Database Management Project

Section-5, Final Report

Group 31

Name	ID
Fardeen Hossain Bhuiyan	1710365
Ragib Yasar Alam	1730648
Shehab Sharar Mazumder	1731663
Sk Asif Faisal	2010454
Adnan Rahman	2120558
Iftakhirul Mohaimin Alif	1611001

Table of Content

CHAPTER-1 INTRODUCTION

A. BACKGROUND OF THE ORGANIZATION - IUB:

IUB is a private university in Bangladesh that was established in 1993. There are currently about 10,000 undergraduate and graduate students enrolled there. More than 450 highly qualified and highly-skilled faculty members are exceptionally good at their departments of teaching, and at least half of them are Ph.D. degree holders. The number of IUB alumni is close to 14,000. There are currently 5 academic schools in IUB

1) School of Business & Entrepreneurship
2) School of Engineering, Technology, and Sciences
3) School of Environment & Life Sciences
4) School of Liberal Arts & Social Sciences
5) School of Pharmacy and Public Health

B. BACKGROUND OF THE PROJECT - SPMS 4.0:

OBE's framework, Student Performance Monitoring System (SPMS 4.0), focuses on student performance. It analyzes student, course instructor, school, department, and program performance and aids the higher authorities of the educational institution in developing improvement strategies.

C. OBJECTIVE OF THE PROJECT - SPMS 4.0:

The current version of a web application is designed to manage student registration, course selection, and academic records. However, it lacks the capability to calculate course outcomes based on student grades. To address this limitation, new features need to be added to the application to capture data points such as student ID, educational year, semester, enrolled course and section, and obtained grades. These data points will be used to calculate course outcomes for students whose information is not already in the system.

D. SCOPE OF THE PROJECT:

The project involves enhancing the functionality of an existing web application to enable the calculation of course outcomes based on student performance data. The project will include the following steps:

- Adding New Data Fields: The first step is to add new data fields to the existing web application to capture the required data points such as student ID, educational year, semester, enrolled course and section, and obtained grades. The user must be able to input the data using a manual form or import a CSV file from which the data points can be extracted and inputted into the database.

- Calculating Course Outcome Percentage: The second step is to implement a new feature that calculates the course outcome percentage based on the obtained for each course. The course outcome percentage will be calculated based on the grading policy defined by the institution, such as a letter grade system or a numerical grade system.

- Displaying Course Outcome Percentage: The third step is to display the calculated course outcome percentage to students and faculty members through the web application. The students will be able to view their course outcome percentage for each enrolled course, and the faculty members will be able to view the course outcome percentage for all the students enrolled in their course.

The new features will be implemented in a way that is user-friendly and intuitive, with clear instructions and guidance provided to the users. The design of the user interface will be kept simple and minimalistic to enhance the user experience. Additionally, appropriate security measures will be implemented to protect the sensitive student data stored in the database.

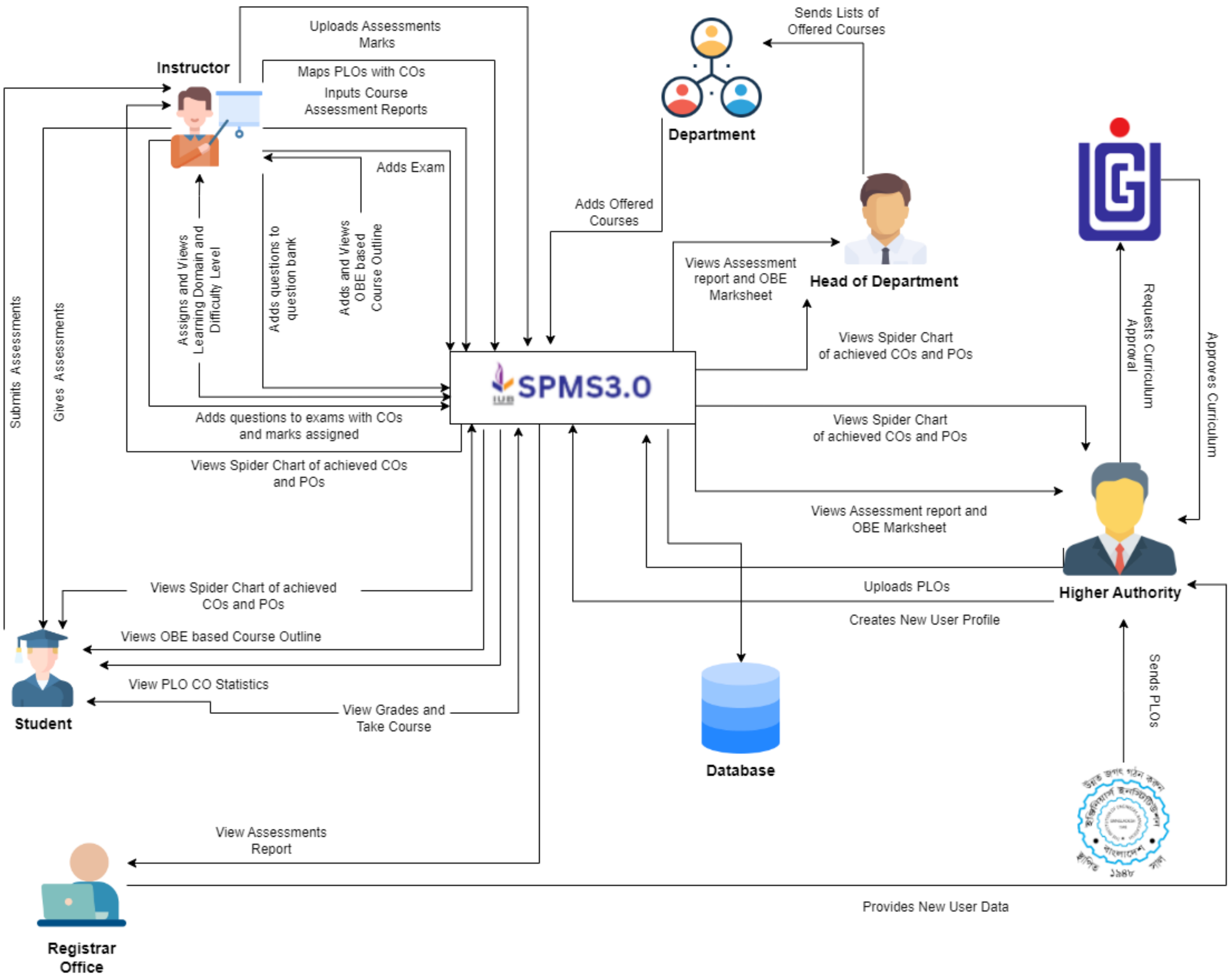
In summary, the project will involve adding new data fields to the existing web application, implementing a new feature to calculate course outcome percentages based on obtained grades, and displaying the calculated percentage to students and faculty members through the web application. The new features will be implemented with a focus on user-friendliness, simplicity, analysis, and security.

CHAPTER-2 REQUIREMENT ANALYSIS

Requirement Analysis is the process of determining what the database is used for. It involves interviews with stakeholders in order to identify the functionality and system requirements they expect and requires from the database, what operations need to be performed, and what data they need to process. By doing so, we can get a proper understanding of the stakeholders and how they interact with each other.

A. RICH PICTURE – EXISTING SYSTEM (SPMS 3.0):

A rich picture is a way to demonstrate processes in a system that is easier to understand for everyone. It consists of pictures, text, symbols, and icons, all of which are used to graphically illustrate the situation. A rich picture helps us see relationships and connections that we may otherwise miss. It helps identify one or more themes that participants may want to further explore and address. Rich pictures are therefore always used in the pre-analysis phase.



Instructor
Uploads Assessments Marks
Maps PLOs with COs
Inputs Course Assessment Reports
Adds Exam
Sends Lists of Offered Courses
Department
Adds Offered Courses
Head of Department
Views Assessment report and OBE Marksheet
Views Spider Chart of achieved COs and POs
Submits Assessments
Gives Assessments
Assigns and Views Learning Domain and Difficulty Level
Adds questions to question bank
Adds and Views OBE based Course Outline
SPMS3.0
Adds questions to exams with COs and marks assigned
Views Spider Chart of achieved COs and POs
Views Spider Chart of achieved COs and POs
Requests Curriculum Approval
Approves Curriculum
Views Assessment report and OBE Marksheet
Views Spider Chart of achieved COs and POs
Uploads PLOs
Creates New User Profile
Higher Authority
Views OBE based Course Outline
View PLO CO Statistics
View Grades and Take Course
Student
Database
Sends PLOs
View Assessments Report
Provides New User Data
Registrar Office

In this rich picture, the stakeholders are:
1) UGC
2) IEB
3) Higher Authority (VC, Dean etc)
4) Department Head

5) Department Office
6) SPMSV3.0 Admin (SPMS Manager)
7) Registers Office
8) Faculty
9) Student

The Main Storage is
1) SPMS V3.0

B. SIX ELEMENT ANALYSIS – EXISTING SYSTEM (SPMS 3.0):

From the rich picture, we can see that there are nine key processes:

1) Creating storing and giving Course Outline
2) Add Questions to the question bank and grading the answer script
3) Course based student performance trend according to GPA
4) Faculty based student performance according to GPA
5) Course wise PLO achievement of a student
6) Student performance trend under VC/Dean/Head of Department
7) Course, program, department, school CLO-PLO statistics
8) Course, student, department school wise expected vs achieved PLO
9) Department average of total PLO achieved and attempted students
10) Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.

We can use six element analysis to analyze the impact of six elements in a process here the six elements are
1. Human
2. Non-computing Hardware
3. Computing Hardware
4. Software.
5. Database.
6. Network and Communication

Process	Human	Non commuting Hardware	Computing Hardware	Software	Database	Network and Communication
Preparing, storing and giving Course Outline	**Faculty:** 1) Signs into the system using their ID and password. 2) Select the "Create Course Outline" tab. 3) Select from the options that they wish to add to their course outline. 4) Press the Create button. 5) Store the course outline in the system. **Students:** 1) Signs into the system using their ID and password. 2) Select Course 3)View/Download course outline from the system.		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print hard copies of course outlines if required.	**SPMS 3.0** 1) Used to store data in the database.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0
Add Questions to the question bank and grading the answer script	**Faculty:** 1) Signs into the system using their ID and password. 2) Select the course and choose the sections that have to solve the question. 3) Input the question in the		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print the grades gotten by the whole section.	**SPMS 3.0** 1) Used to store data in the database or generate a result graph using data from the database.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	question bank. 4) Press the Assign Button. 5) Grade the answers submitted by the students **Student:** 1) Signs into the system using their ID and password. 2) Answer the question assigned by the faculty in the answer bank					
Course based student performance trends according to GPA.	**Department Head:** 1) Signs into the system using their ID and password. 2) Input the time period and course ID to be viewed. 3) View student progress through a graph made after analysis and the GPA earned by the maximum/minimum /average students. **Faculty:** 1) Signs into the system using their ID and password. 2) Search for the course that they are teaching using the course ID and time period, and view the progress of those students in that course.		**Computer/ Laptop** 1) Used to sign into SPMS 3.0. **Printer** 1) Used to print a hard copy of the progress of the current semester's students and compare it with the progress of the previous semester's students who took that course.	**SPMS 3.0** 1)Used to store student Data into the database or generate a performance analysis graph using data from the database.	**SPMS 3.0 Database** 1) All valid data are stored here which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	Student: 1) Signs into the system using their ID and password. 2) Search for the course using the course ID and view their progress in that course and the GPA they earned. **Dean/VC:** 1) Signs into the system using their ID and password. 2) Search for the course using course ID and time period and view the progress of the students of that course					
Faculty based student performance according to GPA	**Faculty:** 1) Signs into the system using their ID and password. 2) View the progress of the students who are being taught by them. **Department Head:** 1) Signs into the system using their ID and password. 2) Search for a faculty to be assessed using the faculty's name.		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print hard copy of a report of students who completed most the PLO achievements If needed.	**SPMS 3.0** 1) Used to store data and generate PLO automatically based on the CO provided**.**	**SPMS 3.0 Database** 1) All valid data are stored here which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0 **wad**

	3) View the progress of the students who are being taught by that faculty based on the GPA earned by the students. **Dean/VC:** 1) Signs into the system using their ID and password. 2) Search for a faculty to be assessed using the faculty's name and Department ID. 3) View the progress of the students who are being taught by that faculty based on the GPA earned by the students.					
Course wise PLO achievement of a student	**VC/ Dean:** 1) Signs into the system using their ID and password. 2) Select PLO achievement Tab and search using Course ID 3) View the PLOs achieved by the student. **Department Head:** 1) Signs into the system using their ID and password. 2) Select PLO		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print hard copy of a report of students who completed most the PLO achievements If needed.	**SPMS 3.0** 1)Used to store Data and generate PLO automatically based on the CO provided	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	achievement Tab and search using Course ID 3) View the PLOs achieved by the students. **Faculty:** 1) Signs into the system using their ID and password. 2) Select PLO achievement Tab and search using Course ID 3) View the PLOs achieved by the students in a course. **Student:** 1) Signs into the system using their ID and password. 2) View the PLOs they have achieved so far and how many they need to achieve to complete the course.					
Student performance trend under VC/Dean/Head of Department	**Dean:** 1) Signs into the system using their ID and password. 2) Search for Department Head to be checked using their Name and Department ID. 3) View student		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print the hard copy of the progress report if needed.	**SPMS 3.0** 1) Used to store data in the database or generate a performance analysis graph using data from the database.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	progress under them or them. **VC:** 1) Signs into the system using their ID and password. 2) Search for a Dean or Department Head to be checked using their Name and either School ID or Department ID. 3) View student progress under them. **Department Head:** 1) Signs into the system using their ID and password. 2) View student progress under them.					
Course, program, department, and school CLO-PLO statistics	**Dean/VC:** 1) Signs into the system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students. **Department Head:** 1) Signs into the system using their ID and Password. 2) View CLO-PLO mapped statistics achieved		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print the hard copy of the progress report if needed	**SPMS 3.0** 1) Used to store data in the database and generate CLO-PLO statistical data or graphs.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	by students. **Faculty:** 1) Signs into the system using their ID and password. 2) View CLO-PLO mapped statistics achieved by students. **Student:** 1) Signs into the system using their ID and password. 2) View CLO-PLO mapped statistics achieved by them and other students.					
Course, student, department school wise expected vs achieved PLO	**Dean/VC:** 1) Sign into the system using an ID and password. 2) View the students achieved PLO of the students during time entered that has been inputted and compare it to what was expected and achieved. Department Head: 1) Sign into the system using an ID and password. 2) View the		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print the hard copy of both the previous and current semester's achieved PLO to compare.	**SPMS 3.0** 1)Used to store data into the database or generate a performance analysis graph using data from the database.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved. **Faculty:** 1) Sign into the system using an ID and password. 2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved. **Student:** 1) Sign into the system using an ID and password. 2) View the students achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved.					
Department average of total PLO achieved and attempted students	**Dean/VC:** 1) Sign into the system using an ID and password. 2) Enter the time period of the semester wished to be viewed. 3) View the		**Computer/ Laptop** 1) Used to Sign into SPMS 3.0 **Printer** 1) Used to print the hard copy of PLO reports	**SPMS 3.0** 1)Used to store Data into the database or generate performance analysis graph using data from the	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	departmental average of total PLO achieved along with the number of students who attempted. **Department Head:** 1) Sign into the system using an ID and password. 2) Enter the time period of the semester wished to be viewed. 3) View the departmental average of total PLO achieved along with the number of students who attempted. **Faculty:** 1) Sign into the system using an ID and password. 2) View the total departmental average of the PLO achieved by the students. **Student:** 1) Sign into the system using an ID and password. 2) View the total departmental average of the PLO achieved by the students			database.		

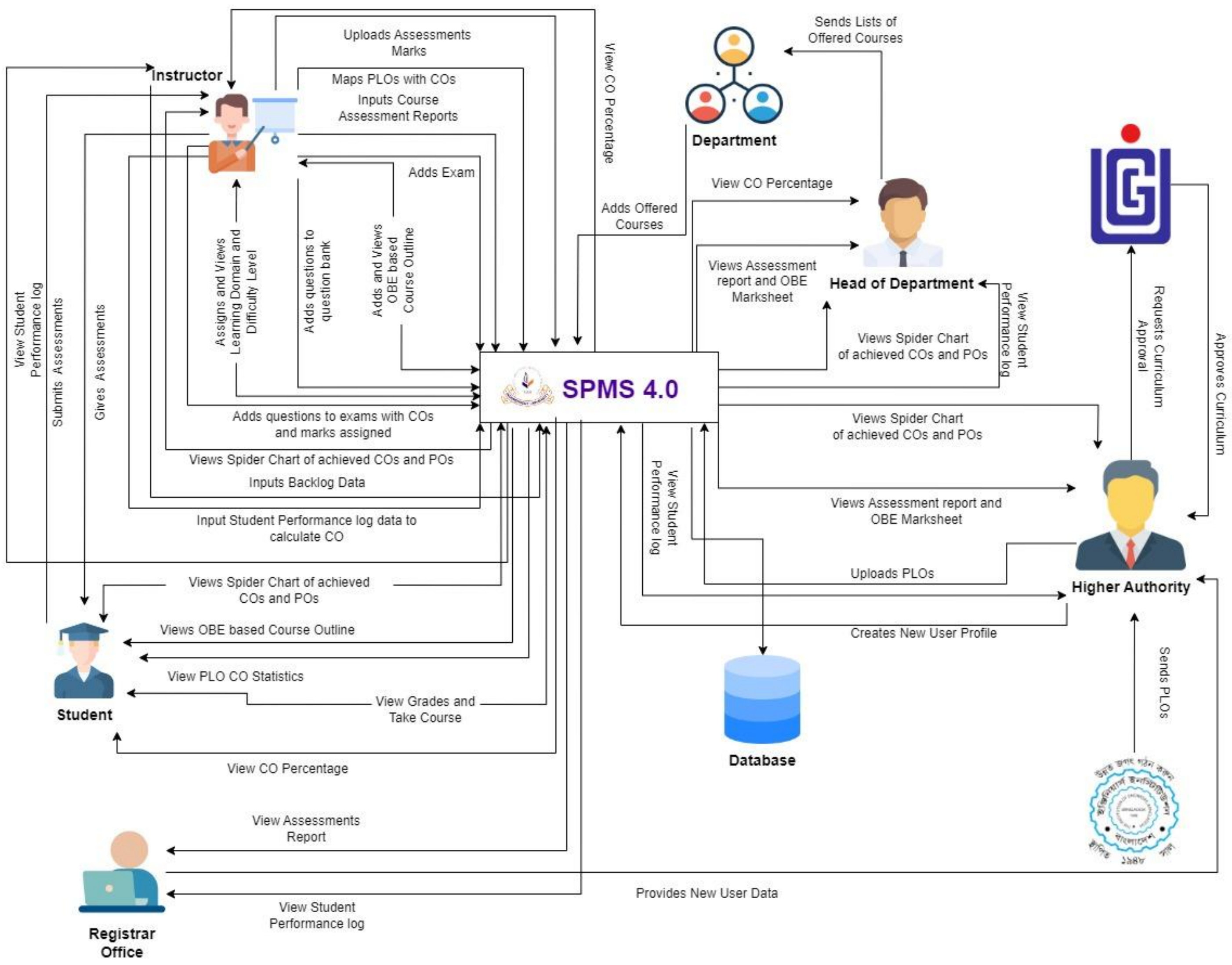
Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.	**VC** 1) Sign into the system using an ID and password. 2) Select Student Enrollment Statistics tab and select Year and Semester under that tab 3) View Student Enrollment Statistics of That Year and Semester. **Dean** 1) Sign into the system using an ID and password. 2) Select Student Enrollment Statistics tab and select Year and Semester under that tab 3) View Student Enrollment Statistics of That Year and Semester. **Department Head** 1) Sign into the system using an ID and password. 2) Select the Student Enrollment Statistics tab and select Year and Semester under that tab.		**Computer/ Laptop** 1) Used to Sign into SPMS2.0 **Printer** 1) Used to print the hard copy of Student enrollment statistics, if needed.	**SPMS 3.0** 1) Used to store data into the database and generate student enrollment statistics graphs.	**SPMS 3.0 Database** 1) All valid data are stored here, which can be updated by SPMS 3.0 admins.	**Internet** 1) Used to Sign into SPMS 3.0

	3) View Student Enrollment Statistics of That Year and Semester.					

