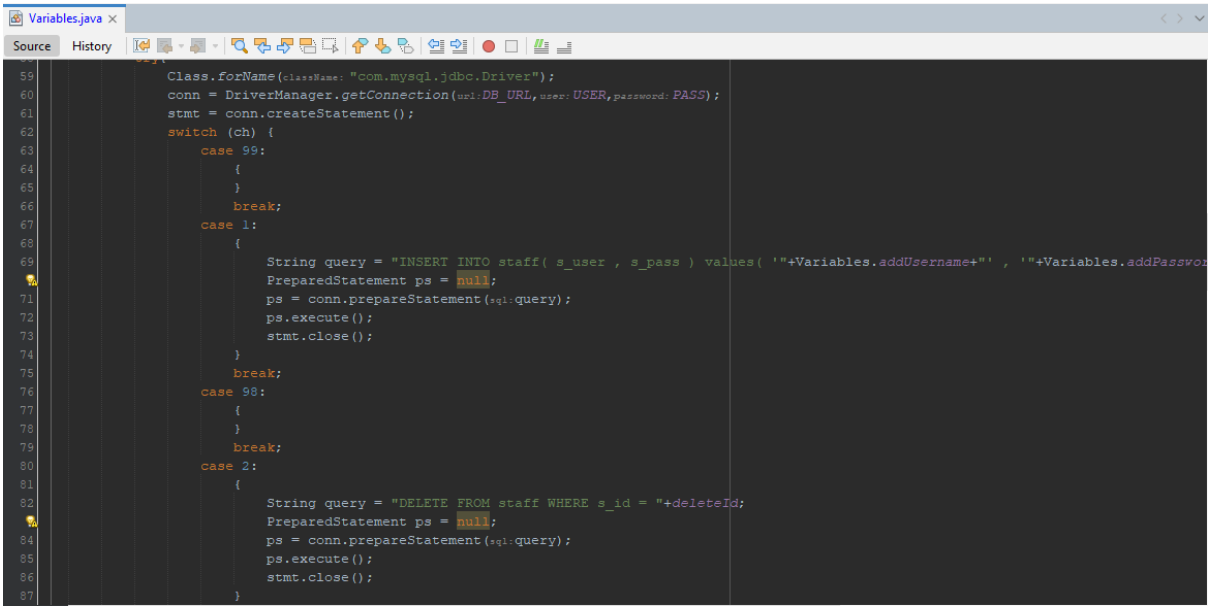
Design Principles Used:

- **Single Responsibility Principle(SRP)**: Each class in the project has a single responsibility, and its behavior is focused on that responsibility only.
- **Open/Closed Principle (OCP)**: The project is designed in a way that allows extension but does not require modification of the existing code.
- **Separation of Concerns(SoC)**: The project separates different concerns and functionalities into different classes, allowing for better maintainability and scalability.

Design Patterns Used:

The **architectural patterns** used are:

- **Model-View-Controller (MVC)** - This pattern is used to separate the application into three interconnected components, namely the Model, View, and Controller.
- **Abstract Factory Pattern:**Single factory class that returns the different subclasses based on the input provided and the factory class uses `if-else` or `switch` statements to achieve this.



