



Database Management Project

Section-2, Final Report

TEAM- DB_Crasher

Group 21

Name	ID
Mohammad Azwad Saadat Sarwar	1910902

Table of Contents

CHAPTER-1 INTRODUCTION.....	4
A. BACKGROUND OF THE ORGANIZATION - IUB.....	4
B. BACKGROUND OF THE PROJECT - SPMS 4.0:	4
C. OBJECTIVE OF THE PROJECT - SPMS 4.0.....	4
D. SCOPE OF THE PROJECT:	5
CHAPTER-2 REQUIREMENT ANALYSIS	5
A. RICH PICTURE – EXISTING SYSTEM (SPMS 3.0).....	6
B. SIX ELEMENT ANALYSIS – EXISTING SYSTEM (SPMS 3.0)	7
C. PROBLEM ANALYSIS – EXISTING SYSTEM (SPMS 3.0)	25
D. RICH PICTURE – PROPOSED SYSTEM (SPMS 4.0)	27
E. SIX ELEMENT ANALYSIS – PROPOSED SYSTEM (SPMS 4.0).....	28
CHAPTER-3 LOGICAL SYSTEM DESIGN	44
A. BUSINESS RULES – SPMS 4.0	44
B. ENTITY RELATIONSHIP DIAGRAM	47
C. ENTITY RELATIONSHIP DIAGRAM TO RELATIONAL SCHEMA	48
D. NORMALIZATION.....	48
1NF:	51
2NF:	52
3NF:	52
E.DATA DICTIONARY.....	52
CHAPTER-4 PHYSICAL SYSTEM DESIGN	62
A. INPUT FORMS.....	62
B. OUTPUT FORMS.....	72
CHAPTER-5 CONCLUSION	101
A. PROBLEM AND SOLUTION.....	101
Analysis Phase:	101
Designing Phase:.....	102
Implementation Phase:	102
Additional Features and Future Development	102
References.....	102

CHAPTER-1 INTRODUCTION

A. BACKGROUND OF THE ORGANIZATION - IUB:

IUB is a private university in Bangladesh which was established in 1993. It has a current enrollment of around 10 000 at undergraduate and graduate levels [1]. More than 450 highly qualified and highly-skilled faculty members who are exceptionally good at their department of teaching and where at least half of them are PhD degree holders [2]. IUB also has an alumni strength near 14000 people. There are currently 5 academic schools in IUB [1]

- 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 3.0:

Student Performance Monitoring System (SPMS 3.0) is a framework for Outcome-Based Education (OBE). It evaluates the performance of students, course instructors, schools, departments and programs and helps the Higher Authorities of the education institution to make strategies for improvements.

C. OBJECTIVE OF THE PROJECT - SPMS 3.0:

SPMS 3.0 monitors and analyzes the performance of its stakeholders such as Students, Course instructors, Departments, Schools, Programs through the database of assessments such as quizzes, midterm exams, final term exams etc. In order to evaluate the performance of the stakeholders, SPMS 3.0 stores necessary documents and data in the database such as all the exam question papers, answer scripts, course outlines and marks of the exams and assessments with respect to their Course Outcomes (CO), Program Learning Outcomes (PLO) and Program Outcomes (PO) achieved by the students. Hence, students can statistically monitor their own performance. SPMS 3.0 also creates opportunities for Higher Authorities to draw conclusions and make further

improvements by providing them with a wide range of analytical reports based on the performance of students, course instructors, departments, schools, and programs.

D. SCOPE OF THE PROJECT:

We have done a complete analysis of the existing system (SPMS 2.0) and identified some issues in the business processes which can cause the process to become slow, inefficient and cause lapses in communication.

The proposed solution to overcome those issues is to create a more improved version of the system called SPMS 3.0 (Student Performance Monitoring System 3.0) which uses a Relational Database Management System (RDBMS) to store, update and retrieve necessary documents such as Course Outlines, Exam Question Papers and Answer Scripts as well as other necessary data required to monitor student performance and produce other OBE (outcome-Based Education) reports.

We have identified all the users of the system (SPMS 3.0), how they would be accessing the necessary data and information and how they would interact with each other etc.

We want to build interfaces for all the users to be able to access their required data and generate, view, and download their desired reports and documents using the system.

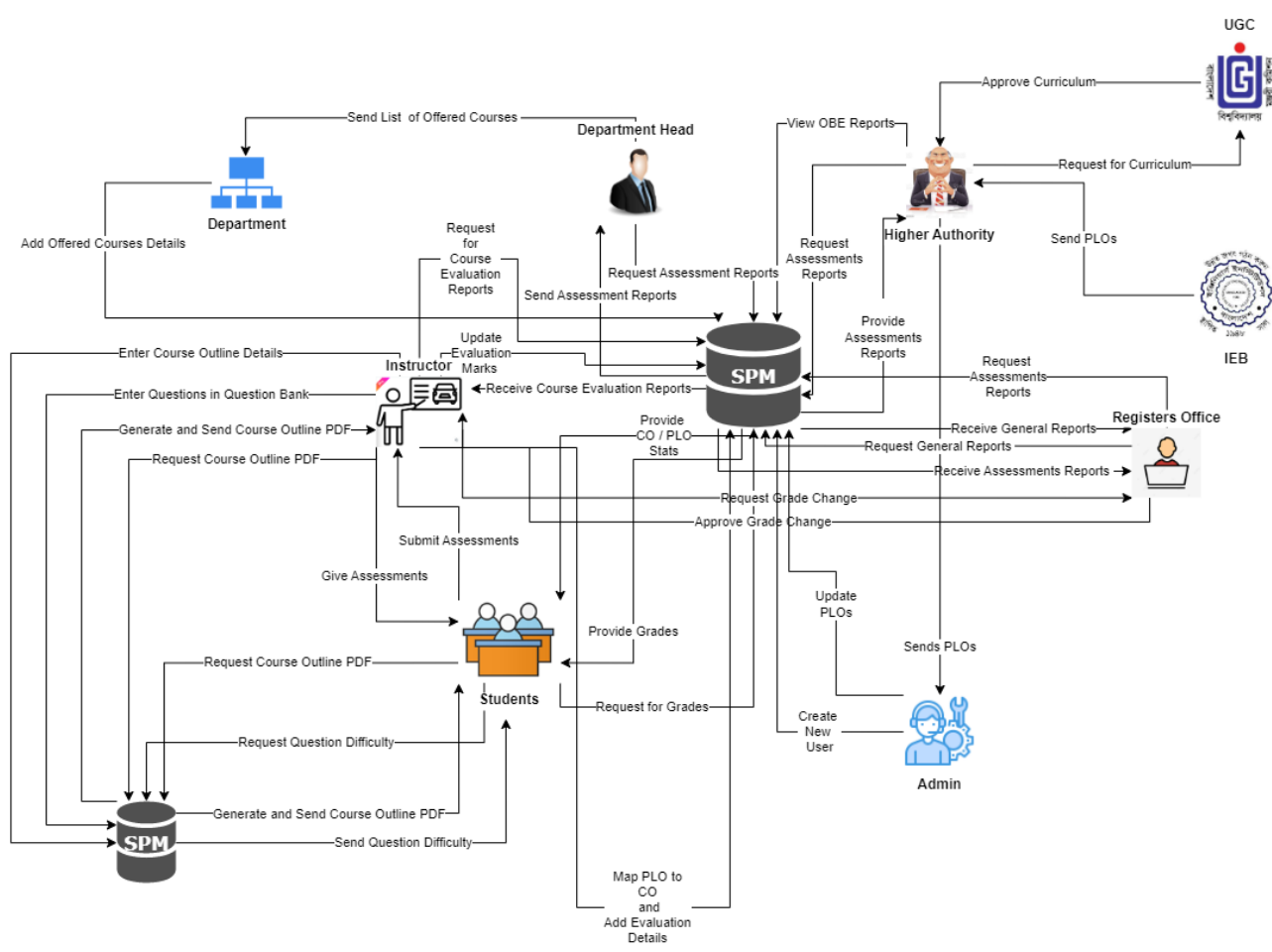
We also want to build an interface for the course instructors/faculties to be able to collaborate with each other on developing of course outlines, exam papers, marksheets etc.

