

**Outcome Based Education System**

**CSE303**

**Database Management System**

**FINAL REPORT**

**Group 5**

Abdullah Amin Sadat (1721439)  
Golam Rabbanie Tanhid (1710475)  
Zawad Newaz (1720155)  
Mahabub Hasan Rafi (1830235)   
MD Abdullah Al Muhit (1830479)

**Table of Contents**

Contents

[**Chapter 1** 3](#_Toc72622184)

[Background of the project 4](#_Toc72622185)

[Objective of the project 5](#_Toc72622186)

[Scope of the project 6](#_Toc72622187)

[**Chapter 2** 7](#_Toc72622188)

[Rich Picture (AS-IS) 8](#_Toc72622189)

[Six Element Analysis (AS-IS) 9](#_Toc72622190)

[Process Diagram (AS-IS) 15](#_Toc72622191)

[Problem Analysis 18](#_Toc72622192)

[Rich Picture (TO-BE) 19](#_Toc72622193)

[Six Element Analysis (TO-BE) 20](#_Toc72622194)

[Process Diagram (TO-BE) 27](#_Toc72622195)

[**Chapter 3** 30](#_Toc72622196)

[Business Rule 31](#_Toc72622197)

[ERD 32](#_Toc72622198)

[Relational Schema 33](#_Toc72622199)

[Normalization 34](#_Toc72622200)

[Data Dictionary 37](#_Toc72622201)

[**Chapter 4** 42](#_Toc72622202)

[**Chapter 5** 59](#_Toc72622203)

[Problems & Solutions 60](#_Toc72622204)

[Additional Features & Future Development 60](#_Toc72622205)

[Conclusion & Recommendation 60](#_Toc72622206)

# Chapter 1

INTRODUCTION

1. Background of the project
2. Objective of the project
3. Scope of the project

# Background of the project

The Student Performance Monitoring System focuses on performance monitoring of student’s continuous assessment (tests) and examination scores in order to predict their final achievement status upon graduation. The main idea is to evaluate the COs achieved and mapped PLOs achieved by each student in each of the enrolled courses as that would be necessary for monitoring the student performance.

Our project aims to design, build and deliver software that we believe will help universities promote a more productive and effective way of evaluating students. At the very core of our project, we have introduced the idea of Course Outcomes (COs) and Program Learning Outcomes (PLOs), Each CO is mapped to a PLO. Each PLO represents a specific valuable skill that the students are expected to gain or enhance at the end of that course, such as problem analysis, design, implementation of a craft, etc.

To evaluate the students efficiently, the project intends to check whether the PLOs mapped to the COs requirement are fulfilled or not for each student. The faculties then input the COs for each of their students so that the system can map the COs to PLO accordingly.

In addition, our software hopes to benefit the institutional bodies and faculty members, administrative bodies, and departmental bodies to track students' progress and departmental performance and help them distribute and allocate resources better.

# Objective of the project

* Our project intends to create an interactive, user-friendly software that will act as a platform for students, faculties and other members of the university to help improve the quality of education and revolutionize the way we integrate technology into our education.
* One of the goals of this Project is to provide insight about how learning might improve in a given program-whether it be online, in a classroom, or happening in another context.
* To provide insight into what students are actually learning in relation to the big ideas of the courses and the program they aim to complete.
* To automate the process of monitoring student performance so as to reduce the manual processing involved in it.
* To analyse how student populations are learning inside of their programs so that the departments can focus more strategically on equity and success.

# Scope of the project

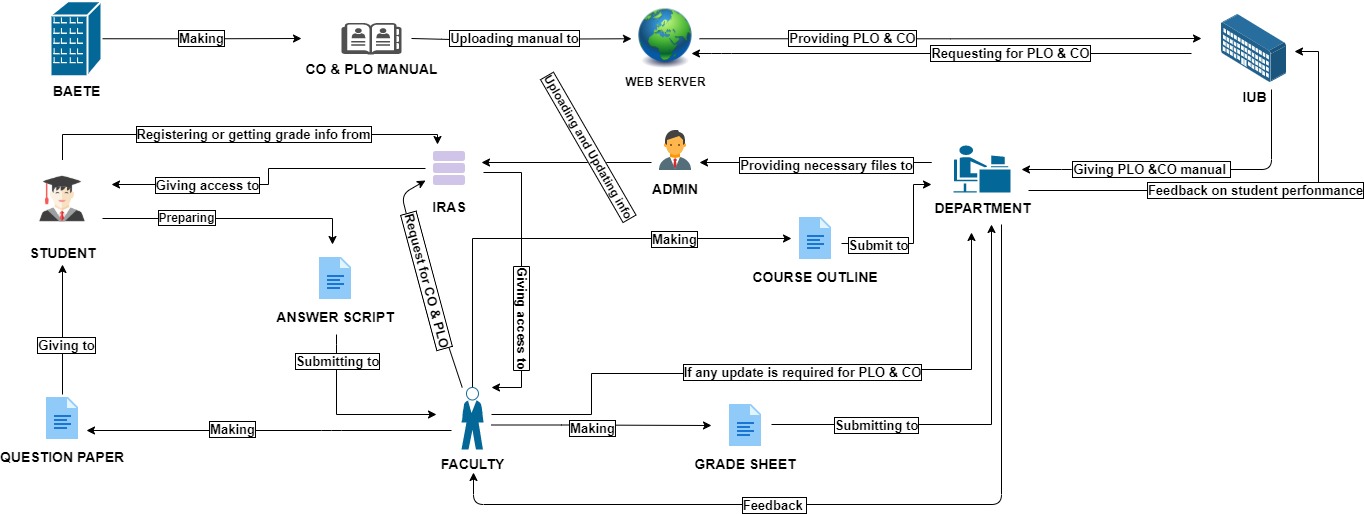
Scope of the project is a necessity to ensure the success of a project. As we are changing an existing system, we have to ensure that the proposed system will be more effective than the existing one. The proposed system would include evaluating the COs achieved, mapping the COs with the PLOs achieved and storing them as records, all of these were done manually in the existing system. The records can also be used to generate reports for analysis purpose. The system can be accessed by the instructors, students, BAETE and Higher Management (VC, Deans, Heads). It is very inefficient to maintain detailed records of student’s performance, and therefore there is a need of an improved and automated student performance monitoring system. Primarily we focused on IUB as the organization for which we are doing this project but the project has the potential of being useful to other universities as well, and future prospects seems likely to also cater to all universities that conform to OBE regulations.

# Chapter 2

System Analysis

1. RICH PICTURE AS-IS
2. SIX ELEMENTS AS IS
3. PROCESS DIAGRAM AS-IS
4. PROBLEM ANALYSIS
5. RICH PICTURE TO-BE
6. SIX ELEMENTS TO-BE
7. PROCESS DIAGRAM TO BE

# Rich Picture (AS-IS)



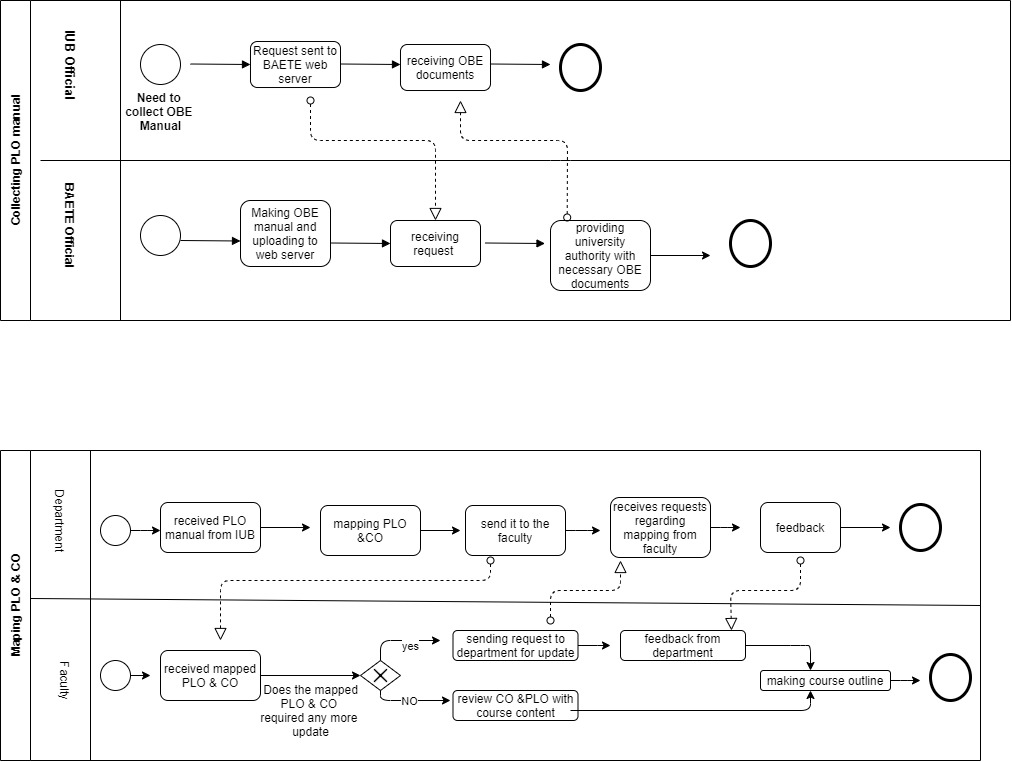
# Six Element Analysis (AS-IS)

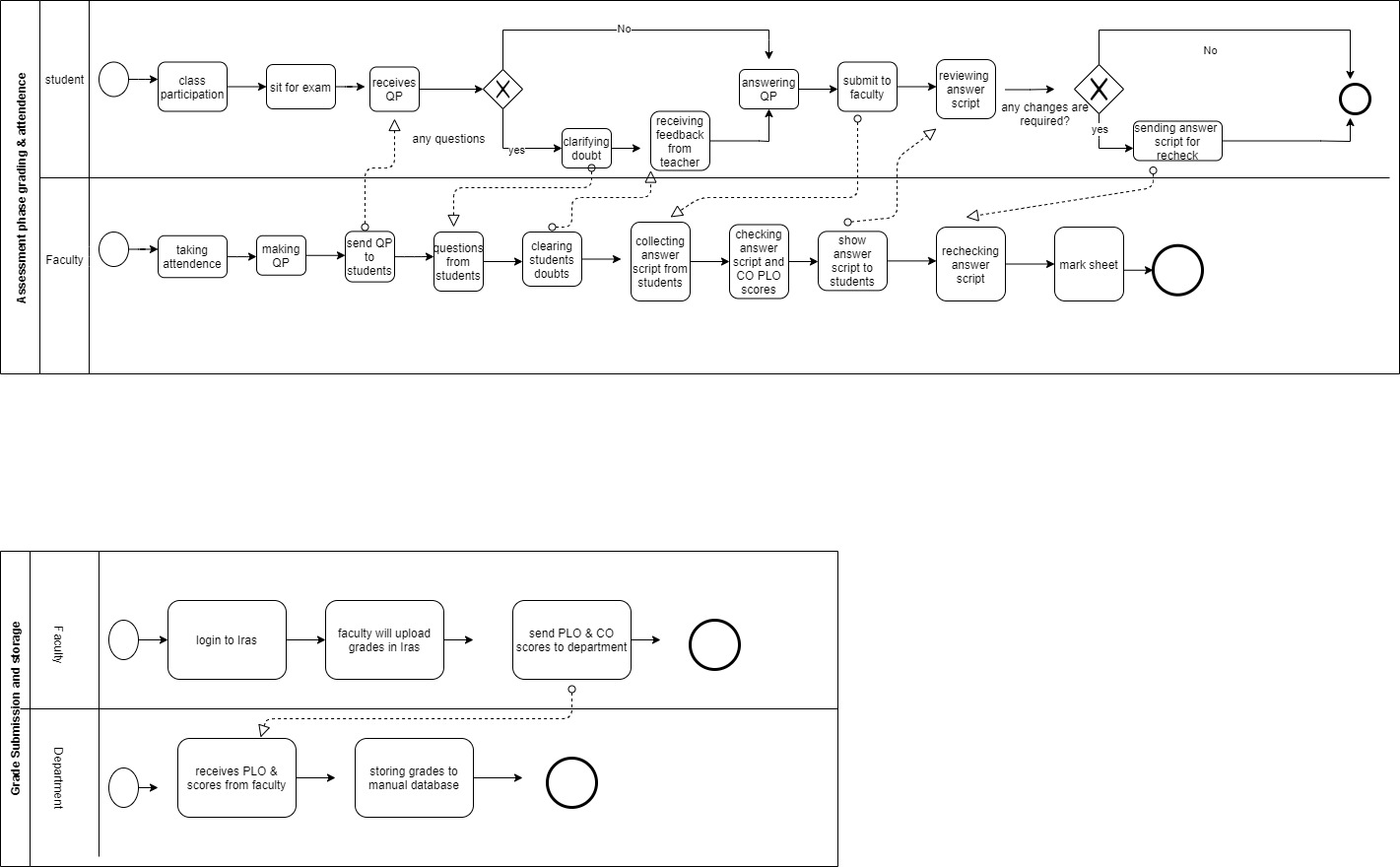
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **1) Checking**  **CO/PLO**  **Manual or OBE document** | **A) IUB officials:**  1. Specific IUB department sends request to BAETE to get info on OBE documents.  2. IUB can receives guidelines and feedbacks regarding courses and departments.  **B) BAETE officials:**  1. BAETE officials can communicate.  2. BAETE officials make the necessary documents on OBE and provide it to university authority  3. BAETE officials can provide feedbacks and guidelines that should be followed by the departments. | **A) Pen, papers:**  All documents are managed by papers.  Pen for taking notes and writing purposes. | **A) Digital devices:**  Such as computers, laptops, phone may be used to store soft copies for the guidelines.  Many other digital devices are used for communication between internal/external parties. | **A) IRAS:**  To access necessary information for different individuals such as faculties, students etc. | **A) Manual database:**  To store all sort of PLO scores which are stored in documents. | **A) WAN, LAN WIFI, BROADBAND:**  Any other communication platform to organize meetings between parties. |