C. PROBLEM ANALYSIS – EXISTING SYSTEM (SPMS 3.0):

Process Name	Stakeholders	Concern (Problems)	Analysis(reason of the Problem)	Proposed Solution
No record of Backlog Data	**Faculty:** **Student:** **Head of Department** **Higher Authority:**	1) No records of students before the system is stored in the database 2) Unreliable Storage 3) Records can be destroyed 4) If the records are kept as hardcopy then the paper may get damaged and the data will be lost forever.	1) There was no other way of storing the data before. 2) Physical Storage space may become full very quickly as more records are stored in local machines.	The backlog data is stored in the database with the help of a form that is integrated into the system with which the faculty can input the data of the students which are not stored in the database. In this way, there is no problem with storage space in local machines or the condition of the paper. Once the data is stored in the database, the course outcome percentage of the student can be converted from grade. This will show the data COs earned from courses the students took.

D. RICH PICTURE – PROPOSED SYSTEM (SPMS 4.0):



Instructor
Uploads Assessments Marks
Maps PLOs with COs
Inputs Course Assessment Reports
View CO Percentage
Adds Exam
Assigns and Views Learning Domain and Difficulty Level
Adds questions to question bank
Adds and Views OBE based Course Outline
View Student Performance log
Submits Assessments
Gives Assessments
Adds questions to exams with COs and marks assigned
Views Spider Chart of achieved COs and POs
Inputs Backlog Data
Input Student Performance log data to calculate CO
Views Spider Chart of achieved COs and POs
Views OBE based Course Outline
View PLO CO Statistics
View Grades and Take Course
View CO Percentage
Student
View Assessments Report
View Student Performance log
Registrar Office
Provides New User Data
SPMS 4.0
Sends Lists of Offered Courses
Department
Adds Offered Courses
View CO Percentage
Views Assessment report and OBE Marksheet
Head of Department
View Student Performance log
Views Spider Chart of achieved COs and POs
Views Spider Chart of achieved COs and POs
Views Assessment report and OBE Marksheet
View Student Performance log
Uploads PLOs
Creates New User Profile
Database
Requests Curriculum Approval
Approves Curriculum
Higher Authority
Sends PLOs

In this rich picture the stakeholders are:

1) UGC
2) IEB
3) Higher Authority (VC, Dean etc)
4) Department Head
5) Department Office
6) SPMSV4.0 Admin (SPMS Manager)
7) Registers Office

8) Faculty
9) Student

The Main Storage is
1) SPMS V4.0

E. SIX ELEMENT ANALYSIS – PROPOSED SYSTEM (SPMS 4.0):

From the rich picture we can see that there are 9 key processes:

1) Creating storing and giving Course Outline
2) Add Questions to the question bank and grading the answer script
3) Course based student performance trend according to GPA
4) Faculty based student performance according to GPA
5) Course wise PLO achievement of a student
6) Student performance trend under VC/Dean/Head of Department
7) Course, Program, department, school CLO-PLO statistics
8) Course, student, department school wise expected vs achieved PLO
9) Department average of total PLO achieved and attempted students
10) Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.

We can use six element analysis to analyze the impact of six elements in a process here the six elements are

1. Human
2. Non-computing Hardware
3. Computing Hardware
4. Software.
5. Database.
6. Network and Communication

Process	Human	Non-computing Hardware	Computing Hardware	Software	Database	Network and Communication
Preparing, storing, and giving Course Outline	**Faculty:** 1) Signs into the System using their ID and Password. 2) Select Create Course Outline Tab. 3) Select From the options that they wish to add to their course outline. 4) Press the Create button. 5) Store course outline into the system. **Students:** 1) Signs into the System using their ID and Password. 2) Select Course 3)View/Download Course Outline From the System.		**Computer/ Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print a hard copy of course outlines if required.	**SPMS 4.0** 1) Used to store Data into the database	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS4.0
Add Questions to the question bank and grade the answer script	**Faculty:** 1) Signs into the System using their ID and Password. 2) Select a course and choose sections that solve the question. 3) Input the question in the question bank. 4) Press the Assign Button.		**Computer/ Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the grades gotten by the whole section	**SPMS 4.0** 1) Used to store Data into the database or generate result graph using data from the database.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins	**Internet** 1) Used to Sign into SPMS 4.0

	5) Grade the answers submitted by the students **Student:** 1) Signs into System using their ID and Password. 2) Answer the question assigned by the faculty in the answer bank					
Course based student performance trend according to GPA	**Department Head:** 1) Signs into System using their ID and Password. 2) Input the time period and course ID to be viewed. 3) View student progress through a graph made after analysis and the GPA earned by maximum/minimum/average students. **Faculty:** 1) Signs into system using their ID and Password. 2) Search for the course that they are teaching using course ID and time period and		**Computer/ Laptop** 1) Used to Sign into SPMS4.0 **Printer** 1) Used to print hard copy of the progress of current semester's students and compare with the progress of the previous semester's students who did that course.	**SPMS 4.0** 1)Used to store student Data into the database or generate performance analysis graph using data from the database.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	view the progress of that students of that course. **Student:** 1) Signs into System using their ID and Password. 2) Search for the course using course ID and View their progress of that course and the GPA they earned. **Dean/VC:** 1) Signs into system using their ID and Password. 2) Search for the course using course ID and time period and view the progress of the students of that course					
Faculty based student performance according to GPA	**Faculty:** 1) Signs into system using their ID and Password. 2) View the Progress of the students who are being taught by them.		**Computer/ Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print hard copy of a report of students who	**SPMS 4.0** 1) Used to store Data and generate PLO automatically based on the CO provided.	**SPMS4.0Database** 1) All valid data are stored here which can be updated by SPMS 4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	Department Head: 1) Signs into system using their ID and Password. 2) Search for a faculty to be assessed using the faculty’s name. 3) View the Progress of the students who are being taught under that faculty basing on the GPA earned by the students. **Dean/VC:** 1) Signs into system using their ID and Password. 2) Search for a faculty to be assessed using the faculty’s name and Department ID. 3) View the Progress of the students who are being taught under that faculty basing on the GPA earned by the students.		completed most the PLO achievements If needed.			
Course wise PLO achievement of a student	**VC/ Dean:** 1) Signs into system using their ID and Password.		**Computer/ Laptop** 1) Used to Sign into SPMS 4.0	**SPMS 4.0** 1)Usedto store Data and generate PLO	**SPMS4.0 Database** 1) All valid data are stored here which can	**Internet** 1) Used to Sign into SPMS 4.0

	2) Select PLO achievement Tab and search using Course ID 3) View PLOs achieved by the student. **Department Head:** 1) Signs into system using their ID and Password. 2) Select PLO achievement Tab and search using Course ID 3) View PLOs achieved by the students. **Faculty:** 1) Signs into system using their ID and Password. 2) Select PLO achievement Tab and search using Course ID 3) View PLOs achieved by the students in a course.		**Printer** 1) Used to print hard copy of a report of students who completed most the PLO achievements If needed.	automatically based on the CO provided.	be updated by SPMS4.0 admins.	

	Student: 1) Signs into system using their ID and Password. 2) View PLOs they have achieved so far and how many they need to achieve to complete the course.					
Student performance trend under VC/Dean/Head of Department	**Dean:** 1) Signs into system using their ID and Password. 2) Search for Department Head to be checked using their Name and Department ID. 3) View student progress under them or them. **VC:** 1) Signs into system using their ID and Password. 2) Search for a Dean or Department Head to be checked using their Name and either School ID or Department ID. 3) View student progress under them.		**Computer/ Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the hard copy of the progress report if needed	**SPMS 4.0** 1)Used to store Data into the database or generate performance analysis graph using data from the database.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	Department Head: 1) Signs into system using their ID and Password. 2) View student progress under them.					

Course, Program, department, school CLO-PLO statistics	**Dean/VC:** 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students. **Department Head:** 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students. **Faculty:** 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students. **Student:** 1) Signs into system using their ID and Password.		**Computer /Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the hard copy of the progress report if needed	**SPMS 4.0** 1)Used to store Data into the database and generate CLO-PLO statistical data or graphs.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	2) View CLO-PLO mapped statistics achieved by them and other students.					
Course, student, department school wise expected vs achieved PLO	**Dean/VC:** 1) Sign into the system using ID and Password. 2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved. **Department Head:** 1) Sign into the system using ID and Password. 2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved. **Faculty:** 1) Sign into the system using ID and Password. 2) View the achieved PLO of the students during time		**Computer /Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the hard copy of both the previous and current semester's achieved PLO to compare.	**SPMS 4.0** 1)Used to store Data into the database or generate performance analysis graph using data from the database.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1)Used to Sign into SPMS 4.0

	entered that has been inputted and comparison between expected and achieved. **Student:** 1) Sign into the system using ID and Password. 2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved.					
Department average of total PLO achieved and attempted students.	**Dean/VC:** 1) Sign into the system using ID and Password. 2) Enter the time period of the semester wished to be viewed. 3) View the departmental average of total PLO achieved along with the number of students who attempted. **Department Head:** 1) Sign into the system using ID and Password.		**Computer /Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the hard copy of PLO reports	**SPMS 4.0** 1)Used to store data into the database or generate performance analysis graph using data from the database.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	2) Enter the time period of the semester wished to be viewed. 3) View the departmental average of total PLO achieved along with the number of students who attempted. **Faculty:** 1) Sign into the system using ID and Password. 2) View the total departmental average of the PLO achieved by the students. **Student:** 1) Sign into the system using ID and Password. 2) View the total departmental average of the PLO achieved by the students					
Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.	**VC** 1) Sign into the system using ID and Password. 2) Select Student Enrollment Statistics tab and select Year and Semester under that tab		**Computer /Laptop** 1) Used to Sign into SPMS 4.0 **Printer** 1) Used to print the hard copy of Student Enrollment Statistics If	**SPMS 4.0** 1) Used to store Data into the database and generate Student Enrollment Statistics graphs.	**SPMS 4.0 Database** 1) All valid data are stored here which can be updated by SPMS4.0 admins.	**Internet** 1) Used to Sign into SPMS 4.0

	3) View Student Enrollment Statistics of That Year and Semester. **Dean** 1) Sign into the system using ID and Password. 2) Select Student Enrollment Statistics tab and select Year and Semester under that tab 3) View Student Enrollment Statistics of That Year and Semester. **Department Head** 1) Sign into the system using ID and Password. 2) Select Student Enrollment Statistics tab and select Year and Semester under that tab 3) View Student Enrollment Statistics of That Year and Semester.		Needed.			
View CO percentage	**Student:** 1) Sign into the system using ID and Password. 2) Select View Course Outcome		**Computer /Laptop** 1) Used to Sign into SPMS 4.0	**SPMS 4.0** 1) Used to retrieve Data from the database and	**SPMS 4.0 Database** 1) All valid data is retrieved from here.	**Internet** 1) Used to Sign into SPMS 4.0

	and input Student ID, Course, Year and Semester 3) Press Load Course Outcome which will redirect user to page showing CO percentage in a Chart **Department** 1) Sign into the system using ID and Password. 2) Select View Course Outcome and input Student ID, Course, Year and Semester 3) Press Load Course Outcome which will redirect user to page showing CO percentage in a Chart **Faculty:** 1) Sign into the system using ID and Password. 2) Select View Course Outcome and input Student ID, Course, Year and Semester 3) Press Load Course Outcome which will redirect user to page showing CO percentage in a Chart			generate CO percentage .		

Input Student Performance log data to calculate CO	**Faculty:** 1) Signs into the System using their ID and Password. 2) Select Submit Student Performance Data. 3) Input student performance data 4) Press submit.		**Computer /Laptop** 1) Used to Sign into SPMS 4.0	**SPMS 4.0** 1) Used to store Data of the student into the database	**SPMS 4.0 Database** 1) All valid data are stored here	**Internet** 1) Used to Sign into SPMS 4.0

CHAPTER-3 LOGICAL SYSTEM DESIGN

A. BUSINESS RULES – SPMS 4.0:

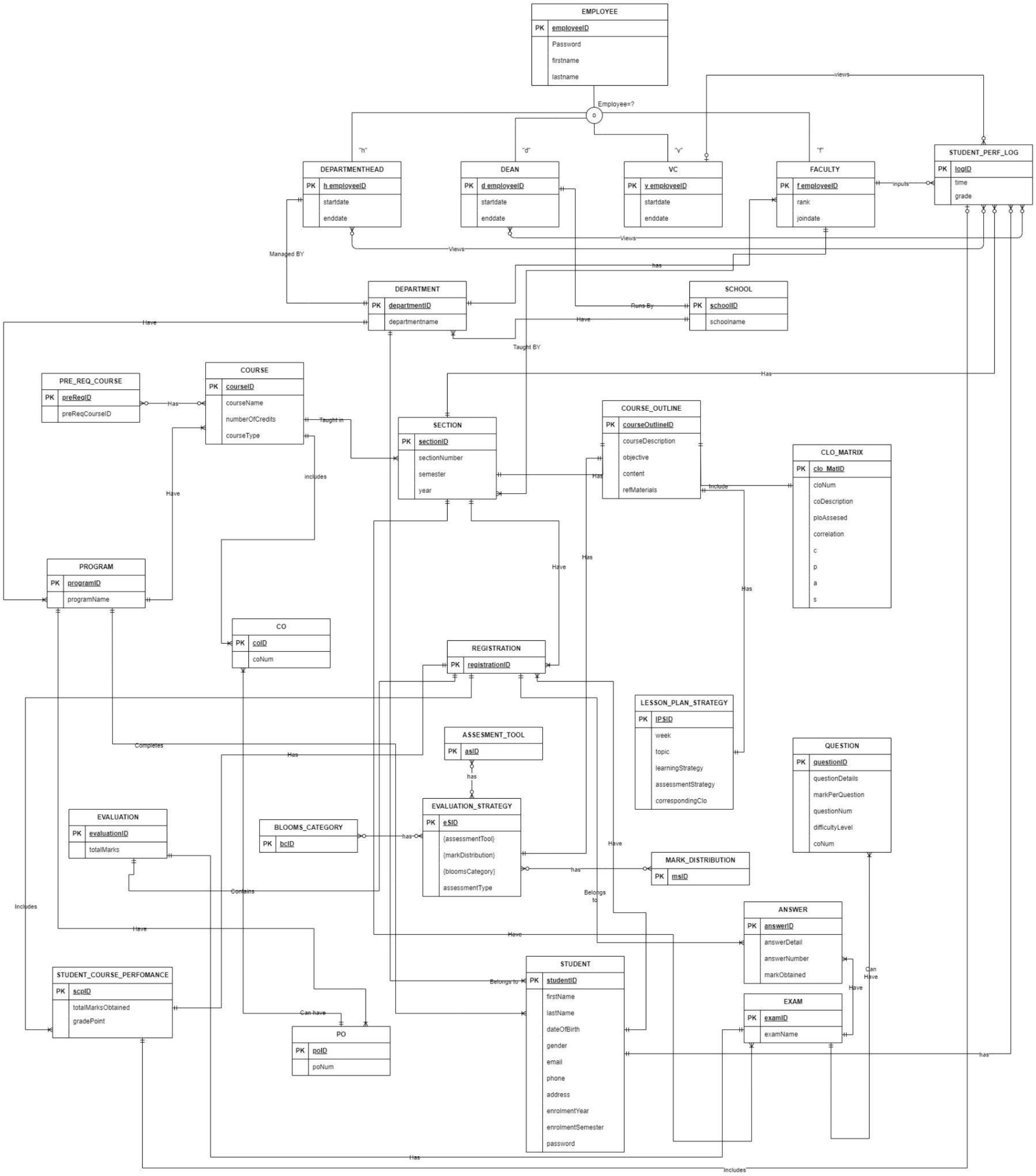
1. A student must have one department. A STUDENT has StudentID, FirstName, LastName, dateOfBirth, gender, email, phone, address, departmentID, programID, enrollmentYear,enrollmentSemester,password. A department must have one or many Students.
2. Student may perform many registrations. A REGISTRATION includes RegistrationID, sectionID, studentID. A registration must be performed by at least one student.
3. A section mandatorily have many registrations. A registration has at least one section. A section includes sectionID, sectionNum, courseID, facultyID, year.
4. A registration may belong to many EVALUATIONS. An evaluation mandatorily belongs to one registration. An EVALUATION contains evaluationID, examID, registrationID,totalMarks.
5. A CO must map with one PLO. A PLO’s must map with one or many CO’s. PLO includes ploID, ploNum, programID.
6. A PLO must contain one program. A program contains one or many PLO’s. A PROGRAM has programID, programName, departmentID. A program must contain one or many courses. A Course must contain one course.