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 require 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 2.0):

A rich picture is a way to demonstrate processes in a system which is easier to understand for everyone. It consists of pictures, text, symbols and icons which are all used to illustrate graphically the situation. [3] A rich picture helps us to see relationships and connections that we may otherwise miss [3]. It helps identifying one or more themes participants may want to further explore and address. Rich pictures are therefore always used in the pre-analysis phase [3].



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 Storages are

- 1) SPMS V3.0
- 2) Physical Storage (Used by the faculty)

B. SIX ELEMENT ANALYSIS – EXISTING SYSTEM (SPMS 2.0)

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

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

- 15) Student performance trend under VC/Dean/Head of Department**
- 16) Course, Program, department, school CLO-PLO statistics**
- 17) Course, student, department school wise expected vs achieved PLO**
- 18) Department average of total PLO achieved and attempted students**
- 19) Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.**
- 20)**

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
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	Pen and Paper 1) Can be Used to create Rough assessment questions. 2) Used to answer assessment questions.	Computer/Laptop 1) Used to Sign into SPMS2.0. 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.	SPMS2.0 1) Used to store student Data into the database or generate performance analysis graph using data from the database.	SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.	Internet 1) Used to Sign into SPMS2.0

	<p>period and View the progress of that students of that course.</p> <p>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.</p> <p>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</p>					
Faculty based student performance according to GPA	<p>Faculty: 1) Signs into system using their ID and Password. 2) View the Progress of the students</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer</p>	<p>SPMS2.0 1)Used to store student Data into the database or generat</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>who are being taught by them.</p> <p>Department Head:</p> <p>1) Signs into system using their ID and Password.</p> <p>2) Search for a faculty to be assessed using the Faculty's name.</p> <p>3) View the Progress of the students who are being taught under that faculty basing on the GPA earned by the students.</p> <p>Dean/VC:</p> <p>1) Signs into system using their ID and Password.</p> <p>2) Search for a faculty to be assessed using the Faculty's name and Department ID.</p> <p>3) View the Progress of the students</p>		<p>1) Used to print hard copy of the progress of students taught by a faculty</p>	<p>performance analysis graph using data from the database.</p>	<p>updated by SPMS2.0 admins.</p>	
--	--	--	---	---	-----------------------------------	--

	who are being taught under that faculty basing on the GPA earned by the students.					
Course wise PLO achievement of a student	<p>VC/ Dean: 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 student.</p> <p>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.</p> <p>Faculty: 1) Signs into system</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print hard copy of a report of students who completed most the PLO achievements If needed.</p>	<p>SPMS2.0 1)Used to store Data and generate PLO automatically based on the CO provided .</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>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.</p> <p>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.</p>					
Student performance trend under VC/Dean/Head of Department	<p>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.</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of the progress report if needed.</p>	<p>SPMS2.0 1)Used to store Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>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.</p> <p>Department Head: 1) Signs into system using their ID and Password. 2) View student progress under them.</p>					
<p>Course, Program, department, school CLO-PLO statistics</p>	<p>Dean/VC : 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students.</p> <p>Department Head: 1) Signs into system</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of the progress report if needed.</p>	<p>SPMS2.0 1)Used to store Data into the database and generate CLO-PLO statistical data or graphs.</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students.</p> <p>Faculty: 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students.</p> <p>Student: 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by them and other students.</p>					
Course, student, department school wise expected vs achieved PLO	<p>Dean/VC : 1) Sign into the system using ID and Password. 2) View the achieved PLO of the students during time entered</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of</p>	<p>SPMS2.0 1)Used to store Data into the database or generate performance analysis</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>that has been inputted and comparison between expected and achieved.</p> <p>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.</p> <p>Faculty: 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</p>		both the previous and current semester's achieved PLO to compare.	graph using data from the database.	SPMS2.0 admins.	
--	---	--	---	-------------------------------------	-----------------	--

	<p>and achieved.</p> <p>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.</p>					
<p>Department average of total PLO achieved and attempted students</p>	<p>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.</p> <p>Department Head: 1) Sign into the system using ID</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of PLO reports</p>	<p>SPMS2.0 1)Used to store Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>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.</p> <p>Faculty: 1) Sign into the system using ID and Password. 2) View the total departmental average of the PLO achieved by the students.</p> <p>Student: 1) Sign into the system using ID and Password. 2) View the total departmental average of the PLO achieved by the students</p>					
Assessments And grading	Faculty: 1) Create Assessment with	Pen and Paper	Computer/Laptop	MS OFFICE 1)Used to store	Physical Storage 1)Used for	Internet 1) Used to Sign into Google

	<p>respect to course outline. 2) Give Assessment to students. 3) Grade the Answer Script papers returned by students. 4) Store Grades To SPMSV2.0</p> <p>Student: 1) Receive Assessment from Faculty. 2) Complete and Return the Answer Scripts to the respective Faculty. 3) Get Grades From SPMSV2.0 when available</p>	<p>1) Can be Used to create Rough assessment questions. 2) Used to answer assessment questions.</p>	<p>2) To Use MS OFFICE</p> <p>Printer 1) Used to print Hard Copy Of The Assessments if required</p>	<p>the Grades in a Spreadsheet 2) Used to create the assessment.</p>	<p>Storing all the answer scripts returned by students</p>	<p>Classroom if communication is required. 2) Used to Sign into SPMS2.0</p>
<p>Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.</p>	<p>VC 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</p>		<p>Computer/Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of Student Enrollment Statistics If Needed.</p>	<p>SPMS2.0 1) Used to store Data into the database and generate Student Enrollment Statistics graphs.</p>	<p>SPMS2.0 Database 1) All valid data are stored here which can be updated by SPMS2.0 admins.</p>	<p>Internet 1) Used to Sign into SPMS2.0</p>

	<p>Statistics Of That Year and Semester.</p> <p>Dean</p> <p>1) Sign into the system using ID and Password.</p> <p>2) Select Student Enrollment Statistics tab and select Year and Semester under that tab</p> <p>3) View Student Enrollment Statistics Of That Year and Semester.</p> <p>Department Head</p> <p>1) Sign into the system using ID and Password.</p> <p>2) Select Student Enrollment Statistics tab and select Year and Semester under that tab</p> <p>3) View Student Enrollment Statistics Of That Year and Semester.</p>					
--	---	--	--	--	--	--

Preparing and storing Course Outline	Faculty: 1) Request Course Material Of Previous Semester to the Department Office. 2) Create Course Outline 3) Store Course Outline In Physical Storage. 4) Give Course Outline To Students. Department Office: 1) Give Course Material Of Previous Semester Students: 1) Receive Course Outline From Faculty	Pen and Paper 1) Used to Create Rough Course Outline Overview	Computer/Laptop 1) To Use MS OFFICE Printer 1) Used to print Hard Copy Of Course Outline	MS OFFICE 1) Used to store the Grades in a Spreadsheet 2) Used to create the assessment.	Physical Storage 1) Course Outlines are Stored Here.	Internet 1) Used to Sign into SPMS2.0

Process	Human	Non-computing Hardware	Computing Hardware	Software	Database	Network and Communication
Preparing and storing	Faculty:		Computer/Laptop	SPMS2.0	SPMS 2.0	Internet

and giving Course Outline	<p>1) Signs into System using their ID and Password. 2) Select Create Course Outline Tab. 3) Select From the options that they wish to add in their course outline. 4) Press the Create button. 4) Store course outline into system.</p> <p>Students: 1) Signs into System using their ID and Password. 2) Select Course 3)View/Download Course Outline From System.</p>		<p>1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print hard copy of course outlines if required.</p>	1)Used to store Data into the database.	Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.	1)Used to Sign into SPMS2.0
Add Questions to the question bank and grading the answer script	<p>Faculty: 1) Signs into System using their ID and Password. 2) Select course and choose section's that has to solve the question. 3) Input the question in the question bank. 4) Press the Assign Button. 4) Grade the answers submitted by the students</p> <p>Student: 1) Signs into System using their ID and Password. 2) Answer the question assigned by the faculty in the answer bank</p>		<p>Computer/ Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the grades gotten by the whole section</p>	SPMS2.0 1)Used to store Data into the database or generate result graph using data from the database.	SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.	Internet 1)Used to Sign into SPMS2.0