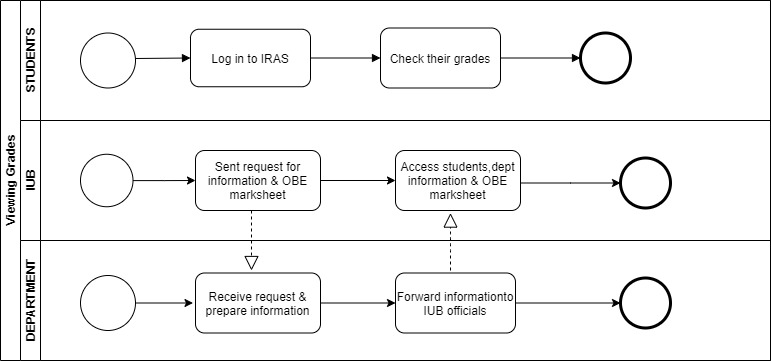
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **2)Making course outline and assessment plan** | **A) Department:**  1. After receiving the manual from BAETE department will map CO and PLO for each course and then provide the mapped guideline to faculty.  2. Department may receive an update from faculties if there any change need to do with CO and PLO mapping  **B) Faculty:**  1. Faculties can plan how each of the CO can be used for questions and assessments tests.  2. He/she will check the mapping of CO and PLO.  3. If any change is necessary about the mapping faculty will contact the department.  3. And finally the course outline is created. | **A) Pen, paper:**  Pen, Paper for taking notes and writing purposes. | **A) Computer, Laptop:**  These are used to make or edit softcopies of course outline.  It is also used for storing softcopies of the course outline and curriculum. | **A) MS Word:**  Used for writing soft copies and making changes to them. We can also save softcopies in text format |  | **A) WAN, LAN WIFI, BROADBAND:**  These are the communication platforms to transfer softcopies. |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **3) Assessment phase and grading and attendance** | **A) Students:**  1. Initially they participate in a class and sit for certain test on certain date.  2. They may ask to clarify if they have any questions.  3. Once they are done with the exam, they submit their answer script.  4. After checking assessed answer script, student can give a feedback to the faculty if there is any correction needed.  **B) Faculties:**  1. They take attendance, prepare assessment papers (take tests online in any emergency situation).  2. They also invigilate or monitor the test given by the students.  3. They often need to clarify the questions asked by the students.  4. They collect the scripts.  5. Each question is connected to a certain CO.  6. By calculating the CO, PLO scores are also calculated.  7. Show the assessed script to the student.  8. Receive feedback from students about their grades. If any correction is about grades required faculty will do that.  9. Thus the faculties calculate the final grade for the students. | **A) Pen, paper, rubber, pencil:**  1. Paper is supplied to students when the exams are taken in offline.  2. It is also used as an answer sheet.  3. Students need pen, pencil, eraser, ruler to complete the answer script. | **A) Computers, Laptops:**  1. Used to conduct particular test in courses related to computer science.  2. They are used to conduct online classes as well | **A) MS Word, GOOGLE DOC,**  **MS Excel,**  **Google classroom,**  **Google meet:**  1. Google classroom is used for taking classes and conducting exams.  2. Google meet is used for monitoring and gather information about the participants of the exam.  3. MS Word or Google DOCS are used for writing answers.  4. Students submit their answers through the panels in google classroom.  5. MS Excel is used for storing each student marks and final grades.  6. It is also used to evaluate PLO and CO score to store them. | **A) Manual database, CITS:**  1. It is used to keep records of the participants of the test.  2. Used for online classes in case of a crisis situation. | **A) WAN,LAN WIFI,BROADBAND:**  1. Used to allow online examinations to be conducted. |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **4) Grade submission and storage** | **A) Faculty:**  1. He/she will upload the grade of the students to the IRAS after log in to IRAS.  2. They also need to submit PLO and CO scores to the department.  3. The admin then avails a form to the faculty and available the changing option.  **B) Department:**  1. They receive OBE mark sheet from the faculties.  2. They then store the hard and soft copies. | **A) Paper:**  Hard copies of the grade submission and PLO and CO scores. | **A) Computer, Laptops:**  1. Use to store soft copies of the score grade in Excel sheets.  2. They are also use to access IRAS. | **A) MS EXCEL:**  It is used to store grades, PLO and CO scores as an Excel Sheet. | **A) Manual database, IRAS:**  1. It is used for faculty to keep records of the students and their grades.  2. IRAS is used for grade submission.  3. It is also used for viewing student’s credit, CGPA and all other information. | **A) WAN, LAN WIFI,BROADBAND:**  Network is used for transferring excel files. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **5) Viewing grades** | **A) Students:**  They can login into IRAS to check their grades.  **B) Department:**  Department can provide OBE mark sheet upon request of IUB officials.  **C) IUB officials:**  IUB officials access the PLO and Co scores through the department. | **A) Paper:**  Hard copies of the grade submission and PLO and CO scores. | **A) Computers, Laptops, Phone:**  They are used for accessing IRAS.  They are used for storing PLO and CO scores, grades of the students. | **A) Browser (Chrome, Mozilla, Opera):**  To access the website of the IRAS. | **A) IRAS:**  IRAS is used by the admin, students and other officials with the id and Password. | **A) WAN, LAN WIFI,BROADBAND:**  Any network to access IRAS from a digital device such as computer, laptop, mobile. |

# Process Diagram (AS-IS)



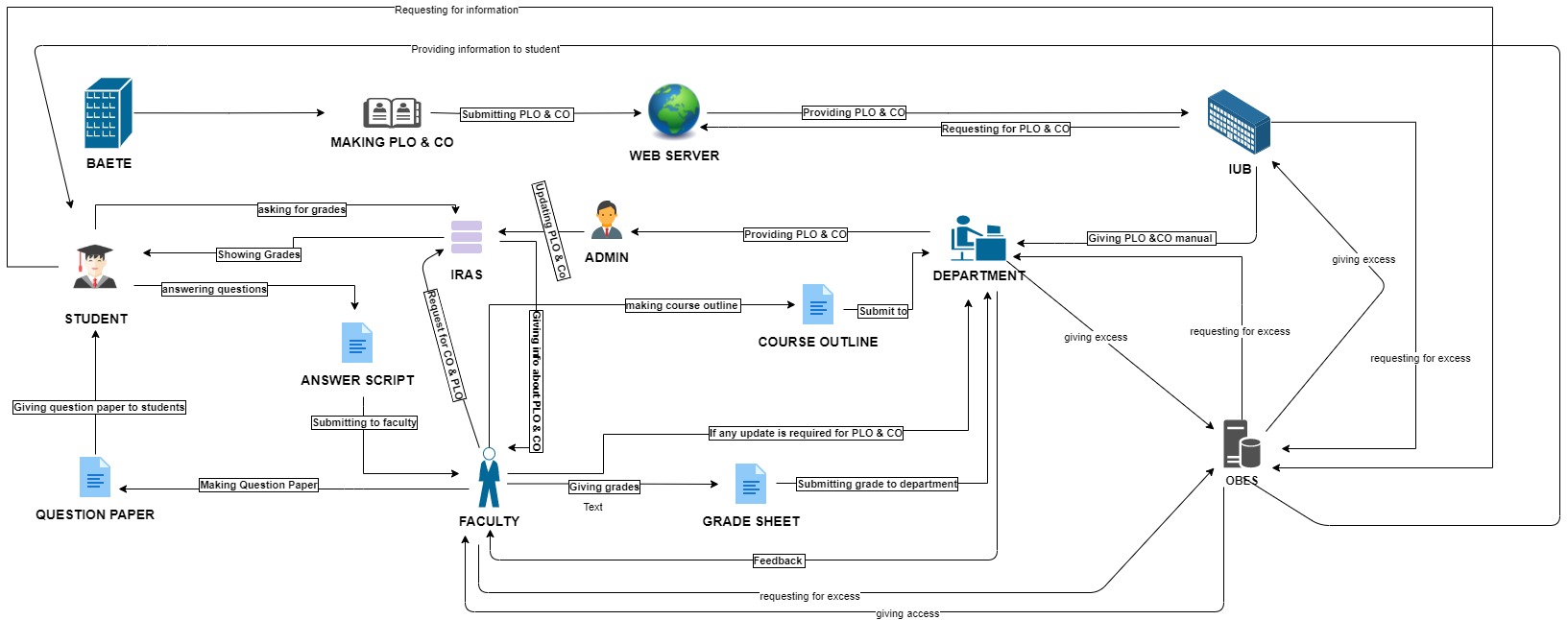




# Problem Analysis

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| PROCESS | STAKEHOLDERS | CONCERNS | ANALYSIS | SOLUTION |
| **CHECKING**  **CO/PLO**  **MANUAL OR OBE DOCUMENTS** | 1. IUB Officials  2. BAETE Officials | In case of PLO checking, IUB officials may have to dig very deep into documents, which can be difficult and time consuming.  Each faculty have to manually prepare the questions based on course outline. | Lacking of a system, where every single data can be stored automatically.  They also don’t have any proper data Management system / procedure. | A system will track everything individually, for an instance the system will keep track of PLO scores of each student as well as overall performance of each course and all of the courses of the department the IUB officials, department, faculty, students can have their own personal account in the system, so that can log in anytime to track the overall PLO performance of a department or even for an individual student if needed. |
| **MAKING COURSE OUTLINE AND ASSESSMENT PLAN** | 1. Department  2. Faculty | Department heads/experts have to provide PLO & CO guideline to the faculties manually. Faculties need to wait longer in order to make their course outline, because of the whole PLO & CO procedure. The specific departments have to wait for BAETE officials to provide them the OBE manual. | They don’t have direct system where the department faculty members can directly discuss with the BAETE officials. They have to do the plan and store OBE mapped documents into a manual database. | If a system is build where CO/PLO mapping can be stored both department and faculties can keep track of the necessities to make assessment plan for every semester. |
| **ASSESSMENT PHASE AND GRADING AND ATTENDANCE** | 1.Faculties  2.Students | Calculating PLO & CO scores manually can be difficult and time consuming. Errors might also occur calculating it manually. | They do have a lack of flexibility, in terms of preparing questions for exam. | The faculties can take coordinate exams, so that they can make the questions with ease. |
| **GRADE SUBMISSION AND STORAGE** | 1.Faculties  2.Department | Faculty members has to send a request to the department to review previous record of a student, and even after requesting they have to manually search all the previous records of that student. This is not an efficient process at all. Faculty has to request the department in terms of any changes, which can be time consuming.  Department has to store every grading documents manually which is time consuming. | There are lots of calculation & data entrance happening here, and not having an automatic system; it’s very obvious for human error. Having no specific arrangement, where all of the previous and present records can be stored. | The calculation & data entering can be done automatically in the system. They can use the web server to keep the record of each & every student, rather than searching it manually. Faculties should have the direct authority in order to make any changes in final grade. |
| **VIEWING GRADES** | 1. Students  2. Department  3. IUB Officials | Student can’t get any info about their CO/PLO scores.  It is difficult and time consuming for a department to arrange all the reports on CO/PLO and hand it over to higher authority. | There are many reports that a department is making manually.  They don’t have any system which can produce these reports. | A system can be build where students can check their OBE performance.  The system will produce necessary reports according to the given command. |

# Rich Picture (TO-BE)



# Six Element Analysis (TO-BE)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **1) Checking**  **CO/PLO**  **Manual or OBE documents** | **A) IUB officials:**  1. Specific IUB department sends request to BAETE to get info on OBE documents.  2. They collect the OBE documents from the BAETE web server.  3. IUB can receives guidelines and feedbacks regarding courses and departments.  **B) BAETE officials:**  1. BAETE officials can communicate.  2. BAETE officials make the necessary documents on OBE and provide it to university authority  3. BAETE officials can provide feedbacks and guidelines that should be followed by the departments. | **A) Pen, papers:**  All documents are managed by papers.  Pen for taking notes and writing purposes. | **A) Digital devices:**  Such as computers, laptops, phone may be used to store soft copies for the guidelines.  Many other digital devices are used for communication between internal/external parties. | **A) IRAS, OBE System:**  To access necessary information for different individuals such as faculties, students etc. | **A) Manual database, OBE System:**  Manual databases to keep information about students.  SPE keeps track of all PLO & CO Reports. | **A) WAN, LAN WIFI, BROADBAND:**  Any other communication platform to organize meetings between parties. |