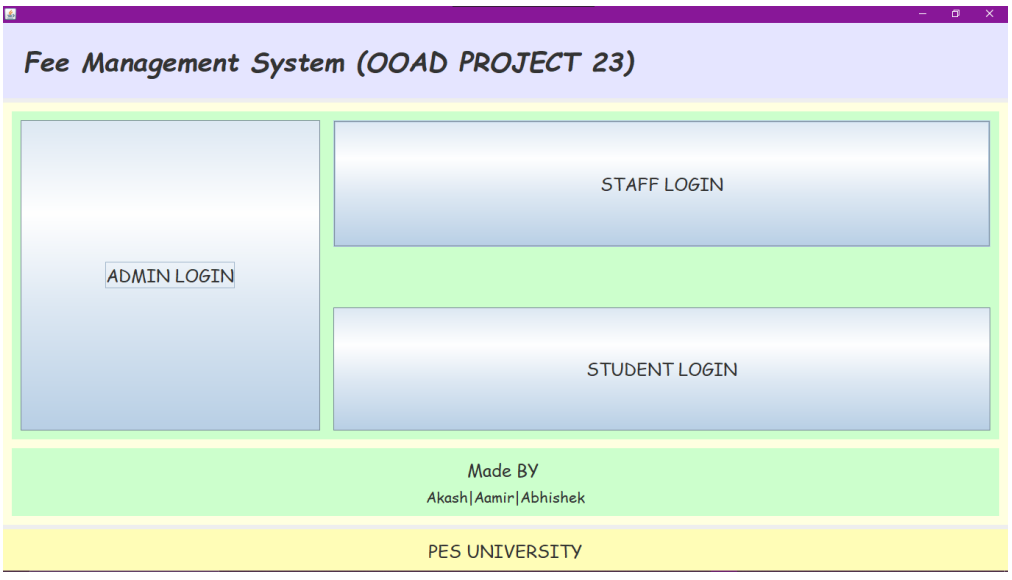
```
Variables.java
Source  History
                case 3:
                    {
//
                        String sql;
                        String n_name = Variables.nName;
                        String n_dept = Variables.nDept;
                        String n_year = Variables.nYear;
                        String n_div = Variables.nDiv;
                        int n_fees = Variables.nFees;
                        int remaining = 70500 - n_fees;
                        sql = "INSERT INTO `student`(`name`, `dept`, `year`, `div`, `fees` , `remaining`) "
                              +"VALUES ('"+n_name+"' , '"+n_dept+"' , '"+n_year+"' , '"+n_div+"' , '"+n_fees+"', '"+remaining+"')";
                        PreparedStatement preparedStmt = conn.prepareStatement(sql);
                        PreparedStatement ps = null;
                        ps = conn.prepareStatement(sql);
                        ps.execute();
                        stmt.close();
                    }
                    break;
```

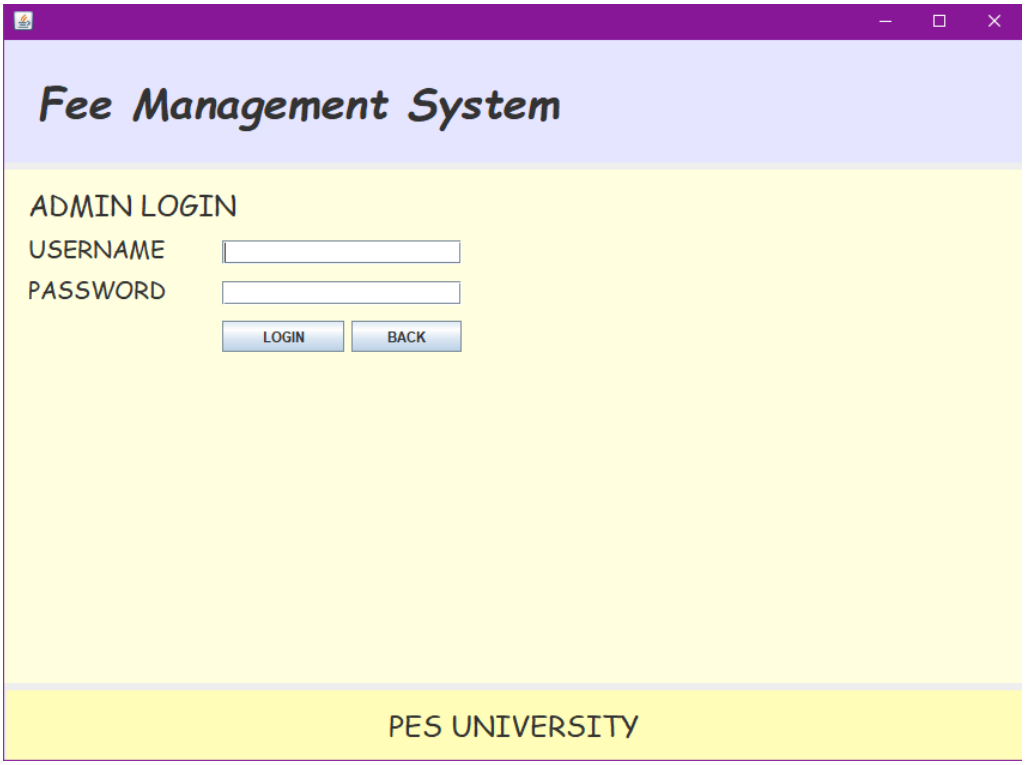
- **Command Pattern:**
  Used to implement loose-coupling in a request-response model. In this pattern, the request is sent to the invoker and the invoker passes it to the encapsulated command object. The command object passes the request to the appropriate method of receiver to perform the specific action.

