STUDENT PERFORMANCE MONITORING SYSTEM

**Final Report**

**Section: 2**

**Faculty Name: Ms. Sadita Ahmed**

|  |  |
| --- | --- |
| NAME | ID |
| Md Navidul Islam | **1821952** |
| Tiasha Swarnaker | **1820874** |
| Abdullah Al-Noman | **1630081** |
| Sumiya Afrin | **1711012** |
| Sabrina Yasmin | **1821172** |
| Md. Faysal Chowdhury | **1721071** |

Table of Contents

[Chapter 1: Introduction 4](#_Toc62429141)

[Section 1.1: Background of the Organization 4](#_Toc62429142)

[Section 1.2: Background of the Project 4](#_Toc62429143)

[Section 1.3: Objectives of the Project 4](#_Toc62429144)

[Section 1.4: Scope of the Project 5](#_Toc62429145)

[Chapter 2: Requirement Analysis 6](#_Toc62429146)

[Section 2.1: Rich Picture (Existing) 6](#_Toc62429147)

[Section 2.2: Six Elements System Analysis (Existing) 7](#_Toc62429148)

[Section 2.3: Process Diagram (Existing) 13](#_Toc62429149)

[Section 2.4: Existing Problems & Analysis of the Problem 14](#_Toc62429150)

[Section 2.5: Rich Picture (Proposed) 16](#_Toc62429151)

[Section 2.6: Six Elemet System Analysis (Proposed) 17](#_Toc62429152)

[Section 2.7: Process Diagram (Proposed) 23](#_Toc62429153)

[Chapter 3: Logical System Design 24](#_Toc62429154)

[Section 3.1: Business Rules 24](#_Toc62429155)

[Section 3.2: ERD 26](#_Toc62429156)

[Section 3.3: ERD to Relations 27](#_Toc62429157)

[Section 3.4: Normalization 28](#_Toc62429158)

[Section 3.5: Data Dictionary 29](#_Toc62429159)

[Chapter 4: Physical System Design 34](#_Toc62429160)

[Section 4.1.1: Input Forms – Add User 34](#_Toc62429161)

[Purpose 34](#_Toc62429162)

[Related SQL Used 34](#_Toc62429163)

[Section 4.1.2: Input Forms – Add Program and PLO 34](#_Toc62429164)

[Purpose 34](#_Toc62429165)

[Related SQL Used 35](#_Toc62429166)

[Section 4.1.3: Input Forms – Add Course with CO 35](#_Toc62429167)

[Purpose 35](#_Toc62429168)

[Related SQL Used 35](#_Toc62429169)

[Section 4.1.4: Input Forms – Add Marks Individual 36](#_Toc62429170)

[Purpose 36](#_Toc62429171)

[Related SQL Used 36](#_Toc62429172)

[Section 4.1.5: Input Forms – Add Marks Mass 37](#_Toc62429173)

[Purpose 37](#_Toc62429174)

[Related SQL Used 37](#_Toc62429175)

[Section 4.2.1: Output Query and Reports – Higher Management Dashboard 39](#_Toc62429176)

[Purpose and Use 39](#_Toc62429177)

[Description along with SQL 39](#_Toc62429178)

[Section 4.2.2: Output Query and Reports – PLO Achievement 39](#_Toc62429179)

[Purpose and Use 39](#_Toc62429180)

[Description along with SQL 39](#_Toc62429181)

[Section 4.2.3: Output Query and Reports – Progress View 41](#_Toc62429182)

[Purpose and Use 41](#_Toc62429183)

[Description along with SQL 41](#_Toc62429184)

[Section 4.2.4: Output Query and Reports – Student Result 45](#_Toc62429185)

[Purpose and Use 45](#_Toc62429186)

[Description along with SQL 45](#_Toc62429187)

[Section 4.3: System Design Architecture 46](#_Toc62429188)

[Chapter 5: Conclusion 52](#_Toc62429189)

[Section 5.1: Problem & Solution 52](#_Toc62429190)

[Section 5.2: Additional Features and Future Development 52](#_Toc62429191)

[Section 5.3: Conclusion & Recommendation 52](#_Toc62429192)

[Contribution of Each Member 53](#_Toc62429193)

[References 54](#_Toc62429194)

[Figure 1: Existing System Rich Picture 6](#_Toc62429195)

[Figure 2: Existing System BPMN 13](#_Toc62429196)

[Figure 3: Proposed System Rich Picture 17](#_Toc62429197)

[Figure 4: Proposed System BPMN 23](#_Toc62429198)

[Figure 5: Entity Relationship Diagram 26](#_Toc62429199)

[Figure 6: Relational Schema Diagram 27](#_Toc62429200)

[Figure 7: Normalization Diagram 28](#_Toc62429201)

[Figure 8: Add User 46](#_Toc62429202)

[Figure 9: Add Program and PLO 47](#_Toc62429203)

[Figure 10: Add Course with CO 47](#_Toc62429204)

[Figure 11: Add Marks 48](#_Toc62429205)

[Figure 12: Higher Management Dashboard 48](#_Toc62429206)

[Figure 13: PLO Achievement View 1 49](#_Toc62429207)

[Figure 14: PLO Achievement View 2 49](#_Toc62429208)

[Figure 15: PLO Achievement View 3 50](#_Toc62429209)

[Figure 16: Progress View 1 50](#_Toc62429210)

[Figure 17: Progress View 2 51](#_Toc62429211)

[Figure 18: Student Result 51](#_Toc62429212)

# Chapter 1: Introduction

## Section 1.1: Background of the Organization

Independent University, Bangladesh (IUB) established in 1993 is the leading private university in Bangladesh with an explicit focus on Research and Global partnerships. IUBs mission is to achieve the goals of higher education and of sustainable economic growth in the country through a two-way relationship between community and university. It’s goals are to produce graduates of international standards within the local environment, with knowledge and relevant skills to provide leadership in enterprise, public service and welfare; encourage and support useful research; create knowledge; and provide further learning opportunities.

IUB currently have six academic schools:

* School of Business
* School of Engineering and Computer Science
* School of Environmental Sciences and Management
* School of Liberal Arts and Social Sciences
* School of Life Sciences
* School of Public Health

IUB is rapidly expanding its portfolio and is in the process of introducing Architecture and Biotechnology. The University curriculum and course of study are progressively revised and adjusted on the basis of their relevance to national needs and the global market demand. [1]

## Section 1.2: 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.

## Section 1.3: Objectives of the Project

* 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 analyze how student populations are learning inside of their programs so that the departments can focus more strategically on equity and success.

## Section 1.4: 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, UGC, IEB 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 UGC/IEB as well, and future prospects seems likely to also cater to all universities that conform to OBE regulations.