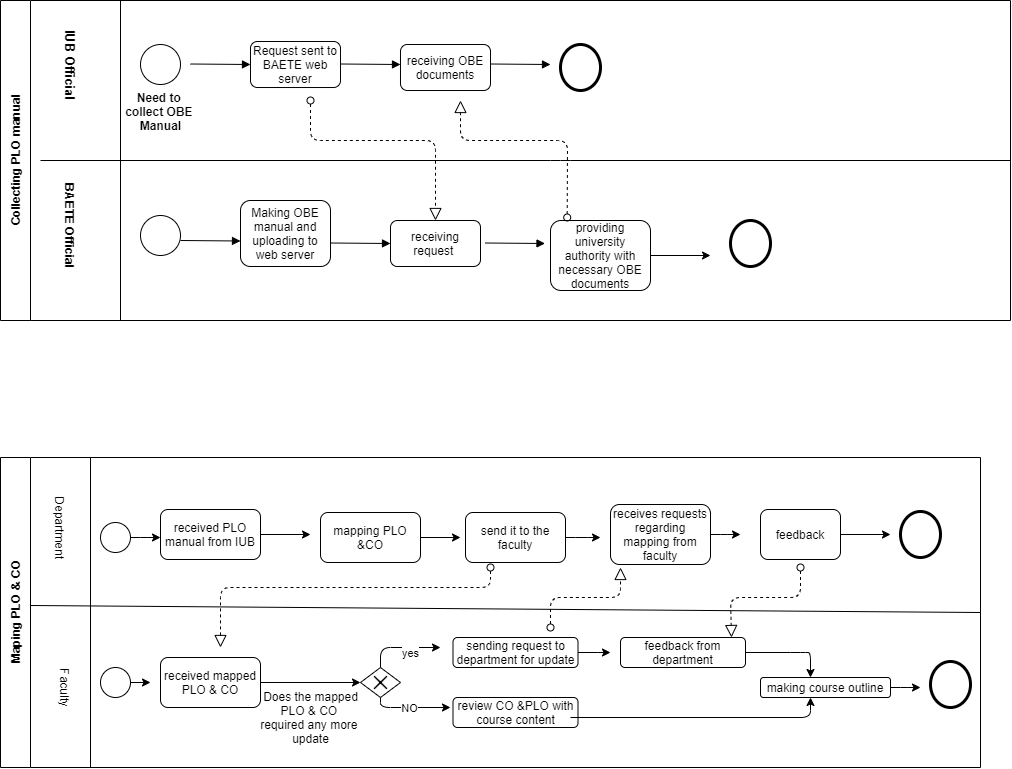
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **2)Making course outline and assessment plan** | **A) Department:**  1. After receiving the manual from BAETE department will map CO and PLO for each course and then provide the mapped guideline to faculty.  2. Department may receive an update from faculties if there any change need to do with CO and PLO mapping  **B) Faculty:**  2. He/she will check the mapping of CO and PLO.  3. If any change is necessary about the mapping faculty will contact the department.  3. And finally the course outline is created. | **A) Pen, paper:**  Pen, Paper for taking notes and writing purposes. | **A) Computer, Laptop:**  These are used to make or edit softcopies of course outline.  It is also used for storing softcopies of the course outline and curriculum. | **A) MS Word:**  Used for writing soft copies and making changes to them. We can also save softcopies in text format. | **A) OBE System:**  Departments and faculties can check the previous records of CO/PLO mapping. | **A) WAN, LAN WIFI, BROADBAND:**  These are the communication platforms to transfer softcopies. |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **3) Assessment phase and attendance** | **A) Students:**  1. Initially they participate in a class and sit for certain test on certain date.  2. They may ask to clarify if they have any questions.  3. Once they are done with the exam, they submit their answer script.  4. Student will give feedback to the faculty after reviewing their answer script.  **B) Faculties:**  1. They take attendance, prepare assessment papers (take tests online in any emergency situation).  2. They also invigilate or monitor the test given by the students.  3. They often need to clarify the questions asked by the students.  4. They collect the scripts.  5. They will assess the answer script. Initially they will calculate the marks achieved by a student.  6. They will show the scripts to the students.  7. They will receive updates from students if any correction needed.  8. Then faculty will finalize a mark sheet. | **A) Pen, paper, rubber, pencil:**  1. Paper is supplied to students when the exams are taken in offline.  2. It is also used as an answer sheet.  3. Students need pen, pencil, eraser, ruler to complete the answer script. | **A) Computers, Laptops:**  1. Used to conduct particular test in courses related to computer science.  2. They are used to conduct online classes as well. | **A) MS Word, GOOGLE DOC,**  **MS Excel,**  **Google classroom,**  **Google meet:**  1. Google classroom is used for taking classes and conducting exams.  2. Google meet is used for monitoring and gather information about the participants of the exam.  3. MS Word or Google DOCS are used for writing answers.  4. Students submit their answers through the panels in google classroom.  5. MS Excel is used for storing each student marks and final grades.  6. It is also used to evaluate PLO and CO score to store them. | **A) Manual database, CITS:**  1. It is used to keep records of the participants of the test.  2. Used for online classes in case of a crisis situation. | **A) WAN,LAN WIFI,BROADBAND:**  1. Used to allow online examinations to be conducted. |

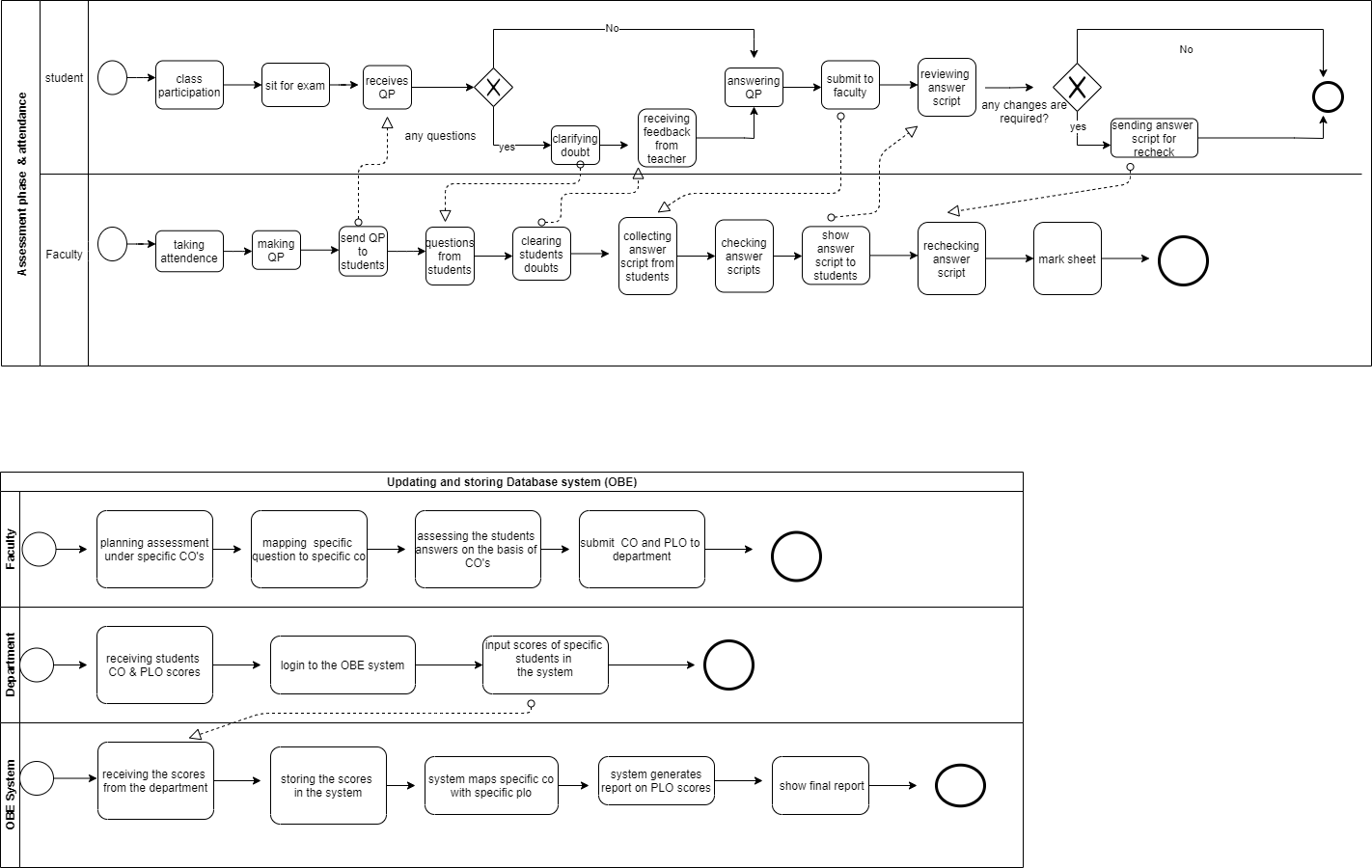
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **4) Updating & storing**  **Database system (OBE)** | **A) Faculty:**  1. Faculties can plan how each of the CO can be used for questions and assessments tests  2. Mapping each question with certain CO  3. Faculties assess the students for each & every questions on the basis of CO/PLO.  4. They will submit PLO and CO scores to the department.  **B) Department:**  1. receiving students CO & PLO scores  2. department will log in to the system  3. A department  data entry operator input all the score received from faculty directly to  OBE system.  **C) OBE system :**  1. system will receive input  2. storing data in the system  1. Each CO is mapped with certain PLOs.  2. OBE system  generates the  PLO scores & charts by clicking.  3. End of the semester overall  Performance of student, department and  course report  & verdict will be  generated by  OBE system. | **A) Paper, Pen:**  Hard copies of answer scripts. Pen is used for marking scripts | **A) Computers, Laptops:**  These devices are used to check the scripts in case of a crisis. Also used to store the answer scripts. | **A) MS Excel, OBE system:**  MS Excel is used to enter and store the marks of each student in each question. Also used to enter and store the grades of the students. OBE system is used to evaluate the PLO & CO scores and store them. | **A) OBE system:**  This system will be used to keep the records of PLO & CO scores. | **A) WAN, LAN, Wi-Fi, Broadband:**  Any communication  Or Network used. |

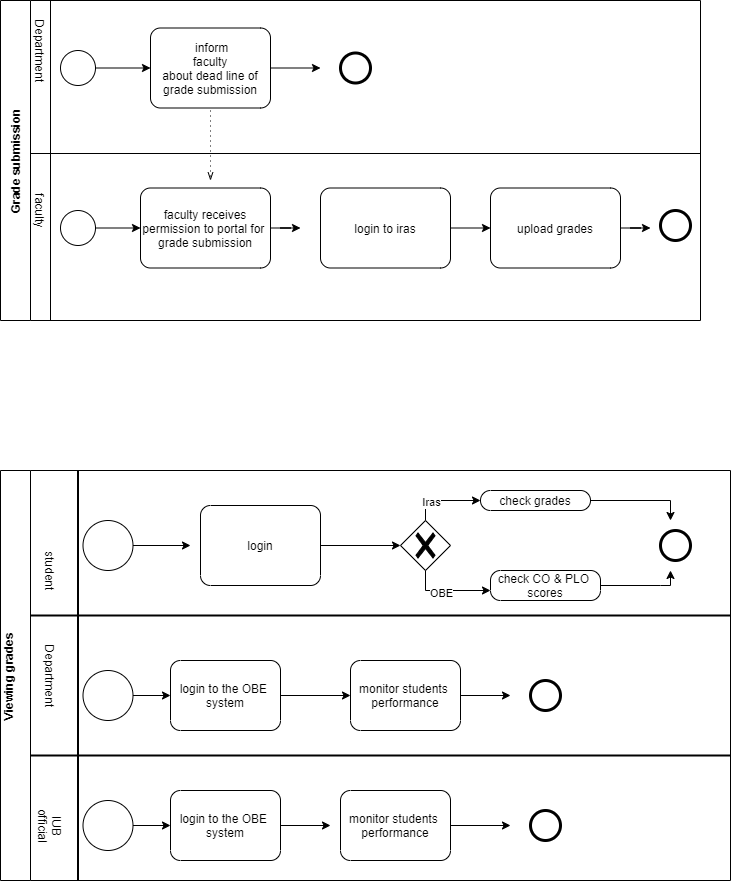
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **5) Grade submission** | **A) Faculty:**  1. Faculty will receive permission or dead line for grade submission  2. He/she will upload the grade of the students to the IRAS after log in to IRAS.  **B) Department:**  1. Department will inform the faculty about the deadline of grade submission. | **A) Paper:**  Hard copies of the grade submission and PLO and CO scores. | **A) Computer, Laptops:**  1. Use to store soft copies of the score grade in Excel sheets.  2. They are also use to access IRAS. | **A) MS EXCEL, IRAS:**  Excel sheet is used to storing student marks.  Grades will be uploaded to IRAS. | **A) Manual database, IRAS, OBE system:**  1. It is used for faculty to keep records of the students and their grades.  2. IRAS is used for grade submission.  3. It is also used for viewing student’s credit, CGPA and all other information.  4. OBE will keep track of student entire performance through-out the semester. | **A) WAN, LAN WIFI,BROADBAND:**  Network is used for transferring excel files. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System role** | | | | | |
| **Human** | **Non-**  **Computing**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Communication** |
| **6) Viewing grades** | **A) Students:**  They can login into IRAS to check their grades. They also can log in to OBE system to check their performance.  **B) Department:**  Department have access to the OBE system. They will monitor student progress.  **C) IUB officials:**  IUB officials access the OBE system to see PLO and CO scores. | **A) Paper:**  Hard copies of the grade submission and PLO and CO scores. | **A) Computers, Laptops, Phone:**  They are used for accessing IRAS, OBE.  They are used for storing PLO and CO scores, grades of the students. | **A) Browser (Chrome, Mozilla, Opera), IRAS, OBE:**  To access the website of the IRAS.  IRAS to view grades.  OBE to view performance. | **A) IRAS, OBE:**  IRAS, OBE system is used by the students and other officials with the id and Password. | **A) WAN, LAN WIFI,BROADBAND:**  Any network to access IRAS from a digital device such as computer, laptop, mobile. |