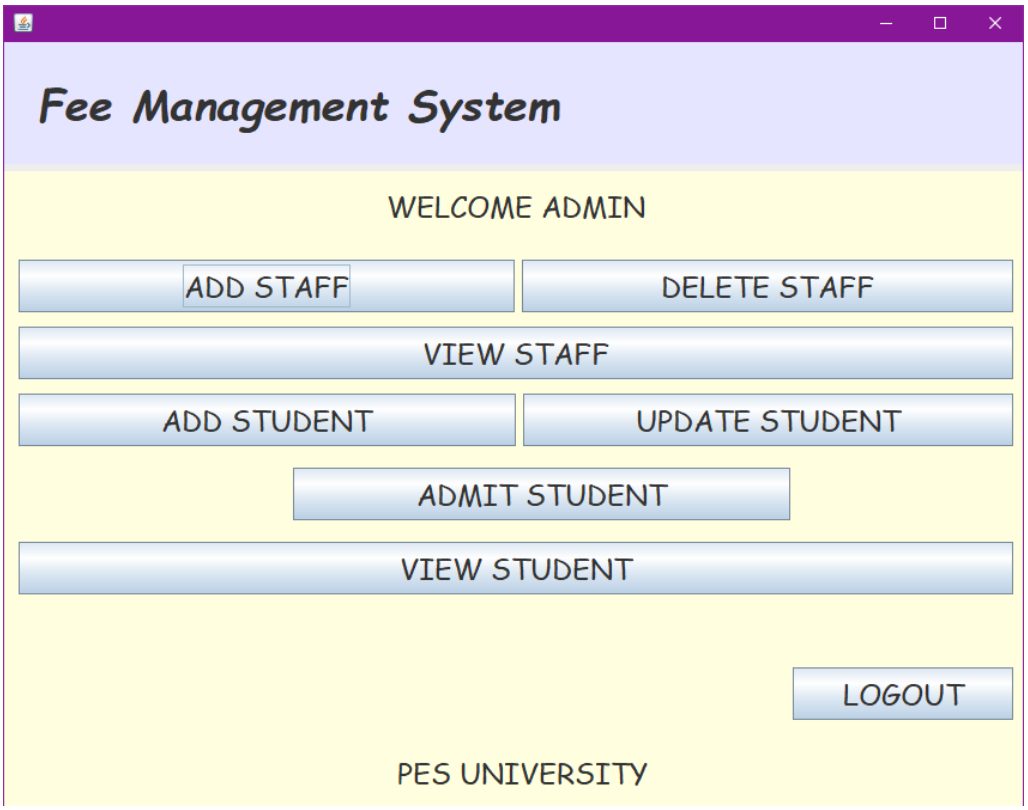
```
public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new AdminLogin().setVisible(b: true);
        }
    });
}
```