7. A program must belong to one department. A department must belong to one or many programs. A DEPARTMENT contains departmentID, departmentName, schoolID.
8. A department must contain one school. A SCHOOL must contain one or many departments. A school includes schoolID, schoolName.
9. An employee has four sub-type(Dean, Department Head, Faculty, VC). An EMPLOYEE includes employeeID,password, firstName, lastName.
10. A school must be run by exactly one. A dean must run exactly one school. A DEAN has schoolID, startDate, endDate.
11. A Department must be run by exactly one Department head. A department head must manage exactly one department. A DEPARTMENTHEAD includes departmentID, startDate, endDate.
12. A Faculty must have exactly one Department. A department must have one or many Faculties. A FACULTY includes departmentID, rank, joinDate. A faculty may teach many sections. A section must be taught by exactly one faculty.
13. A course outline belongs to exactly one section. A section must have exactly one course outline. A COURSE_OUTLINE includes courseOutlineID ,sectionID, contactHours, courseDescription ,objective ,content ,refMaterials ,courseType, courseTitle, prerequisiteCode ,creditValue.
14. A Course outline must have exactly one CLO Matrix. A CLO matrix belongs to exactly one course outline. A CLO_MATRIX includes clo_MatID, cloNum, coDescription, ploAssessed, correlation, courseOutlineID , c ,p ,a ,s.
15. A Lesson Plan Strategy must have exactly one Evaluation strategy. An Evaluation strategy must have exactly one Lesson Plan Strategy. A LESSON_PLAN_STRATEGY includes lPSID ,week ,topic ,learningStrategy , assessmentStrategy, corresponding CLO, courseOutlineID.
16. An exam has exactly one evaluation. An Evaluation for an exam is done exactly once. An exam belongs to exactly one section. An EXAM includes examID, examName, sectionID. A section must have one or many exams.
17. An exam must have one or many questions. Every question must belong to exactly one exam. A QUESTION includes questionID , questionDetails , marksPerQuestion , questionNum , difficultyLevel , examID , coNum. A Question is answered exactly once. An answer has exactly one question.
18. A PO belongs to exactly one program A program must have one or many PO.PO includes poID , poNum , programID. A PO must belong to one or many CO. A CO must have exactly one PO.
19. A student course performance evaluation is done for registration exactly once. A registration has student course performance evaluation done exactly once. A

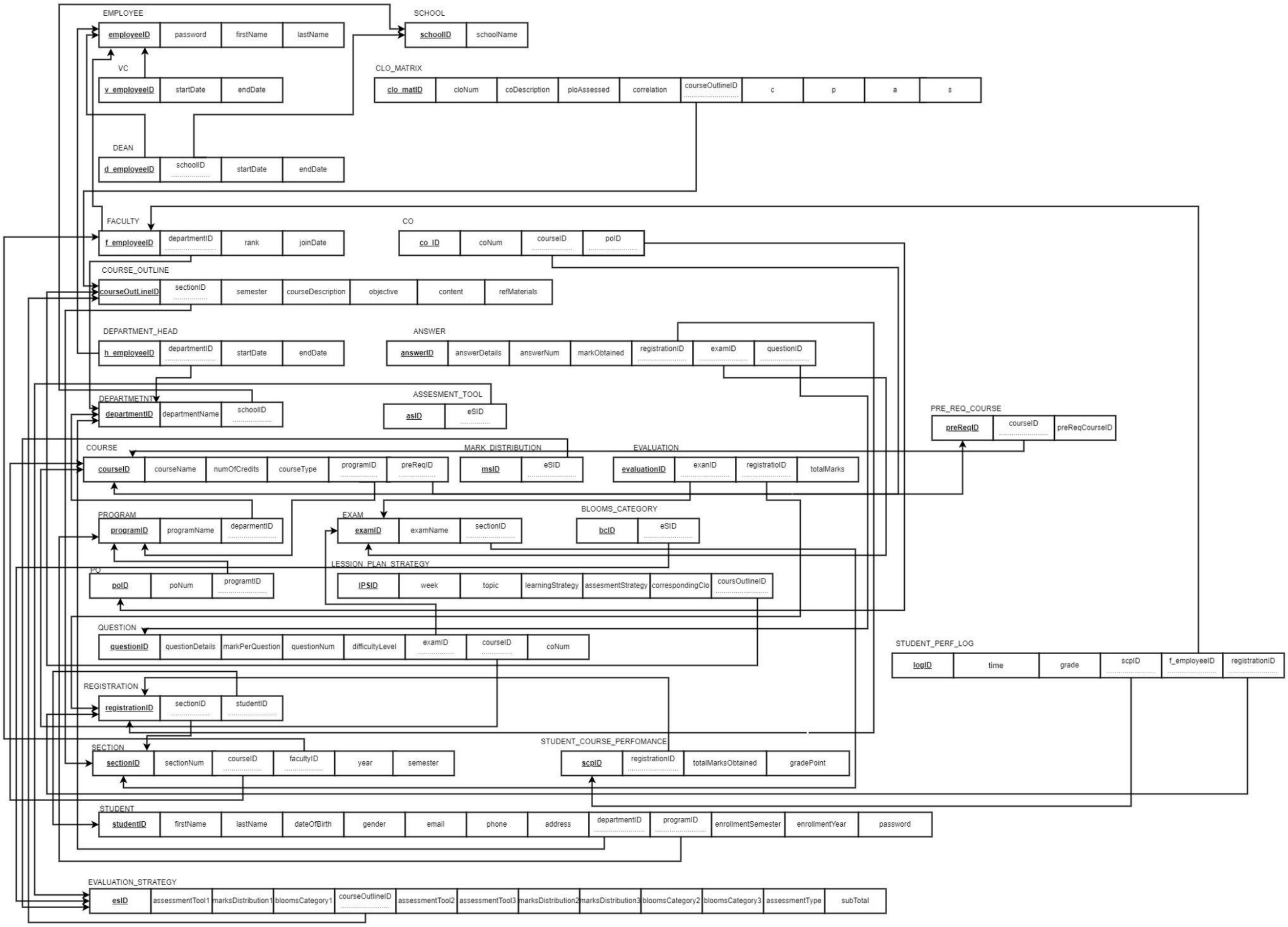
registration has exactly one evaluation. An Evaluation has exactly one registration.

20. A faculty may input many student data. A backlog data has studentID, enrollmentYear, enrollmentSemester, enrolledCourse, enrolledSection and obtainedGrade.
21. A Backlog data may be viewed by many deans, many department heads and many faculties

B. ENTITY RELATIONSHIP DIAGRAM:



C. ENTITY RELATIONSHIP DIAGRAM TO RELATIONAL SCHEMA:


EMPLOYEE
employeeID | password | firstName | lastName
SCHOOL
schoolID | schoolName
VC
v_employeeID | startDate | endDate
CLO_MATRIX
clo_matID | cloNum | coDescription | ploAssessed | correlation | courseOutlineID | c | p | a | s
DEAN
d_employeeID | schoolID | startDate | endDate
FACULTY
f_employeeID | departmentID | rank | joinDate
CO
co_ID | coNum | courseID | poID
COURSE_OUTLINE
courseOutLineID | sectionID | semester | courseDescription | objective | content | refMaterials
DEPARTMENT_HEAD
h_employeeID | departmentID | startDate | endDate
ANSWER
answerID | answerDetails | answerNum | markObtained | registrationID | examID | questionID
DEPARTMETNT
departmentID | departmentName | schoolID
ASSESMENT_TOOL
asID | eSID
PRE_REQ_COURSE
preReqID | courseID | preReqCourseID
COURSE
courseID | courseName | numOfCredits | courseType | programID | preReqID
MARK_DISTRIBUTION
msID | eSID
EVALUATION
evaluationID | exanID | registratioID | totalMarks
PROGRAM
programID | programName | deparmentID
EXAM
examID | examName | sectionID
BLOOMS_CATEGORY
bcID | eSID
PO
poID | poNum | programID
LESSION_PLAN_STRATEGY
lPSID | week | topic | learningStrategy | assesmentStrategy | correspondingClo | coursOutlineID
QUESTION
questionID | questionDetails | markPerQuestion | questionNum | difficultyLevel | examID | courseID | coNum
STUDENT_PERF_LOG
logID | time | grade | scpID | f_employeeID | registrationID
REGISTRATION
registrationID | sectionID | studentID
STUDENT_COURSE_PERFOMANCE
scpID | registrationID | totalMarksObtained | gradePoint
SECTION
sectionID | sectionNum | courseID | facultyID | year | semester
STUDENT
studentID | firstName | lastName | dateOfBirth | gender | email | phone | address | departmentID | programID | enrollmentSemester | enrollmentYear | password
EVALUATION_STRATEGY
esID | assessmentTool1 | marksDistribution1 | bloomsCategory1 | courseOutlineID | assessmentTool2 | assessmentTool3 | marksDistribution2 | marksDistribution3 | bloomsCategory2 | bloomsCategory3 | assessmentType | subTotal

D. NORMALIZATION:

EMPLOYEE (i)	employeeID	i1
	password	i2
	firstname	i3
	lastname	i4

VC (v)	v_employeeID	v1
	startDate	v2
	endDate	v3
DEAN (w)	d_employeeID	w1
	schoolID	h1
	startDate	w2
	endDate	w3
FACULTY (f)	f_employeeID	f1
	departmentID	d1
	rank	f2
	joinDate	f3
Course_Outline (c)	courseOutlineID	c1
	sectionID	y1
	contactHours	c2
	courseDescription	c3
	objective	c4
	content	c5
	refMaterials	c6
	courseType	c7
	courseTitle	c8

	reprequisiteCode	c9
	creditValue	c10
DEPARTMENTHEAD (k)	h_employeeID	k1
	departmentID	d1
	startDate	k2
	endDate	k3
DEPARTMENT (d)	departmentID	d1
	departmentName	d2
	schoolID	h1
COURSE (u)	courseID	u1
	courseName	u2
	numOfCredits	u3
	courseType	u4
	programID	r1
PROGRAM (r)	programID	r1
	programName	r2
	departmentID	d1
PO (x)	poID	x1
	poNum	x2
	programID	r1
QUESTIONS (q)	questionID	q1
	questionDetails	q2
	marksPerQuestion	q3
	questionNum	q4
	difficultyLevel	q5
	examID	e1
	courseID	u1

	coNum	q6
REGISTRATION (g)	registrationID	g1
	sectionID	y1
	studentID	s1
SECTION (y)	sectionID	y1
	sectionNum	y2
	semester	y3
	courseID	u1
	facultyID	f1
	year	y4
STUDENT (s)	studentID	s1
	firstName	s2
	lastName	s3
	dateOfBirth	s4
	gender	s5
	email	s6
	phone	s7
	address	s8
	departmentID	d1
	programID	r1
	enrollmentSemester	s9
	enrollmentYear	s10
	password	s11
SCHOOL (h)	schoolID	h1
	schoolName	h2
CLO_MATRIX (m)	clo_MatID	m1
	cloNum	m2

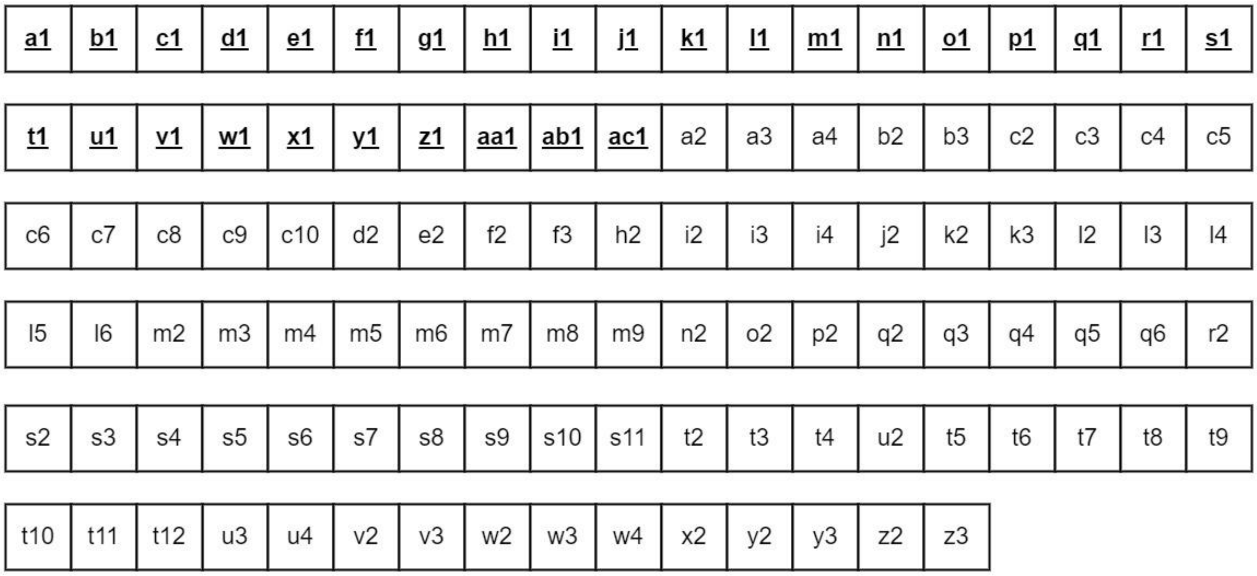
	coDescription	m3
	ploAssessed	m4
	correlation	m5
	courseOutlineID	c1
	c	m6
	p	m7
	a	m8
	s	m9
PLO (p)	ploID	p1
	ploNum	p2
	programID	r1
CO (o)	co_ID	o1
	coNum	o2
	courseID	u1
	ploID	p1
	poID	x1
ANSWER (a)	answerID	a1
	answerDetails	a2
	answerNum	a3
	markObtained	a4
	registrationID	g1
	examID	e1
EVALUATION_STRATEGY (t)	eSID	t1
	assessmentTool1	t2
	assessmentTool2	t3
	assessmentTool3	t4
	marksDistribution1	t5

	marksDistribution2	t6
	marksDistribution3	t7
	bloomsCategory1	t8
	bloomsCategory2	t9
	bloomsCategory3	t10
	assessmentType	t11
	subTotal	t12
	courseOutlineID	c1
PRE_REQ_COURSE(j)	preReqID	j1
	courseID	u1
	preReqCourseID	j2
EVALUATION (n)	evaluationID	n1
	examID	e1
	registrationID	g1
	totalMarks	n2
EXAM (e)	examID	e1
	examName	e2
	sectionID	y1
LESSON_PLAN_STRATEGY (l)	lPSID	l1
	weel	l2
	topic	l3
	learningStrategy	l4
	assessmentStrategy	l5
	correspondingClo	l6
	courseOutlineID	c1
STUDENT_COURSE_PERFORMANCE (z)	scpID	z1
	registrationID	g1

	totalMarksObtained	z2
	gradePoint	z3
BLOOMS_CATEGORY	bcID	aa1
	eSID	t1
ASSESMENT_TOOL	asID	ab1
	eSID	t1
MARK_DISTRIBUTION	msID	ac1
	eSID	t1
STUDENT_PERF_LOG (b)	logID	b1
	time	b2
	grade	b3
	scpID	z1
	registrationID	g1
	f_employeeID	f1

1NF:

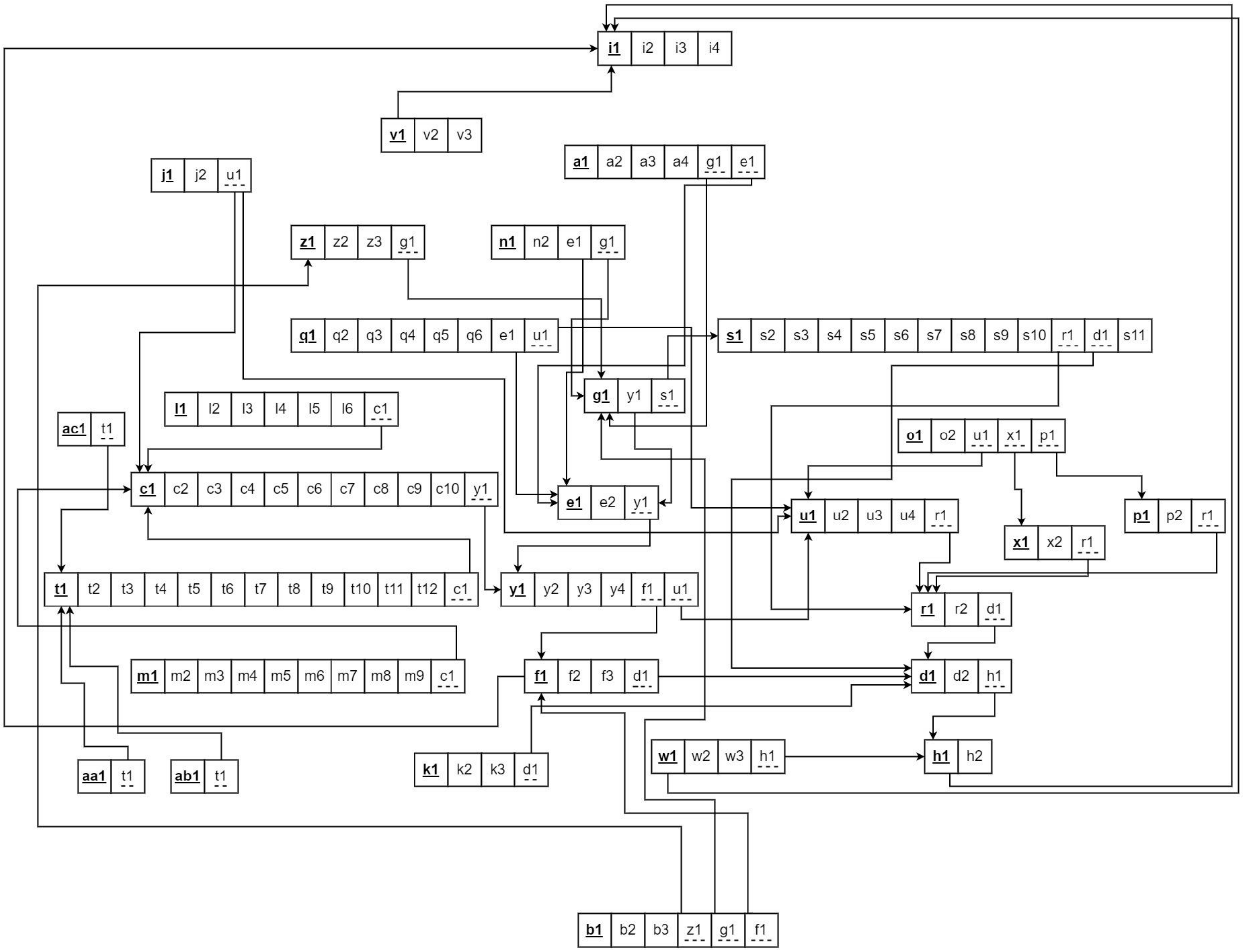
1) There are no repeating groups
2) There are multiple candidate keys



a1	**b1**	**c1**	**d1**	**e1**	**f1**	**g1**	**h1**	**i1**	**j1**	**k1**	**l1**	**m1**	**n1**	**o1**	**p1**	**q1**	**r1**	**s1**
t1	**u1**	**v1**	**w1**	**x1**	**y1**	**z1**	**aa1**	**ab1**	**ac1**	a2	a3	a4	b2	b3	c2	c3	c4	c5
c6	c7	c8	c9	c10	d2	e2	f2	f3	h2	i2	i3	i4	j2	k2	k3	l2	l3	l4
l5	l6	m2	m3	m4	m5	m6	m7	m8	m9	n2	o2	p2	q2	q3	q4	q5	q6	r2
s2	s3	s4	s5	s6	s7	s8	s9	s10	s11	t2	t3	t4	u2	t5	t6	t7	t8	t9
t10	t11	t12	u3	u4	v2	v3	w2	w3	w4	x2	y2	y3	z2	z3				

2NF:

1)Partial dependency has been removed



3NF:

1) Already in 3NF as there are no transitive dependencies.