# Process Diagram (TO-BE)







# Chapter 3

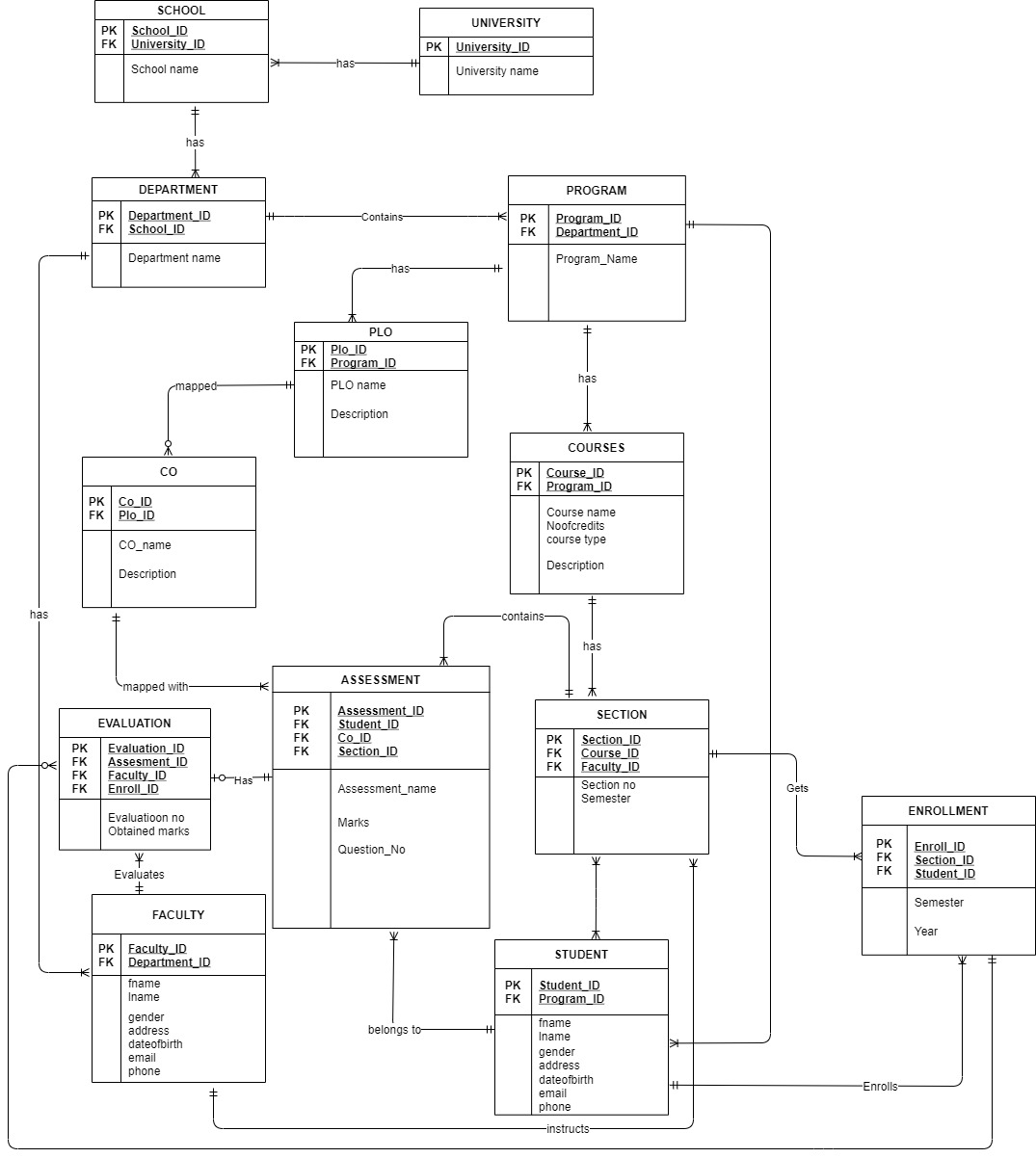
SYSTEM DESIGN

1. Business Rule
2. ERD (Entity Relationship Diagram)
3. Relational Schema
4. Normalization
5. Data Dictionary

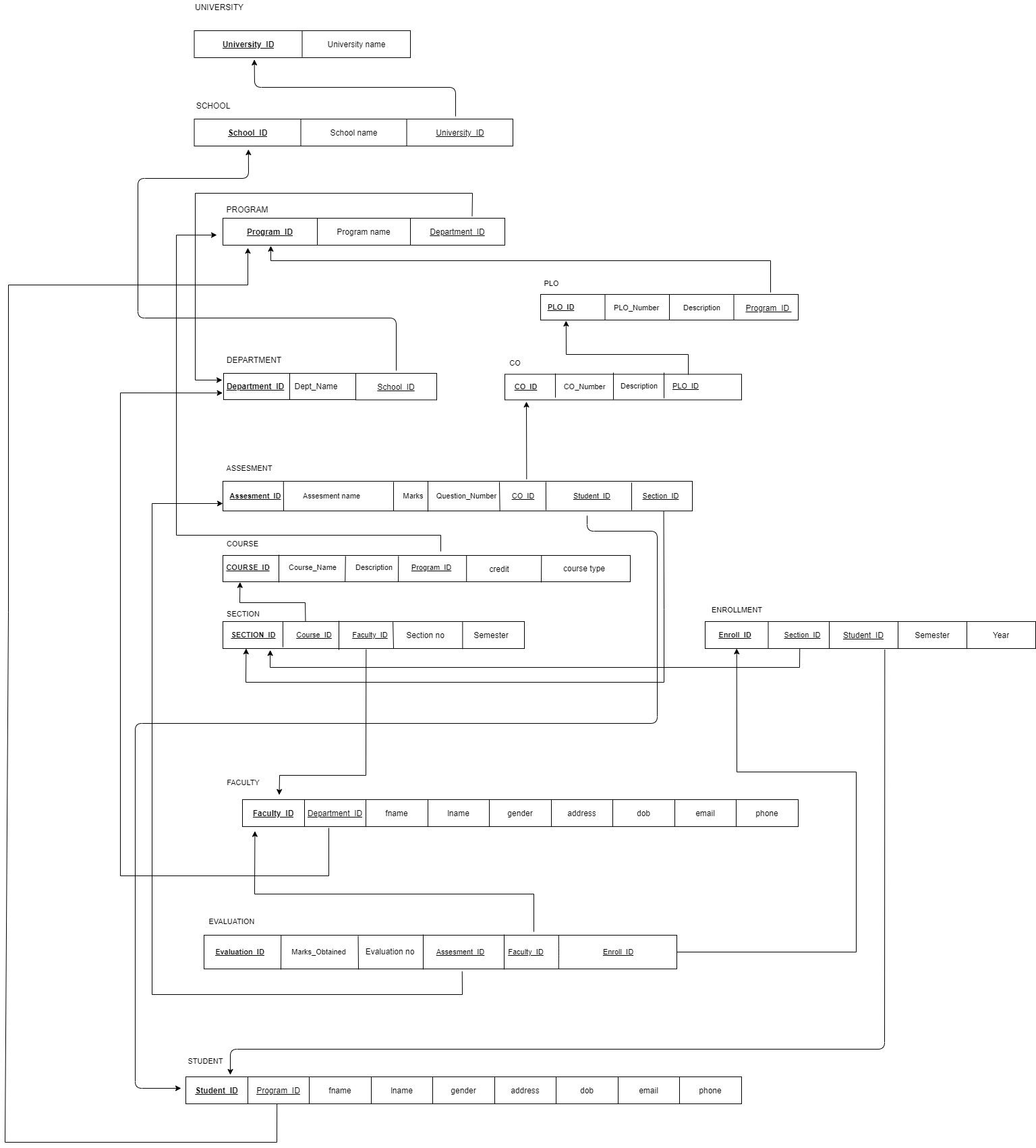
# Business Rule

1. A university has unique id and name. There are many schools in a university with id and name. Under each school there are many department with id and name.
2. BAETE makes a standard OBE manual which then uploaded to their website. Each department then gathers information from the BAETE manual/ website.
3. Each of the Degree Programs under which a student gets admitted belongs to a department. A Program has exactly one department. Each program consists of many courses and a course belongs to exactly one program.
4. Under the OBE model for each program there will be a set of program learning outcomes (PLO). A PLO has a PLO id, PLO name and description.
5. A program has many students but a student must have exactly one program. Each of the department and program has a name and id. Each department has a particular faculty as a head. Department offers many courses but a course is offered by exactly one department.
6. To evaluate the students in each course, courses have a set of course outcomes (CO’s) that are mapped with the PLO’s of the degree program. A CO must be mapped with exactly one PLO. A PLO may be mapped with one or more CO’s. The CO’s are measured through different assessment techniques (e.g., quiz, mid, final, project, presentation). A CO has a CO id, CO name and description.
7. An assessment is mapped with exactly one CO and a CO is mapped with one or more assessments. Each of the assessments are identified uniquely using an Assessment id. Assessment has an assessment name (Mid Q1, Mid Q2, Final Q1, etc.), COID, Section Number and the total marks that is achievable in that particular assessment. An assessment contains exactly one section. A section must have one or more assessments.
8. Faculties assess the COs achieved and mapped PLOs achieved by each student in a course. An assessment is done by one faculty and a faculty must do many assessment.
9. Faculties have academic qualifications (i.e. highest degree certificate so far), area/s of specialization, job position (e.g. Lecturer, Professor). A faculty has exactly one department and a department has multiple faculties.
10. A Course have a course id that uniquely identifies the course. Course also has a course title. A course can be a prerequisite of one or more courses and a course may have one or more prerequisites. A course may be mapped with many other courses and multiple courses may be mapped with exactly one course.
11. Each course must be taught by at least one faculty. A Faculty may teach multiple courses. Every faculty has a teaching schedule i.e. teaching days and teaching time for a course. There may be multiple sections for each course in a particular semester but a particular section teaches exactly one course.
12. A section has a section number; however, different courses may have sections with the same number assigned to them. Hence, a semester and course id are required along with the section number to identify a section. A section also has a schedule and a maximum capacity. A student may enroll in one or more sections and a section must have many student. There is a registration date for the courses taken by a student in a semester. If a course has no student enrolled, then all of its sections along with the course are removed.