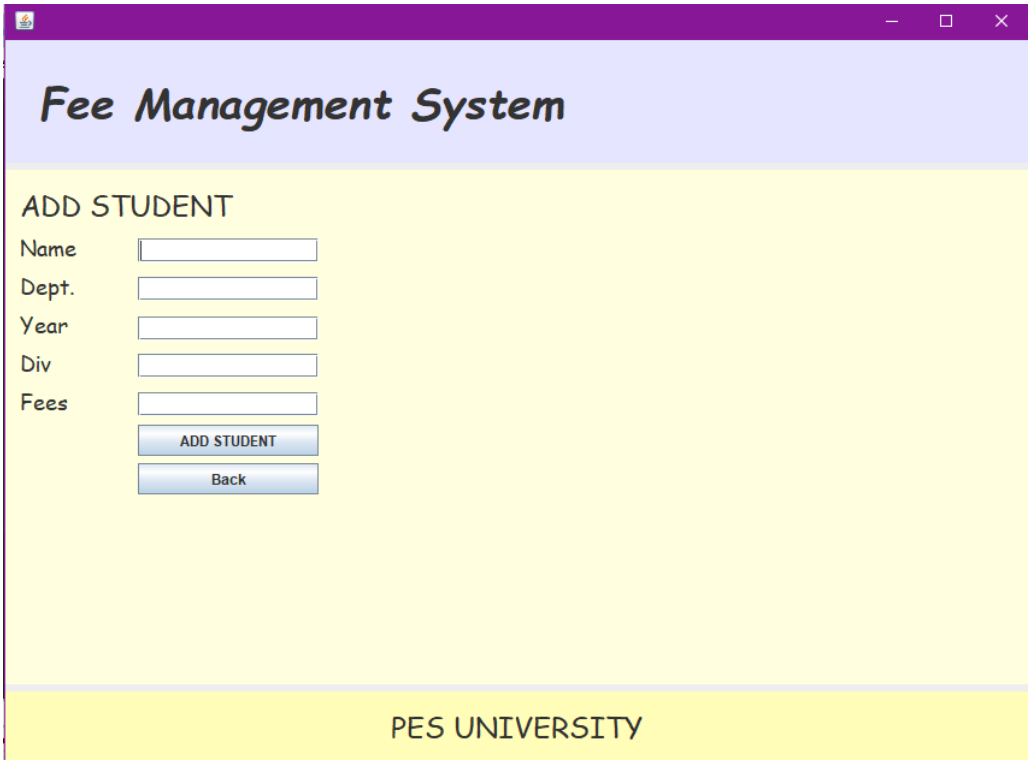
Screenshots of the GUI

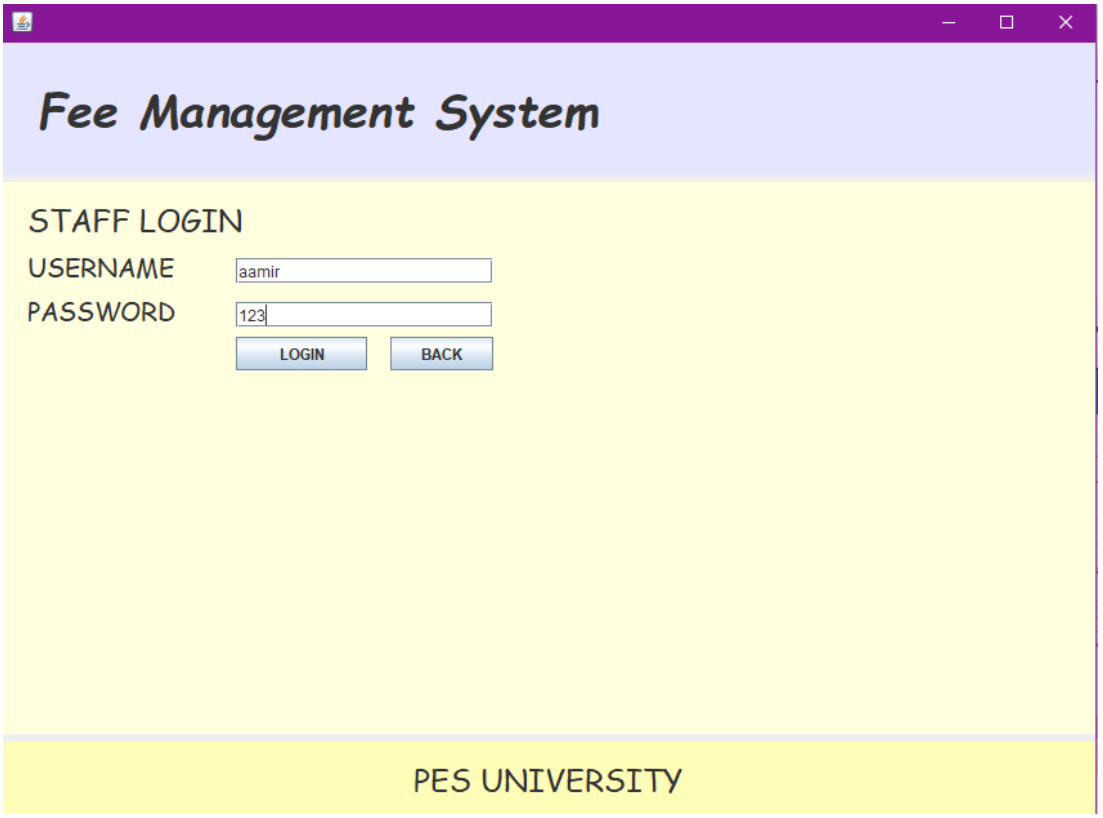
Landing Page:

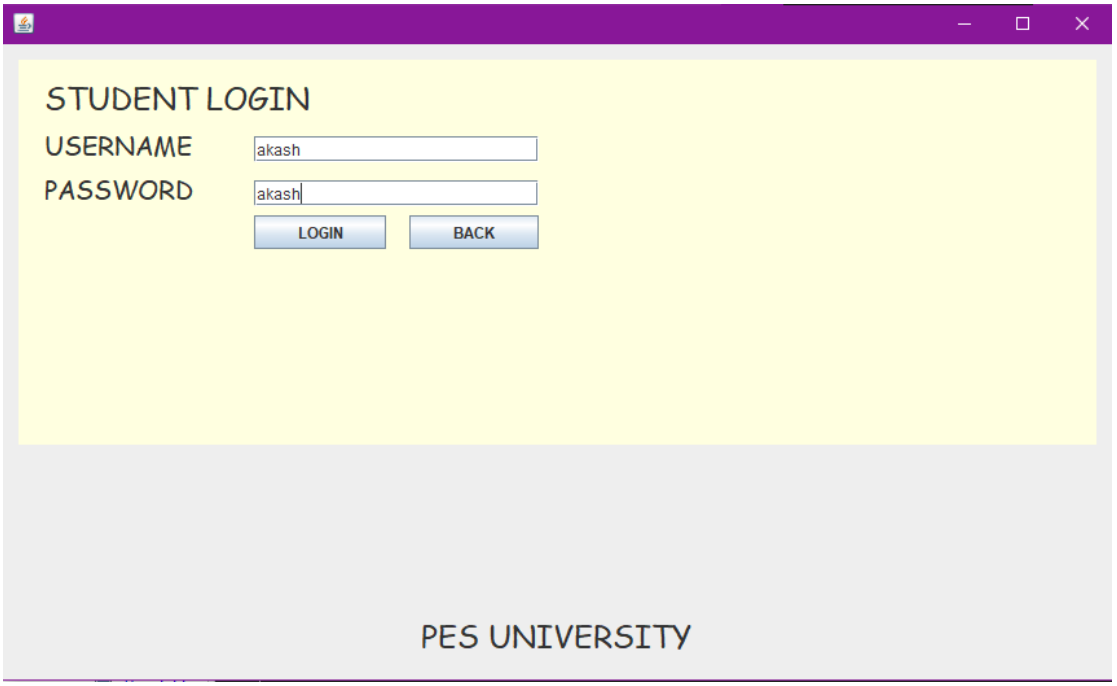


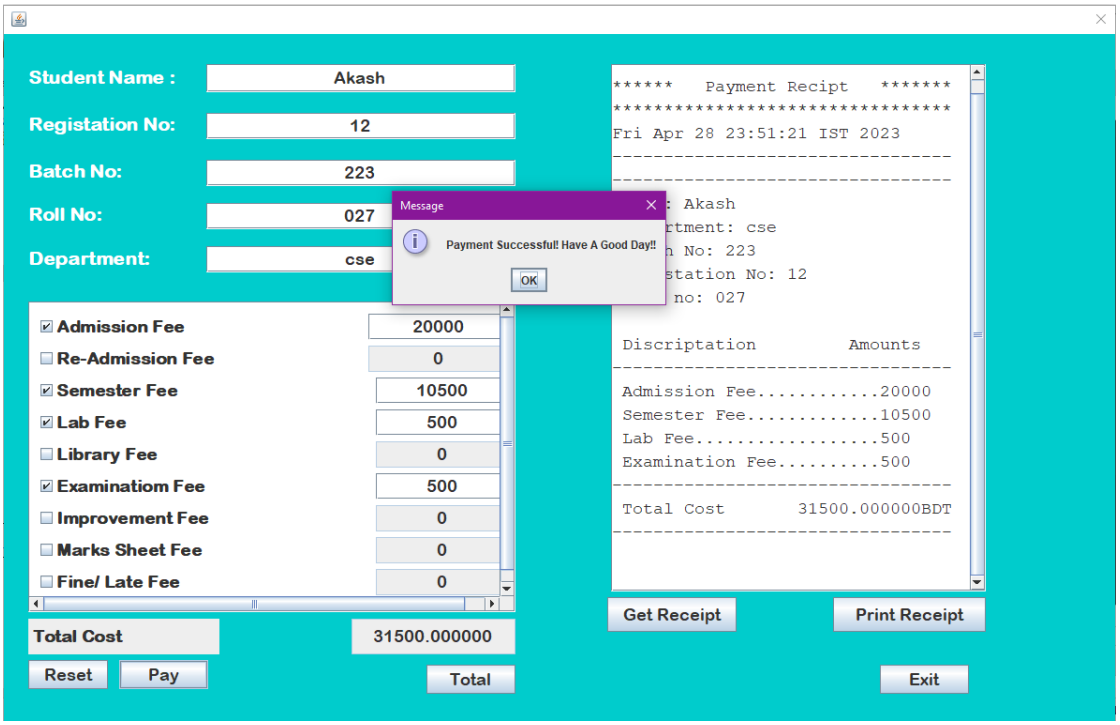

Fee Management System
ADMIN LOGIN
USERNAME
PASSWORD
LOGIN
BACK
PES UNIVERSITY


Fee Management System
WELCOME ADMIN
ADD STAFF
DELETE STAFF