# Chapter 2: Requirement Analysis

## Section 2.1: Rich Picture (Existing)

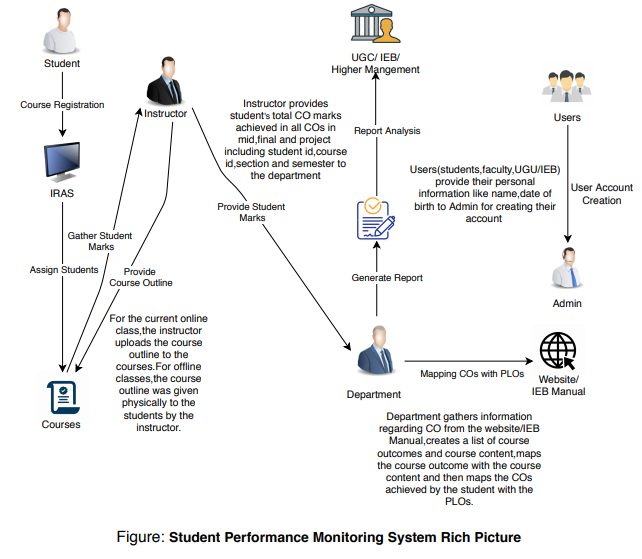


Figure 1: Existing System Rich Picture

In the existing business system, users create account for new users of the system by collecting user’s name, DOB and assigns a certain id and password. IEB/UGC sends accreditation manual with PLO’s defined to the department and then the department gathers CO from the corresponding PLO. Department creates a list of the course content if it is not available in the course outline and a list of course outcomes and maps the CO with the course content. The department then maps the received CO’s to specific PLO’s and hence mapping the CO’s to specific questions of mid-term, final & project. For the current online system, the instructor provides the course outline to online classroom and for the physical classes, course outline is printed and distributed among the students. The instructor receives CO’s to be achieved in a particular course and takes exams for assessing students in various course outcomes. Hence, the instructor converts the total marks and calculate total for all the CO’s. The instructor sends the CO’s marks to the department. The department calculates CO percentages and if the percentage is greater than or equal to 40% a student passes that specific PLO and hence otherwise fails. The corresponding CO’s are mapped against PLO’s and PLO achievements are recorded. In order to generate a report, the department retrieves the student information of PLO achievement from previously calculated OBE mark sheet and makes report based on the requirements of UGC/ IEB/ Higher management.

## Section 2.2: Six Elements System Analysis (Existing)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Process | System Roles | | | | | |
| Human | Non-Computing Hardware | Computing Hardware | Software | Database | Network & Communication |
| User Account Creation | Admin:  1. Can create new users for the system.  2. Collect user information such as name, DOB based on their specific role.  3. Assigns a certain id and password for each individual user type.  Users:  1. Provide personal information such as name, DOB to the admin for their specific role. | Pen & Papers:  1. The information sometimes are recorded manually. | Computer:  1. All related data is searched and stored using computer. | MS Excel:  1. All related information are stored. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Mapping COs with PLOs | Department:  1. IEB/UGC sends accreditation manual with PLO’s defined to the department  2. Department then gathers information regarding Course Outcome from the website/ IEB manual.  3. If the course content exist in the course outline, review the course content else create a list of the course content.  4. Creates a list of the course outcomes.  5. Mapping the CO with the course content.  6. Mapping the received CO’s to specific PLOs.  7. Mapping CO’s to specific questions of mid-term, final & project. | None. | Computer:  1. All related data is searched and stored using computer. | None. | Other Sources: 1. All related information are stored in the specific location. | All related data are searched through the internet. |
| Provide Course Outline | Instructor:  1. For the current online class, Course Outline is uploaded to the online classroom (Google Classroom).  2. For offline classes, Course Outline is printed and distributed among the students.  3. At the end of the semester, Course Outline is submitted to the department. | None. | Computer:  1. All related data is searched and stored using computer. | None. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Course Registration | Student:  1. On the reserved day for registration log in to IRAS with their id and password.  2. Go to registration page, load courses and select courses. Can only select courses of which prerequisites have been completed and the course capacity is not exceeded.  3. Click on ‘DONE’ button to complete the registration process.  4. A message prompts saying ‘’Registration Successfully Done”. | Pen & Paper:  1. Sometimes course registration is done manually. | Computer:  1. All related data is searched and stored using computer. | IRAS:  1. IRAS has an interface for doing registration. | MS SQL Server:  1. IRAS is integrated with MS SQL Server.  Other Sources: 1. All related information are stored in the specific location. | Internet:  IRAS is a web-based application and requires internet to access. |
| Gather Student Marks from Courses they have enrolled | Instructor:  1. Receives CO’s to be achieved in that particular course.  2. Takes exams such as mid-term, final & project to assess students in various course outcomes.  3. Gather marks for different assessment including student id, course id, section, semester.  4. Convert the total marks for mid-term, final.  5. Calculate total for all the CO’s. | Pen & Paper:  1. Written exams conducted by the faculty to evaluate the students. | Computer:  1. All related data such as student name, id is searched and stored using computer. | MS Excel: 1. All related information are stored. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Provide Student Marks | Instructor:  1. Provides students total CO marks achieved in all CO’s in mid-term, final & project to the department including student id, course id, section, semester.  Department:  1. Gather students marks in all CO’s achieved based on mid-term, final & project with student id, course id, section, semester.  2. Calculates CO percentages.  3. If greater than or equal to 40% CO’s have been achieved, a student passes that certain CO otherwise fails.  4. The corresponding CO’s are mapped against PLO’s and PLO achievements are recorded. | None. | Computer:  1. All related data is searched and stored using computer. | MS Excel:  1. All related information are stored. | Other Sources: 1. All related information are stored in the specific location. | All related data are provided through the internet. |
| Generate Report | Department:  1. Retrieve the student information of PLO achievement from previously calculated OBE marksheet.  2. Make reports after comparing results of multiple students.  3. Percentage of successfully passed or failed to achieve are calculated based on the total number of students. | None. | Computer:  1. All related data is searched and stored using computer. | MS Excel:  1. All related information are stored and calculated. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Report Analysis | UGC/ IEB/ Higher Management:  Does analysis by 1a. Viewing number of students passing or failing in a certain PLO.  1b. Viewing progression of students with charts and graphs. | Paper:  1. Paper is required for the printing purpose. | Printer:  1. Print the report based on users’ request. | None. | Other Sources: 1. All related information are stored in the specific location. | None. |