# ERD



# Relational Schema



# Normalization

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| UNIVERSITY | **University\_ID** | U1 | SCHOOL | **School\_ID** | S1 |
| University name | U2 | School name | S2 |
| University\_ID | U1 |
| DEPARTMENT | **Department\_ID** | D1 | PROGRAM | **Program\_ID** | P1 |
| Dept. name | D2 | Program name | P2 |
| School\_ID | S1 | Department\_ID | D1 |
| CO | **CO\_ID** | C1 | PLO | **PLO\_ID** | P11 |
| CO\_Number | C2 | PLO\_Number | P12 |
| Description | C3 | Description | P13 |
| PLO\_ID | P11 | Program\_ID | P1 |
| COURSE | **Course\_ID** | C11 | ASSESSMENT | **Assessment\_ID** | A1 |
| Course name | C12 | Assessment name | A2 |
| Description | C13 | Marks | A3 |
| Credit | C14 | Ques. No | A4 |
| Course type | C15 | CO\_ID | C1 |
| Program\_ID | P1 | Student\_ID | S111 |
| Section\_ID | S11 |
| SECTION | **Section\_ID** | S11 | ENROLLMENT | **Enroll\_ID** | E1 |
| Course\_ID | C11 | **Section\_ID** | S11 |
| Faculty\_ID | F1 | Student\_ID | S111 |
| Section no | S12 | Semester | E2 |
| Semester | S13 | Year | E3 |
| FACULTY | **Faculty\_ID** | F1 | STUDENT | **Student\_ID** | S111 |
| Department\_ID | D1 | Program\_ID | P1 |
| Fname | F2 | Fname | S112 |
| Lname | F3 | Lname | S113 |
| Gender | F4 | Gender | S114 |
| Address | F5 | Address | S115 |
| Date of Birth | F6 | Date of birth | S116 |
| Email | F7 | Email | S117 |
| Phone | F8 | Phone | S118 |
| EVALUATION | **Evaluation\_ID** | E11 |  |  |  |
| Marks obtained | E12 |
| Evaluation no | E13 |
| Assessment\_ID | A1 |
| Faculty\_ID | F1 |
| Enroll\_ID | E1 |

|  |  |
| --- | --- |
| **U1→** | U2 |
| **S1→** | S2, U1 |
| **D1→** | D2, S1 |
| **P1→** | P2, D1 |
| **C1→** | C2, C3, P11 |
| **P11→** | P12, P13, P1 |
| **C11→** | C12, C13, C14, C15, P1 |
| **S11→** | S12, S13, C11, F1 |
| **E1→** | E2, E3, S11, S111 |
| **S111→** | S112, S113, S114, S115, S116, S117, S118, P1 |
| **F1→** | F2, F3, F4, F5, F6, F7, F8, D1 |
| **A1→** | A2, A3, A4, C1, S111, S11 |
| **E11→** | E12, E13, A1, F1, E1 |

|  |  |
| --- | --- |
| **University\_ID→** | University name |
| **School\_ID→** | School name, University\_ID |
| **Department\_ID→** | Department name, School\_ID |
| **Program\_id→** | Program name, Department\_ID |
| **CO\_ID→** | CO number, Description, PLO\_ID |
| **PLO\_ID→** | PLO number, Description, Program\_ID |
| **Course\_ID→** | Course name, Description, Credit, Course type, Program\_ID |
| **Section\_ID→** | Section no, Semester, Course\_ID, Faculty\_ID |
| **Enroll\_ID→** | Semester, Year, Section\_ID, Student\_ID |
| **Student\_ID→** | Fname, Lname, Gender, Address, Date of Birth, Email, Phone, Program\_ID |
| **Faculty\_ID→** | Fname, Lname, Gender, Address, Date of Birth, Email, Phone, Department\_ID |
| **Assessment\_ID→** | Assessment name, Marks, Ques. no, CO\_ID, Student\_ID, Section\_ID |
| **Evaluation\_ID→** | Marks Obtained, Evaluation no, Assessment\_ID, Faculty\_ID, Enroll\_ID |

**1NF**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **E11** | E12 | E13 | U1 | U2 | S1 | S2 | D1 | D2 | P1 | P2 | C1 | C2 | C3 | P11 | P12 |
| P13 | C11 | C12 | C13 | C14 | C15 | S11 | S12 | S13 | E1 | E2 | E3 | S111 | S112 | S113 | S114 |
| S115 | S116 | S117 | S118 | F1 | F2 | F2 | F4 | F5 | F6 | F7 | F8 | A1 | A2 | A3 | A4 |

**2NF**

The relations are already in 2nf.

|  |  |
| --- | --- |
| **U1** | U2 |

**3NF**

|  |  |  |
| --- | --- | --- |
| **S1** | S2 | U1 |

|  |  |  |
| --- | --- | --- |
| **D1** | D2 | S1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **P11** | P12 | P13 | P1 |

|  |  |  |
| --- | --- | --- |
| **P1** | P2 | D1 |

|  |  |  |  |
| --- | --- | --- | --- |
| **C1** | C2 | C3 | P11 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **S111** | S112 | S113 | S114 | S115 | S115 | S117 | S118 | P1 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **A1** | A2 | A3 | A4 | C1 | S111 | S11 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **S11** | S12 | S13 | C11 | F1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **E11** | E12 | E13 | A1 | F1 | E1 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **E1** | E2 | E3 | S11 | S111 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **F1** | F2 | F3 | F4 | F5 | F6 | F7 | F8 | D1 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| P1 | C12 | C13 | C14 | C15 | **C11** |

**BCNF**

No non-key can identify any primary key or part of the primary key. Therefore, all the relations are in BCNF.

# Data Dictionary

**University\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data type | Size | Remarks |
| cuniversityID | VARCHAR | 5 | This is the PRIMARY KEY of the University |
| cuniversityName | VARCHAR | 225 | This is the name of the University |

**School\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| cschool\_id | VARCHAR | 5 | This is the PRIMARY KEY of School |
| cuniversityID | VARCHAR |  | This is the FOREIGN KEY from University Table |
| cschoolName | VARCHAR | 30 | This is the name of the School  Example: “School of Engineering and Science” |

**Department\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| cdepartmentID | VARCHAR | 5 | This is the PRIMARY KEY of the Department  Example: “CSE” |
| cschool­\_id | VARCHAR | 5 | This is the FOREIGN KEY of the School |
| cdepartmentName | VARCHAR |  | This the Name of the Department  Example: “Computer Science and Engineering” |

**Program\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| cprogramID | VARCHAR | 5 | This is the PRIMARY KEY for a program  Example: “B.Sc” |
| cdepartment\_id | VARCHAR | 5 | This is the FOREIGN KEY from the Department table  Example: “BS.C” |
| cprogramName | VARCHAR | 30 | This is the name of the Degree program  Example: “Bachelor of Science” |

**Course\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccourseID | VARCHAR | 7 | This is the PRIMARY KEY for the course  Example: “CSE203” |
| cprogramID | VARCHAR | 5 | This is the FOREIGN KEY from the Program Key |
| ccourseName | VARCHAR | 40 | This is the name of the course  Example: ”Data Base” |
| nnoOfCredits | integer |  | This is the number of credits for the course  Example: “3” |
| ccourseType | VARCHAR | 10 | This is the type of the course  Example: Core |
| cdescription | VARCHAR |  | This is the description of the course |

**PLO\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | remarks |
| cploID | VARCHAR | 5 | This is the PRIMARY KEY of the Program Learning Outcome  Example: “PLO1” |
| cprogram\_id | VARCHAR | 5 | This is the FOREIGN KEY from the Program Table  Example: “B.Sc” |
| cploName | VARCHAR |  | This is the name of the Program Learning Outcome |
| cdescription | VARCHAR |  | This the description of the Program Learning Outcome |

**CO\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| ccoID | VARCHAR | 5 | This is the PRIMARY KEY for the Course Outcome |
| cplo\_id | VARCHAR | 5 | This is the FOREIGN KEY from the Program Learning Outcome  Example: “PLO1” |
| ccoName | VARCHAR |  | This is the name of the Course Outcome |
| cdescription | VARCHAR |  | This is the name of the Course Outcome |

**Section\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| nsectionID | integer |  | This is the PRIMARY KEY for the section |
| course\_id | VARCHAR | 7 | This is the FOREIGN KEY for the section  Example: “ CSE213” |
| cfaculty\_id | VARCHAR |  | This is the FOREIGN KEY from the faculty table |
| nsectionNO | INTEGER |  | This is the section number |
| csemester | VARCHAR |  | This is the semester number |

**Student\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| cstudentID | VARCHAR | 10 | This is the PRIMARY KEY for the student  Example: “1710237” |
| cprogram­\_id | VARCHAR | 5 | This is the FOREIGN KEY for the student |
| cfname | VARCHAR | 80 | This is first name of the student  Example: “Mohammad Tasiful” |
| clname | VARCHAR | 20 | This is the last name of the student  Example: “Amin” |
| cgender | VARCHAR |  | This is the gender of the student  Example:”M” |
| caddress | VARCHAR | 50 | This is the address of the student  Example: “ College road, Chawkbazar, Chattagram, Keari Elysium” |
| cdateofbirth | DATE | DD-MM-YYYY | This is the date of birth of the student  Example: 02-02-1997 |
| cemail | VARCHAR | 30 | This is the email of the student  Example: “1710575@iub.edu.bd” |
| cphone | VARCHAR | 20 | This is the phone number of the student  Example: “01819686321” |

**Enrollment\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| cenrollID | VARCHAR |  | This is the PRIMARY KEY for enrollment |
| nsectionID | INTEGER |  | This is the FOREIGN KEY for enrollment |
| cstudent\_id | VARCHAR | 7 | This is the FOREIGN KEY for enrollment  Example: “1715437” |
| csemester | VARCHAR | 6 | This is the semester of enrollment  Example: “Autumn” |
| cyear | year | yyyy | This is the year of enrollment  Example:2019 |

**Faculty\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| cfacultyID | VARCHAR | 4 | This is the PRIMARY KEY for faculty  Example: “1801” |
| cdepartment\_id | VARCHAR | 5 | This is the FOREIGN KEY for faculty |
| cfname | VARCHAR | 30 | This is first name of the faculty  Example: “ Mahady” |
| clname | VARCHAR | 20 | This is the last name of the faculty  Example: “Hasan” |
| cgender | VARCHAR | 1 | This is the gender of the faculty  Example: “M” |
| caddress | VARCHAR | 30 | This is the address of the faculty  Example: “House 1, Road 1, Sector 1, Uttara, Dhaka, Bangladesh” |
| cdateOfBirth | DATE | DD-MM-YYYY | This is the date of birth of the faculty  Example: “15-05-1980 |
| cemail | VARCHAR | 30 | This is the email of the faculty  Example: “mahady@iub.edu.bd” |
| cphone | VARCHAR | 15 | This is the phone number of the faculty  Example: 01855247992 |

**Assessment\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| nassessmentID | INTEGER |  | This is the PRIMARY KEY for assessment |
| cstudentID | VARCHAR | 7 | This is the FOREIGN KEY for assessment |
| nco\_id | INTEGER |  | This is the FOREIGN KEY from Course |
| nsection\_id | Integer |  | This is the FOREIGN KEY from Section Table |
| cassessmentName | VARCHAR |  | This is the name of the assessment |
| cmarks | VARCHAR | 6 | This is the marks of the assessment |
| cquestionNo | VARCHAR |  | This is the number of question in the assessment |

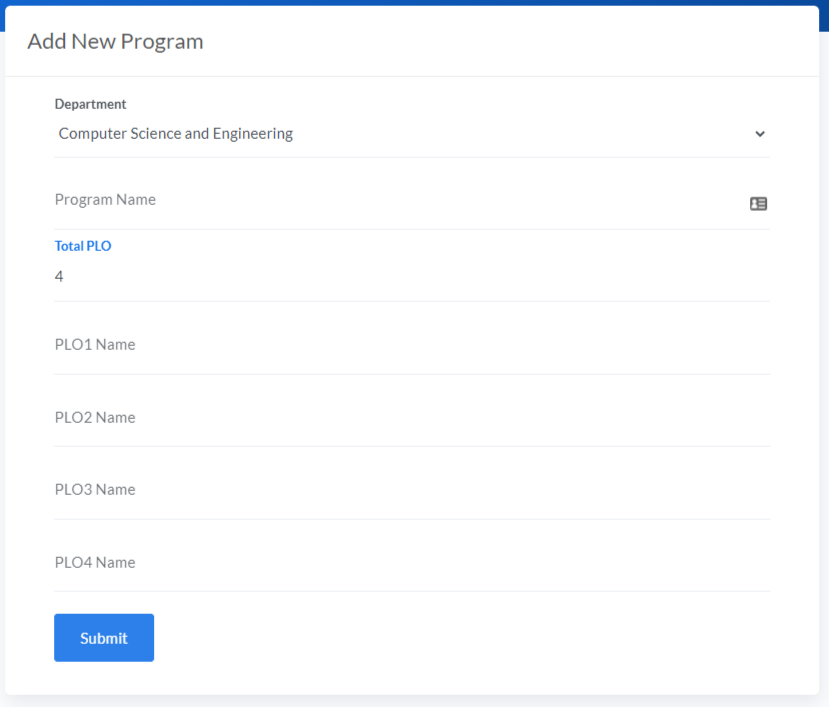
**Evaluation\_T**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remarks |
| nevaluationID | INTEGER |  | This is the PRIMARY KEY for evaluation |
| nassessment\_id | INTEGER |  | This is the FOREIGN KEY from the assessment table |
| cfacultyID | VARCHAR | 4 | This is the FOREIGN KEY from the faculty table  Example: “1801” |
| cenrollID | VARCHAR |  | This is the FOREIGN KEY from the enrollment table |
| nevaluationNO | INTEGER |  | This is the number of evaluations |
| nobtainedMarks | FLOAT |  | This is the marks obtained by the student  Example: “25.5” |