	3) Press the Submit button 4) Check grade in SPMS3.0 after faculty is done checking					
Course based student performance trend according to GPA	<p>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.</p> <p>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 view the progress of that students of that course.</p> <p>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.</p> <p>Dean/VC : 1) Signs into system using their ID and Password.</p>		<p>Computer/ Laptop 1) Used to Sign into SPMS2.0.</p> <p>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.</p>	<p>SPMS2.0 1)Used to store student Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	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	<p>Faculty: 1) Signs into system using their ID and Password. 2) View the Progress of the students who are being taught by them.</p> <p>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.</p> <p>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.</p>		<p>Computer/ Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print hard copy of the progress of students taught by a faculty</p>	<p>SPMS2.0 1)Used to store student Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

Course wise PLO achievement of a student	VC/ Dean: 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 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. 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.		Computer/ Laptop 1) Used to Sign into SPMS2.0 Printer 1) Used to print hard copy of a report of students who completed most the PLO achievements If needed.	SPMS2.0 1)Used to store Data and generate PLO automatically based on the CO provided.	SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.	Internet 1)Used to Sign into SPMS2.0
Student performance trend	Dean : 1) Signs into system using their ID and Password.		Computer/ Laptop	SPMS2.0 1)Used to store	SPMS 2.0 Database	Internet 1)Used to Sign into SPMS2.0

under VC/Dean/ Head of Department	<p>2) Search for Department Head to be checked using their Name and Department ID.</p> <p>3) View student progress under them or them.</p> <p>VC:</p> <p>1) Signs into system using their ID and Password.</p> <p>2) Search for a Dean or Department Head to be checked using their Name and either School ID or Department ID.</p> <p>3) View student progress under them.</p> <p>Department Head:</p> <p>1) Signs into system using their ID and Password.</p> <p>2) View student progress under them.</p>		<p>1) Used to Sign into SPMS2.0</p> <p>Printer</p> <p>1) Used to print the hard copy of the progress report if needed</p>	<p>Data into the database or generate performance analysis graph using data from the database.</p>	<p>1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	
Course, Program, department, school CLO-PLO statistics	<p>Dean/VC :</p> <p>1) Signs into system using their ID and Password.</p> <p>2) View CLO-PLO mapped statistics achieved by students.</p> <p>Department Head:</p> <p>1) Signs into system using their ID and Password.</p> <p>2) View CLO-PLO mapped statistics achieved by students.</p> <p>Faculty:</p>		<p>Computer/ Laptop</p> <p>1) Used to Sign into SPMS2.0</p> <p>Printer</p> <p>1) Used to print the hard copy of the progress report if needed</p>	<p>SPMS2.0</p> <p>1)Used to store Data into the database and generate CLO-PLO statistical data or graphs.</p>	<p>SPMS 2.0 Database</p> <p>1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet</p> <p>1)Used to Sign into SPMS2.0</p>

	<p>1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by students.</p> <p>Student: 1) Signs into system using their ID and Password. 2) View CLO-PLO mapped statistics achieved by them and other students.</p>					
<p>Course, student, department school wise expected vs achieved PLO</p>	<p>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.</p> <p>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.</p> <p>Faculty: 1) Sign into the system using ID and Password.</p>		<p>Computer/ Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of both the previous and current semester's achieved PLO to compare.</p>	<p>SPMS2.0 1)Used to store Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

	<p>2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved.</p> <p>Student:</p> <p>1) Sign into the system using ID and Password.</p> <p>2) View the achieved PLO of the students during time entered that has been inputted and comparison between expected and achieved.</p>					
Department average of total PLO achieved and attempted students	<p>Dean/VC :</p> <p>1) Sign into the system using ID and Password.</p> <p>2) Enter the time period of the semester wished to be viewed.</p> <p>3) View the departmental average of total PLO achieved along with the number of students who attempted.</p> <p>Department Head:</p> <p>1) Sign into the system using ID and Password.</p> <p>2) Enter the time period of the semester wished to be viewed.</p> <p>3) View the departmental average of total PLO achieved along with the number of</p>		<p>Computer/ Laptop</p> <p>1) Used to Sign into SPMS2.0</p> <p>Printer</p> <p>1) Used to print the hard copy of PLO reports</p>	<p>SPMS2.0</p> <p>1)Used to store Data into the database or generate performance analysis graph using data from the database.</p>	<p>SPMS 2.0 Database</p> <p>1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet</p> <p>1)Used to Sign into SPMS2.0</p>

	<p>students who attempted.</p> <p>Faculty: 1) Sign into the system using ID and Password.</p> <p>2) View the total departmental average of the PLO achieved by the students.</p> <p>Student: 1) Sign into the system using ID and Password. 2) View the total departmental average of the PLO achieved by the students</p>					
<p>Student Enrollment Statistics VC-wise, Dean-wise, Department Head-wise.</p>	<p>VC 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.</p> <p>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</p>		<p>Computer/ Laptop 1) Used to Sign into SPMS2.0</p> <p>Printer 1) Used to print the hard copy of Student Enrollment Statistics If Needed.</p>	<p>SPMS2.0 1)Used to store Data into the database and generate Student Enrollment Statistics graphs.</p>	<p>SPMS 2.0 Database 1) All valid data are stored here which can be updated by SPMS 2.0 admins.</p>	<p>Internet 1)Used to Sign into SPMS2.0</p>

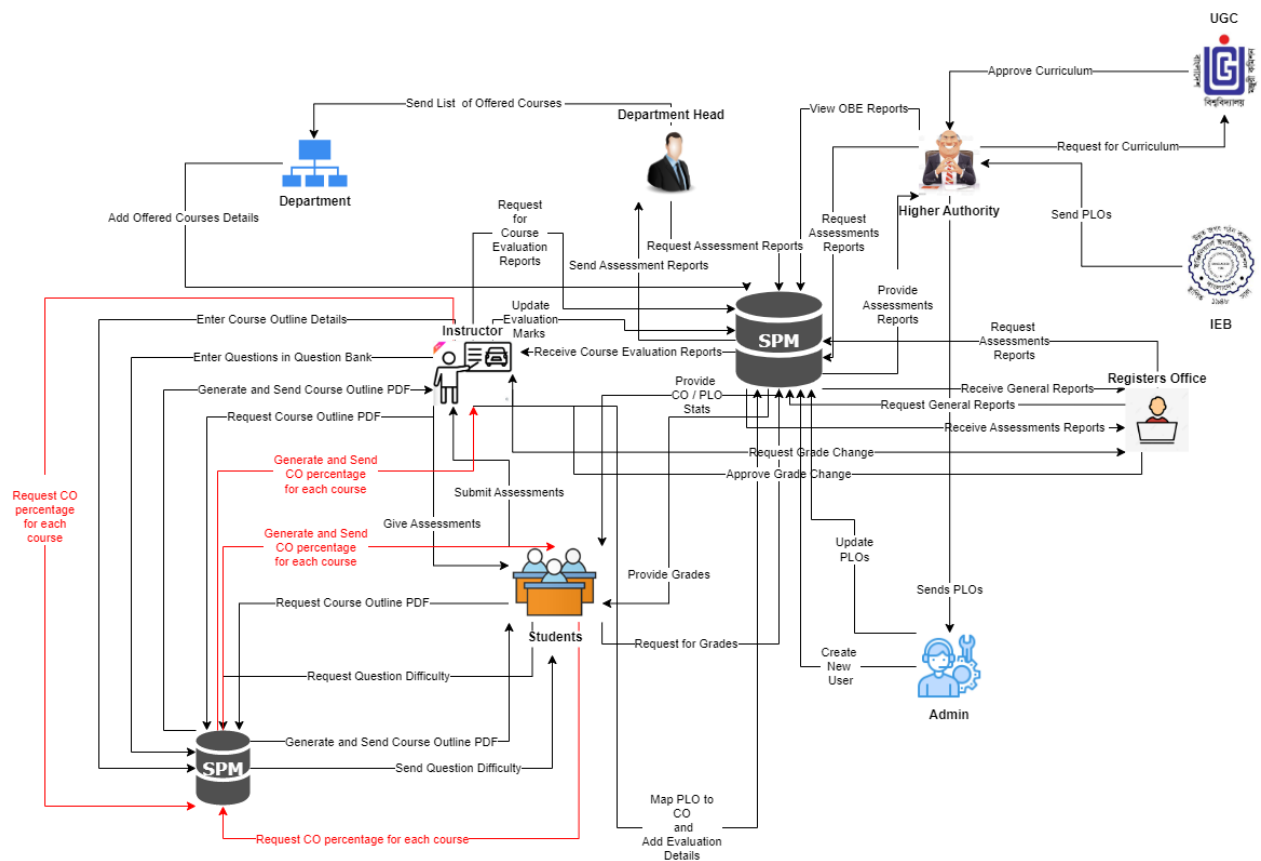
	<p>Year and Semester.</p> <p>Department Head</p> <p>1) Sign into the system using ID and Password.</p> <p>2) Select Student Enrollment Statistics tab and select Year and Semester under that tab</p> <p>3) View Student Enrollment Statistics Of That Year and Semester.</p>					
--	--	--	--	--	--	--