## Section 2.3: Process Diagram (Existing)

Graphical user interface

Description automatically generated

Figure 2: Existing System BPMN

## Section 2.4: Existing Problems & Analysis of the Problem

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Process Name | Stakeholders | Concerns (Isuues/ Problems) | Analysis (Reason of the Problems) | Proposed Solution |
| User Account Creation | Users | The users name might be similar. | There might be several persons with the same name. | In addition to the name and DOB the user provides for creating an account, they must also provide their contact numbers since contact numbers are unique and all these information must be stored in a system. |
| Mapping CO’s with PLO’s | Department | 1. Manually creates a list of the course content if it does not exist in course outline.  2. Manually creates a list of the course outcomes.  3. Manually mapped the CO’s with the course content.  4. Manually mapping CO’s to PLO’s. | All the processes for mapping CO’s with PLO’s are manually done by the department since there is no such system which can do the work without any manual task. | There can be a system which will gather the course content and course outcome from the department and then map the CO’s with the course content and hence mapping the received CO’s to specific PLO’s. |
| Course Registration | Student | 1. Course capacity gets filled up due to which students cannot register on the day they intended to.  2. Server gets jammed. | There is a specific capacity for any particular course and if the capacity exceeds then students cannot enroll in that course. During registration system cannot take huge load so it creates delay in the process. | Increasing the capacity of courses automatically as soon as it gets filled so that there is no delay in the registration process. There can be an information system where the load distribution is handled properly when there is extensive load.  . |
| Gather Student Marks from courses they have enrolled | Instructor | 1. Instructor gathers students marks from courses they have enrolled by taking assessments (mid, final, project) and converts the total marks for mid-term, final, project manually using MS Excel.  2. Using MS Excel calculate total for all the CO’s. | The process of converting marks for each student in each of the courses is very inefficient and takes a lot of time.  There is a chance of manual error as well during input or when calculating the marks. | A system can be introduced which can take total marks for mid-term, final as an input and convert it and hence automatically calculate total for all the CO’s. |
| Provide Student Marks | Instructor | .Student marks that are provided to the department by the instructor are calculated manually. CO percentages are calculated manually using MS Excel.  2. Manually checking if a student passed or failed.  3. Manually recording PLO achievement from the CO received. | There is no such specific system which can automatically calculate CO percentages and determine whether the COs and PLOs have been achieved or not. | A system can be introduced which can calculate CO percentages automatically with the marks provided as an input .The system will show by what percentages the PLOs and COs have been achieved and also all the COs and PLOs that the student failed to achieve. |
| Generate Report | Department | Manually retrieving information from MS Excel. | The OBE marksheet submitted by the instructor to the department for generating report is stored in MS Excel so manually retrieving every information from it might be very ineffcient and can cause manual errors. | This issue can be resolved by introducing a system in which the faculty provides the OBE marksheet and through that system the department can automatically generate a report. |

## Section 2.5: Rich Picture (Proposed)



Figure 3: Proposed System Rich Picture

The system gathers the CO data from the website / IEB manual. Then the system creates a list of the course content and a list of the course outcomes and hence mapping the CO’s with the course content. The CO’s received are then mapped to specific PLO. The instructor receives CO’s to be achieved in a particular course and takes exams for assessing the students in that course. The instructor than converts the total marks and calculates total for all CO’s. The CO marks are provided to the system and the system stores the marks. Based on the stored information, the system then creates reports for analysis by the UGC/ IEB/ Higher management.

## Section 2.6: Six Elemet System Analysis (Proposed)

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Process | System Roles | | | | | |
| Human | Non-Computing Hardware | Computing Hardware | Software | Database | Network & Communication |
| Mapping COs with PLOs | Department:  1. IEB/UGC sends accreditation manual with PLO’s defined to the department  2. Department then gathers information regarding Course Outcome from the website/ IEB manual.  3. If the course content exist in the course outline, review the course content else create a list of the course content.  4. Creates a list of the course outcomes.  5. Login to the system with user ID and password. Map the CO with the course content.  6. Map the received CO’s to specific PLOs.  7. Map CO’s to specific questions of mid-term, final & project. | None. | Computer:  1. All related data is searched and stored using computer. | None. | Other Sources: 1. All related information are stored in the specific location. | All related data are searched through the internet. |
| Course Registration | Student:  1. On the reserved day for registration log in to IRAS with their id and password.  2. Go to registration page, load courses and select courses.  Can only select courses of which prerequisites have been completed and the course capacity is not exceeded.  3. Click on ‘DONE’ button to complete the registration process.  4. A message prompts saying ‘’Registration Successfully Done”. | Pen & Paper:  1. Sometimes course registration is done manually. | Computer:  1. All related data is searched and stored using computer. | IRAS:  1. IRAS has an interface for doing registration. | MS SQL Server:  1. IRAS is integrated with MS SQL Server.  Other Sources: 1. All related information are stored in the specific location. | Internet:  IRAS is a web-based application and requires internet to access. |
| Gather Student Marks from Courses they have enrolled | Instructor:  1. Receives CO’s to be achieved in that particular course.  2. Takes exams such as mid-term, final & project to assess students in various course outcomes.  3. Gather marks for different assessment including student id, course id, section, semester.  4. Convert the total marks for mid-term, final.  5. Calculate total for all the CO’s. | Pen & Paper:  1. Written exams conducted by the faculty to evaluate the students. | Computer:  1. All related data such as student name, id is searched and stored using computer. | MS Excel: 1. All related information are stored. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Enter Student Marks | Instructor:  1. Login to the system with ID and password. Enter students total CO marks achieved in all CO’s in mid-term, final & project into the system including student id, course id, section, semester.  2. System stores students marks achieved in all CO’s based on mid-term, final & project with student id, course id, section, semester.  3. Calculates CO percentages.  4. If greater than or equal to 40% CO’s have been achieved, a student passes that certain CO otherwise fails.  5. The corresponding CO’s are mapped against PLO’s and PLO achievements are recorded. | None. | Computer:  1. All related data is searched and stored using computer. | MS Excel:  1. All related information are stored. | Other Sources: 1. All related information are stored in the specific location. | All related data are provided through the internet. |
| Generate Report | Department:  1. User will login to the system and retrieve the student information of PLO achievement from previously calculated OBE marksheet.  2. Generate reports after comparing results of multiple students.  3. Percentage of successfully passed or failed to achieve are calculated based on the total number of students. | None. | Computer:  1. All related data is searched and stored using computer. | MS Excel:  1. All related information are stored and calculated. | Other Sources: 1. All related information are stored in the specific location. | None. |
| Request to Generate Report | UGC/ IEB/ Higher Management:  Login to the system with user ID and password.  Asking to generate report for:  1a. Viewing number of students passing or failing in a certain PLO.  1b. Viewing progression of students with charts and graphs. | Paper:  1. Paper is required for the printing purpose. | Printer:  1. Print the report based on users’ request. | None. | Other Sources: 1. All related information are stored in the specific location. | None. |