# Chapter 4

PHYSICAL SYSTEM DESIGN

1. Input
2. Output



<?php

    include 'mysql.php';

    $program\_name = $\_POST['program\_name'];

    $department\_id = $\_POST['department\_id'];

    $query = "INSERT INTO program (program\_name, department\_id)

                VALUES ('$program\_name', '$department\_id')";

    $conn->query($query);

    $program\_id = $conn->insert\_id;

    $plo\_total = $\_POST['plo\_total'];

    foreach(range(1,$plo\_total) as $plo){

        $plo\_name = $\_POST['plo'.$plo];

        $query = "INSERT INTO plo (plo\_num, plo\_name, program\_id)

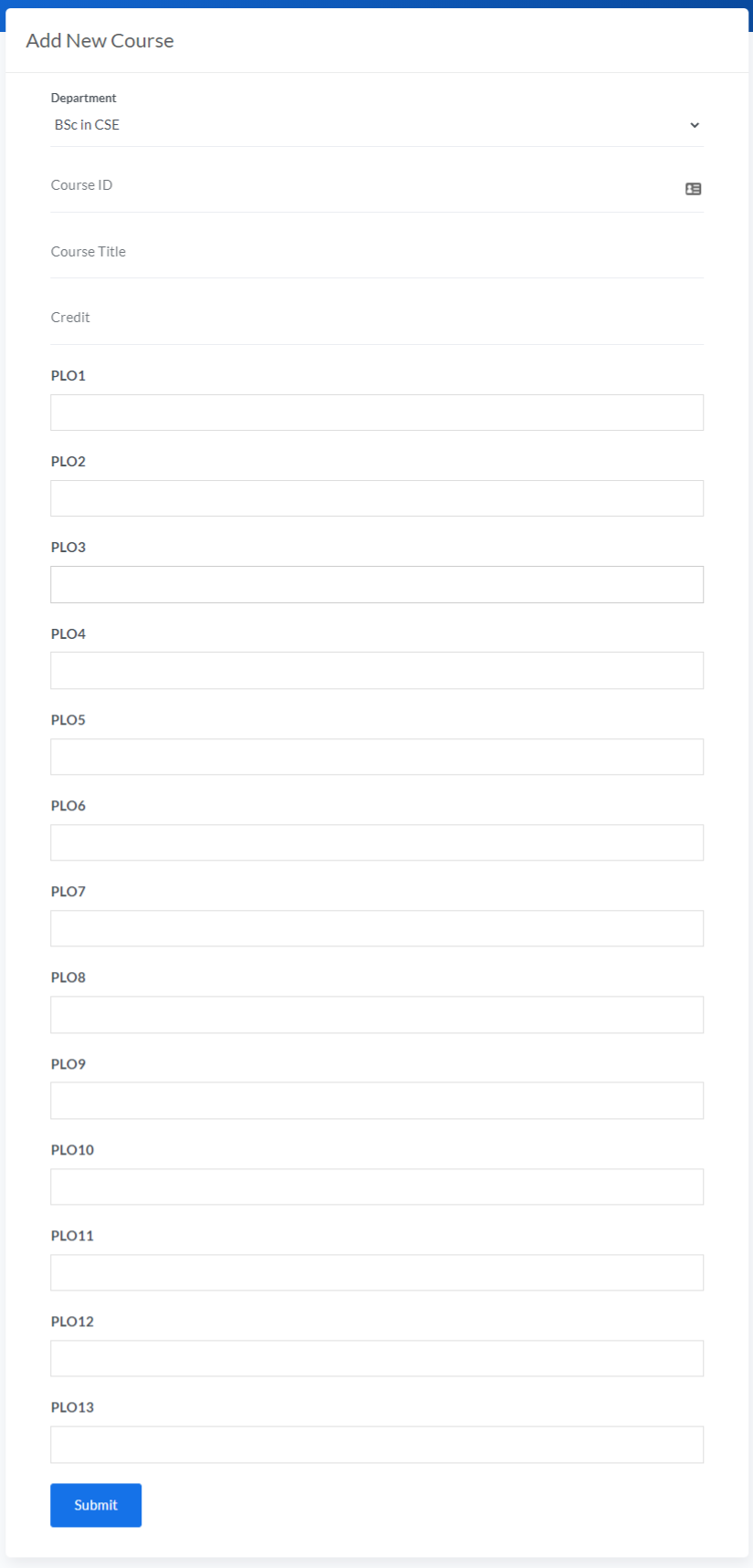
                    VALUES($plo, '$plo\_name', $program\_id)";

        $conn->query($query);

    }

    header("Location: ../department/add-program.php");

?>



<?php

    include 'mysql.php';

    $course\_id = strtolower($\_POST['course\_id']);

    $course\_name = $\_POST['course\_name'];

    $no\_credits = $\_POST['no\_credits'];

    $program\_id = $\_POST['program\_id'];

    $query = "INSERT INTO course (course\_id, course\_name, no\_credits, program\_id)

                VALUES ('$course\_id', '$course\_name', $no\_credits, $program\_id)";

    $conn->query($query);

    foreach(range(1, 13) as $plo){

        if ($\_POST['plo'.$plo]!=NULL){

            $plo\_data = json\_decode($\_POST['plo'.$plo]);

            foreach($plo\_data as $p){

                $query = "SELECT \* FROM plo WHERE plo\_num = $plo AND program\_id = $program\_id";

                $plo\_id = $conn->query($query)->fetch\_assoc()['plo\_id'];

                $co = substr($p->value, 2);

                $query = "INSERT INTO co (co\_num, plo\_id) VALUES ('$co', $plo\_id)";

                $conn->query($query);

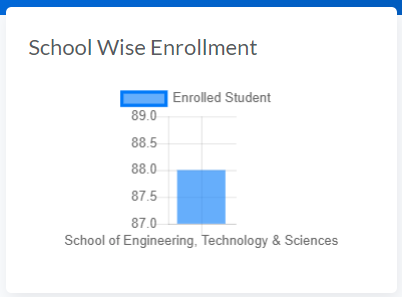
            }

        }

    }

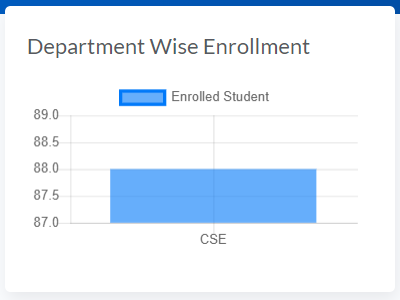
    header ("Location: ../department/add-course.php");

?>



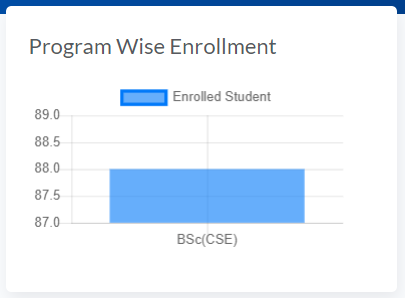
$query = "SELECT school.school\_name as 'name', COUNT(DISTINCT(enrollment.student\_id)) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN enrollment GROUP BY school.school\_id";

        $scls = $conn->query($query);



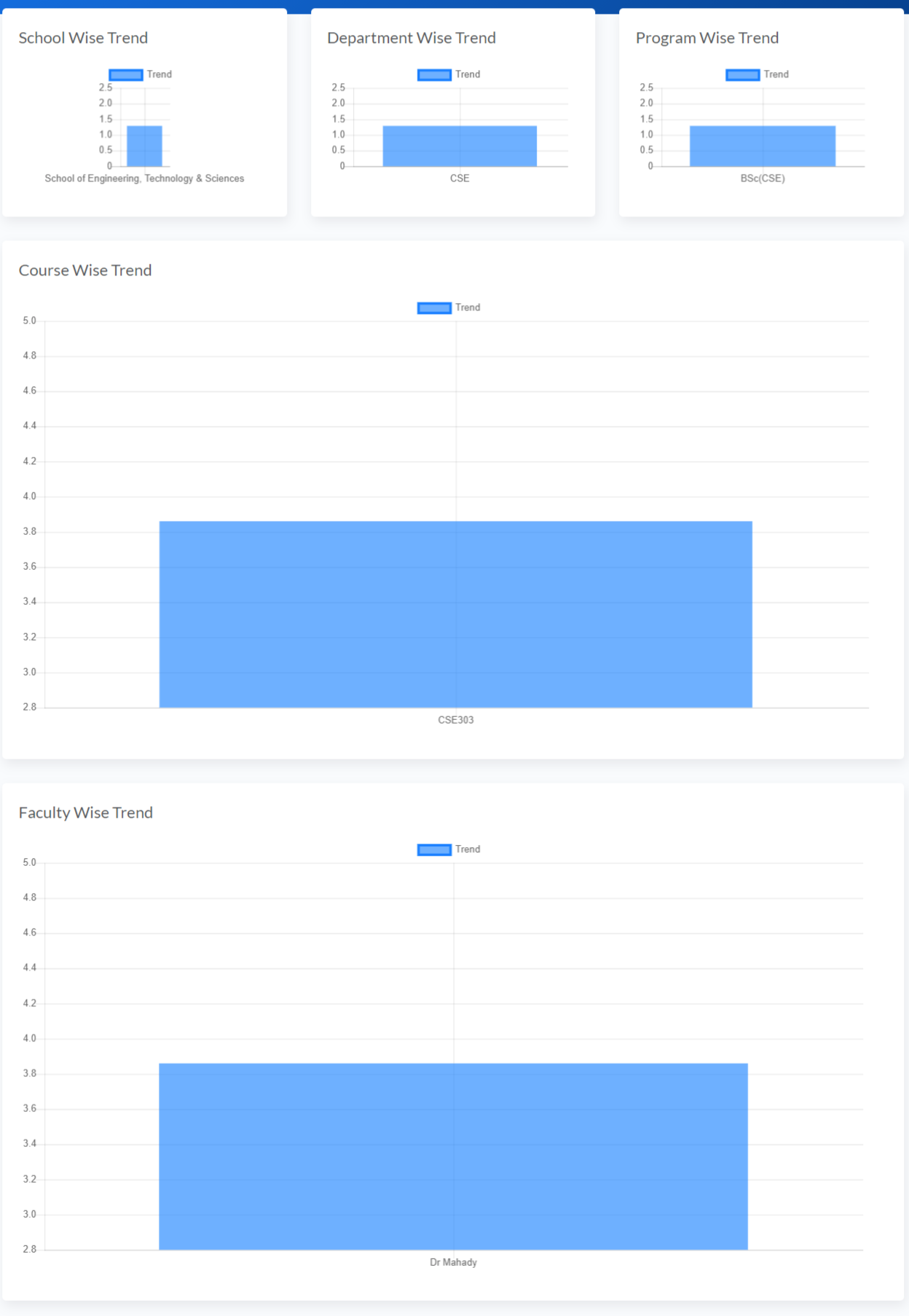
$query = "SELECT UPPER(department.department\_id) as 'name', COUNT(DISTINCT(enrollment.student\_id)) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN enrollment GROUP BY department.department\_id";

        $deps = $conn->query($query);



$query = "SELECT UPPER(department.department\_id) as 'dep', program.program\_name as 'name', COUNT(DISTINCT(enrollment.student\_id)) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN enrollment GROUP BY program.program\_id";

        $progs = $conn->query($query);



1. SCHOOL WISE STUDENT PEFORMACE TREND BASED ON CGPA
2. DEPARTMENT WISE STUDENT PEFORMACE TREND BASED ON CGPA
3. PROGRAM WISE STUDENT PEFORMACE TREND BASED ON CGPA
4. COURSE WISE STUDENT PEFORMACE TREND BASED ON GPA
5. INSTRUCTOR WISE STUDENT PEFORMACE TREND BASED ON GPA

code in the next page, for all 4 graphs shown above

$semester = $\_GET['semester'];

        $query = "SELECT school\_name, department\_id, program\_name,  semester, faculty, course\_id, no\_credits, student\_id, SUM(marks) as 'marks' FROM (SELECT school.school\_name, department.department\_id, program.program\_name, section.semester, CONCAT(faculty.fname, ' ', faculty.lname) as 'faculty', course.course\_id, course.no\_credits, enrollment.student\_id, IF(assessment.assessment\_name = 'final', (SUM(evaluation.obtained\_marks) / SUM(assessment.marks)) \* 40, (SUM(evaluation.obtained\_marks) / SUM(assessment.marks)) \* 30) as 'marks', assessment.assessment\_name FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN faculty LEFT JOIN co ON assessment.co\_number = co.co\_num AND section.section\_id = co.section\_id LEFT JOIN plo on co.plo\_id = plo.plo\_id LEFT JOIN enrollment ON enrollment.enroll\_id = evaluation.enroll\_id WHERE section.semester = LOWER('$semester') GROUP BY course.course\_id, enrollment.student\_id, assessment.assessment\_name, course.course\_id) as subQuery GROUP BY semester, student\_id, course\_id";