C. PROBLEM ANALYSIS – EXISTING SYSTEM (SPMS 2.0):

The problems in SPMS2.0 were analyzed, and the proposed solution are given in the following.

Process Name	Stake Holders	Concerns(Problems)	Analysis(Reason of The Problem)	Proposed Solution
CO Analysis by grade	Faculty: Student:	1) Cannot view CO earned per course by students according to their grades.	1) No input forum created that takes grades as input and displays a chart to analyze a student's CO achieved according to the grade received.	There is an input forum that takes course details, student details and grade into account and generates a spider chart to display the outcome.

D. RICH PICTURE – PROPOSED SYSTEM (SPMS 3.0):



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 3.0):

From the rich picture we can see that there are 2 key processes:

- 1) Calculate CO percentage per course as per grade.**
- 2) Generate spider chart according to percentage calculated.**

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.

CO achievement ent per course	Student: a) Logs into th esystem using Student -ID and password. b) Select sCO achievement ent per course c) View CO Achieve ment per course.		Compute r/Laptop a)User willneed a computer to acce ssSPMS Printer a)Used t o print out the repor tif need be. Networki ng Devices	SPMS a) A CO achievement t will be generated bythe software.	SPMS Database a) Here, the performanc e will be storedand updated.	Internet a) To login into and access the SPMS it is used.
--	--	--	--	---	--	--

	<p>Admin:</p> <p>a) Logs into the System using user-ID and password.</p> <p>b) Select s CO achievem e nt</p> <p>c) View CO Achievem ent per course.</p> <p>Faculty:</p> <p>a) Logs into the System using Faculty-ID and password.</p> <p>b) Select s</p>		<p>(Router, Switch, Bridge, Hub):</p> <p>a)Used to access the Internet.</p>			
--	--	--	---	--	--	--

	CO Achievem ent. c) View CO Achievem ent per course.					
--	---	--	--	--	--	--

CHAPTER-3 LOGICAL SYSTEM DESIGN

A. BUSINESS RULES – SPMS 3.0:

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 IPSID ,week ,topic ,learningStrategy , assessmentStrategy, correspondingClo, courseOutlineID.

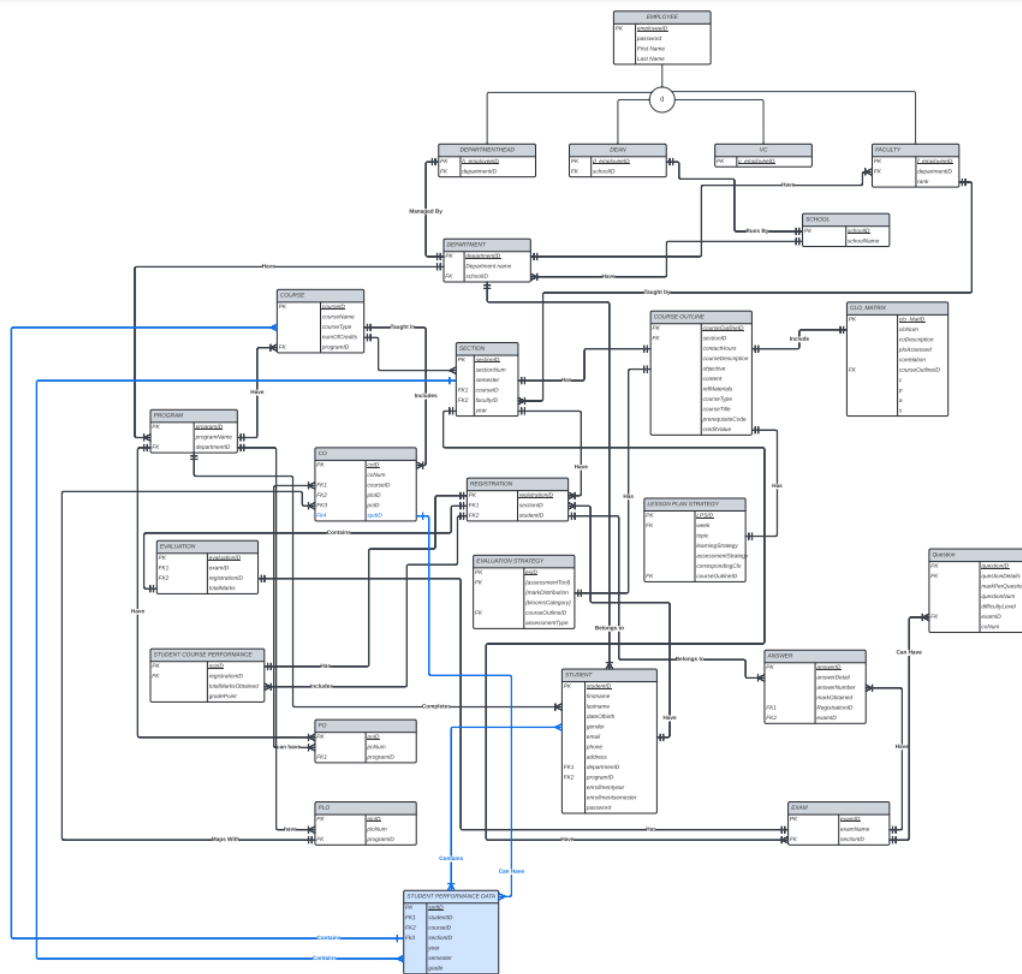
16. An exam has exactly one evaluation. An Evaluation for an exam is done exactly once. An exam belongs to exactly one section. An EXAM includes examID, examName, sectionID. A section must have one or many exams.

17. An exam must have one or many questions. Every question must belong to exactly one exam. A QUESTION includes questionID , questionDetails , marksPerQuestion , questionNum , difficultlyLevel , examID , coNum. A Question is answered exactly once. An answer has exactly one question.