## Section 2.7: Process Diagram (Proposed)

Graphical user interface, text

Description automatically generated

Figure 4: Proposed System BPMN

# Chapter 3: Logical System Design

## Section 3.1: Business Rules

1. IEB sends accreditation manual with PLO’s defined to the departments. Each department then gathers information from the IEB manual/ website.
2. 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.
3. Under the OBE model for each program there will be a set of program learning outcomes (PLO). A PLO has a PLOId, PLO name and description.
4. A Department has many students but a student must have exactly one department.Each of the departments has a name and a phone. Each department has a particular faculty as a head. Department offers many courses but a course is offered by exactly one department.
5. To evaluate the students in each course, courses have a set of course outcomes(COs) that are mapped with the PLOs of the degree program. A CO must be mapped with exactly one PLO. A PLO may be mapped with one or more COs. The COs are measured through different assessment techniques(e.g., quiz, mid, final, project, presentation). A CO has a COId, CO name and description.
6. 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 AssessmentID. Assessment has an assessment name (eg.Mid Q1, Mid Q2, Final Q1, Final Q3 etc), CourseID, 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.
7. Faculties evaluate the COs achieved and mapped PLOs achieved by each student in a course.An Evaluation is done by one or more faculties and a faculty must do atleast one evaluation. There is a date of evaluations done by a faculty. Each of the evaluations are identified uniquely using a StudentID and an AssessmentID. Evaluation also has the total marks obtained by a student in an assessment. An assessment has one or more evaluation but an evaluation is done for exactly one assessment. A student belongs to one or more evaluation but an evaluation belongs to exactly one student.
8. Accounts are maintained for two kinds of users of the system: students and faculties.An account has an ID, name (first name, last name), birth date, gender, email address, contact number, address.A student must log in to Iras for registration of course.
9. Faculties have academic qualifications (i.e. highest degree certificate so far), area/s of specialization, job position (e.g. Lecturer, Professor), salary. A student must have an enroll date and the total course completed. A faculty has exactly one department and a department has multiple faculties.
10. A Course have a CourseID that uniquely identifies the course. Course also has a CourseTitle. 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 atleast 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 CourseID 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 atleast one 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 it's sections along with the course are removed.

## Section 3.2: ERD



Figure 5: Entity Relationship Diagram

## Section 3.3: ERD to Relations



Figure 6: Relational Schema Diagram

## Section 3.4: Normalization



Figure 7: Normalization Diagram

## Section 3.5: Data Dictionary

tblaccount:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| naccountid | Number | 7 | This is the primary key of this relation. This contain the ID of students and instructors. Example: ‘1821952’ for student and ‘4321’ for faculty. |
| cfirstname | Text |  | This is the first name of the students or instructors. Example: ‘Navid’. |
| clastname | Text |  | This is the last name of the students or faculty. Example: ‘Islam’. |
| dbirthdate | Datetime | “dd/mm/yy” | This contains date of birth of the students or faculty. Example: ‘26/07/97’. |
| cgender | Text | 6 | This contains the gender of the students or instructors. Example: ‘Male’ or ‘Female’. |
| cemailaddress | Text |  | This is the email address of the students or faculty. |
| ncontactnumber | Number | 11 | This is the contact number of the students or faculty. Example: ‘01712345678’. |
| caddress | Text |  | This is the home address of the students or faculty. |

tblfaculty:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nfaccountid | Number | 4 | This is the primary key of this relation. This contain the ID of faculty. Example: ‘4321’. |
| cacademicqualifications | Text |  | This is the latest academic qualifications of the faculty. |
| cspecialization | Text |  | This is the specialization of the faculty. |
| cjobposition | Text | 9 | This is the job position of the faculty. Example: ‘Professor’ or ‘Lecturer’. |
| nsalary | Number |  | This is the monthly salary of the faculty. |

tbldepartment:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| cdepartmentname | Text |  | This is the primary key of this relation. This contains the department name. |
| nfacultyid | Number | 4 | This contains the name of the department head. |
| nphonenumber | Number | 11 | This contains the phone number of the department. |

tblsection:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccourseid | Text | 7 | This is the composite key of this relation. This contains the course id. Example: ‘CSE303’. |
| nsectionnumber | Number | 2 | This is the composite key of this relation. This contains the section number of a particular course in a particular semester. |
| csemester | Text | 6 | This is the composite key of this relation. This contains the semester name. Example: ‘Spring’, ‘Summer’, ‘Autumn’. |
| nschedule | Text |  | This is the schedule of a section which is the timing of a course. |
| nmaximumcapacity | Number | 2 | This is the maximum capacity a section can hold. |

tblstudentsection:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nsaccountid | Number | 7 | This is the composite key of this relation. This contains the account id of the students. Example: ‘1821952’. |
| ccourseid | Text | 7 | This is the composite key of this relation. This contains the course id. Example: ‘CSE303’. |
| nsectionnumber | Number | 2 | This is the composite key of this relation. This contains the section number of a particular course in a particular semester. |
| csemester | Text | 6 | This is the composite key of this relation. This contains the semester name. Example: ‘Spring’, ‘Summer’, ‘Autumn’. |
| dregistrationdate | Datetime | “dd/mm/yy” | This contains registration date of the students in a particular course. |

tblfacultycourse:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nfaccountid | Number | 4 | This is the composite key of this relation. This contains the account id of the instructors. Example: ‘4321’. |
| ccourseid | Text | 7 | This is the composite key of this relation. This contains the course id. Example: ‘CSE303’. |
| cteachingdays | Text | 9 | This contains the days a particular faculty teaches. |
| nteachingtime | Number |  | This contains the teaching time of a faculty. |

tblcourseoutcome:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccourseid | Text | 7 | This is the composite key of this relation. This contains the course id of a particular course. |
| ccourseoutcome | Text |  | This is the composite key of this relation. This contains the course outcome from a particular course. |

tblstudent:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nsaccountid | Number | 7 | This is the primary key of this relation. This contains the account id of the students. Example: ‘1821952’. |
| denrolldate | Datetime | “dd/mm/yy” | This contains the enroll date of the students. |
| ntotalcoursecompleted | Number | 2 | This contains the total course completed by the students. |
| cdepartmentname | Text |  | This is a foreign key from Table DEPARTMENT. |

tblcourse:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccourseid | Text | 7 | This is the primary key of this relation. This contains the course id. Example: ‘CSE303’. |
| ccoursetitle | Text |  | This contains the course title of a particular course. Example: ‘Database Management’. |

tblprogram:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nprogramid | Number | 1 | This is the primary key of this relation. This contains the program id. |
| ccprogramname | Text |  | This contains the program name. |
| cdepartmentname | Text |  | This is a foreign key from Table DEPARTMENT. |

tplo:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| cploid | Number | 1 | This is the primary key of this relation. This contains the plo id. |
| cploname | Text |  | This contains the plo name of a particular course. Example: ‘Database Management’. |
| cplodescription | Text |  | This is the plo description. |
| nprogramid | Number | 1 | This is a foreign key from Table PROGRAM. |