VIEW STAFF
ADD STUDENT
UPDATE STUDENT
ADMIT STUDENT
VIEW STUDENT
LOGOUT
PES UNIVERSITY

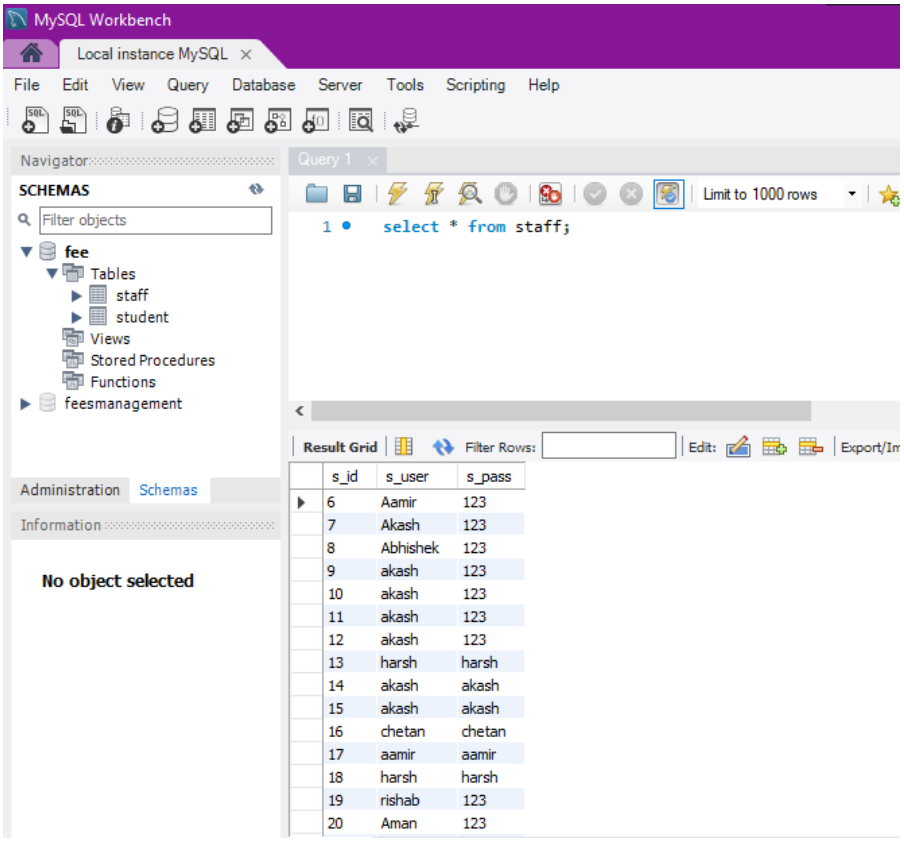

Fee Management System
ADD STUDENT
Name
Dept.
Year
Div
Fees
ADD STUDENT
Back
PES UNIVERSITY