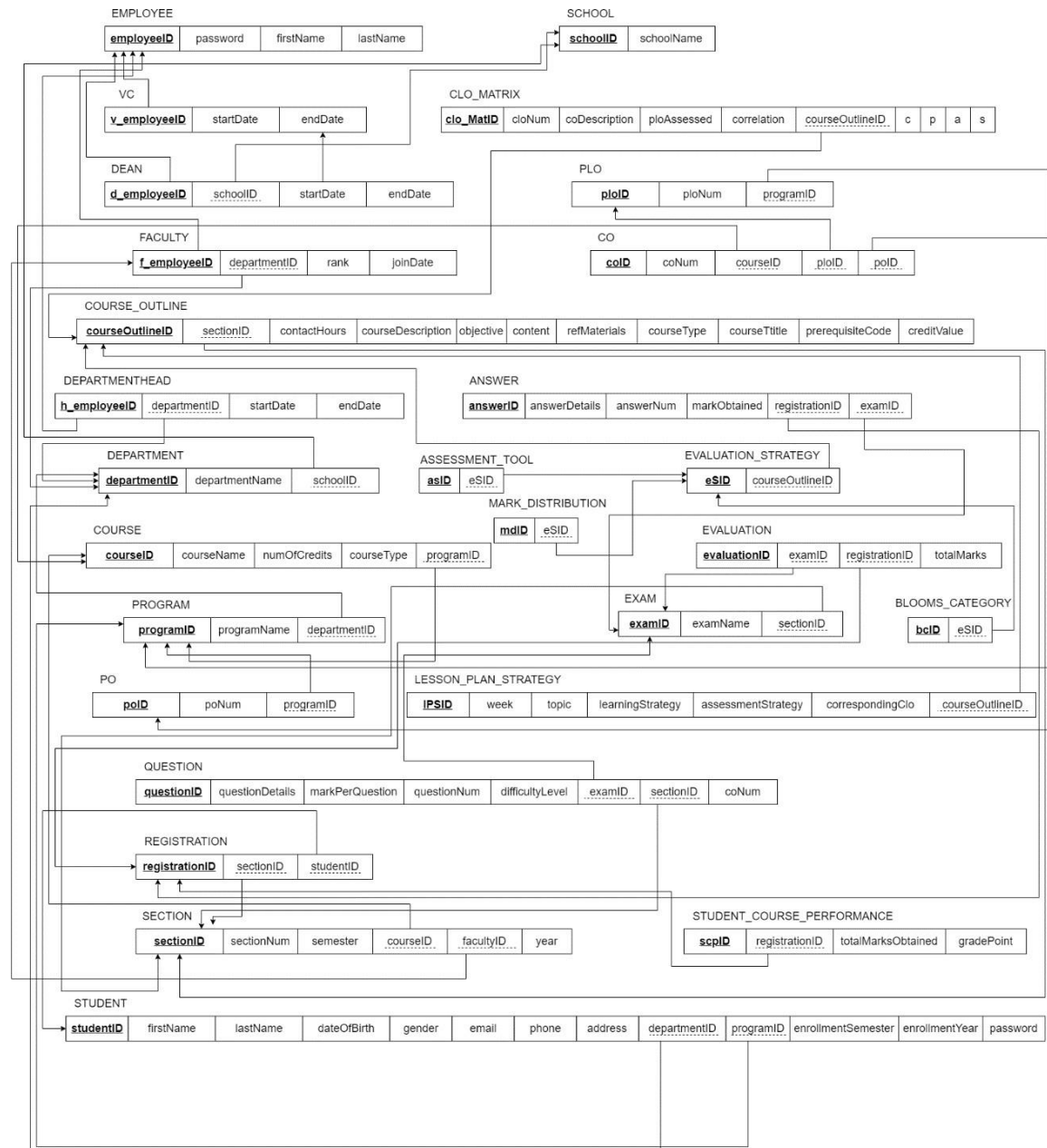
18) A PO belongs to exactly one program A program must have one or many PO.PO includes poID , poNum , programID. A PO must belong to one or many CO. A CO must have exactly one PO.

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.

B. ENTITY RELATIONSHIP DIAGRAM:



C. ENTITY RELATIONSHIP DIAGRAM TO RELATIONAL SCHEMA:



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
	prerequisiteCode	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
QUESTION (q)	questionID	q1
	questionDetails	q2
	markPerQuestion	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
SCHOOL (h)	enrollmentYear	s10
	password	s11
CLO_MATRIX (m)	schoolID	h1
	schoolName	h2
	clo_MatID	m1
	cloNum	m2
	coDescription	m3
	ploAssessed	m4
	correlation	m5
	courseOutlineID	c1
	c	m6
	p	m7
PLO (p)	a	m8
	s	m9
CO (o)	ploID	p1
	ploNum	p2
	programID	r1
ANSWER (a)	colD	o1
	coNum	o2
	courseID	u1
	ploID	p1
	polD	x1
	spdID	aa1
EVALUATION_STRATEGY (t)	answerID	a1
	answerDetails	a2
	answerNum	a3
	markObtained	a4
	registrationID	g1
	examID	e1
EVALUTION (n)	eSID	t1
	assessmentTool	t2
	markDistribution	t3
	bloomsCategory	t4
	courseOutlineID	c1
	evaluationID	n1
	examID	e1
	registrationID	g1
	totalMarks	n2

EXAM (e)	examID	e1
	examName	e2
	sectionID	y1
LESSON_PLAN_STRATEGY (l)	IPSID	l1
	week	l2
	topic	l3
	learningStrategy	l4
	assessmentStrategy	l5
	correspondingClo	l6
	courseOutlineID	c1
STUDENT_COURSE_PERFORMANCE (z)	scplD	z1
	registrationID	g1
	totalMarksObtained	z2
	gradePoint	z3

STUDENT PERFORMANCE DATA(aa)	spdID	aa1	
	studentID	s1	
	courseID	u1	
	sectionID	y1	
	year	aa2	
	semester	aa3	
	grade	aa4	

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: "21-12-1996".
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: "01XXXXXXXXXX".
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
scplID	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"

PLO_T

Name	Data Type	Size	Remark
ploID	INTEGER	11	This is the primary key for Program Learning Outcome. E.g: "PLO1"
ploNum	INTEGER	11	This is the PLO 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: "01-03-2024"

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: "01-03-2024"

--	--	--	--

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
prerequisiteCode	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_PERFORMANCE_DATA_T

Name	Data Type	Size	Remark
spdID	INTEGER	11	This is the primary key for this table
studentID	MEDIUMTEXT		This is the answer details
courseID	VARCHAR	6	This is the foreign key from the section table e.g: "CSC303"
sectionID	INTEGER	11	This is the foreign key from the section table e.g: "Section -1"
year	INT	11	This is the year e.g: "2021"
semester	INT	11	This is the enrolled semester e.g: "Autumn, Spring, Summer"
grade	INT	2	This the grade received in a course e.g: "A, B, C, etc"

CHAPTER-4 PHYSICAL SYSTEM DESIGN

A. INPUT FORMS:

```
<?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">
<div class="login-box">
    

    <form action="login.php" method="post">
    <div>
        <label style="">
            User Type
        </label>
        <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: 0%;">
        ID
    </label>
```

```
<div class="mainContainer">
<div class="login-box">
    

    <form action="login.php" method="post">
    <div>
        <label style="">
            User Type
        </label>
        <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:0%;">
        ID
    </label>
    <input class="ID" style="margin-left:5%;" type="text" name="ID" placeholder="Enter Your ID">
    <br>
    <label>
        Password
    </label>
    <input class="ID" style="margin-left:5%;" type="password" name="password" placeholder="Enter Your Password"><br>
    <input type="submit" name="submit" value="Login" class="submitButton">
    </form>
    </div>
</div>
</div>

</body>
</html>
```

[illegible][illegible]


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

if(isset($_POST['submit'])){

$level1=array("arrange","count","define","describe","draw",
"duplicate","identify","label","list","match","name",
"order","point","quote","read","recall","recite",
"recognize","record","repeat","reproduce","select",
"state","write");

$level2=array("associate","classify","compare","compute","contrast",
"convert","describe","differentiate","discuss","distinguish","explain",
"express","extend","generalize","give examples","identify","indicate",
"locate","listing","matching","paraphrase","predict",
"recognize","report","restate","review","rewrite","select","sort","summarize",
"tell","translate");

$level3=array("add","apply","calculate","choose","change","classify",
"compete","compute","demonstrate","determine","develop","discover","divide",
"dramatize","employ","examine","formulate","graph","illustrate","interpret","manipulate",
"modify","multiply","operate","organize","perform","practice","predict","prepare",
"produce","relate","schedule","sketch","shop","show","solve","subtract","translate","use");
```