tco:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccoid | Number | 1 | This is the primary key of this relation. This contains the co id. |
| cconame | Text |  | This contains the co name. |
| ccodescription | Text |  | This contains the description of co. |
| ccourseid | Number | 7 | This is a foreign key from Table COURSE. |
| cploid | Number | 1 | This is a foreign key from Table PLO. |

tcourseprerequisite:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| ccourseid | Text | 7 | This is the composite key of this relation. This contains the course id. Example: ‘CSE303’. |
| ccprerequisitecoureseid | Text | 7 | This is the composite key of this relation. This contains the course id of the prerequisite course. |

tassessment:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nassessmentid | Number |  | This is the primary key of this relation. This contains the assessment id. |
| ccourseid | Text | 7 | This is a foreign key from Table COURSE. |
| nsectionnumber | Number | 2 | This is a foreign key from Table SECTION. |
| ncoid | Number | 1 | This is a foreign key from Table CO. |
| nassessmenttype | Number | 2 | This contains the assessment marks. |
| ntotalmarksachievable | Number | 3 | This contains the total marks achievable. |

tevaluation:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nstudentid | Number | 7 | This is the composite key of this relation. This contains the student id. |
| nassessmentid | Number |  | This is the composite key of this relation. This contains the assessment id. |
| ntotalmarksobtained | Number | 3 | This contains the total marks achieved. |

tevaluationfaculty:

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Data Type | Size | Remark |
| nfaccountid | Number | 4 | This is the composite key of this relation. This contains the faculty id. |
| nstudentid | Number | 7 | This is the composite key of this relation. This contains the student id. |
| nassessmentid | Number |  | This is the composite key of this relation. This contains the assessment id. |
| devaluationdate | Datetime | “dd/mm/yy” | This contains the evaluation date. |

# Chapter 4: Physical System Design

## Section 4.1.1: Input Forms – Add User

### Purpose

The purpose of adding user is that these users will be able to use the system. So, without adding any kinds of users this system would not exist as this system depends on storing and retrieving data.

### Related SQL Used

1. **<?php**
2. require 'mysql.php';
3. *// geting post requests*
4. $id = $\_POST['id'];
5. $fName = $\_POST['fName'];
6. $lName = $\_POST['lName'];
7. $prog = $\_POST['prog'];
8. $email = $\_POST['email'];
9. $password = $\_POST['password'];
10. $role = [" target="\_blank" title="http://www.php.net/strtolower">">http://www.php.net/strtolower">](%3ca%20href=)strtolower($\_POST['role']);
12. $store = "INSERT INTO user (id, first\_name, last\_name, program\_id, email, password, role) VALUES
13. ('**$id**', '**$fName**', '**$lName**', '**$prog**', '**$email**', '**$password**' , '**$role**')";
15. if($mysql->query($store)){
16. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-user.php");
17. }else{
18. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-user.php");
19. }

22. **?>**

## Section 4.1.2: Input Forms – Add Program and PLO

### Purpose

Since this system tracks PLO achievement of students of any courses in any program, so we must input the PLO data with their corresponding program. The system cannot collect the data automatically, so manually inputting the programs with their PLO’s are done using input form.

### Related SQL Used

1. **<?php**
2. require 'mysql.php';
4. $id = $\_POST['program\_id'];
5. $program\_name = $\_POST['program\_name'];
6. $school = $\_POST['school'];
7. $sql = "INSERT INTO program(id, program\_name, school) VALUES ('**$id**', '**$program\_name**', '**$school**')";
8. *//echo $sql;*
9. if($mysql->query($sql) == **FALSE**){
10. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-program.php?failed=1");
11. }
13. $i = 1;
15. while([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_POST['title'.$i])){
16. $name = $\_POST['title'.$i];
17. $sql = "INSERT INTO plo(program\_id, plo\_no, plo\_name) VALUES
18. ('**$id**', **$i**, '**$name**')";
19. if($mysql->query($sql) == **FALSE**){
20. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-program.php?failed=1");
21. }
22. $i++;
23. }
25. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-program.php?success=1");
27. **?>**

## Section 4.1.3: Input Forms – Add Course with CO

### Purpose

Each individual course must be added to the system with their respective CO’s. In order to do that, an input form must be used to give inputs to the system.

### Related SQL Used

1. **<?php**
2. require 'mysql.php';
4. $id = $\_POST['course\_id'];
5. $program\_id = $\_POST['program\_id'];
6. $credit = $\_POST['credit'];
7. $total\_co = $\_POST['total-co'];
8. $title = $\_POST['course\_title'];
10. $sql = "INSERT INTO course (id, program\_id, title, credit, total\_co) VALUES
11. ('**$id**', '**$program\_id**', '**$title**', **$credit**, **$total\_co**)";
13. if($mysql->query($sql) == **FALSE**){
14. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-course.php?failed=1");
15. }
17. for($i=1; $i<=15; $i++){
18. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_POST["plo-co".$i])){
19. $sql = "SELECT sl FROM plo WHERE program\_id = '**$program\_id**' AND plo\_no = **$i**";
20. $plo\_id = $mysql->query($sql)->fetch\_assoc()['sl'];
21. $data = $\_POST["plo-co".$i];
22. $field = ""; $val ="";
23. foreach($data as $co){
24. $field .= 'co'. $co . ', ';
25. $val .= '1, ';
26. }
27. $sql = "INSERT INTO co (course\_id, plo\_id, ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($field, 0, -2).") VALUES ('**$id**', **$plo\_id**, ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($val,0, -2).")";
28. if($mysql->query($sql) == **FALSE**){
29. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-course.php?failed=1");
30. }
31. }
32. }
34. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../admin/add-course.php?success=1");
36. **?>**

## Section 4.1.4: Input Forms – Add Marks Individual

### Purpose

This input form requires marks entry process with which a students’ data can be entered based on a course in a particular semester. This is then stored in the database which can be retrieved when required. Marks entry form can input all the assessments of a particular student.

### Related SQL Used

1. **<?php**
2. require 'mysql.php';
4. $student\_id = $\_POST['student\_id'];
5. $course\_id = $\_POST['course\_id'];
6. $exam\_name = $\_POST['exam\_name'];
7. $semester = $\_POST['semester'];
8. $section = $\_POST['section'];
10. $field = "";
11. $val = "";
12. $i=1;
14. while([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_POST['co'.$i])){
15. $field .= 'q'.$i.'\_mark, ' . 'q'.$i.'\_co, ' . 'q'.$i.'\_max, ';
16. $val .= $\_POST['mark'.$i].', '.$\_POST['co'.$i].', '.$\_POST['max'.$i].', ';
17. $i++;
18. }
20. $sql = "INSERT INTO marks(student\_id, course\_id, exam\_name, semester, section, ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($field, 0, -2).") VALUES
21. ('**$student\_id**', '**$course\_id**', '**$exam\_name**', '**$semester**', '**$section**', ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($val, 0, -2).")";
23. echo $sql . '<br>';
24. $mysql->query($sql);

27. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../faculty/entry-marks.php?success=1");
29. **?>**

## Section 4.1.5: Input Forms – Add Marks Mass

### Purpose

This input form requires marks entry process with which many students data can be entered on any course in any semester. This is then stored in the database which can be retrieved when required. Marks entry form can input all the assessments of all the students using a .csv format file.