        $data = $conn->query($query);

if($rows !=0 ){

            $scl = array();

            $dep = array();

            $prog = array();

            $crs = array();

            $fac = array();

            foreach($data as $d){

                $school = $d['school\_name']; $department = $d['department\_id']; $program = $d['program\_name']."($department)"; $course = strtoupper($d['course\_id']); $faculty = $d['faculty'];

                if(array\_key\_exists($school, $scl)==false){

                    $scl[$school] = array();

                }

                if(array\_key\_exists($department, $dep)==false){

                    $dep[$department] = array();

                }

                if(array\_key\_exists($program, $prog)==false){

                    $prog[$program] = array();

                }

                if(array\_key\_exists($course, $crs)==false){

                    $crs[$course] = array();

                    $crs[$course]['gp'] = 0;

                    $crs[$course]['count'] = 0;

                }

                if(array\_key\_exists($faculty, $fac)==false){

                    $fac[$faculty] = array();

                    $fac[$faculty]['gp'] = 0;

                    $fac[$faculty]['count'] = 0;

                }

                $student = $d['student\_id'];

                if(array\_key\_exists($student, $scl[$school]) == false){

                    $scl[$school][$student] = array();

                    $scl[$school][$student]['cg'] = 0;

                    $scl[$school][$student]['cr'] = 0;

                }

                if(array\_key\_exists($student,  $dep[$department]) == false){

                    $dep[$department][$student] = array();

                    $dep[$department][$student]['cg'] = 0;

                    $dep[$department][$student]['cr'] = 0;

                }

                if(array\_key\_exists($student,  $prog[$program]) == false){

                    $prog[$program][$student] = array();

                    $prog[$program][$student]['cg'] = 0;

                    $prog[$program][$student]['cr'] = 0;

                }

                $cr = $d['no\_credits'];

                $scl[$school][$student]['cr']+=$cr; $dep[$department][$student]['cr']+=$cr; $prog[$program][$student]['cr']+=$cr; $crs[$course]['count']++; $fac[$faculty]['count']++;

                $cg = 0; $marks = $d['marks'];

                if($marks>=85){

                    $cg =($cr \* 4.0);

                }else if($marks>=80){

                    $cg =($cr \* 3.7);

                }else if($marks>=75){

                    $cg =($cr \* 3.3);

                }else if($marks>=70){

                    $cg =($cr \* 3.0);

                }else if($marks>=65){

                    $cg =($cr \* 2.7);

                }else if($marks>=60){

                    $cg =($cr \* 2.3);

                }else if($marks>=55){

                    $cg =($cr \* 2.0);

                }else if($marks>=50){

                    $cg =($cr \* 1.7);

                }else if($marks>=45){

                    $cg =($cr \* 1.3);

                }else if($marks>=40){

                    $cg =($cr \* 1.0);

                }

                $scl[$school][$student]['cg']+=$cg; $dep[$department][$student]['cg']+=$cg; $prog[$program][$student]['cr']+=$cg; $crs[$course]['gp']+=$cg; $fac[$faculty]['gp']+=$cg;

            }

            $scls = array();

            $deps = array();

            $progs = array();

            foreach($scl as $k => $v){

                $total = sizeof($v);

                $sum = 0;

                foreach($v as $m){

                    $sum+=($m['cg'] / $m['cr']);

                }

                $scls[$k] = round(($sum / $total), 2);

            }

            foreach($dep as $k => $v){

                $total = sizeof($v);

                $sum = 0;

                foreach($v as $m){

                    $sum+=($m['cg'] / $m['cr']);

                }

                $deps[$k] = round(($sum / $total), 2);

            }

            foreach($prog as $k => $v){

                $total = sizeof($v);

                $sum = 0;

                foreach($v as $m){

                    $sum+=($m['cg'] / $m['cr']);

                }

                $progs[$k] = round(($sum / $total), 2);

            }

        }



1. PLO TOTAL PERCENTAGE
2. PLO ATTEMPTION COMPARISON

$query = "SELECT plo, COUNT(stat) as 'stat' FROM (SELECT plo.plo\_num as 'plo', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id GROUP BY enrollment.student\_id, section.course\_id, plo.plo\_num ORDER BY stat DESC, plo ASC) as testQ WHERE stat=1 GROUP BY plo";

    $achieved = $conn->query($query);

    $query = "SELECT plo, COUNT(plo) as 'stat' FROM (SELECT plo.plo\_num as 'plo', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id GROUP BY enrollment.student\_id, section.course\_id, plo.plo\_num ORDER BY stat DESC, plo ASC) as testQ GROUP BY plo";

    $attempted = $conn->query($query);

    $p\_perc = array();

    foreach($attempted as $a){

        $p\_perc[$a['plo']]=array();

        $p\_perc[$a['plo']]['achieved'] = 0;

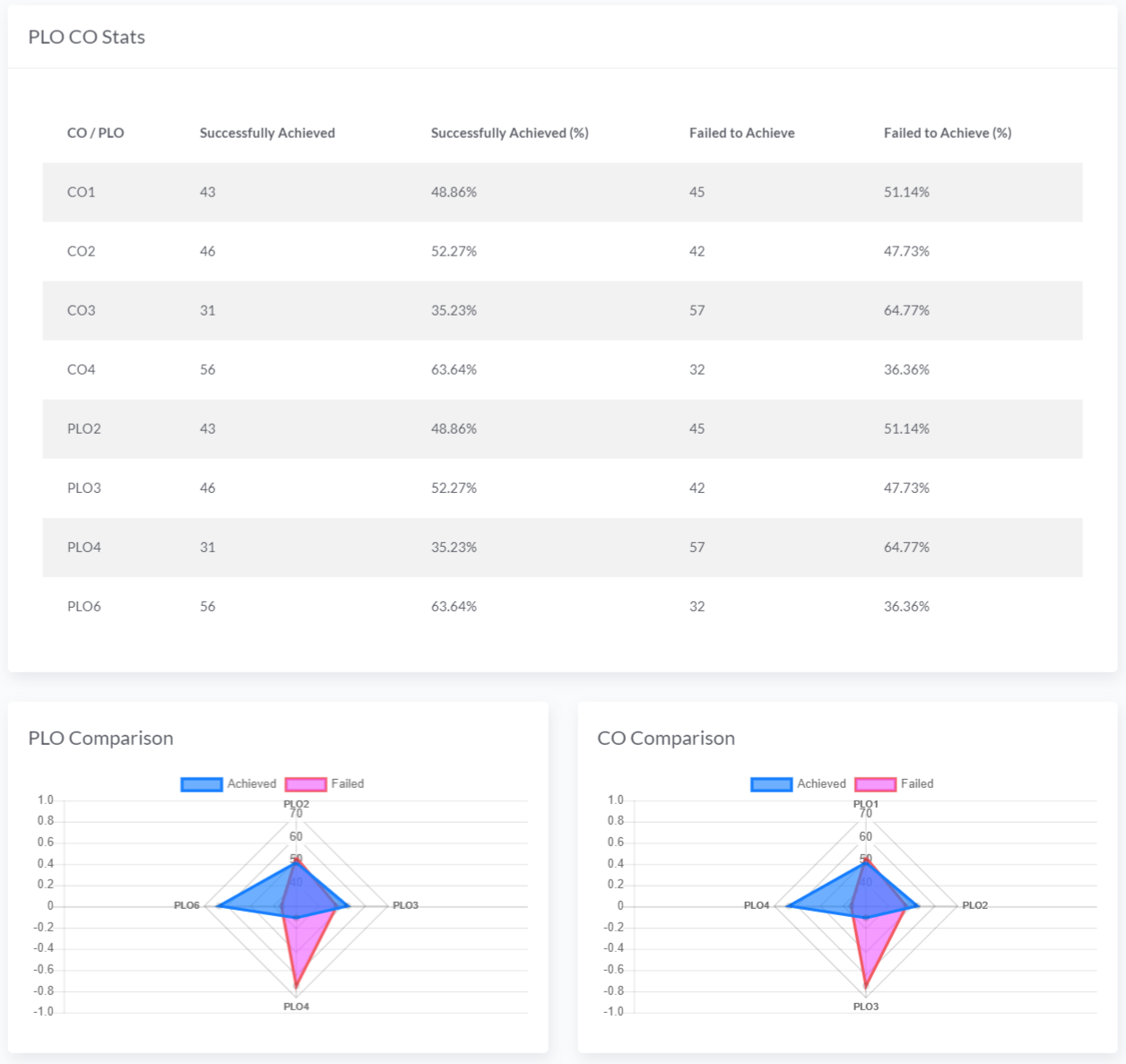
        $p\_perc[$a['plo']]['attempted'] = $a['stat'];

    }

    foreach($achieved as $a){

        $p\_perc[$a['plo']]['achieved'] = $a['stat'];

    }



1. CO-PLO ACHIEVEMENT STATS
2. COPMARISON OF PLO ACHIEVEMENT AND FAILURE
3. COPMARISON OF CO ACHIEVEMENT AND FAILURE

if($\_GET['qt']=='school'){

            $school = $\_GET['qs'];

            $query = "SELECT  school, department, program, co, COUNT(stat) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE school.school\_name LIKE '%$school%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ WHERE stat = 1 GROUP BY co";

            $achieved = $conn->query($query);

            $query = "SELECT school, department, program, co, COUNT(co) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE school.school\_name LIKE '%$school%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ GROUP BY co";

            $attempted = $conn->query($query);

        }else if($\_GET['qt']=='department'){

            $department = $\_GET['qs'];

            $query = "SELECT  school, department, program, co, COUNT(stat) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE department.department\_id LIKE '%$department%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ WHERE stat = 1 GROUP BY co";

            $achieved = $conn->query($query);

            $query = "SELECT school, department, program, co, COUNT(co) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE department.department\_id LIKE '%$department%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ GROUP BY co";

            $attempted = $conn->query($query);

        }else if($\_GET['qt']=='program'){

            $program = $\_GET['qs'];

            $query = "SELECT  school, department, program, co, COUNT(stat) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE program.program\_name LIKE '%$program%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ WHERE stat = 1 GROUP BY co";

            $achieved = $conn->query($query);

            $query = "SELECT school, department, program, co, COUNT(co) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE program.program\_name LIKE '%$program%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ GROUP BY co";

            $attempted = $conn->query($query);

        }else if($\_GET['qt']=='course'){

            $course = $\_GET['qs'];

            $query = "SELECT  school, department, program, co, COUNT(stat) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE course.course\_id LIKE '%$course%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ WHERE stat = 1 GROUP BY co";

            $achieved = $conn->query($query);

            $query = "SELECT school, department, program, co, COUNT(co) as 'stat' FROM (SELECT school.school\_name as 'school', program.department\_id as 'department', program.program\_name as 'program', co.co\_num as 'co', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE course.course\_id LIKE '%$course%' GROUP BY enrollment.student\_id, section.course\_id, co.co\_num ORDER BY stat DESC, co ASC) as testQ GROUP BY co";

            $attempted = $conn->query($query);

        }

        $c\_stat = array();

        foreach($attempted as $a){

            $c\_stat[$a['co']]=array();

            $c\_stat[$a['co']]['achieved'] = 0;

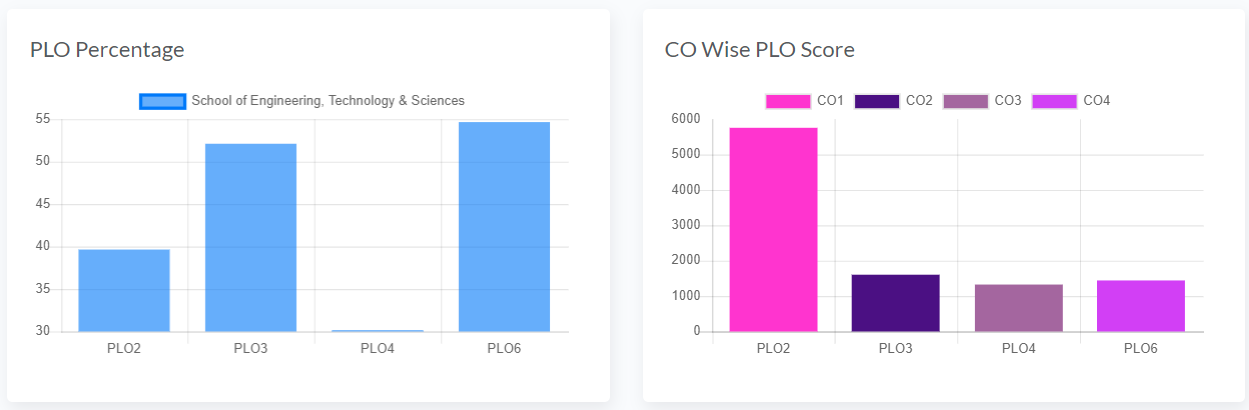
            $c\_stat[$a['co']]['attempted'] = $a['stat'];

        }

        foreach($achieved as $a){

            $c\_stat[$a['co']]['achieved'] = $a['stat'];