```
$level3=array("add","apply","calculate","choose","change","classify",
"compete","compute","demonstrate","determine","develop","discover","divide",
"dramatize","employ","examine","formulate","graph","illustrate","interpret","manipulate",
"modify","multiply","operate","organize","perform","practice","predict","prepare",
"produce","relate","schedule","sketch","shop","show","solve","subtract","translate","use");

$level4=array("diagram","differentiate","discriminate","distinguish",
"examine","estimate","experiment","extrapolate","formulate","identify",
"illustrate","infer","inspect","inventory","outline","point out",
"question","relate","select",
"analyze","appraise","arrange","breakdown","calculate","combine",
"contrast","compare","criticize","design","detect","determine",
"develop","separate","subdivide","test","utilize");

$level5=array("appraise","argue","assess","attack","choose",
"compare","conclude","contrast","criticize","critique",
"defend","determine","estimate","evaluate","grade","interpret",
"judge","justify","measure","predict","rank","rate","revise",
"score","select","support","test","value","weigh");

$level6=array("arrange","assemble","categorize","collect",
"combine","compile","compose","construct","create","debate",
"derive","design","devise","explain","formulate","generate",
```

```
$level6=array("arrange","assemble","categorize","collect",
"combine","compile","compose","construct","create","debate",
"derive","design","devise","explain","formulate","generate",
"group","integrate","manage","modify","order","organize","plan",
"prepare","prescribe","produce","propose","rearrange","reconstruct",
"relate","reorganize","revise","rewrite","specify","summarize",
"synthesize","tell","transform");

$examName=$_POST['examName'];
$sectionNum=$_POST['sectionNum'];
$questionCount=$_POST['questionCount'];
$courseID=$_POST['courseID'];
$semester=$_POST['semester'];
$year=$_POST['year'];

//Getting section ID from database
$result=mysqli_query($con,"SELECT sec.sectionID AS sectionID
FROM section_t AS sec
WHERE sec.sectionNum='$sectionNum' AND sec.courseID='$courseID'
AND sec.semester='$semester' AND sec.year='$year'");
$row=mysqli_fetch_assoc($result);
$sectionID=$row['sectionID'];
```

```
//storing exam in database
$query="INSERT INTO `exam_t` (`examID`, `examName`, `sectionID`)
VALUES (NULL, '$examName', '$sectionID')";
$result=mysqli_query($con,$query);

//getting the exam ID from database
$result=mysqli_query($con,"SELECT MAX(examID) AS examID
FROM exam_t");
$row=mysqli_fetch_assoc($result);
$examID=$row['examID'];

//Getting course ID from database
$result=mysqli_query($con,"SELECT sec.courseID AS courseID
FROM section_t AS sec
WHERE sec.sectionID='$sectionID'");
$row=mysqli_fetch_assoc($result);
$courseID=$row['courseID'];

//Storing questions in database
for($i=1;$i<=$questionCount;$i++){

    $difficultyLevel=0;
```

```

    $difficultyLevel=0;

    $questionNum=$_POST["questionNum".$i];
    $questionDetails=$_POST["questionDetails".$i];
    $mark=$_POST["mark".$i];
    $coNum=$_POST["coNum".$i];

    if($difficultyLevel<=0){
        for($k=0;$k<sizeof($level1);$k++){
            if(strpos($questionDetails,$level1[$k])!==false){
                $difficultyLevel=1;
                break;
            }
        }
    }
    if($difficultyLevel<=0){
        for($k=0;$k<sizeof($level2);$k++){
            if(strpos($questionDetails,$level2[$k])!==false){
                $difficultyLevel=2;
                break;
            }
        }
    }
}
```

```

    }

    if($difficultyLevel<=0){
        for($k=0;$k<sizeof($level3);$k++){
            if(strpos($questionDetails,$level3[$k])!==false){
                $difficultyLevel=3;
                break;}
            }
        }
    if($difficultyLevel<=0){
        for($k=0;$k<sizeof($level4);$k++){
            if(strpos($questionDetails,$level4[$k])!==false){
                $difficultyLevel=4;
                break;}
            }
        }
    if($difficultyLevel<=0){
        for($k=0;$k<sizeof($level5);$k++){
            if(strpos($questionDetails,$level5[$k])!==false){
                $difficultyLevel=5;
                break;
            }
        }
    }
}
```

```

        break;
    }
}
}
if($difficultyLevel<=0){
    for($k=0;$k<sizeof($level6);$k++){
        if(strpos($questionDetails,$level6[$k])!==false){
            $difficultyLevel=6;
            break;
        }
    }
}

$query="insert into question_t (questionID,questionDetails,markPerQuestion,
questionNum,difficultyLevel,examID,courseID,coNum)
values('','$questionDetails','$mark','$questionNum',
'$difficultyLevel','$examID','$courseID','$coNum')";

$result=mysqli_query($con,$query);
}
header('location:addExam.php');
}
?>

```

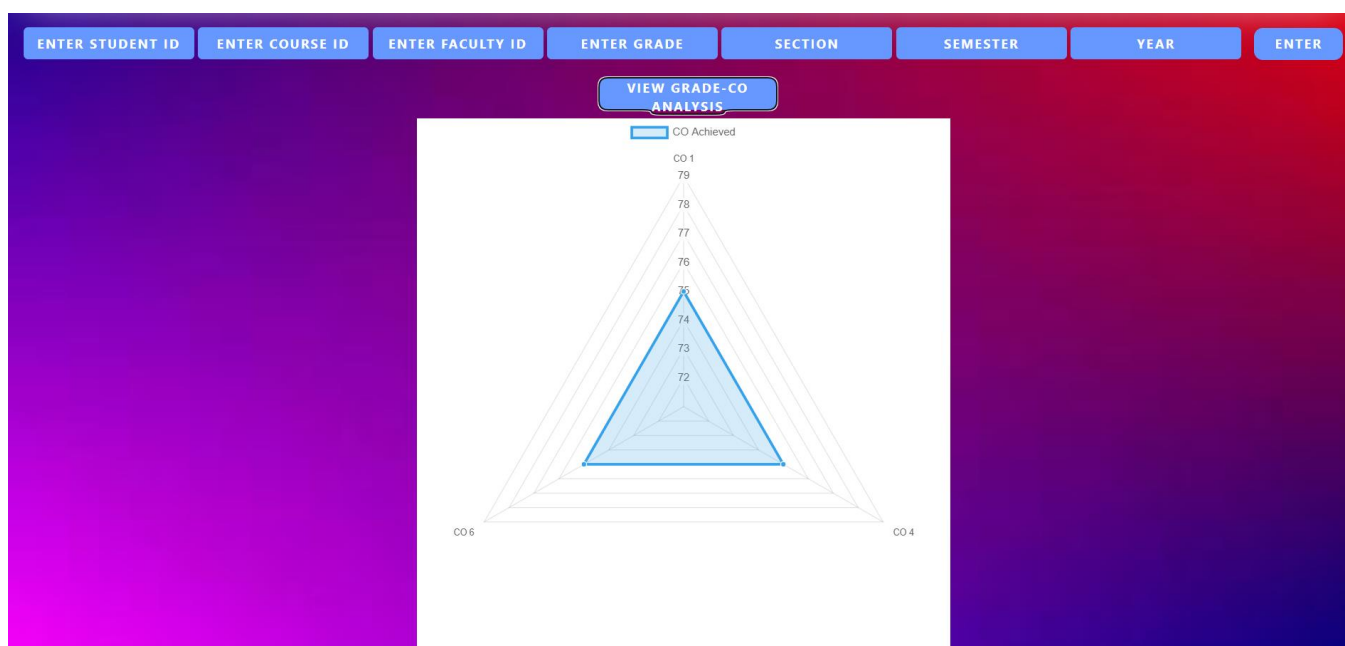
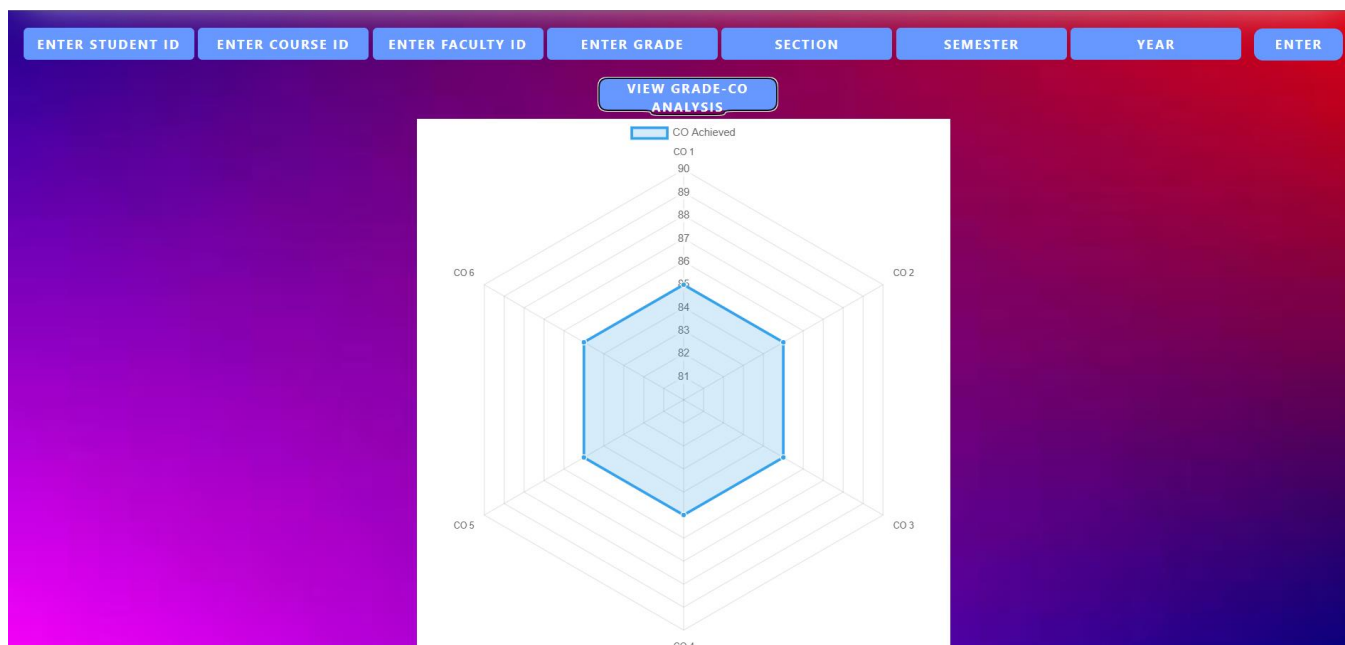
A. OUTPUT FORMS:

Dashboard Logout

CO grade Chart CO Analysis Input CO Analysis Import CSV

ENTER STUDENT ID ENTER COURSE ID ENTER FACULTY ID ENTER GRADE SECTION SEMESTER YEAR ENTER

VIEW GRADE-CO ANALYSIS