### Related SQL Used

1. **<?php**
2. require 'mysql.php';
3. $course\_id = $\_POST['course\_id'];
4. $exam\_name = $\_POST['exam\_name'];
5. $semester = $\_POST['semester'];
6. $section = $\_POST['section'];
8. $file = [" target="\_blank" title="http://www.php.net/fopen">">http://www.php.net/fopen">](%3ca%20href=)fopen($\_FILES['file']['tmp\_name'], "r");
10. [" target="\_blank" title="http://www.php.net/fgetcsv">">http://www.php.net/fgetcsv">](%3ca%20href=)fgetcsv($file,200); *// dumping header*
11. $co = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
12. $f = 1; $i=1;
13. foreach([" target="\_blank" title="http://www.php.net/fgetcsv">">http://www.php.net/fgetcsv">](%3ca%20href=)fgetcsv($file, 200) as $c){
14. if($f == 1){
15. $f = 0;
16. continue;
17. }
18. $co[$i] = $c;
19. $i++;
20. }
21. $max = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
22. $f = 1; $i=1;
23. foreach([" target="\_blank" title="http://www.php.net/fgetcsv">">http://www.php.net/fgetcsv">](%3ca%20href=)fgetcsv($file, 200) as $m){
24. if($f == 1){
25. $f = 0;
26. continue;
27. }
28. $max[$i] = $m;
29. $i++;
30. }
32. while($marks = [" target="\_blank" title="http://www.php.net/fgetcsv">">http://www.php.net/fgetcsv">](%3ca%20href=)fgetcsv($file, 200)){
34. $student\_id;
35. $field = "";
36. $val = "";
37. $f = 1; $i=1;
39. foreach($marks as $m){
40. if($f==1){
41. $student\_id = $m;
42. $f=0;
43. continue;
44. }
45. $field .= 'q'.$i.'\_mark, ' . 'q'.$i.'\_co, ' . 'q'.$i.'\_max, ';
46. $val .= $m.', '.$co[$i].', '.$max[$i].', ';
47. $i++;
48. }
50. $sql = "INSERT INTO marks(student\_id, course\_id, exam\_name, semester, section, ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($field, 0, -2).") VALUES
51. ('**$student\_id**', '**$course\_id**', '**$exam\_name**', '**$semester**', '**$section**', ".[" target="\_blank" title="http://www.php.net/substr">">http://www.php.net/substr">](%3ca%20href=)substr($val, 0, -2).")";
53. $mysql->query($sql);
54. }
56. [" target="\_blank" title="http://www.php.net/header">">http://www.php.net/header">](%3ca%20href=)header("Location: ../faculty/entry-marks-mass.php?success=1");
57. **?>**

## Section 4.2.1: Output Query and Reports – Higher Management Dashboard

### Purpose and Use

It gives the overall summary of the system which includes the total number of courses that uses OBE model, total students who are evaluated with OBE model, number of faculty evaluated using OBD, total number of PLO set in the program. It can be used to gather information about an institution by the higher management.

### Description along with SQL

1. **<?php**
2. require 'php/mysql.php';
3. $sql = "SELECT \* FROM course";
4. $course = $mysql->query($sql)->num\_rows;
5. $sql = "SELECT \* FROM user WHERE role='student'";
6. $student = $mysql->query($sql)->num\_rows;
7. $sql = "SELECT \* FROM user WHERE role='faculty'";
8. $faculty = $mysql->query($sql)->num\_rows;
9. $sql = "SELECT \* FROM plo";
10. $plo = $mysql->query($sql)->num\_rows;
11. **?>**

## Section 4.2.2: Output Query and Reports – PLO Achievement

### Purpose and Use

It is used to show student wise PLO analysis which includes PLO total percentage score for each PLO calculated from the scores achieved in each CO associated with the corresponding PLO among all the courses the student has done so far. Upon entering a student id course wise PLO analysis can be viewed. Also, a tabular view of student wise PLO achievement can be viewed.

### Description along with SQL

1. **<?php**
2. require 'mysql.php';
3. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_GET['id'])){
4. $id = $\_GET['id'];
5. $sql = "SELECT \* FROM marks WHERE student\_id = **$id**";
6. $sMarks = $mysql->query($sql);
8. *//course based total co marks*
9. $cMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
10. $cTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
11. foreach($sMarks as $marks){
12. $course = $marks['course\_id'];
13. for($i=1; $i<=10; $i++){
14. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($marks["q".$i."\_co"]) &&$marks["q".$i."\_co"]!=0){
15. $co = $marks["q".$i."\_co"];
16. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($cMarks[$course][$co])){
17. $cMarks[$course][$co] += $marks["q".$i."\_mark"];
18. $cTotal[$course][$co] += $marks["q".$i."\_max"];
19. }else{
20. $cMarks[$course][$co] = $marks["q".$i."\_mark"];
21. $cTotal[$course][$co] = $marks["q".$i."\_max"];
22. }
23. }
24. }
25. }
27. *//course base total plo marks*
28. $pMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
29. $pTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
30. foreach($cMarks as $c => $v){
32. $sql = "SELECT \* FROM co WHERE course\_id = '**$c**'";
33. $plos = $mysql->query($sql);
34. foreach($plos as $plo){
35. $pId = $plo['plo\_id'];
36. for($i=1; $i<=10; $i++){
37. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($plo["co".$i]) && $plo["co".$i]==1){
38. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($pMakrs[$c][$pId])){
39. $pMarks[$c][$pId] += $cMarks[$c][$i];
40. $pTotal[$c][$pId] += $cTotal[$c][$i];
41. }else{
42. $pMarks[$c][$pId] = $cMarks[$c][$i];
43. $pTotal[$c][$pId] = $cTotal[$c][$i];
44. }
45. }
46. }
47. }
48. }
50. *//total marks in plo*
51. $pfMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
52. $pfTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
53. foreach($pMarks as $c => $v){
54. foreach($v as $i => $j){
55. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($pfMarks[$i])){
56. $pfMarks[$i] += $j;
57. $pfTotal[$i] += $pTotal[$c][$i];
58. }else{
59. $pfMarks[$i] = $j;
60. $pfTotal[$i] = $pTotal[$c][$i];
61. }
62. }
63. }
65. *//student info*
66. $sql = "SELECT \* FROM user WHERE id = **$id**";
67. $student = $mysql->query($sql)->fetch\_assoc();
68. *//total plo*
69. $sql = "SELECT \* FROM plo WHERE program\_id = '".$student['program\_id']."'";
70. $ploNum = $mysql->query($sql)->num\_rows;
72. $color = ["", "#1FE7C4", "#E45C17", "#06B97B", "#8CE026", "#E1CCFF", "#5BA2CC", "#0A2E82","#957107", "#80CF18"];
73. }
74. **?>**

## Section 4.2.3: Output Query and Reports – Progress View

### Purpose and Use

It contains student and course progress views. For a given student, it shows the count of PLO’s expected to be achieved and the counts actually achieved at the end of each semester. Upon selecting a certain course, it shows the number of students in that particular course with the percentages of CO achieved of failed.