        }



1. PLO PERCENTAGE ACHIEVED FROM EACH CO
2. PLO SCORE ACHIEVED FROM ASSOCIATED CO

if($\_GET['qt']=='school'){

            $bas = 'school';

            $qs = $\_GET['qs'];

            $query = "SELECT school.school\_name AS 'school', department.department\_id AS 'department', program.program\_name AS 'program', plo.plo\_num as 'plo', co.co\_num as 'co', SUM(evaluation.obtained\_marks) as 'mark',  SUM(assessment.marks) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN faculty LEFT JOIN co ON assessment.co\_number = co.co\_num AND section.section\_id = co.section\_id LEFT JOIN plo on co.plo\_id = plo.plo\_id WHERE school.school\_name LIKE '%$qs%' GROUP BY school.school\_name, plo.plo\_num, co.co\_num";

            $res = $conn->query($query);

        }else if($\_GET['qt']=='department'){

            $bas = 'department';

            $qs = $\_GET['qs'];

            $query = "SELECT school.school\_name AS 'school', department.department\_id AS 'department', program.program\_name AS 'program', plo.plo\_num as 'plo', co.co\_num as 'co', SUM(evaluation.obtained\_marks) as 'mark',  SUM(assessment.marks) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN faculty LEFT JOIN co ON assessment.co\_number = co.co\_num AND section.section\_id = co.section\_id LEFT JOIN plo on co.plo\_id = plo.plo\_id WHERE department.department\_id LIKE '%$qs%' GROUP BY department.department\_id, plo.plo\_num, co.co\_num";

            $res = $conn->query($query);

        }else if($\_GET['qt']=='program'){

            $bas = 'program';

            $qs = $\_GET['qs'];

            $qs = $\_GET['qs'];

            $query = "SELECT school.school\_name AS 'school', department.department\_id AS 'department', program.program\_name AS 'program', plo.plo\_num as 'plo', co.co\_num as 'co', SUM(evaluation.obtained\_marks) as 'mark',  SUM(assessment.marks) as 'total' FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN faculty LEFT JOIN co ON assessment.co\_number = co.co\_num AND section.section\_id = co.section\_id LEFT JOIN plo on co.plo\_id = plo.plo\_id WHERE program.program\_name LIKE '%$qs%' GROUP BY program.program\_id, plo.plo\_num, co.co\_num";

            $res = $conn->query($query);

        }

        $data = array();

        foreach($res as $r){

            $base = $r[$bas];

            if(!array\_key\_exists($base, $data)){

                $data[$base] = array();

                $data[$base]['p'] = array();

                $data[$base]['c'] = array();

            }

            if(!array\_key\_exists($r['plo'], $data[$base]['p'])){

                $data[$base]['p'][$r['plo']] = array();

                $data[$base]['p'][$r['plo']]['m'] = 0;

                $data[$base]['p'][$r['plo']]['t'] = 0;

            }

            $data[$base]['p'][$r['plo']]['m'] += $r['mark'];

            $data[$base]['p'][$r['plo']]['t'] += $r['total'];

            if(!array\_key\_exists($r['co'], $data[$base]['c'])){

                $data[$base]['c'][$r['co']] = array();

            }

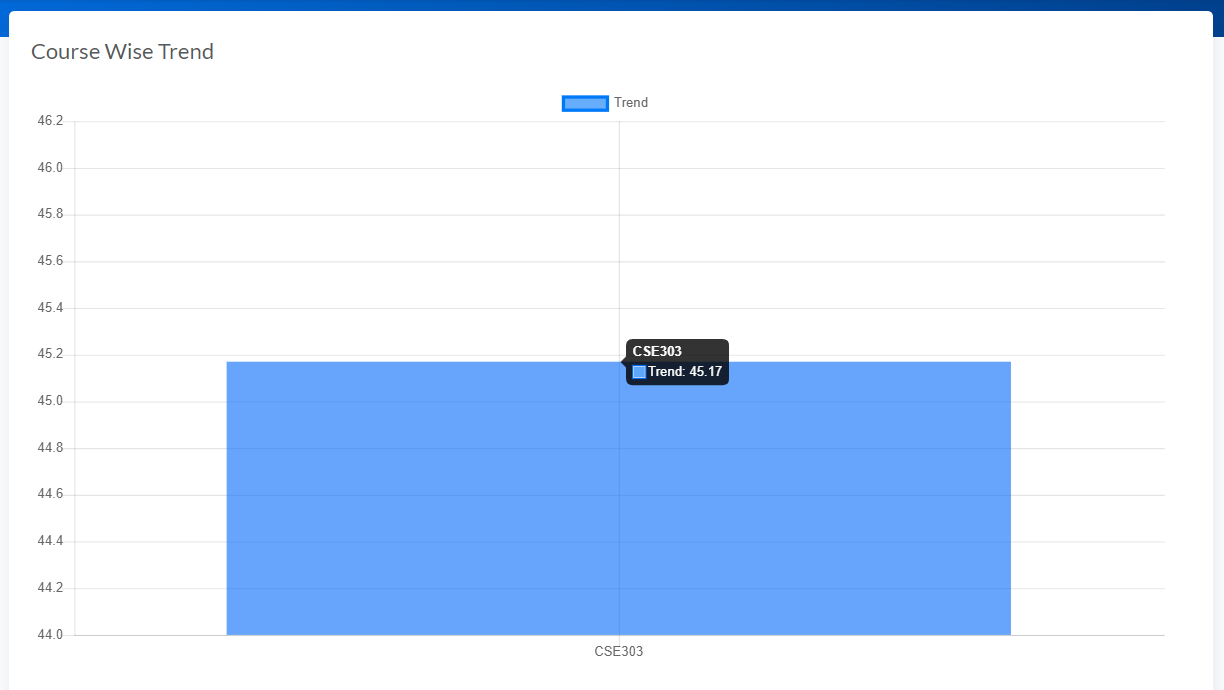
            if(!array\_key\_exists($r['plo'], $data[$base]['c'][$r['co']])){

                $data[$base]['c'][$r['co']][$r['plo']] = 0;

            }

            $data[$base]['c'][$r['co']][$r['plo']] += $r['mark'];

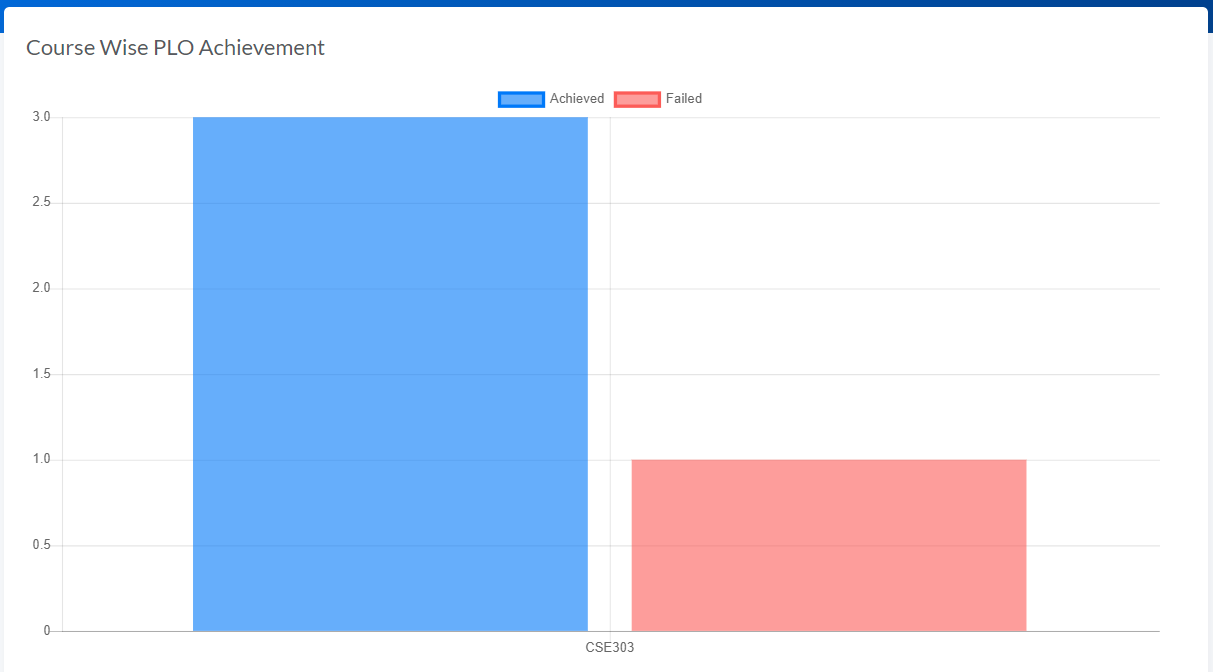
        }



1. COURSE WISE STUDENT PERFORMANCE TREND UNDER INSTRUCTOR

$query = "SELECT UPPER(course\_id) as 'course', SUM(marks) as 'marks', COUNT(marks)\*100 as 'student' FROM (SELECT school\_name, department\_id, program\_name,  semester, faculty, course\_id, no\_credits, student\_id, SUM(marks) as 'marks' FROM (SELECT school.school\_name, department.department\_id, program.program\_name, section.semester, CONCAT(faculty.fname, ' ', faculty.lname) as 'faculty', course.course\_id, course.no\_credits, enrollment.student\_id, IF(assessment.assessment\_name = 'final', (SUM(evaluation.obtained\_marks) / SUM(assessment.marks)) \* 40, (SUM(evaluation.obtained\_marks) / SUM(assessment.marks)) \* 30) as 'marks', assessment.assessment\_name FROM school NATURAL LEFT JOIN department NATURAL LEFT JOIN program NATURAL LEFT JOIN course NATURAL LEFT JOIN section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN faculty LEFT JOIN co ON assessment.co\_number = co.co\_num AND section.section\_id = co.section\_id LEFT JOIN plo on co.plo\_id = plo.plo\_id LEFT JOIN enrollment ON enrollment.enroll\_id = evaluation.enroll\_id WHERE section.faculty\_id = $id GROUP BY course.course\_id, enrollment.student\_id, assessment.assessment\_name, course.course\_id) as subQuery GROUP BY semester, student\_id, course\_id) as query2 GROUP BY course";

    $data = $conn->query($query);



1. STUDENT WISE PLO ACHIEVEMENT COMPARISON

$id = $\_SESSION['id'];

    $query = "SELECT course, COUNT(stat) as 'passed' FROM (SELECT UPPER(section.course\_id) as 'course', section.faculty\_id as 'faculty',  plo.plo\_num as 'plo', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE enrollment.student\_id = $id GROUP BY section.course\_id, enrollment.student\_id, section.course\_id, plo.plo\_num) as sQuery WHERE stat = 1 GROUP BY course ORDER BY course";

    $pls = $conn->query($query);

    $query = "SELECT course, COUNT(stat) as 'passed' FROM (SELECT UPPER(section.course\_id) as 'course', section.faculty\_id as 'faculty',  plo.plo\_num as 'plo', IF(SUM(evaluation.obtained\_marks)/SUM(assessment.marks)>=0.40, 1, 0) AS 'stat' FROM section NATURAL LEFT JOIN assessment NATURAL LEFT JOIN evaluation NATURAL LEFT JOIN enrollment LEFT JOIN co ON assessment.co\_number = co.co\_num AND assessment.section\_id = co.section\_id LEFT JOIN PLO ON co.plo\_id = plo.plo\_id WHERE enrollment.student\_id = $id GROUP BY section.course\_id, enrollment.student\_id, section.course\_id, plo.plo\_num) as sQuery WHERE stat = 0 GROUP BY course ORDER BY course";

    $mns = $conn->query($query);

# Chapter 5

CONCLUSION

1. Problems & Solution
2. Additional Features & Future Development
3. Conclusion & Recommendation

# Problems & Solutions

There were some problems that we have faced while creating the Student Performance Monitoring System. The major issue was we had lack of knowledge on the languages such as (PHP, CSS, JAVASCRIPT, HTML, MYSQL) that we must use while creating the system.

We came across this problem by seeking help from our faculty members through email or by doing problem solving sessions, who were experienced enough to guide us in creating the system.

We had issues on using Github which was a new platform for us. So we tried doing some internet searches and gaining information ourselves and using it properly.

# Additional Features & Future Development

The current proposed system does not include a process with which we can track a failed PLO that were previously achieved in a particular course. Moreover, the system can be made more secured by adding two-factor authentication so that only specific users can access the data stored in the application. An additional AI feature can be included to make it more ease in inputting the data by giving voice input rather than typing.

# Conclusion & Recommendation

This Student Performance Monitoring System would provide an insight about how learning might improve in a given program. We have created the system through which a user can automatically store and retrieve data that were previously done manually. It is more user friendly as gathering and collecting data manually was a tiresome task and it required more manpower. Now, these things can be done with ease. Primarily, we have focused on IUB as the organization for which we have done this project but the project has the potential of being useful to other universities as well.