BCNF:

Already in BCNF as there are no determinants that are not a unique identifier.

E.DATA DICTIONARY

VC_T

Name		Data Type	Size	Remark
v_employeeID		INTEGER	11	This is the foreign key from the Employee table. E.g: “4250”
startDate		DATE		This is starting date for the VC. E.g: “01-03-2020”
endDate		DATE		This is the date VC retire from his post. E.g: “01-03-2024”

STUDENT_T

Name	Data Type	Size	Remark
studentID	INTEGER	11	This is the primary key for the Student table. E.g: “1921834”.
firstName	VARCHAR	30	This is the first name of the student. E.g: “Rakibul”.
lastName	VARCHAR	30	This is the last name of the student. E.g: “Hasan”.
dateOfBirth	DATE		This is the birth date of the student. E.g: “2112-1996”.

gender	VARCHAR	6	This is the gender of the student. E.g: “Female”.
email	VARCHAR	30	This is the email of the student. E.g: “1921834@iub.edu.bd”
phone	VARCHAR	11	This is the phone of the student. E.g: “01XXXXXXXXX”.
address	VARCHAR	50	This is the address of the student. E.g: “House 1,Road 4,Block D, Bashundhara RA
departmentID	VARCHAR	3	This is the foreign key from the Department table. E.g: “CSE”
programID	INTEGER	11	This is the foreign key from the Program table. E.g: “1”
enrollmentSemester	VARCHAR	10	This is the enrollment semester of the student.
enrollmentYear	VARCHAR	4	This is enrollment year of the student.

STUDENT_COURSE_PERFORMANCE_T

Name	Data Type	Size	Remark
scpID	INTEGER	11	This is the primary key for this table
registrationID	INTEGER	11	This is the foreign key from registration table
totalMarksObtained	INTEGER	11	This is the total marks obtained by the student
gradePoint	FLOAT		This is the grade point achieved by the student

SECTION_T

Name	Data Type	Size	Remark
sectionID	INTEGER	11	This is the Primary Key for Section. E.g: “1”
sectionNum	INTEGER	11	This is the section number. E.g: “1”
semester	VARCHAR	6	This is the semester of the section. E.g: “Summer”
courseID	VARCHAR	6	This is the foreign key from the Course table. E.g: “CSE101”
facultyID	INTEGER	11	This is the foreign key from Faculty table. E.g: “1801”
year	YEAR	4	This is the year this section of this course was taken by this specific faculty

SCHOOL_T

Name	Data Type	Size	Remark
schoolID	VARCHAR	5	This is the primary key of School. E.g: “SETS”
schoolName	VARCHAR	50	This is the name of the School. E.g: “School of Engineering,
			Technology & Science”.

REGISTRATION_T

Name	Data Type	Size	Remark
registrationID	INTEGER	11	This is the Primary Key for Registration. E.g: “0101010101”
sectionID	INTEGER	11	This is the foreign key from section table
studentID	INTEGER	11	This is the foreign key from student table

QUESTION_T

Name	Data Type	Size	Remark
questionID	INTEGER	11	This is the primary key of this table
questionDetails	MEDIUMTEXT		This is the question
markPerQuestion	INTEGER	11	This is the mark each question contains
questionNum	INTEGER	11	This is the number of the question
difficultyLevel	INTEGER	11	This is the difficulty level of the question
examID	VARCHAR	20	This is the foreign key from exam table

courseID	VARCHAR	6	This is the foreign key from course table
coNum	INTEGER	11	This is the CO number of the question

PROGRAM_T

Name	Data Type	Size	Remark
programID	INTEGER	11	This is the primary key for a program. E.g: “1”
programName	VARCHAR	50	This is the name of the program. E.g: “Bachelor of Science”
departmentID	VARCHAR	3	This is the foreign key from the Department table.
			E.g: “CSE”

PO_T

Name	Data Type	Size	Remark
poID	VARCHAR	5	This is the primary key for Program Outcome. E.g: “PO1”
poNum	INTEGER	11	This is the PO number. E.g: “1”
programID	INTEGER	11	This is a foreign key from Program table. E.g: “1”

LESSON_PLAN_STRATEGY_T

Name	Data Type	Size	Remark
lpsID	INTEGER	11	This is the primary key of the table
week	INTEGER	11	This is the week number
topic	MEDIUMTEXT		This is the topic name
learningStrategy	MEDIUMTEXT		This is the lesson plan strategy of that topic
assessmentStrategy	VARCHAR	10	This is the assessment strategy of that topic
courseOutlineID	INTEGER	11	This is the foreign key from course outline table

FACULTY_T

Name	Data Type	Size	Remark
f_employeeID	INTEGER	11	This is the foreign key from the Employee table. E.g: “4250”
departmentID	VARCHAR	3	This is the DepartmentID of the department faculty belongs to. E.g: “CSE”
rank	VARCHAR	30	This is the rank of the faculty. E.g: “Assistant Professor”
joinDate	DATE		This is starting date. E.g: “01-03-2020”

EXAM_T

Name	Data Type	Size	Remark
examID	INTEGER	11	This is the primary key for this table
examName	VARCHAR	30	This is the name of the exam
sectionID	INTEGER	11	This is the foreign key from exam table

EVALUATION_T

Name	Data Type	Size	Remark
evaluationID	INTEGER	11	This is the primary key for this table
examID	VARCHAR	20	This is the foreign key from exam table
registrationID	INTEGER	11	This is the foreign key from registration table
totalMarks	INTEGER	11	This is the total marks achieved by the student in a specific exam

EVALUATION_STRATEGY_T

Name	Data Type	Size	Remark
eSID	INTEGER	11	This is the primary key for this table
courseOutlineID	INTEGER	11	This is the foreign key from course outline table

ASSESSMENT_TOOL_T

Name	Data Type	Size	Remark
asID	INTEGER	11	This is the primary key for this table
eSID	INTEGER	11	This is the foreign key from evaluation strategy table

MARK_DISTRIBUTION_T

Name	Data Type	Size	Remark
mdID	INTEGER	11	This is the primary key for this table
eSID	INTEGER	11	This is the foreign key from evaluation strategy table

BLOOMS_CATEGORY_T

Name	Data Type	Size	Remark
bcID	INTEGER	11	This is the primary key for this table
eSID	INTEGER	11	This is the foreign key from evaluation strategy table

EMPLOYEE_T

Name	Data Type	Size	Remark
employeeID	INTEGER	11	This is the primary key for Employee table. E.g: “1801”
password	VARCHAR	10	This is the password of the employee
firstName	VARCHAR	50	This is the last name of the faculty. E.g: “Ahmed”
lastName	VARCHAR	50	This is the last name of the faculty. E.g: “Ahmed”

DEPARTMENTHEAD_T

Name	Data Type	Size	Remark
h_employeeID	INTEGER	11	This is the foreign key from the Employee table.
			E.g: “4250”
departmentID	VARCHAR	3	This is the DepartmentID of the department HEAD manages. E.g: “CSE”
startDate	DATE		This is starting date. E.g: “01-03-2020”
endDate	DATE		This is the date HEAD retire from his post. E.g: “0103-2024”

DEPARTMENT_T

Name	Data Type	Size	Remark
departmentID	VARCHAR	3	This is the primary key for the Department table. E.g: “CSE”
departmentName	VARCHAR	50	This is the name of the department. E.g: “Computer Science and Engineering”.
schoolID	VARCHAR	5	This is a foreign key from the School table. E.g: “SETS”.

DEAN_T

Name	Data Type	Size	Remark
d_employeeID	INTEGER	11	This is the foreign key from the Employee table. E.g: “4250”
schoolID	VARCHAR	5	This is the SchoolID of the school DEAN manages. E.g: “SETS”
startDate	DATE		This is starting date. E.g: “01-03-2020”
endDate	DATE		This is the date DEAN retire from his post. E.g: “0103-2024”

COURSE_T

Name	Data Type	Size	Remark
courseID	VARCHAR	6	This is the Primary Key for the Course. E.g: “CSE203”
courseName	VARCHAR	40	This is the name of the Course. E.g: “Discreet Mathematics”
numOfCredits	INTEGER	11	This is the number of credits for the Course. E.g: “3”
courseType	VARCHAR	10	This is the type of the Course. E.g: “Core”

programID	INTEGER	11	This is the foreign key from the program table. E.g: “1”

COURSE_OUTLINE_T

Name	Data Type	Size	Remark
courseOutlineID	INTEGER	11	This is the primary key for this table
sectionID	INTEGER	11	This is the foreign key from the section table
courseDescription	MEDIUMTEXT		This is the description of the course
objective	MEDIUMTEXT		This is the objective of the course
content	MEDIUMTEXT		This is the content of the course
refMaterials	MEDIUMTEXT		This is the reference material
courseTitle	VARCHAR	1000	This is the title of the course
prerequsiteCode	VARCHAR	6	This is the prerequisite course code
creditValue	INTEGER	11	This is the credit value of the course

CO_T

Name	Data Type	Size	Remark
coID	INTEGER	11	This is the primary key for the CO table. E.g: “CO1”.
coNum	INTEGER	11	This is the CO number. E.g: 1,2 etc.

courseID	VARCHAR	6	This is the foreign key from the Course table. E.g: “CSE303”
ploID	VARCHAR	5	This is the foreign key from the PLO table. E.g: “PLO1”
poID	VARCHAR	6	This is the foreign key from the PLO table. E.g: “PO1”

CLO_MATRIX_T

Name	Data Type	Size	Remark
clo_MatID	INTEGER	11	This is the primary key for this table
cloNum	INTEGER	11	This is the clo number
coDescription	MEDIUMTEXT		This is the co description
ploAssessed	VARCHAR	10	This is the name of the plo assessed
correlation	INTEGER	11	This is the correlation value or number
courseOutlineID	INTEGER	11	This is the foreign key from the course outline table
c	INTEGER	11	This is the bloom’s category level
p	INTEGER	11	This is the bloom’s category level
a	INTEGER	11	This is the bloom’s category level
s	INTEGER	11	This is the bloom’s category level

ANSWER_T

Name	Data Type	Size	Remark
answerID	INTEGER	11	This is the primary key for this table
answerDetails	MEDIUMTEXT		This is the answer details
answerNum	INTEGER	11	This is the number of the answer
markObtained	INTEGER	11	This is the mark obtained by the student for each answer
registrationID	INTEGER	11	This is the foreign key from registration table
examID	INTEGER	11	This is the foreign key from the exam table

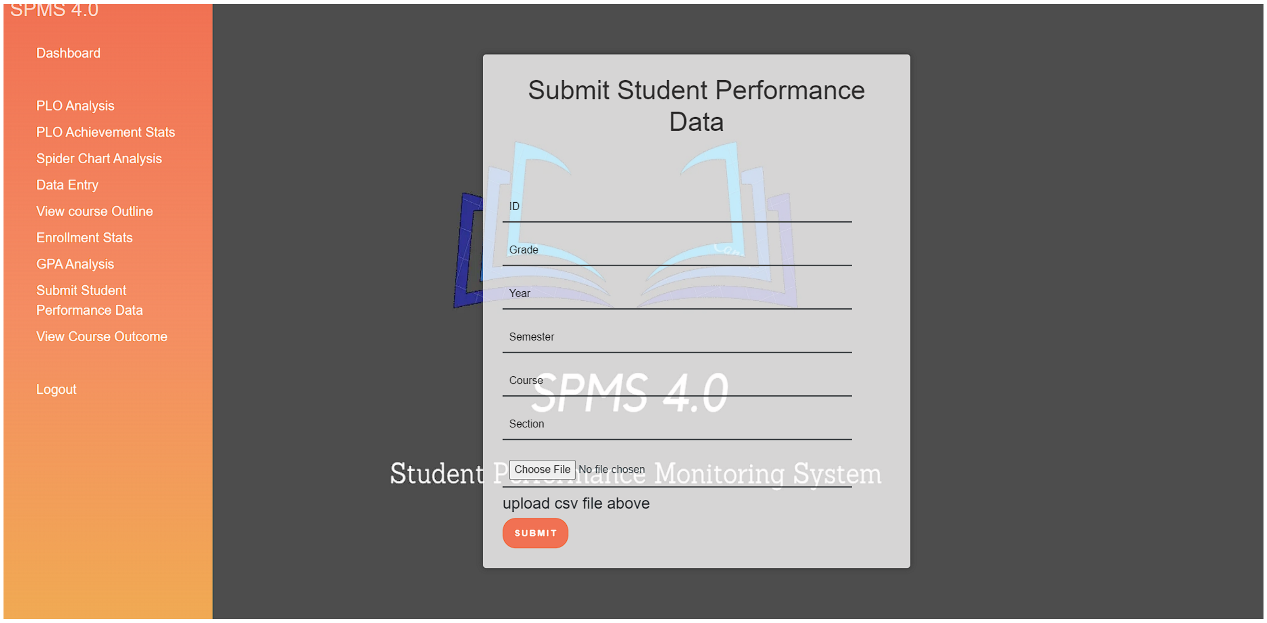
STUDENT_PERF_LOG

Name	Data Type	Size	Remark
logID	INTEGER	11	This is the primary key for this table
time	TIMESTAMP		Time at which student performance log form was submitted
scpID	INTEGER	11	This is the foreign key from student course performance table
f_employeeID	INTEGER	11	This is the foreign key from faculty table
registrationID	INTEGER	11	This is the foreign key from registration table

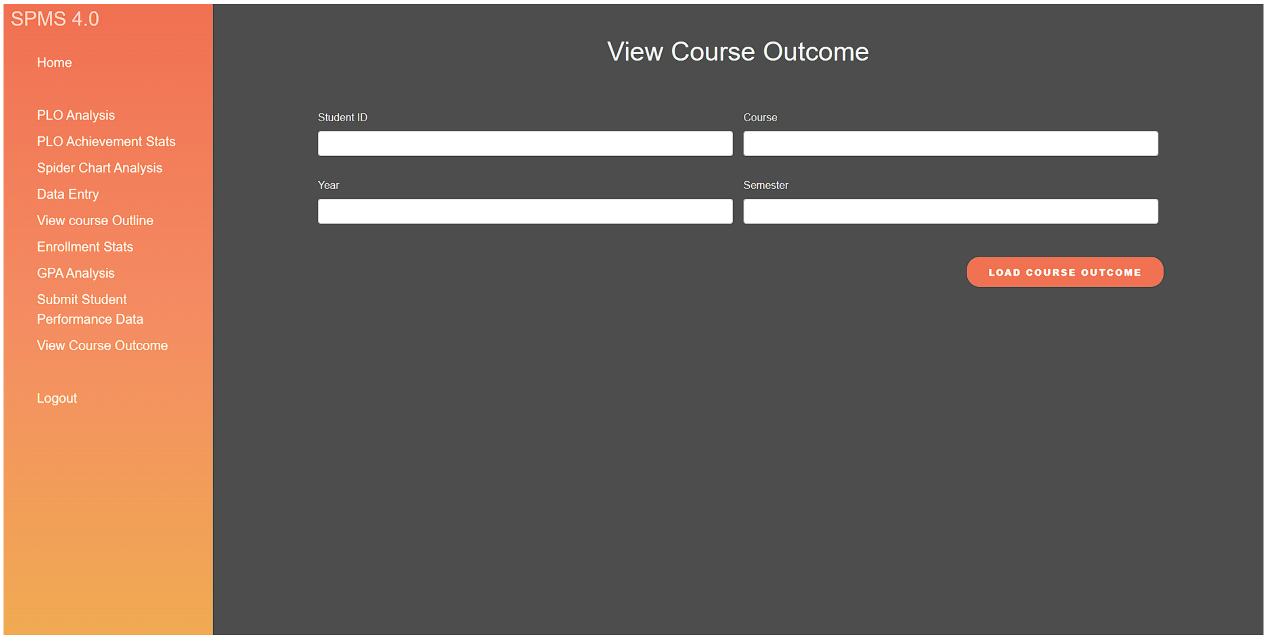
CHAPTER- 4 PHYSICAL SYSTEM DESIGN

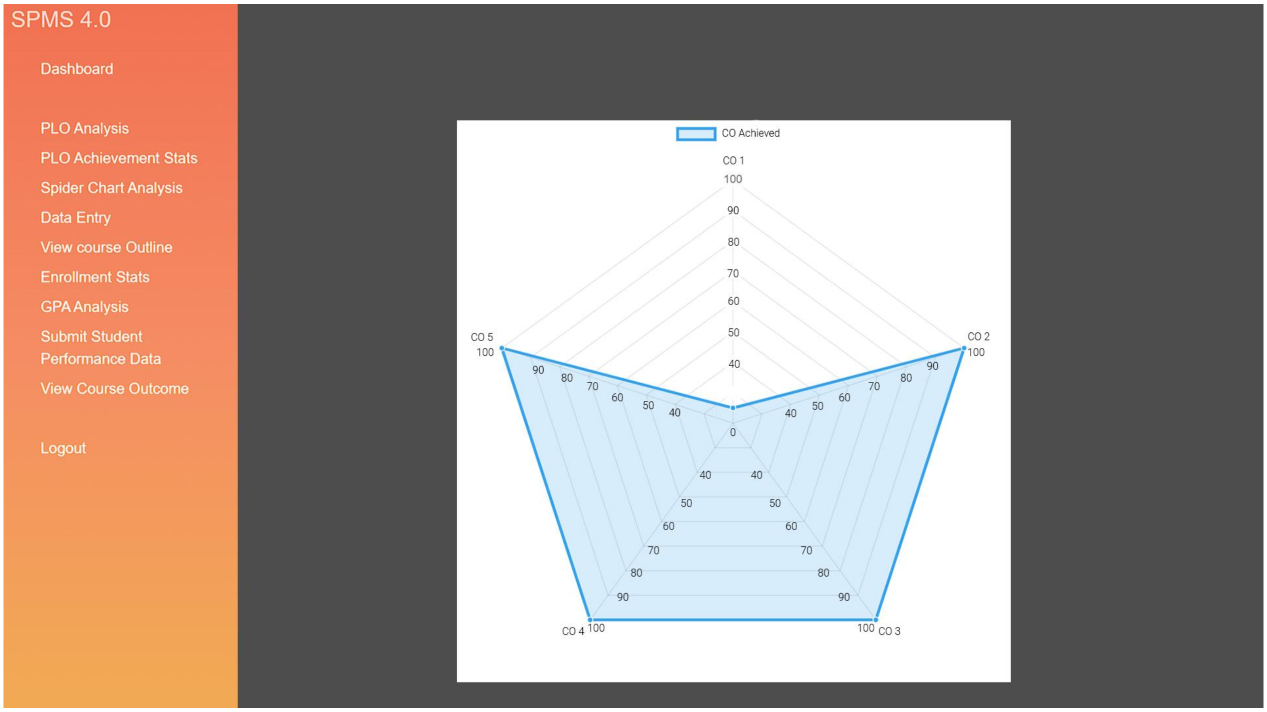
A. INPUT FORMS:

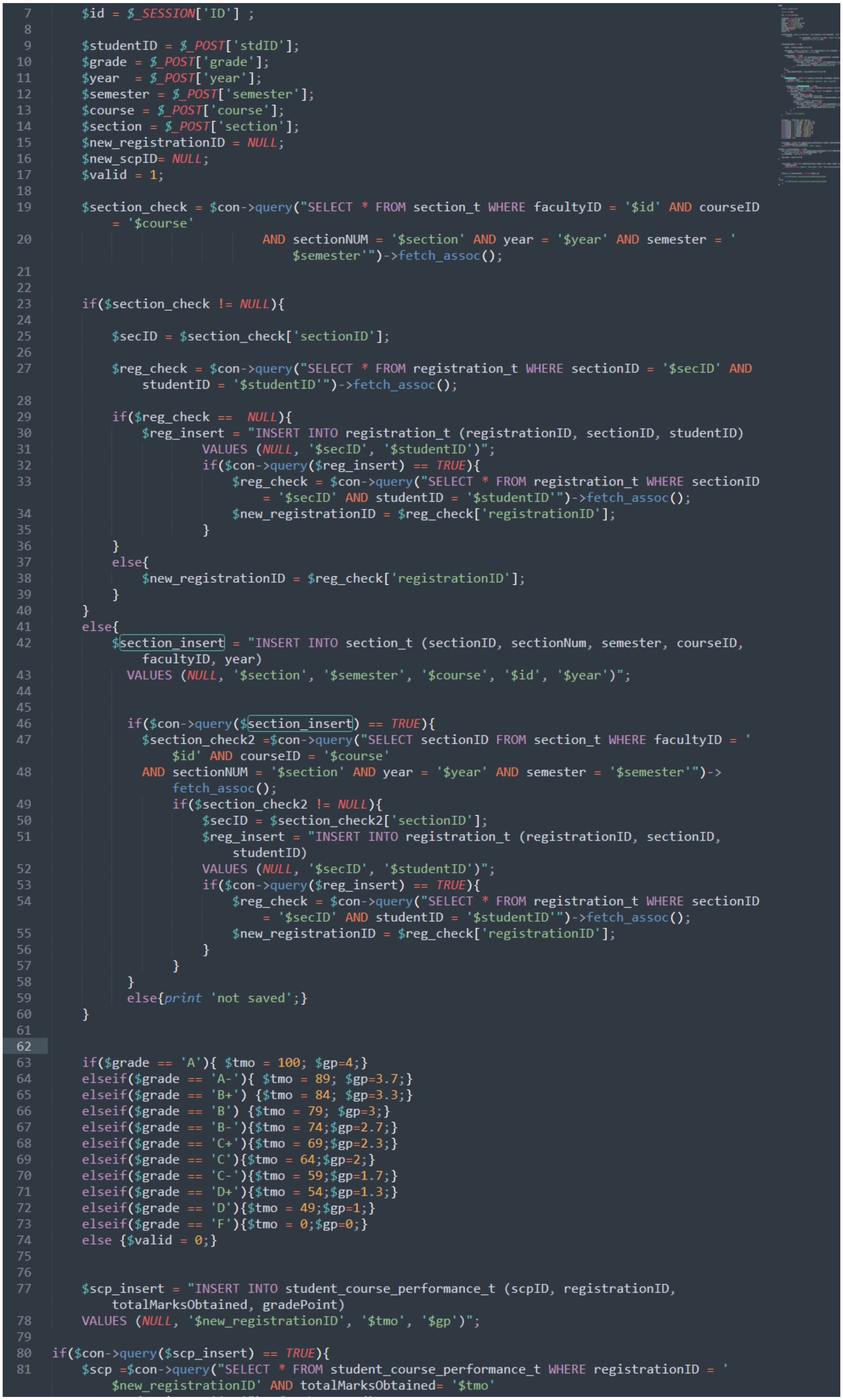
Input form for taking student performance data to convert to Course Outcome



Input form to load for Student performance data as Course Outcome percentage

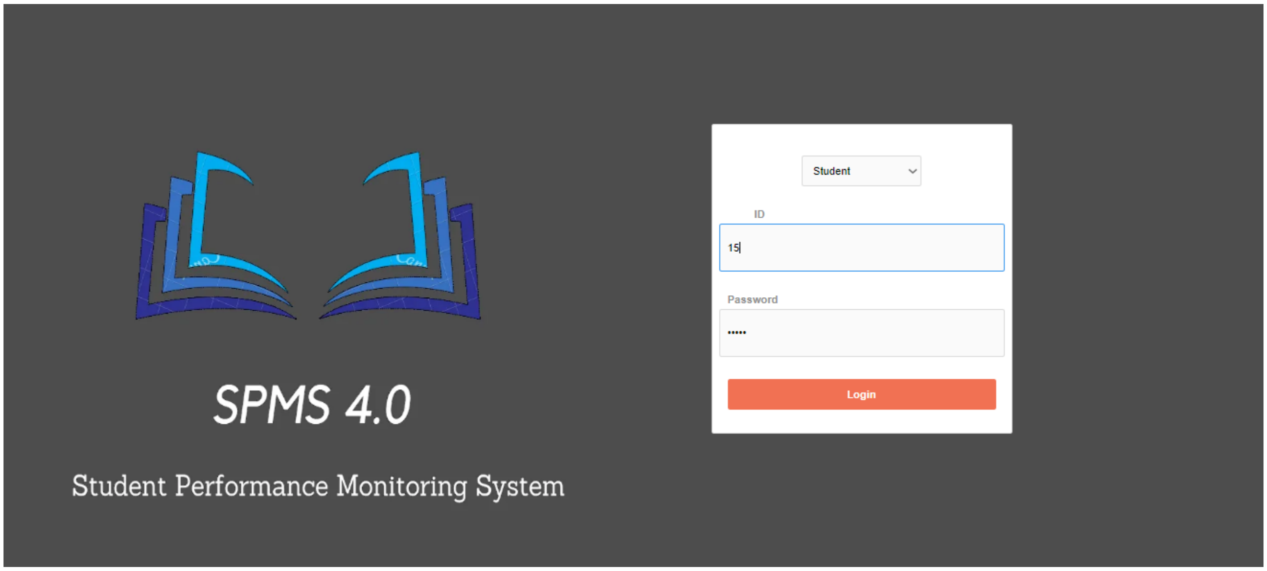



SPMS 4.0
Dashboard
PLO Analysis
PLO Achievement Stats
Spider Chart Analysis
Data Entry
View course Outline
Enrollment Stats
GPA Analysis
Submit Student Performance Data
View Course Outcome
Logout
CO Achieved
CO 1
CO 2
CO 3
CO 4
CO 5
100
90
80
70
60
50
40
0



```php
    $id = $_SESSION['ID'] ;

    $studentID = $_POST['stdID'];
    $grade = $_POST['grade'];
    $year  = $_POST['year'];
    $semester = $_POST['semester'];
    $course = $_POST['course'];
    $section = $_POST['section'];
    $new_registrationID = NULL;
    $new_scpID= NULL;
    $valid = 1;

    $section_check = $con->query("SELECT * FROM section_t WHERE facultyID = '$id' AND courseID = '$course'
                            AND sectionNUM = '$section' AND year = '$year' AND semester = '$semester'")->fetch_assoc();


    if($section_check != NULL){

        $secID = $section_check['sectionID'];

        $reg_check = $con->query("SELECT * FROM registration_t WHERE sectionID = '$secID' AND studentID = '$studentID'")->fetch_assoc();

        if($reg_check ==  NULL){
            $reg_insert = "INSERT INTO registration_t (registrationID, sectionID, studentID)
                    VALUES (NULL, '$secID', '$studentID')";
                    if($con->query($reg_insert) == TRUE){
                        $reg_check = $con->query("SELECT * FROM registration_t WHERE sectionID = '$secID' AND studentID = '$studentID'")->fetch_assoc();
                        $new_registrationID = $reg_check['registrationID'];
                    }
        }
        else{
            $new_registrationID = $reg_check['registrationID'];
        }
    }
    else{
        $section_insert = "INSERT INTO section_t (sectionID, sectionNum, semester, courseID, facultyID, year)
      VALUES (NULL, '$section', '$semester', '$course', '$id', '$year')";


      if($con->query($section_insert) == TRUE){
            $section_check2 =$con->query("SELECT sectionID FROM section_t WHERE facultyID = '$id' AND courseID = '$course'
            AND sectionNUM = '$section' AND year = '$year' AND semester = '$semester'")->fetch_assoc();
                if($section_check2 != NULL){
                    $secID = $section_check2['sectionID'];
                    $reg_insert = "INSERT INTO registration_t (registrationID, sectionID, studentID)
                    VALUES (NULL, '$secID', '$studentID')";
                    if($con->query($reg_insert) == TRUE){
                        $reg_check = $con->query("SELECT * FROM registration_t WHERE sectionID = '$secID' AND studentID = '$studentID'")->fetch_assoc();
                        $new_registrationID = $reg_check['registrationID'];
                    }
                }
      }
      else{print 'not saved';}
    }


    if($grade == 'A'){ $tmo = 100; $gp=4;}
    elseif($grade == 'A-'){ $tmo = 89; $gp=3.7;}
    elseif($grade == 'B+') {$tmo = 84; $gp=3.3;}
    elseif($grade == 'B') {$tmo = 79; $gp=3;}
    elseif($grade == 'B-'){$tmo = 74;$gp=2.7;}
    elseif($grade == 'C+'){$tmo = 69;$gp=2.3;}
    elseif($grade == 'C'){$tmo = 64;$gp=2;}
    elseif($grade == 'C-'){$tmo = 59;$gp=1.7;}
    elseif($grade == 'D+'){$tmo = 54;$gp=1.3;}
    elseif($grade == 'D'){$tmo = 49;$gp=1;}
    elseif($grade == 'F'){$tmo = 0;$gp=0;}
    else {$valid = 0;}


    $scp_insert = "INSERT INTO student_course_performance_t (scpID, registrationID, totalMarksObtained, gradePoint)
    VALUES (NULL, '$new_registrationID', '$tmo', '$gp')";

if($con->query($scp_insert) == TRUE){
    $scp =$con->query("SELECT * FROM student_course_performance_t WHERE registrationID = '$new_registrationID' AND totalMarksObtained= '$tmo'
```