```

CO_grade Chart.txt - Notepad
File Edit Format View Help
</div> <!-- div row-3 ends here -->

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

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

<script>

function coview(){
<?php

$sql="SELECT g.mark, co.coNum,
AVG((g.mark/100)*100) AS percent
FROM registration_t AS r, answer_t AS ans, question_t AS q,
co_t AS co, po_t AS po, student_course_performance_t AS stp, grade_t AS g
WHERE r.registrationID=ans.registrationID
AND r.registrationID=stp.registrationID
AND co.courseID='$courseID'
AND g.grade='$grade'
GROUP BY co.coNum";

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

$co=array();
$percent=array();

while($data=mysqli_fetch_array($result)){

    array_push($co,"CO ". $data['coNum']);
    array_push($percent,$data['percent']);

}

?>

```

```

CO_grade Chart.txt - Notepad
File Edit Format View Help
var percent=<?php echo json_encode($percent); ?>;

for(var i=0;i<percent.length;i++){
    percent[i]=parseFloat(percent[i]);
}

const ctx = document.getElementById('myChart');

new Chart(ctx, {
    type: 'radar',
    data: {
        labels: co,
        datasets: [{
            label: 'CO Achieved',
            data: percent,
            fill: true,
            backgroundColor: 'rgba(54, 162, 235, 0.2)',
            borderColor: 'rgb(54, 162, 235)',
            pointBackgroundColor: 'rgb(54, 162, 235)',
            pointBorderColor: '#fff',
            pointHoverBackgroundColor: '#fff',
            pointHoverBorderColor: 'rgb(54, 162, 235)'}]
    },
    options: {
        elements: {
            line: {
                borderWidth: 3
            }
        }
    }
});

</script>

</body>

</html>

```

Dashboard
Logout

CO_grade Chart
CO Analysis Input
CO Analysis Import CSV

1910876
MKT101
4546
A
2
AUTMUN
2019
ENTER

```

smth.txt - Notepad
File Edit Format View Help
<?php

$con = mysqli_connect('localhost', 'root');

if($con){
    echo "Connection successful";
}
else{
    echo "No connection";
}

mysqli_select_db($con, 'spms');

$studentID = $_POST['studentID'];
$courseID = $_POST['courseID'];
$year = $_POST['year'];
$semester = $_POST['semester'];
$sectionNum = $_POST['sectionNum'];
$grade = $_POST['grade'];
$f_employeeID = $_POST['f_employeeID'];

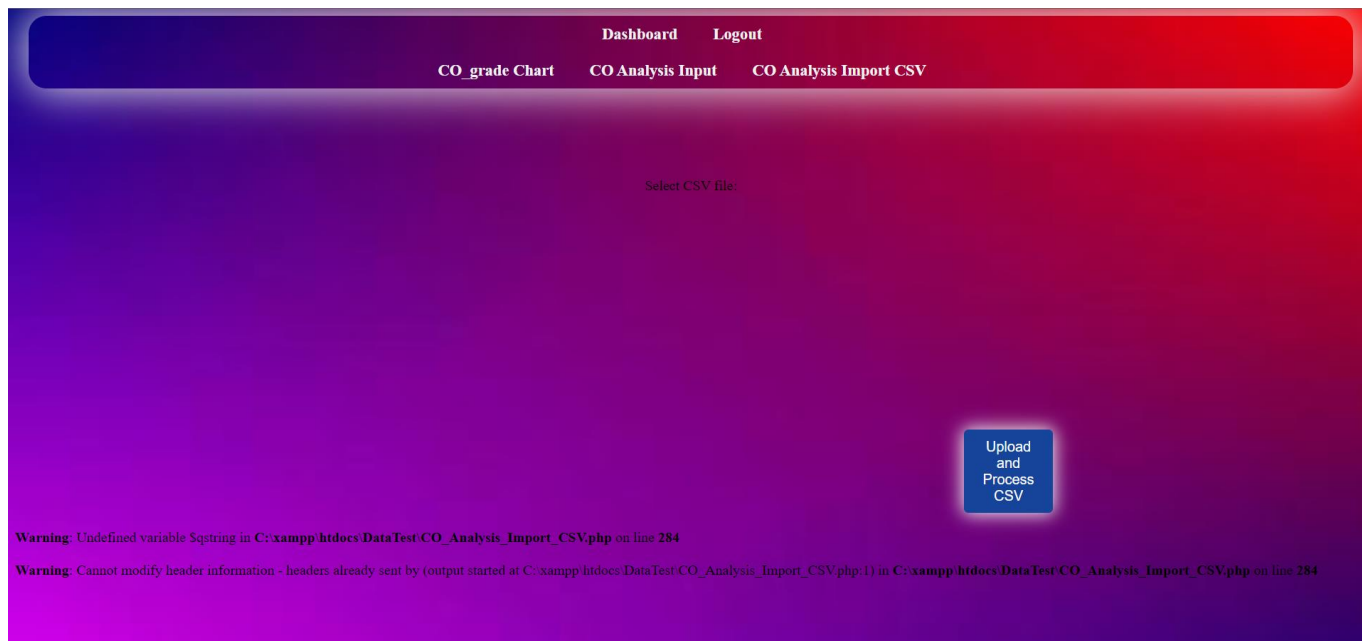
session_start();