### Description along with SQL

1. **<?php**
2. require 'mysql.php';
3. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_GET['id'])){
4. $id = $\_GET['id'];
5. $sql = "SELECT DISTINCT semester FROM marks WHERE student\_id = **$id**";
6. $sems =  $mysql->query($sql);
8. $ploProg = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
9. foreach($sems as $sem){
10. $res = seeker($sem['semester'], $id);
11. $ploProg[$sem['semester']]['total'] = $res['p'];
12. $ploProg[$sem['semester']]['com'] = $res['t'];
13. }
15. }if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($\_GET['c'])){
16. $crs = $\_GET['c'];
17. $sql = "SELECT DISTINCT student\_id FROM marks WHERE course\_id = '**$crs**'";
18. $uList = $mysql->query($sql);
19. $totalS = $uList->num\_rows;
20. $report = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
21. foreach($uList as $u){
22. $usr = $u['student\_id'];
23. $ret = seeker2($crs, $usr);
24. foreach($ret["co"] as $i => $j){
25. if($j==1){
26. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($report["co"][$i])){
27. $report["co"][$i]++;
28. }else{
29. $report["co"][$i] = 1;
30. }
31. }
32. }
33. foreach($ret["plo"] as $i => $j){
34. if($j==1){
35. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($report["plo"][$i])){
36. $report["plo"][$i]++;
37. }else{
38. $report["plo"][$i] = 1;
39. }
40. }
41. }
42. }
43. [" target="\_blank" title="http://www.php.net/ksort">">http://www.php.net/ksort">](%3ca%20href=)ksort($report["co"]);
44. [" target="\_blank" title="http://www.php.net/ksort">">http://www.php.net/ksort">](%3ca%20href=)ksort($report["plo"]);
45. }
47. **function** seeker($sem, $uid){
48. require 'mysql.php';
49. $sql;
50. if($sem!="null"){
51. $sql = "SELECT \* FROM marks WHERE student\_id = **$uid** AND semester = '**$sem**'";
52. }else{
53. $sql = "SELECT \* FROM marks WHERE student\_id = **$uid**";
54. }
56. $sMarks = $mysql->query($sql);
58. $cMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
59. $cTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
61. foreach($sMarks as $marks){
62. $course = $marks['course\_id'];
63. for($i=1; $i<=10; $i++){
64. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($marks["q".$i."\_co"]) &&$marks["q".$i."\_co"]!=0){
65. $co = $marks["q".$i."\_co"];
66. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($cMarks[$course][$co])){
67. $cMarks[$course][$co] += $marks["q".$i."\_mark"];
68. $cTotal[$course][$co] += $marks["q".$i."\_max"];
69. }else{
70. $cMarks[$course][$co] = $marks["q".$i."\_mark"];
71. $cTotal[$course][$co] = $marks["q".$i."\_max"];
72. }
73. }
74. }
75. }
77. $pMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
78. $pTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
79. foreach($cMarks as $c => $v){
80. $sql = "SELECT \* FROM co WHERE course\_id = '**$c**'";
81. $plos = $mysql->query($sql);
82. foreach($plos as $plo){
83. $pId = $plo['plo\_id'];
84. for($i=1; $i<=10; $i++){
85. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($plo["co".$i]) && $plo["co".$i]==1){
86. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($pMakrs[$c][$pId])){
87. $pMarks[$c][$pId] += $cMarks[$c][$i];
88. $pTotal[$c][$pId] += $cTotal[$c][$i];
89. }else{
90. $pMarks[$c][$pId] = $cMarks[$c][$i];
91. $pTotal[$c][$pId] = $cTotal[$c][$i];
92. }
93. }
94. }
95. }
96. }
98. $res = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
99. $res['t'] = 0;

102. $pTrack = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
104. foreach($pMarks as $c => $v){
105. foreach($v as $i => $j){
106. $pTrack[$i]=1;
107. if($j \* 100 / $pTotal[$c][$i]>=40){
108. $res['t']++;
109. }
110. }
111. }
112. $res['p'] = [" target="\_blank" title="http://www.php.net/count">">http://www.php.net/count">](%3ca%20href=)count($pTrack);
113. return $res;
114. }
116. **function** seeker2($crs, $uid){
117. require 'mysql.php';
118. $sql = "SELECT \* FROM marks WHERE student\_id = **$uid** AND course\_id = '**$crs**'";
120. $sMarks = $mysql->query($sql);
122. $cMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
123. $cTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
125. foreach($sMarks as $marks){
126. $course = $marks['course\_id'];
127. for($i=1; $i<=10; $i++){
128. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($marks["q".$i."\_co"]) &&$marks["q".$i."\_co"]!=0){
129. $co = $marks["q".$i."\_co"];
130. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($cMarks[$course][$co])){
131. $cMarks[$course][$co] += $marks["q".$i."\_mark"];
132. $cTotal[$course][$co] += $marks["q".$i."\_max"];
133. }else{
134. $cMarks[$course][$co] = $marks["q".$i."\_mark"];
135. $cTotal[$course][$co] = $marks["q".$i."\_max"];
136. }
137. }
138. }
139. }
141. $pMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
142. $pTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
143. foreach($cMarks as $c => $v){
144. $sql = "SELECT \* FROM co WHERE course\_id = '**$c**'";
145. $plos = $mysql->query($sql);
146. foreach($plos as $plo){
147. $pId = $plo['plo\_id'];
148. for($i=1; $i<=10; $i++){
149. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($plo["co".$i]) && $plo["co".$i]==1){
150. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($pMakrs[$c][$pId])){
151. $pMarks[$c][$pId] += $cMarks[$c][$i];
152. $pTotal[$c][$pId] += $cTotal[$c][$i];
153. }else{
154. $pMarks[$c][$pId] = $cMarks[$c][$i];
155. $pTotal[$c][$pId] = $cTotal[$c][$i];
156. }
157. }
158. }
159. }
160. }
162. $stats = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
163. foreach($cMarks as $c => $v){
164. foreach($v as $i => $m){
165. if(($m \* 100 / $cTotal[$c][$i]) >=40 ){
166. $stats["co"][$i] = 1;
167. }else{
168. $stats["co"][$i] = 0;
169. }
170. }
171. }
173. foreach($pMarks as $p => $v){
174. foreach($v as $i => $m){
175. if(($m \* 100 / $pTotal[$p][$i]) >=40 ){
176. $stats["plo"][$i] = 1;
177. }else{
178. $stats["plo"][$i] = 0;
179. }
180. }
181. }
183. return $stats;
184. }
186. **?>**

## Section 4.2.4: Output Query and Reports – Student Result

### Purpose and Use

It is used to show the students result of PLO achievement in a pie chart for all the courses that student has completed upon entering the students id.