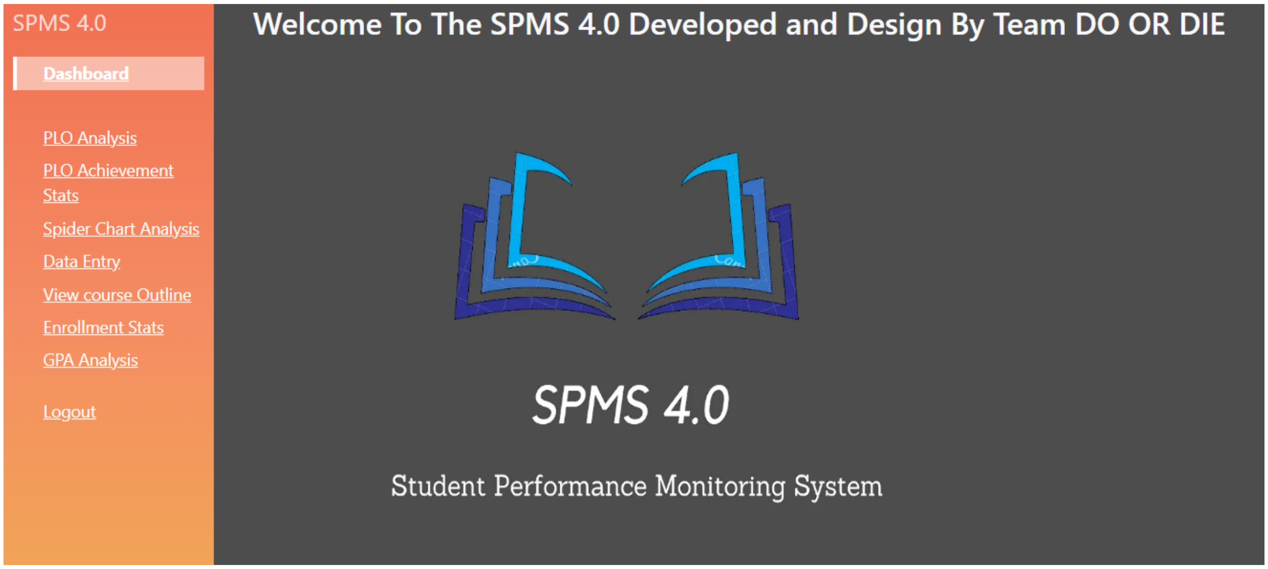
B. Existing System



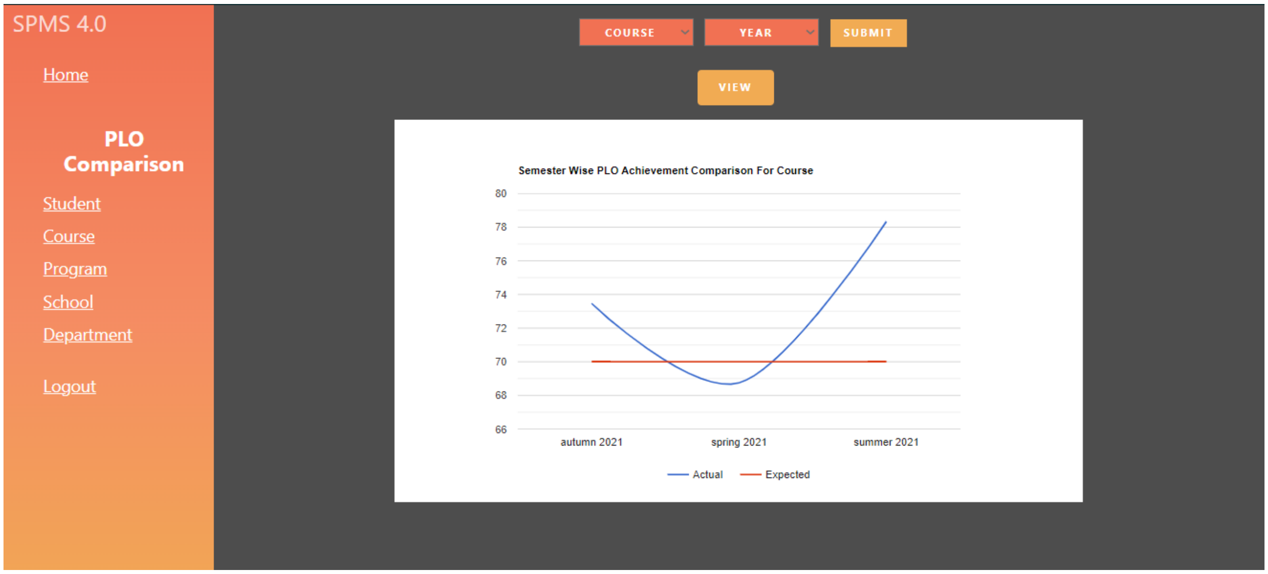
Student

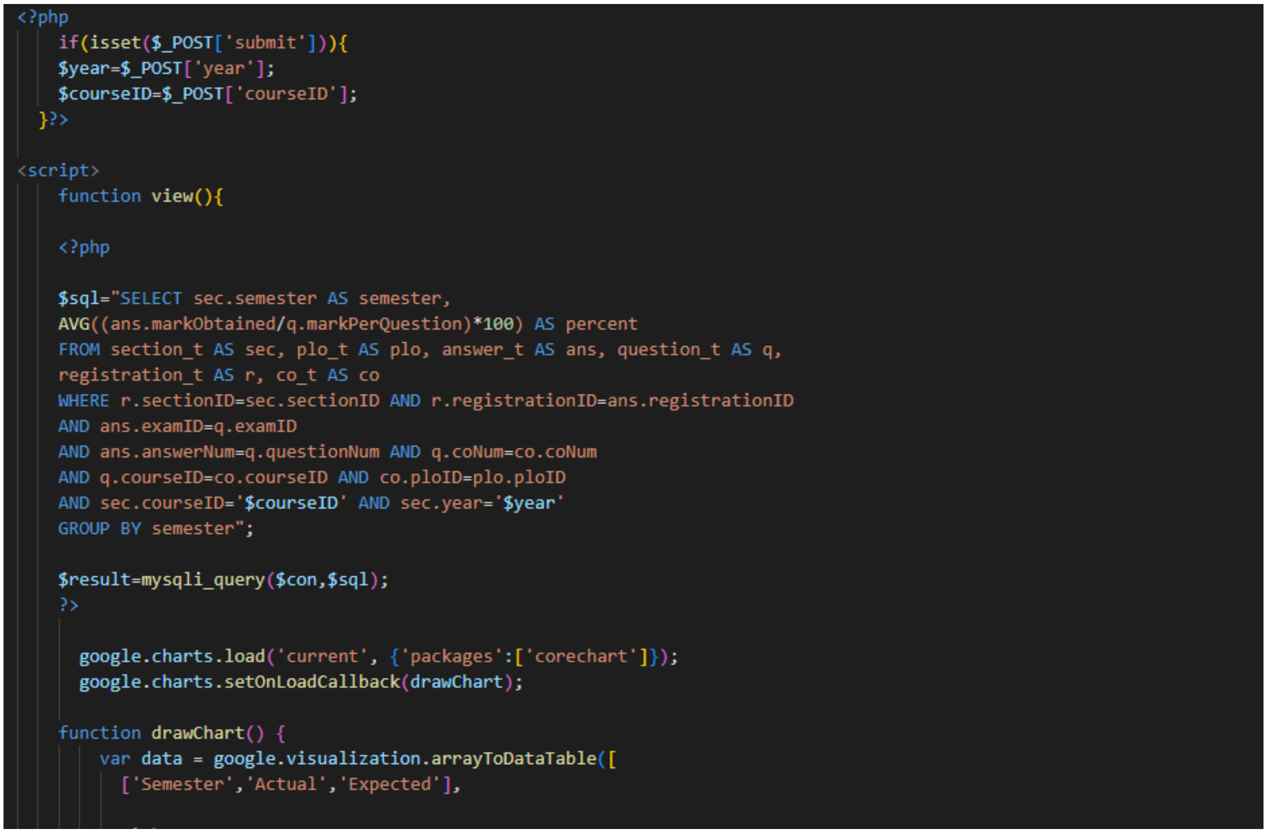
ID
15

Password
•••••

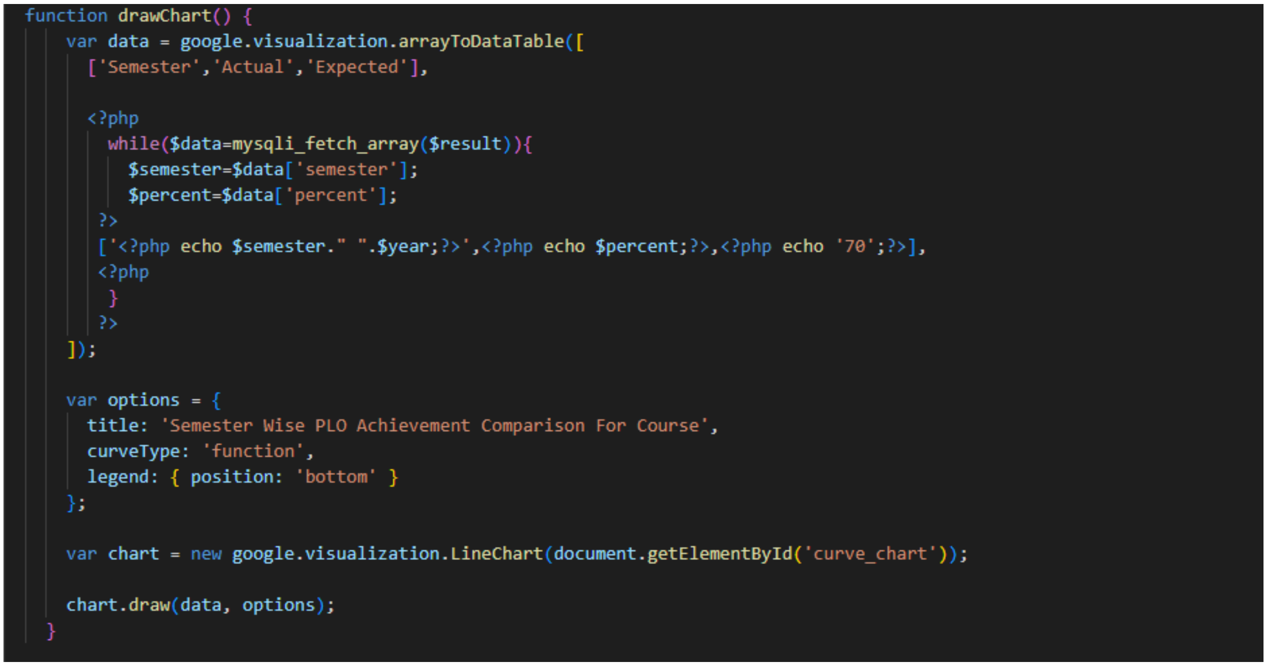


Welcome To The SPMS 4.0 Developed and Design By Team DO OR DIE

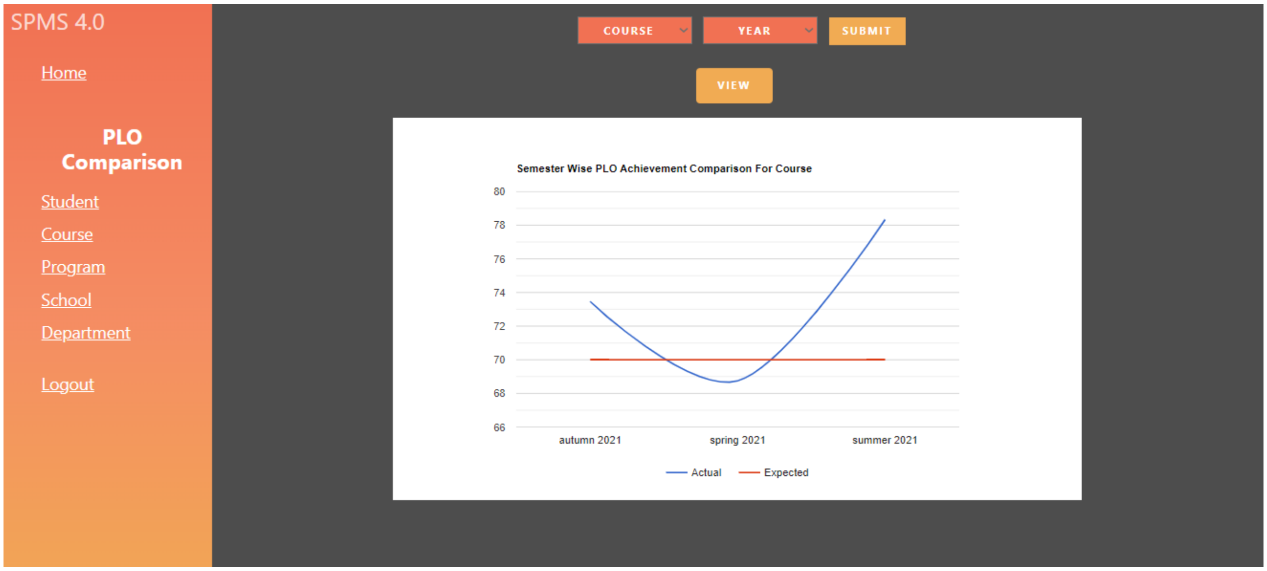


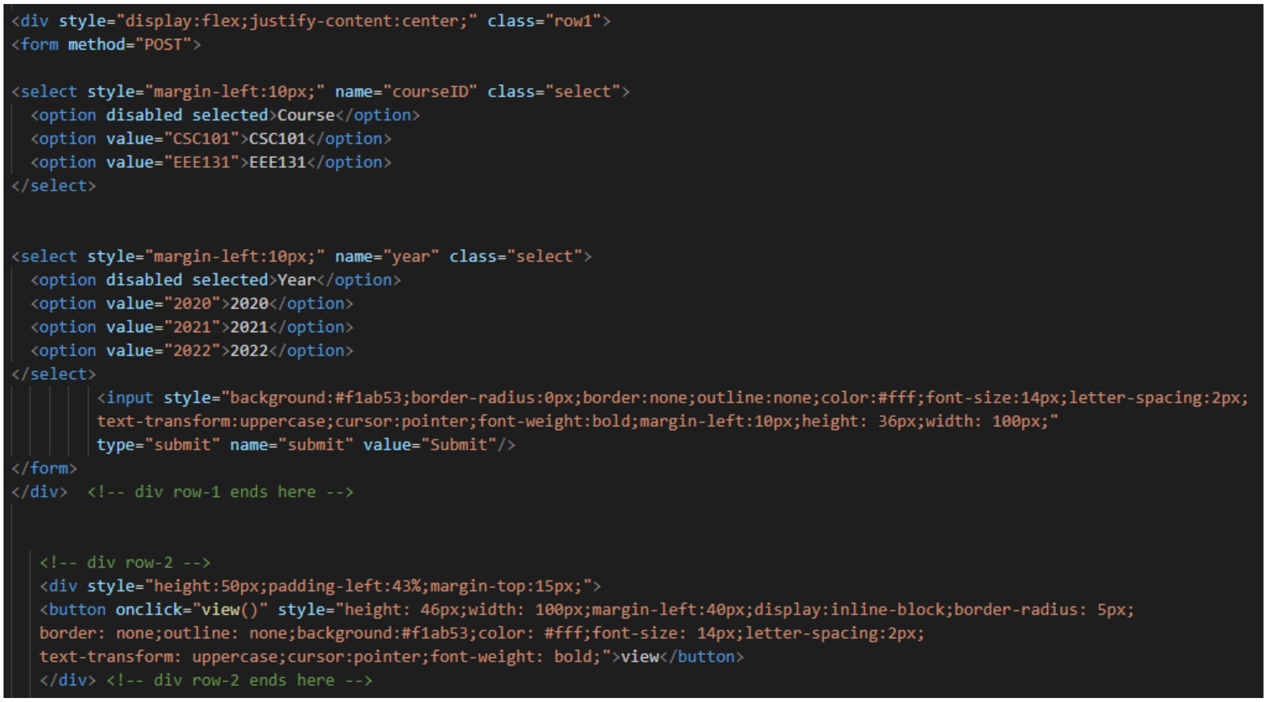


```
<?php
    if(isset($_POST['submit'])){
    $year=$_POST['year'];
    $courseID=$_POST['courseID'];
  }?>

<script>
    function view(){

    <?php

    $sql="SELECT sec.semester AS semester,
    AVG((ans.markObtained/q.markPerQuestion)*100) AS percent
    FROM section_t AS sec, plo_t AS plo, answer_t AS ans, question_t AS q,
    registration_t AS r, co_t AS co
    WHERE r.sectionID=sec.sectionID AND r.registrationID=ans.registrationID
    AND ans.examID=q.examID
    AND ans.answerNum=q.questionNum AND q.coNum=co.coNum
    AND q.courseID=co.courseID AND co.ploID=plo.ploID
    AND sec.courseID='$courseID' AND sec.year='$year'
    GROUP BY semester";

    $result=mysqli_query($con,$sql);
    ?>

      google.charts.load('current', {'packages':['corechart']});
      google.charts.setOnLoadCallback(drawChart);

    function drawChart() {
        var data = google.visualization.arrayToDataTable([
          ['Semester','Actual','Expected'],
```

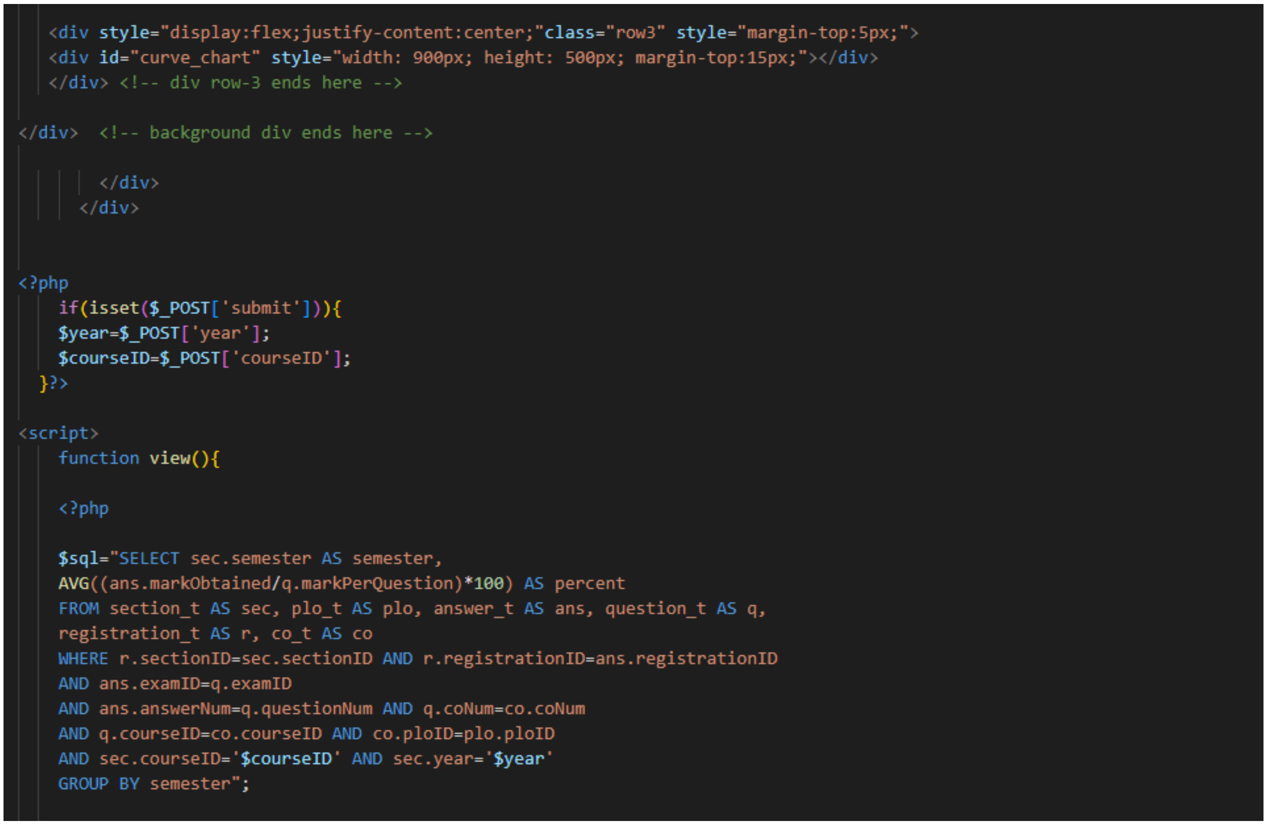


```
function drawChart() {
    var data = google.visualization.arrayToDataTable([
      ['Semester','Actual','Expected'],

      <?php
        while($data=mysqli_fetch_array($result)){
          $semester=$data['semester'];
          $percent=$data['percent'];
       ?>
       ['<?php echo $semester." ".$year;?>',<?php echo $percent;?>,<?php echo '70';?>],
       <?php
        }
       ?>
    ]);

    var options = {
      title: 'Semester Wise PLO Achievement Comparison For Course',
      curveType: 'function',
      legend: { position: 'bottom' }
    };

    var chart = new google.visualization.LineChart(document.getElementById('curve_chart'));

    chart.draw(data, options);
  }
```

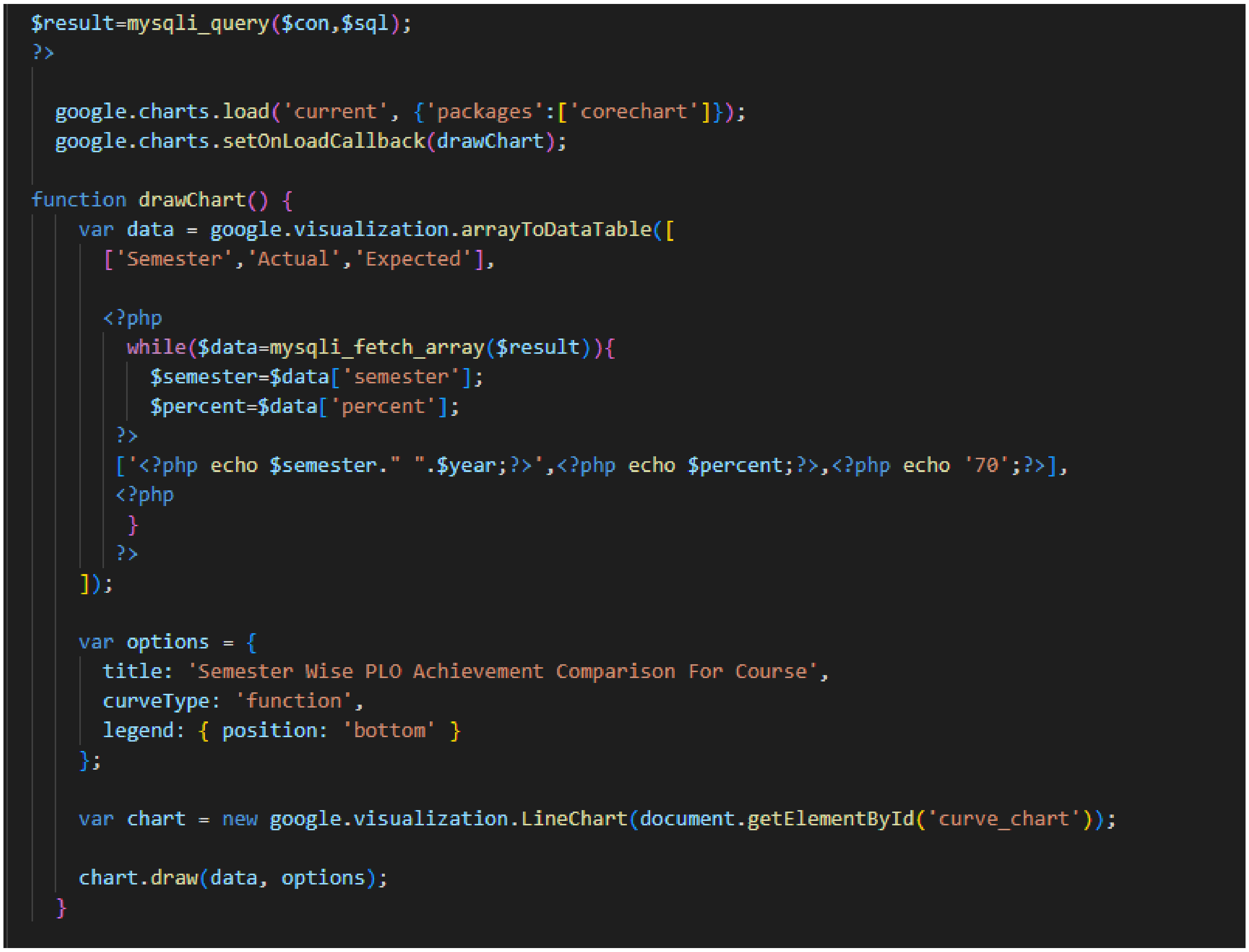




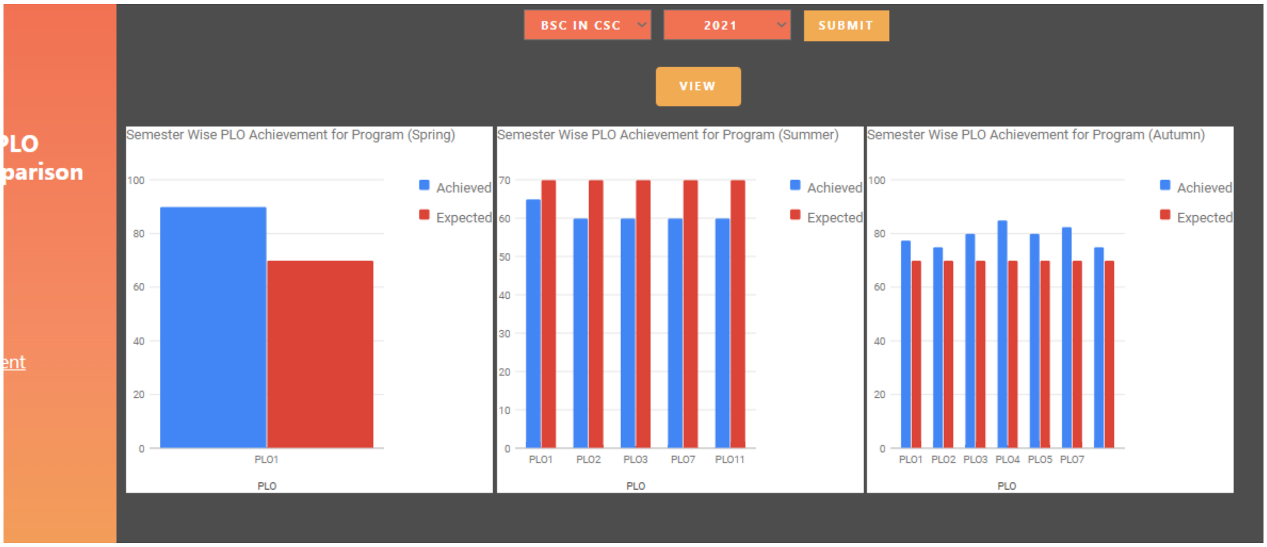
```
<div style="display:flex;justify-content:center;" class="row1">
<form method="POST">

<select style="margin-left:10px;" name="courseID" class="select">
  <option disabled selected>Course</option>
  <option value="CSC101">CSC101</option>
  <option value="EEE131">EEE131</option>
</select>


<select style="margin-left:10px;" name="year" class="select">
  <option disabled selected>Year</option>
  <option value="2020">2020</option>
  <option value="2021">2021</option>
  <option value="2022">2022</option>
</select>
         <input style="background:#f1ab53;border-radius:0px;border:none;outline:none;color:#fff;font-size:14px;letter-spacing:2px;
         text-transform:uppercase;cursor:pointer;font-weight:bold;margin-left:10px;height: 36px;width: 100px;"
         type="submit" name="submit" value="Submit"/>
</form>
</div>  <!-- div row-1 ends here -->


   <!-- div row-2 -->
   <div style="height:50px;padding-left:43%;margin-top:15px;">
   <button onclick="view()" style="height: 46px;width: 100px;margin-left:40px;display:inline-block;border-radius: 5px;
   border: none;outline: none;background:#f1ab53;color: #fff;font-size: 14px;letter-spacing:2px;
   text-transform: uppercase;cursor:pointer;font-weight: bold;">view</button>
   </div> <!-- div row-2 ends here -->
```