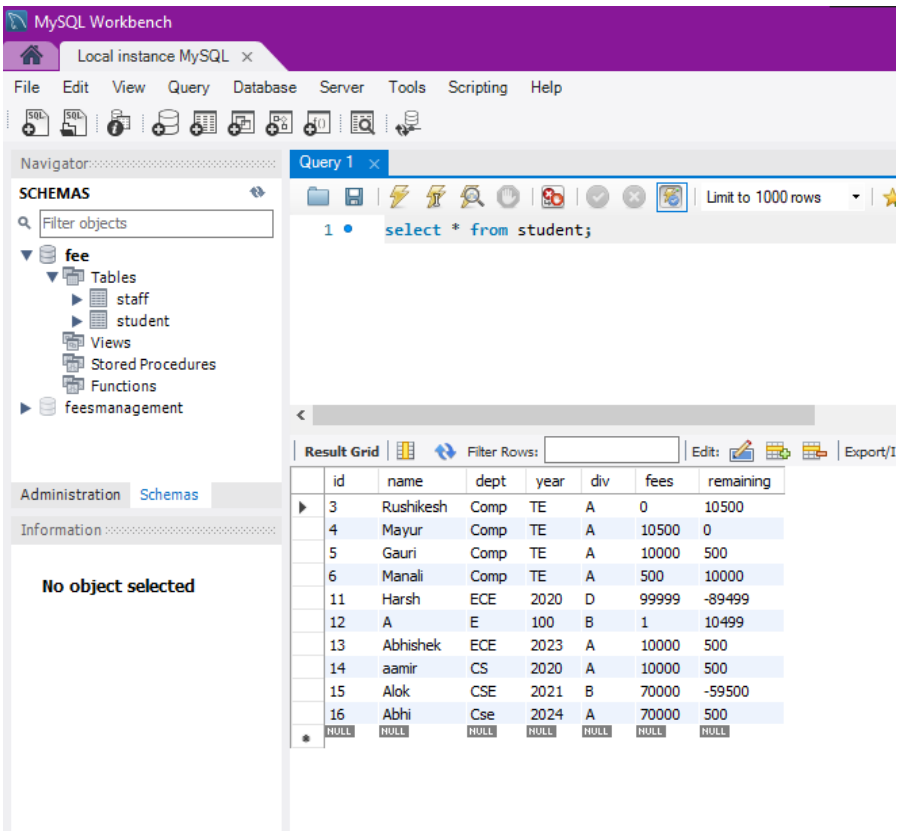

Fee Management System
STAFF LOGIN
USERNAME
aamir
PASSWORD
123
LOGIN
BACK
PES UNIVERSITY


STUDENT LOGIN
USERNAME
akash
PASSWORD
akash
LOGIN
BACK
PES UNIVERSITY


Student Name :
Akash
Registation No:
12
Batch No:
223
Roll No:
027
Department:
cse
Admission Fee
20000
Re-Admission Fee
0
Semester Fee
10500
Lab Fee
500
Library Fee
0
Examinatiom Fee
500
Improvement Fee
0
Marks Sheet Fee
0
Fine/ Late Fee
0
Total Cost
31500.000000
Reset
Pay
Total
Message
Payment Successful! Have A Good Day!!
OK
****** Payment Recipt ******
Fri Apr 28 23:51:21 IST 2023
Discription Amounts
Admission Fee.....20000
Semester Fee.....10500
Lab Fee.....500
Examination Fee.....500
Total Cost 31500.000000BDT
Get Receipt
Print Receipt
Exit

Database:


MySQL Workbench
Local instance MySQL
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
fee
Tables
staff
student
Views
Stored Procedures
Functions
feesmanagement
Administration Schemas
Information
No object selected
Query 1
Limit to 1000 rows
1 select * from staff;
Result Grid
Filter Rows:
Edit:
Export/Im
s_id s_user s_pass
6 Aamir 123
7 Akash 123
8 Abhishek 123
9 akash 123
10 akash 123
11 akash 123
12 akash 123
13 harsh harsh
14 akash akash
15 akash akash
16 chetan chetan
17 aamir aamir
18 harsh harsh
19 rishab 123
20 Aman 123


MySQL Workbench
Local instance MySQL
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
fee
Tables
staff
student
Views
Stored Procedures
Functions
feesmanagement
Administration Schemas
Information
No object selected
Query 1
Limit to 1000 rows
1 select * from student;
Result Grid
Filter Rows:
Edit:
Export/I
id name dept year div fees remaining
3 Rushikesh Comp TE A 0 10500
4 Mayur Comp TE A 10500 0
5 Gauri Comp TE A 10000 500
6 Manali Comp TE A 500 10000
11 Harsh ECE 2020 D 99999 -89499
12 A E 100 B 1 10499
13 Abhishek ECE 2023 A 10000 500
14 aamir CS 2020 A 10000 500
15 Alok CSE 2021 B 70000 -59500
16 Abhi Cse 2024 A 70000 500

Github link:

https://github.com/Akashk21/OOAD-PROJECT