### Description along with SQL

1. **<?php**
2. require '../php/mysql.php';
4. $color = ["", "#1FE7C4", "#E45C17", "#06B97B", "#8CE026", "#E1CCFF", "#5BA2CC", "#0A2E82", "#957107","#80CF18"];
5. *// session\_start();*
6. $id = $\_SESSION['user\_id'];
7. $sql = "SELECT \* FROM user WHERE id = **$id**";
8. $uInfo = $mysql->query($sql)->fetch\_assoc();

11. $sql = "SELECT \* FROM plo WHERE program\_id = '".$uInfo['program\_id']."'";
12. $totalPlo = $mysql->query($sql)->num\_rows;
14. $sql = "SELECT \* FROM marks WHERE student\_id = **$id**";
15. $sMarks = $mysql->query($sql);
17. *//course based total co marks*
18. $cMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
19. $cTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
20. foreach($sMarks as $marks){
21. $course = $marks['course\_id'];
22. for($i=1; $i<=10; $i++){
23. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($marks["q".$i."\_co"]) &&$marks["q".$i."\_co"]!=0){
24. $co = $marks["q".$i."\_co"];
25. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($cMarks[$course][$co])){
26. $cMarks[$course][$co] += $marks["q".$i."\_mark"];
27. $cTotal[$course][$co] += $marks["q".$i."\_max"];
28. }else{
29. $cMarks[$course][$co] = $marks["q".$i."\_mark"];
30. $cTotal[$course][$co] = $marks["q".$i."\_max"];
31. }
32. }
33. }
34. }
36. $pMarks = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
37. $pTotal = [" target="\_blank" title="http://www.php.net/array">">http://www.php.net/array">](%3ca%20href=)array();
38. foreach($cMarks as $c => $v){
40. $sql = "SELECT \* FROM co WHERE course\_id = '**$c**'";
41. $plos = $mysql->query($sql);
42. foreach($plos as $plo){
43. $pId = $plo['plo\_id'];
44. for($i=1; $i<=10; $i++){
45. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($plo["co".$i]) && $plo["co".$i]==1){
46. if([" target="\_blank" title="http://www.php.net/isset">">http://www.php.net/isset">](%3ca%20href=)isset($pMakrs[$c][$pId])){
47. $pMarks[$c][$pId] += $cMarks[$c][$i];
48. $pTotal[$c][$pId] += $cTotal[$c][$i];
49. }else{
50. $pMarks[$c][$pId] = $cMarks[$c][$i];
51. $pTotal[$c][$pId] = $cTotal[$c][$i];
52. }
53. }
54. }
55. }
56. }
58. **?>**

## Section 4.3: System Design Architecture

Graphical user interface, text, application, email

Description automatically generated

Figure 8: Add User

Graphical user interface, text, application, email

Description automatically generated

Figure 9: Add Program and PLO

Graphical user interface, application, table

Description automatically generated

Figure 10: Add Course with CO

Graphical user interface, application

Description automatically generated

Figure 11: Add Marks

Graphical user interface, application, website

Description automatically generated

Figure 12: Higher Management Dashboard

Chart, bar chart

Description automatically generated

Figure 13: PLO Achievement View 1

Chart

Description automatically generated

Figure 14: PLO Achievement View 2

Graphical user interface, application

Description automatically generated

Figure 15: PLO Achievement View 3

A screenshot of a computer

Description automatically generated

Figure 16: Progress View 1

Table

Description automatically generated

Figure 17: Progress View 2

Graphical user interface, application

Description automatically generated

Figure 18: Student Result

# Chapter 5: Conclusion

## Section 5.1: Problem & Solution

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, SQL) 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.

## Section 5.2: Additional Features and 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.

## Section 5.3: 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.

# Contribution of Each Member

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | Md Navidul Islam | Tiasha Swarnaker | Sabrina Yasmin | Sumiya Afrin | Md Faysal Chowdhury | | Abdullah Al-Noman | |
| Cover Page | ü |  |  |  | |  | |  |
| Table of Contents | ü |  |  |  | |  | |  |
| List of Figures | ü |  |  |  | |  | |  |
| Background of the Organization |  |  | ü |  | |  | |  |
| Background of the Project |  | ü |  |  | |  | |  |
| Objectives of the Project |  |  | ü |  | |  | |  |
| Scope of the Project |  | ü |  |  | |  | |  |
| Existing Rich Picture | ü | ü |  |  | |  | |  |
| Existing Six Element System Analysis | ü |  |  |  | |  | |  |
| Existing BPMN | ü | ü |  |  | |  | |  |
| Existing Problems and Analysis of the Problems | ü |  |  |  | |  | |  |
| Proposed Rich Picture | ü |  |  |  | |  | |  |
| Proposed Six Element System Analysis | ü |  |  |  | |  | |  |
| Proposed BPMN | ü |  |  |  | |  | |  |
| Business Rules | ü | ü | ü |  | |  | |  |
| ERD |  |  | ü |  | |  | |  |
| Relational Schema |  | ü | ü |  | |  | |  |
| Normalization |  |  | ü |  | |  | |  |
| Data Dictionary | ü |  |  |  | |  | |  |
| Input Forms - Purpose | ü |  |  |  | |  | |  |
| Output Query and Reports – Purpose and Use | ü |  |  |  | |  | |  |
| Input Forms – Related SQL Used |  |  |  | ü | | ü | |  |
| Output Query and Reports – Description along with SQL | ü |  |  | ü | |  | |  |
| System Design Architecture |  | ü |  |  | |  | |  |
| Problem & Solution | ü |  |  |  | |  | |  |
| Additional Features and Future Development | ü |  |  |  | |  | |  |
| Conclusion and Recommendation | ü |  |  |  | |  | |  |

# References

|  |  |
| --- | --- |
| [1] | “IUB at a glance,” [Online]. |
| [2] | “W3Schools,” [Online]. Available: https://www.w3schools.com/default.asp. |
| [3] | J. Duckett, HTML & CSS: Designing and Building Web Sites and Javascript & jQuery: Interactive Front-End Development (1st Edition), Wiley, 2014. |
| [4] | L. Welling and L. Thomson, PHP and MySQL Web Development (4th Edition), Indianapolis, IN: Sams, 2009. |