//inserting into table
$query0 = "insert into grade_co_t (studentID, courseID, grade, sectionNum, semester, year, f_employeeID) values ('$studentID', '$courseID', '$grade', '$sectionNum', '$semester', '$year', '$f_employeeID')";

mysqli_query($con, $query0);

header('location:CO_Analysis_Input.php');
?>

```

```

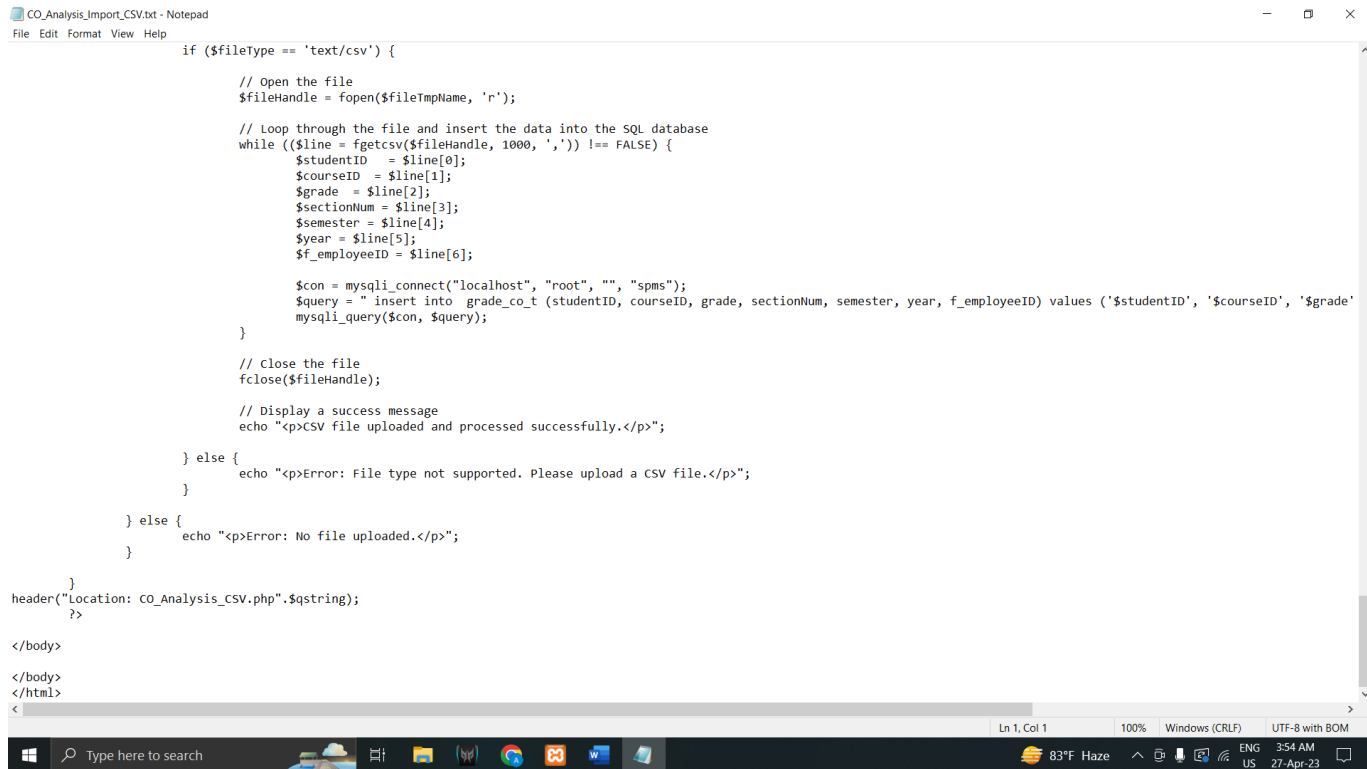
CO_Analysis_Import_CSV.txt - Notepad
File Edit Format View Help
<nav class="bottom-nav">
  <ul>
    <li><a href="CO_grade_Chart.php">CO_grade Chart</a></li>
    <li><a href="CO_Analysis_Input.php">CO Analysis Input</a></li>
    <li><a href="CO_Analysis_Import_CSV.php">CO Analysis Import CSV</a></li>
  </ul>
</nav>
</div>
<body>
  <form method="post" enctype="multipart/form-data">
    <label for="csvfile">Select CSV file:</label>
    <input type="file" id="csvfile" name="csvfile" accept=".csv"><br><br>
    <button type="submit" name="submit">Upload and Process CSV</button>
  </form>

  <?php
  // Check if the form was submitted
  if (isset($_POST['submit'])) {
    // Check if a file was uploaded
    if (isset($_FILES['csvfile']) && $_FILES['csvfile']['error'] == 0) {
      // Get the file information
      $fileName = $_FILES['csvfile']['name'];
      $fileTmpName = $_FILES['csvfile']['tmp_name'];
      $fileSize = $_FILES['csvfile']['size'];
      $fileType = $_FILES['csvfile']['type'];

      // Check if the file is a csv file
      if ($fileType == 'text/csv') {
        // open the file
        $fileHandle = fopen($fileTmpName, 'r');

        // Loop through the file and insert the data into the SQL database
        while (($line = fgetcsv($fileHandle, 1000, ',')) != FALSE) {
          $studentID = $line[0];
          $courseID = $line[1];
          $grade = $line[2];
          $sectionNum = $line[3];
          $semester = $line[4];
        }
      }
    }
  }

```



```
CO_Analysis_Import_CSV.txt - Notepad
File Edit Format View Help

if ($fileType == 'text/csv') {

    // Open the file
    $fileHandle = fopen($fileTmpName, 'r');

    // Loop through the file and insert the data into the SQL database
    while (($line = fgetcsv($fileHandle, 1000, ',')) !== FALSE) {
        $studentID = $line[0];
        $courseID = $line[1];
        $grade = $line[2];
        $sectionNum = $line[3];
        $semester = $line[4];
        $year = $line[5];
        $f_employeeID = $line[6];

        $con = mysqli_connect("localhost", "root", "", "spms");
        $query = "insert into grade_co_t (studentID, courseID, grade, sectionNum, semester, year, f_employeeID) values ('$studentID', '$courseID', '$grade'";
        mysqli_query($con, $query);
    }

    // Close the file
    fclose($fileHandle);

    // Display a success message
    echo "<p>CSV file uploaded and processed successfully.</p>";

} else {
    echo "<p>Error: File type not supported. Please upload a CSV file.</p>";
}

} else {
    echo "<p>Error: No file uploaded.</p>";
}

}

header("Location: CO_Analysis_CSV.php".$qstring);
?>

</body>

</body>
</html>
```

Ln 1, Col 1 100% Windows (CRLF) UTF-8 with BOM

Type here to search 83°F Haze 3:54 AM 27-Apr-23

CHAPTER-5 CONCLUSION

A.PROBLEM AND SOLUTION:

Analysis Phase:

During the Analysis Phase, one of the major problems faced was the confusion around the Rich Picture and Six Element Analysis of the organizational operations since there was no data available regarding those operations. However, Faculty members and other stakeholders were interviewed in order to overcome such confusions, and information received during the interview was collected in order to get a better understanding of the system that was being developed.

Designing Phase:

Some problems were faced while creating the EERD during the Design Phase, However, constant feedback from the faculty were enough to overcome those issues.

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:

One new feature could be added to this system in the near future which can monitor a student's club and extracurricular activities and then provide reports and analytics based on the student's extracurricular activity performance.

References

- [1] <http://www.iub.edu.bd/AboutIUB/ata glance>. [Online].
- [2] <http://www.cse.iub.edu.bd/degrees/1>. [Online].
- [3] t. <https://mspguide.org/2022/03/18/rich-picture/#:~:text=What%20is%20a%20Rich%20Picture>. [Online].
- [4] b. <https://www.trisotech.com/bpmn-introduction-and-history/#:~:text=BPMN%20was%20originally%20developed%20by>. [Online].