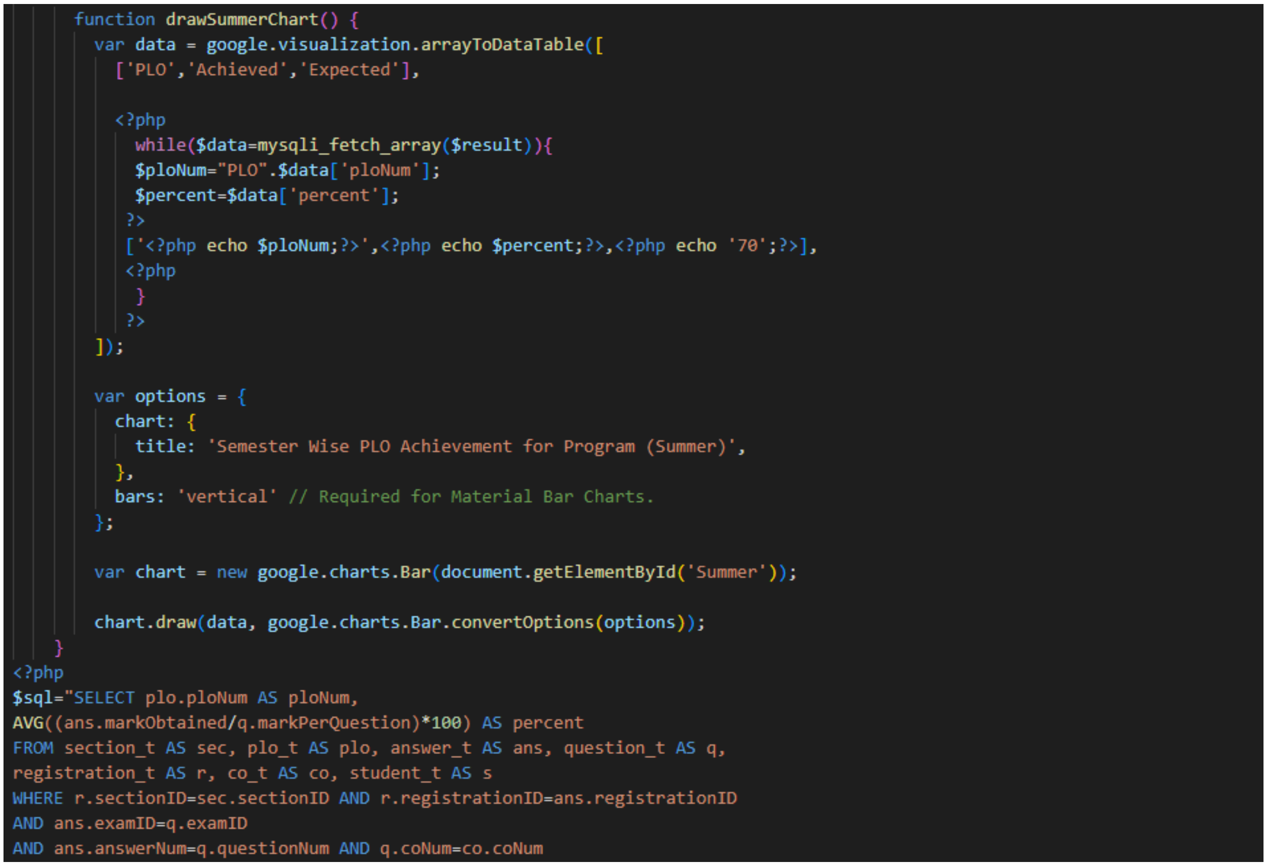


```
    <div style="display:flex;justify-content:center;"class="row3" style="margin-top:5px;">
    <div id="curve_chart" style="width: 900px; height: 500px; margin-top:15px;"></div>
    </div> <!-- div row-3 ends here -->

</div>  <!-- background div ends here -->

        </div>
      </div>


<?php
    if(isset($_POST['submit'])){
    $year=$_POST['year'];
    $courseID=$_POST['courseID'];
  }?>

<script>
    function view(){

    <?php

    $sql="SELECT sec.semester AS semester,
    AVG((ans.markObtained/q.markPerQuestion)*100) AS percent
    FROM section_t AS sec, plo_t AS plo, answer_t AS ans, question_t AS q,
    registration_t AS r, co_t AS co
    WHERE r.sectionID=sec.sectionID AND r.registrationID=ans.registrationID
    AND ans.examID=q.examID
    AND ans.answerNum=q.questionNum AND q.coNum=co.coNum
    AND q.courseID=co.courseID AND co.ploID=plo.ploID
    AND sec.courseID='$courseID' AND sec.year='$year'
    GROUP BY semester";
```

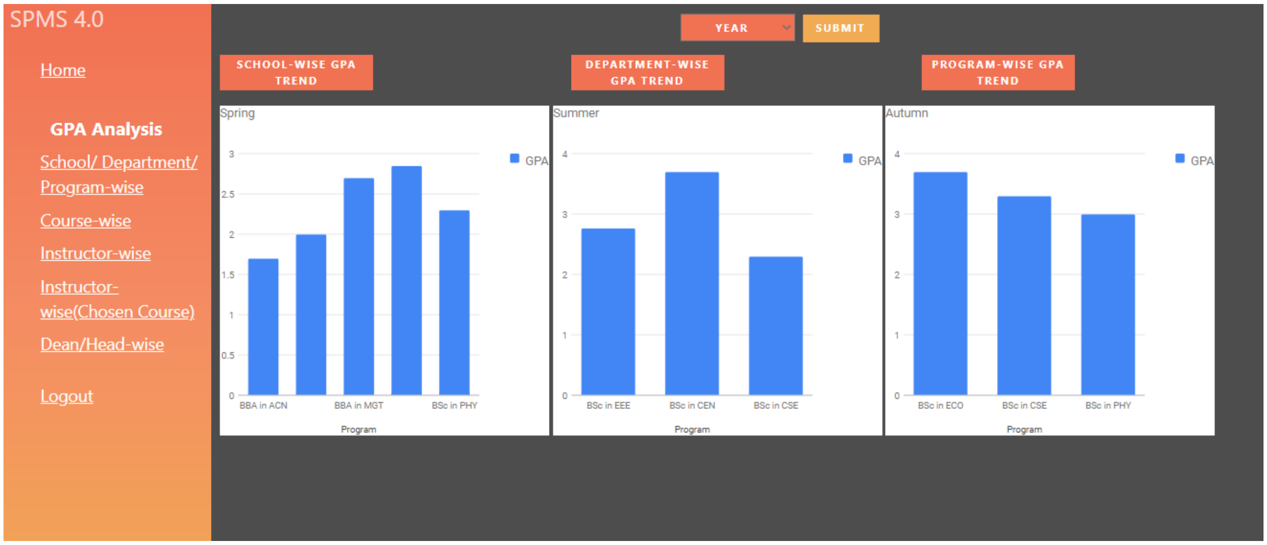


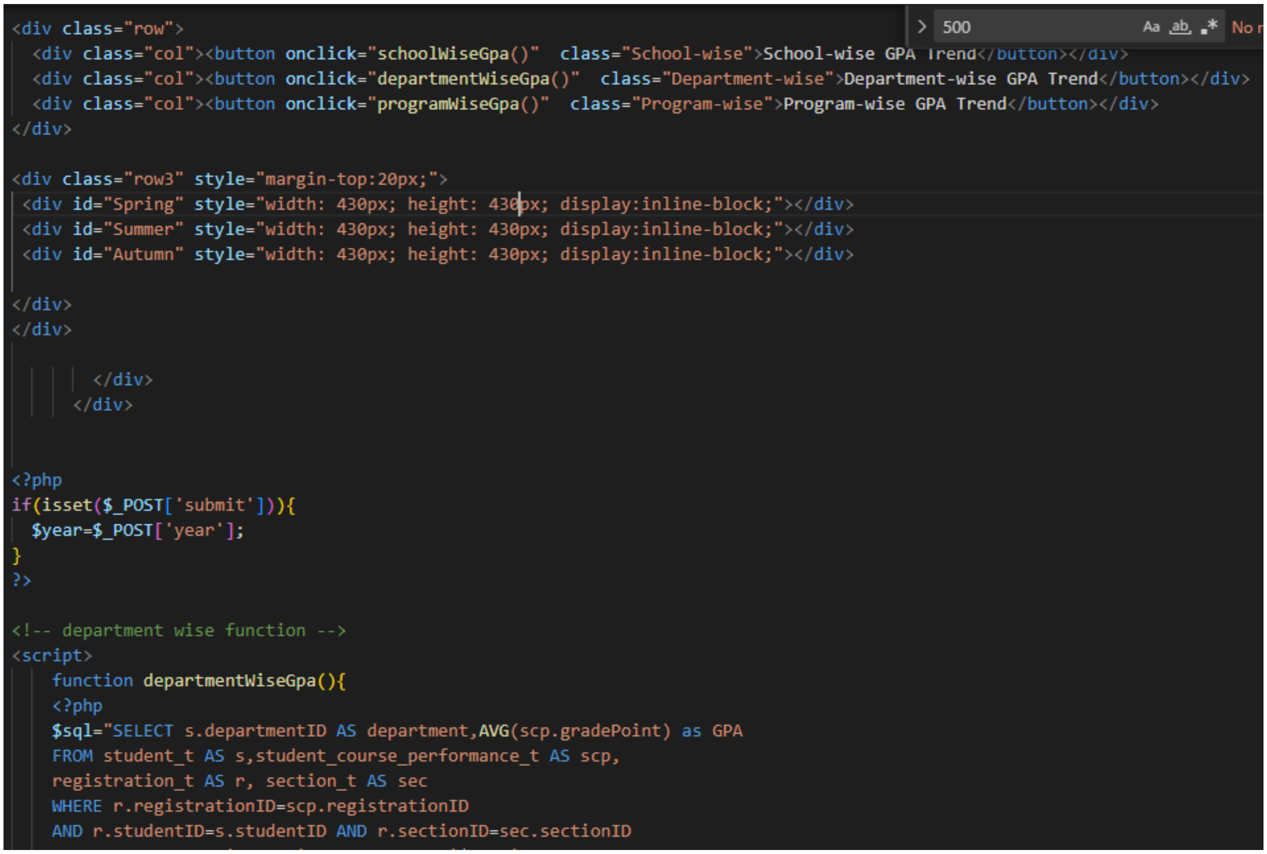
```
$result=mysqli_query($con,$sql);
?>

  google.charts.load('current', {'packages':['corechart']});
  google.charts.setOnLoadCallback(drawChart);

function drawChart() {
    var data = google.visualization.arrayToDataTable([
      ['Semester','Actual','Expected'],

      <?php
        while($data=mysqli_fetch_array($result)){
          $semester=$data['semester'];
          $percent=$data['percent'];
       ?>
       ['<?php echo $semester." ".$year;?>',<?php echo $percent;?>,<?php echo '70';?>],
       <?php
        }
       ?>
    ]);

    var options = {
      title: 'Semester Wise PLO Achievement Comparison For Course',
      curveType: 'function',
      legend: { position: 'bottom' }
    };

    var chart = new google.visualization.LineChart(document.getElementById('curve_chart'));

    chart.draw(data, options);
  }
```



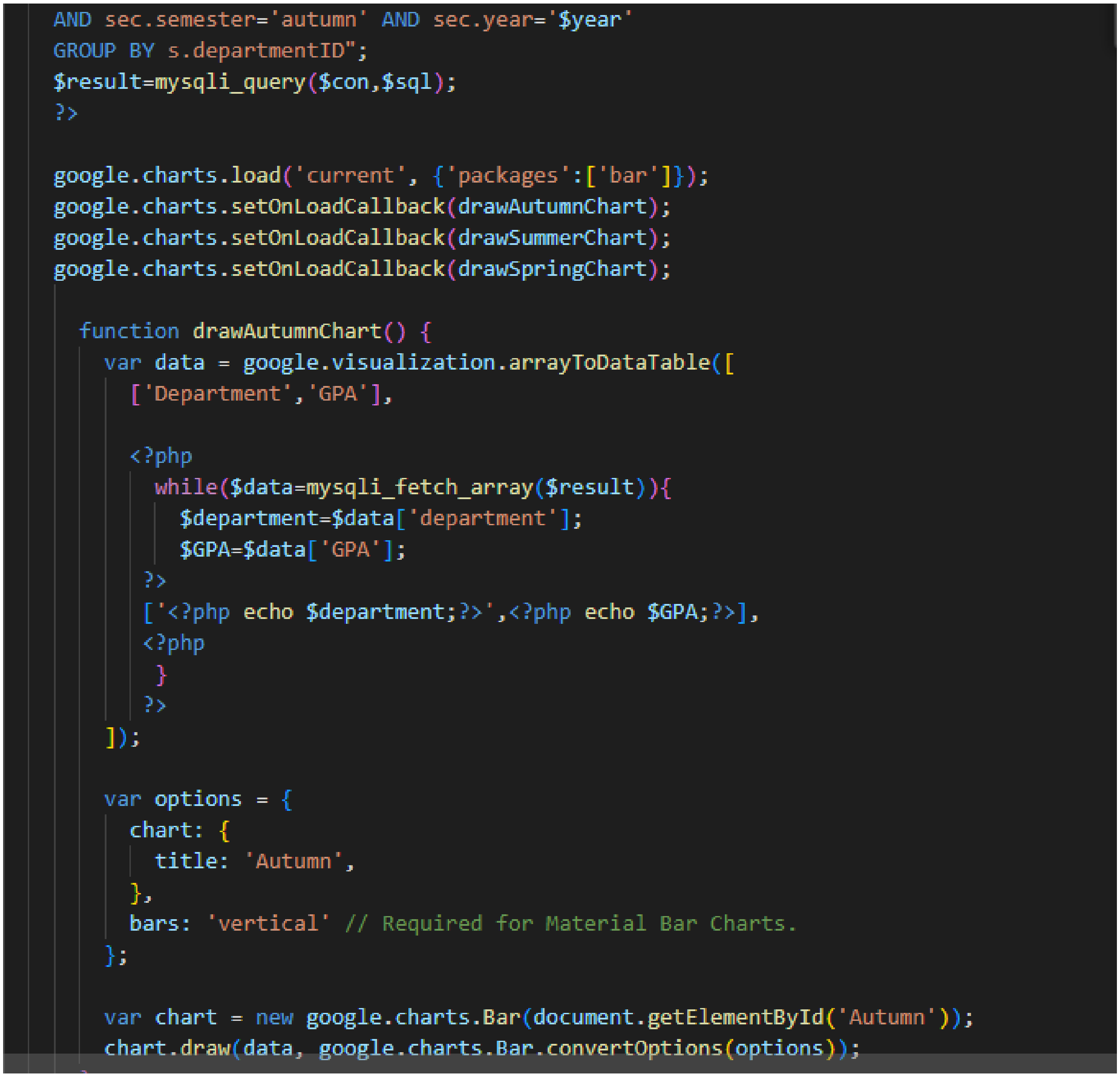


```
        function drawSummerChart() {
          var data = google.visualization.arrayToDataTable([
            ['PLO','Achieved','Expected'],

            <?php
              while($data=mysqli_fetch_array($result)){
              $ploNum="PLO".$data['ploNum'];
              $percent=$data['percent'];
             ?>
             ['<?php echo $ploNum;?>',<?php echo $percent;?>,<?php echo '70';?>],
             <?php
              }
             ?>
          ]);

          var options = {
            chart: {
              title: 'Semester Wise PLO Achievement for Program (Summer)',
            },
            bars: 'vertical' // Required for Material Bar Charts.
          };

          var chart = new google.charts.Bar(document.getElementById('Summer'));

          chart.draw(data, google.charts.Bar.convertOptions(options));
      }
<?php
$sql="SELECT plo.ploNum AS ploNum,
AVG((ans.markObtained/q.markPerQuestion)*100) AS percent
FROM section_t AS sec, plo_t AS plo, answer_t AS ans, question_t AS q,
registration_t AS r, co_t AS co, student_t AS s
WHERE r.sectionID=sec.sectionID AND r.registrationID=ans.registrationID
AND ans.examID=q.examID
AND ans.answerNum=q.questionNum AND q.coNum=co.coNum
```

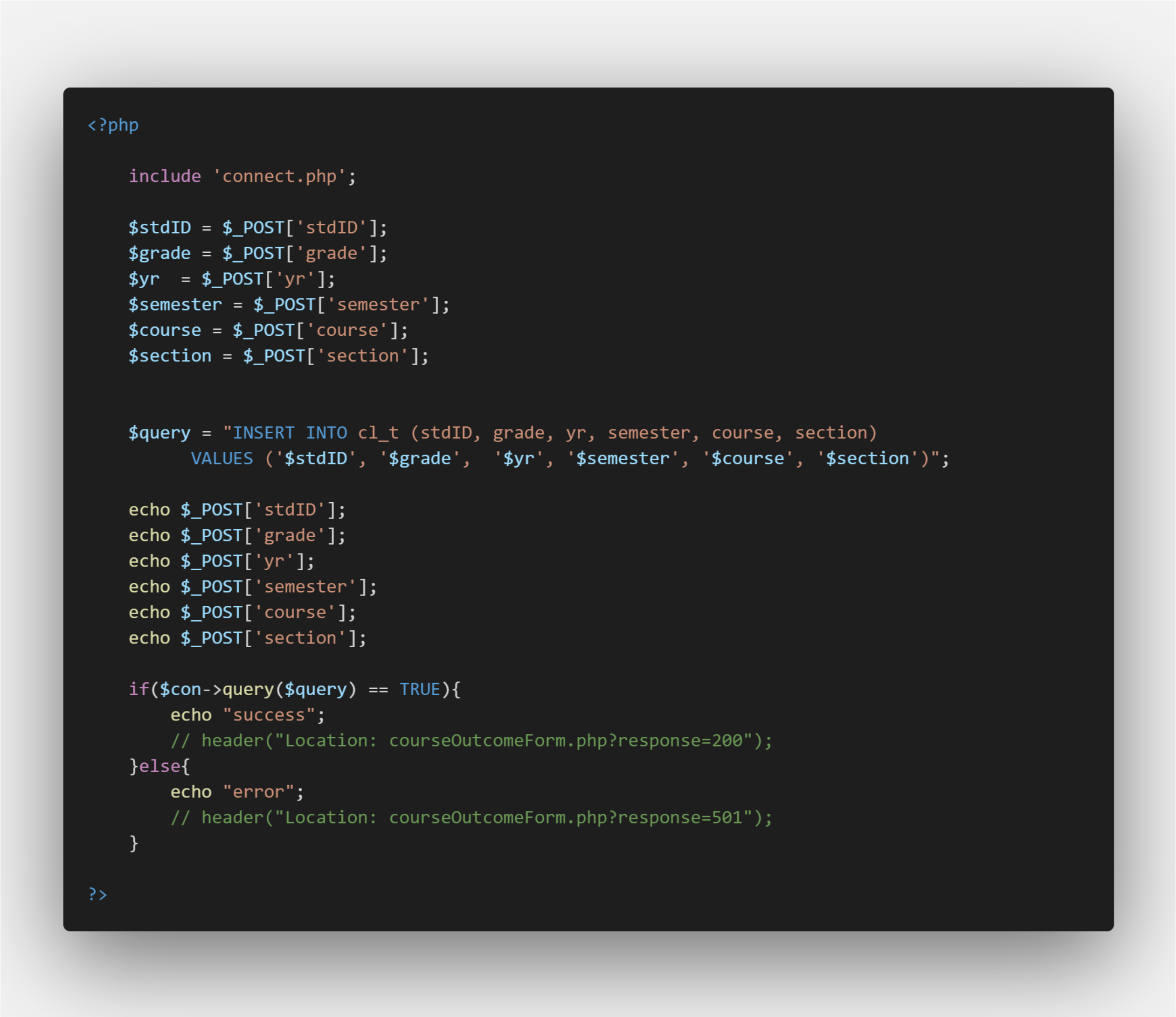




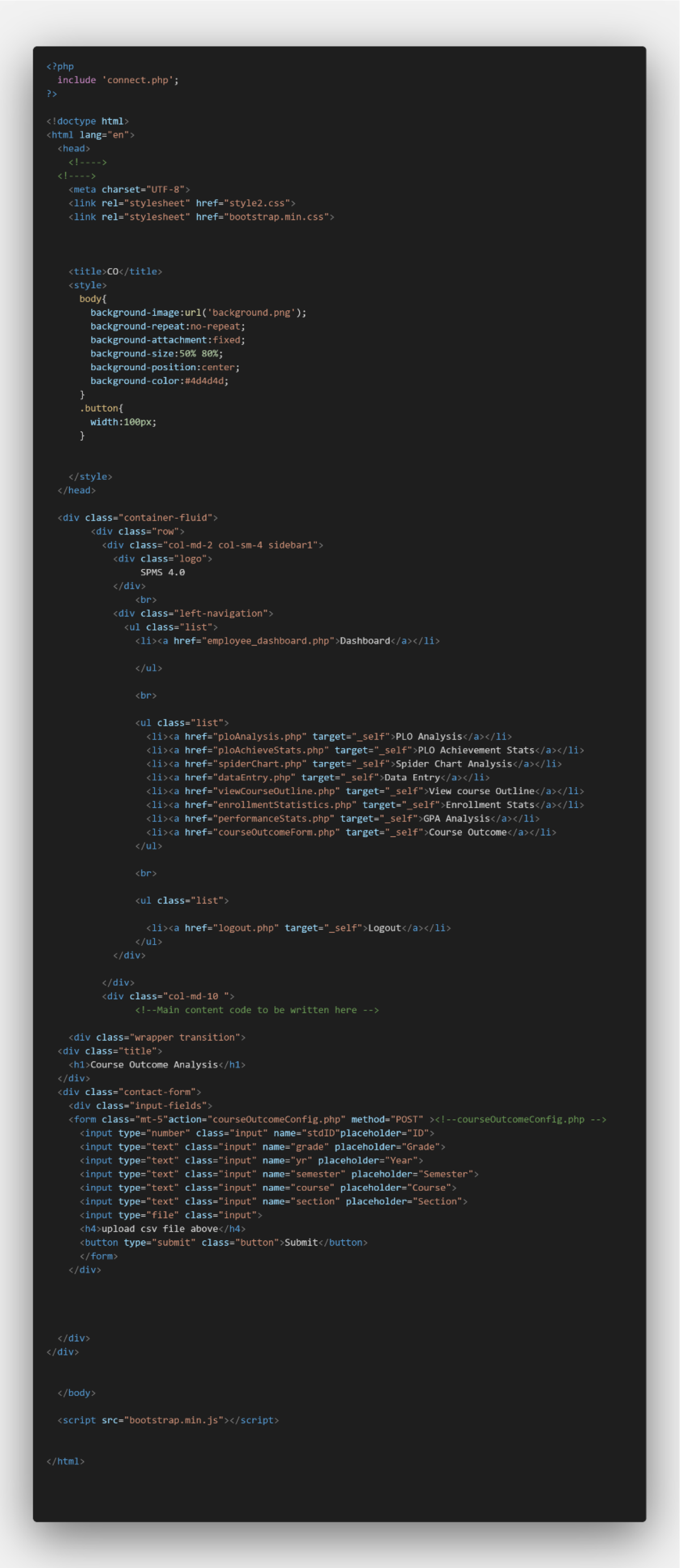
```
<div class="row">
  <div class="col"><button onclick="schoolWiseGpa()"  class="School-wise">School-wise GPA Trend</button></div>
  <div class="col"><button onclick="departmentWiseGpa()"  class="Department-wise">Department-wise GPA Trend</button></div>
  <div class="col"><button onclick="programWiseGpa()"  class="Program-wise">Program-wise GPA Trend</button></div>
</div>

<div class="row3" style="margin-top:20px;">
 <div id="Spring" style="width: 430px; height: 430px; display:inline-block;"></div>
 <div id="Summer" style="width: 430px; height: 430px; display:inline-block;"></div>
 <div id="Autumn" style="width: 430px; height: 430px; display:inline-block;"></div>

</div>
</div>

        </div>
      </div>


<?php
if(isset($_POST['submit'])){
  $year=$_POST['year'];
}
?>

<!-- department wise function -->
<script>
    function departmentWiseGpa(){
    <?php
    $sql="SELECT s.departmentID AS department,AVG(scp.gradePoint) as GPA
    FROM student_t AS s,student_course_performance_t AS scp,
    registration_t AS r, section_t AS sec
    WHERE r.registrationID=scp.registrationID
    AND r.studentID=s.studentID AND r.sectionID=sec.sectionID
```



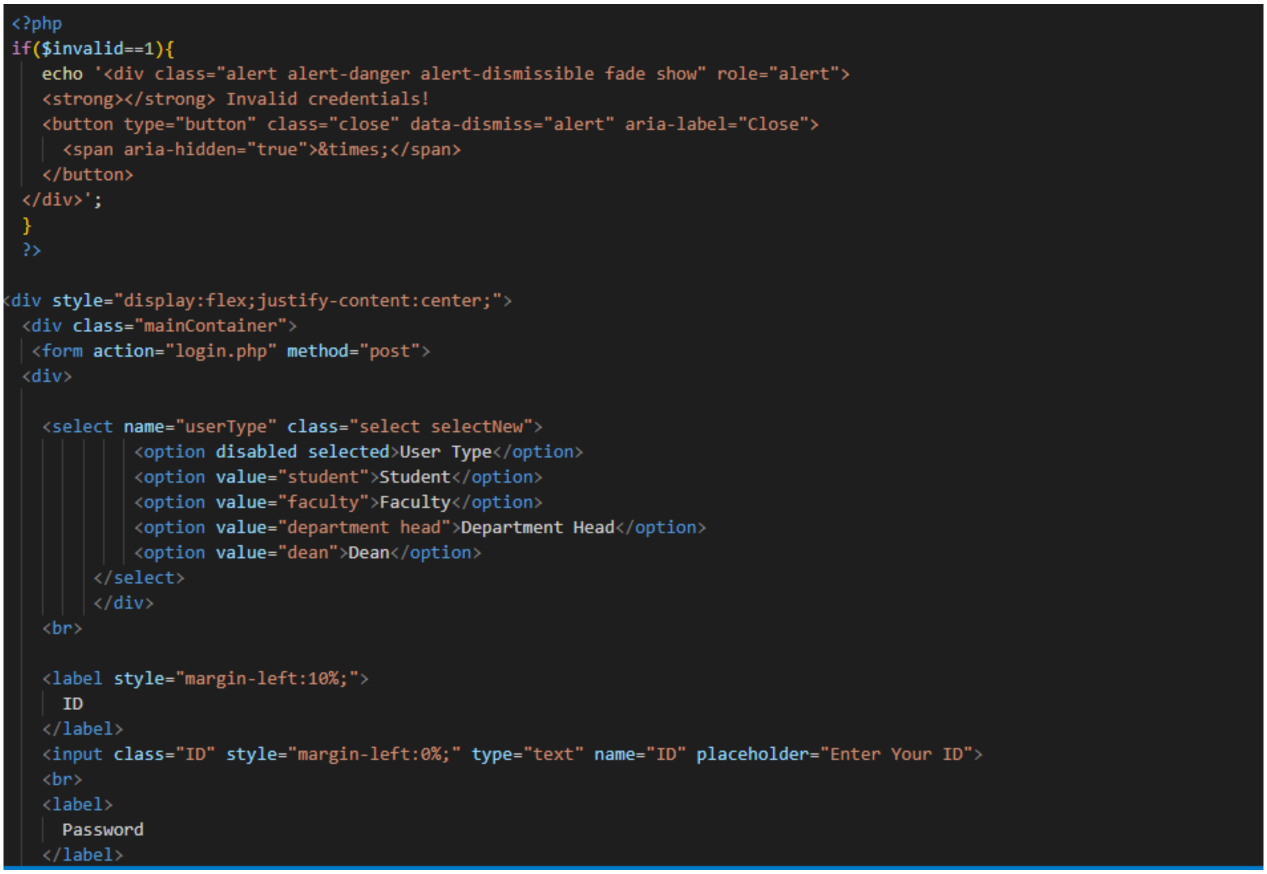
```
AND sec.semester='autumn' AND sec.year='$year'
GROUP BY s.departmentID";
$result=mysqli_query($con,$sql);
?>

google.charts.load('current', {'packages':['bar']});
google.charts.setOnLoadCallback(drawAutumnChart);
google.charts.setOnLoadCallback(drawSummerChart);
google.charts.setOnLoadCallback(drawSpringChart);

  function drawAutumnChart() {
    var data = google.visualization.arrayToDataTable([
      ['Department','GPA'],

      <?php
        while($data=mysqli_fetch_array($result)){
          $department=$data['department'];
          $GPA=$data['GPA'];
       ?>
       ['<?php echo $department;?>',<?php echo $GPA;?>],
       <?php
        }
       ?>
    ]);

    var options = {
      chart: {
        title: 'Autumn',
      },
      bars: 'vertical' // Required for Material Bar Charts.
    };

    var chart = new google.charts.Bar(document.getElementById('Autumn'));
    chart.draw(data, google.charts.Bar.convertOptions(options));
```



```php
<?php

    include 'connect.php';

    $stdID = $_POST['stdID'];
    $grade = $_POST['grade'];
    $yr  = $_POST['yr'];
    $semester = $_POST['semester'];
    $course = $_POST['course'];
    $section = $_POST['section'];


    $query = "INSERT INTO cl_t (stdID, grade, yr, semester, course, section)
          VALUES ('$stdID', '$grade',  '$yr', '$semester', '$course', '$section')";

    echo $_POST['stdID'];
    echo $_POST['grade'];
    echo $_POST['yr'];
    echo $_POST['semester'];
    echo $_POST['course'];
    echo $_POST['section'];

    if($con->query($query) == TRUE){
        echo "success";
        // header("Location: courseOutcomeForm.php?response=200");
    }else{
        echo "error";
        // header("Location: courseOutcomeForm.php?response=501");
    }

?>
```



```
<?php
  include 'connect.php';
?>

<!doctype html>
<html lang="en">
  <head>
    <!---->
  <!---->
    <meta charset="UTF-8">
    <link rel="stylesheet" href="style2.css">
    <link rel="stylesheet" href="bootstrap.min.css">



    <title>CO</title>
    <style>
      body{
        background-image:url('background.png');
        background-repeat:no-repeat;
        background-attachment:fixed;
        background-size:50% 80%;
        background-position:center;
        background-color:#4d4d4d;
      }
      .button{
        width:100px;
      }


    </style>
  </head>

  <div class="container-fluid">
        <div class="row">
          <div class="col-md-2 col-sm-4 sidebar1">
            <div class="logo">
                 SPMS 4.0
            </div>
                <br>
            <div class="left-navigation">
              <ul class="list">
                <li><a href="employee_dashboard.php">Dashboard</a></li>

                </ul>

                <br>

                <ul class="list">
                  <li><a href="ploAnalysis.php" target="_self">PLO Analysis</a></li>
                  <li><a href="ploAchieveStats.php" target="_self">PLO Achievement Stats</a></li>
                  <li><a href="spiderChart.php" target="_self">Spider Chart Analysis</a></li>
                  <li><a href="dataEntry.php" target="_self">Data Entry</a></li>
                  <li><a href="viewCourseOutline.php" target="_self">View course Outline</a></li>
                  <li><a href="enrollmentStatistics.php" target="_self">Enrollment Stats</a></li>
                  <li><a href="performanceStats.php" target="_self">GPA Analysis</a></li>
                  <li><a href="courseOutcomeForm.php" target="_self">Course Outcome</a></li>
                </ul>

                <br>

                <ul class="list">

                  <li><a href="logout.php" target="_self">Logout</a></li>
                </ul>
            </div>

          </div>
          <div class="col-md-10 ">
                <!--Main content code to be written here -->

    <div class="wrapper transition">
  <div class="title">
    <h1>Course Outcome Analysis</h1>
  </div>
  <div class="contact-form">
    <div class="input-fields">
    <form class="mt-5"action="courseOutcomeConfig.php" method="POST" ><!--courseOutcomeConfig.php -->
      <input type="number" class="input" name="stdID"placeholder="ID">
      <input type="text" class="input" name="grade" placeholder="Grade">
      <input type="text" class="input" name="yr" placeholder="Year">
      <input type="text" class="input" name="semester" placeholder="Semester">
      <input type="text" class="input" name="course" placeholder="Course">
      <input type="text" class="input" name="section" placeholder="Section">
      <input type="file" class="input">
      <h4>upload csv file above</h4>
      <button type="submit" class="button">Submit</button>
      </form>
    </div>




  </div>
</div>


  </body>

  <script src="bootstrap.min.js"></script>


</html>
```



```
<?php
if($invalid==1){
   echo '<div class="alert alert-danger alert-dismissible fade show" role="alert">
   <strong></strong> Invalid credentials!
   <button type="button" class="close" data-dismiss="alert" aria-label="Close">
     <span aria-hidden="true">&times;</span>
   </button>
 </div>';
 }
 ?>

<div style="display:flex;justify-content:center;">
  <div class="mainContainer">
   <form action="login.php" method="post">
  <div>

    <select name="userType" class="select selectNew">
             <option disabled selected>User Type</option>
             <option value="student">Student</option>
             <option value="faculty">Faculty</option>
             <option value="department head">Department Head</option>
             <option value="dean">Dean</option>
         </select>
         </div>
    <br>

    <label style="margin-left:10%;">
      ID
    </label>
    <input class="ID" style="margin-left:0%;" type="text" name="ID" placeholder="Enter Your ID">
    <br>
    <label>
      Password
    </label>
```

CHAPTER- 5 CONCLUSION

A.PROBLEM AND SOLUTION:

Analysis Phase:

Lack of information on organizational activities during the analysis phase caused uncertainty regarding the Rich Picture and Six Element Analysis. The project team conducted interviews with faculty members and other stakeholders to collect pertinent data about the system being developed in order to resolve this problem.

The project team was able to clarify any ambiguities or misconceptions regarding the Rich Picture and Six Element Analysis thanks to the information gathered from the interviews. Additionally, it assisted in identifying possible system flaws and limitations that might influence their development and application.

Additionally, by including the stakeholders in the development process, the project team was able to create a sense of ownership and collaboration. This can result in a system that is more successful and effective because it considers the requirements and viewpoints of those who will be impacted by it.

Designing Phase:

The project team faced a number of difficulties when developing the Enhanced Entity-Relationship Diagram (EERD) and Relational Schema during the Design Phase. However, the team was able to resolve these problems thanks to ongoing feedback from the faculty members.

Faculty members' comments gave us important insights into the needs of the system, which helped us better understand the EERD and Relational Schema. The design of the system was made in accordance with the requirements and expectations of the stakeholders thanks to the iterative process of getting feedback and revising the design.

The project team was able to overcome design obstacles while also identifying possible problems that might impair the system's usability or functionality. As a result, the team was able to make the necessary modifications to the EERD and Relational Schema, which eventually enhanced the system's overall functionality.

Additionally, including the faculty members in the design process increased their sense of ownership and participation in the project while also facilitating the resolution of design-related issues. As a result, stakeholders may adopt and accept the system more readily once it is put into place because they will feel that their feedback has been considered.

Implementation Phase:

All the System Requirements were completed successfully.

Front-End Developing tools: HTML, CSS, JavaScript, Google Charts, Chart JS
Back-End Developing tools: PHP, JSON
Database-integration: MySQL

Additional Features and Future Development:

A peer-to-peer learning platform is a function that could be added to a student educational monitoring system. Students could interact with one another and share knowledge using this function on an online platform.

The peer-to-peer learning platform might have tools like discussion boards, online study groups, and team tasks. The forum allowed students to pose questions, share their knowledge and experience, and give constructive criticism to one another's work.

Peer tutoring, in which students who succeed in particular subjects help those who are struggling, could also be made possible by the platform. Students would have the chance to support their own learning while also assisting their classmates.

The peer-to-peer learning platform could promote a feeling of community and collaboration in the classroom by allowing students to learn from one another. Additionally, it might contribute to the development of a learning atmosphere that is more geared toward the interests and needs of the students themselves.

References

http://www.iub.edu.bd/AboutIUB/ataglance. [Online].
http://www.cse.iub.edu.bd/degrees/1. [Online].