

**STUDENT PERFORMANCE MONITORING SYSTEM**

**DATABASE MANAGEMENT**

**GROUP-4**

**GROUP MEMBERS**

**Misbahur Rashid – 1721911**

**Rafid Al Ahsan – 1722006**

**Sadia Afroz Alma - 1730407**

**Puja Bhowmik – 1730791**

**Elan Md Taseen – 1831050**

**Md.Musfiqur Rahaman – 1721684**

**Md.Sakimuzzaman – 1721527**

**CONTENTS**

**INTRODUCTION**

Background of the Organization --------------------------------------------------------------------------------------- 05

Background of the Project ----------------------------------------------------------------------------------------------- 05

Objective of the Project --------------------------------------------------------------------------------------------------- 05

Scope of the Project ------------------------------------------------------------------------------------------------------- 06

**REQUIREMENT ANALYSIS**

Rich Picture (As-Is) -------------------------------------------------------------------------------------------------------- 08

Six Element (As-Is) ---------------------------------------------------------------------------------------------------- 09-21

Process Diagram (As-Is) --------------------------------------------------------------------------------------------- 22-26

Problem Analysis ------------------------------------------------------------------------------------------------------ 27-31

Rich Picture (To-Be) ------------------------------------------------------------------------------------------------------ 32

Six Element (To-Be) -------------------------------------------------------------------------------------------------- 33-40

Process Diagram (To-Be) ------------------------------------------------------------------------------------------- 41-44

**LOGICAL SYSTEM DESIGN**

Business Rule -------------------------------------------------------------------------------------------------------------- 46

Entity Relationship Diagram -------------------------------------------------------------------------------------------- 47

Entity Relationship Diagram to Relational Schema --------------------------------------------------------------- 48

Normalization ----------------------------------------------------------------------------------------------------------- 49-52

Data Dictionary --------------------------------------------------------------------------------------------------------- 53-60

**PHYSICAL SYSTEM DESIGN**

Input Forms ------------------------------------------------------------------------------------------------------------------ 62

Output Forms --------------------------------------------------------------------------------------------------------------- 65

**CONCLUSION**

Problem and Solution ----------------------------------------------------------------------------------------------------- 67

Additional Features & Future Development ------------------------------------------------------------------------- 67

Conclusion & Recommendation ---------------------------------------------------------------------------------------- 67

**CHAPTER 1**

**INTRODUCTION**

BACKGROUND OF THE PROJECT

OBJECTIVE OF THE PROJECT

SCOPE OF THE PROJECT

Background of the organization:

Independent University, Bangladesh (IUB) is one of the leading and oldest private university in Bangladesh where academic excellence is a tradition, teaching a passion and lifelong learning a habit. It was established in 1993. It has an explicit focus on Research and Global partnerships. The IUB campus sprawling over 3 acres, has an amphitheater, the state-of-the-art laboratories, well-equipped library with online access to journals and books, above 70 classrooms, lecture galleries, auditorium, gymnasium, food court, playground, medical Center, counseling Center and an alumni office.

IUB has world-class undergraduate and graduate program accredited by professional national 7 international accreditation bodies, such as University Grants Commission of Bangladesh (UGC), Accreditation Council for Business Schools and Programs (ACBSP), USA, and Institution of Engineers, Bangladesh (IEB). IUB prepares graduates for a successful career and this is central to the design of courses and the support we provide. The programs and the courses are designed in such a way that prepare the students for a successful career. The faculty members of IUB are actively engaged in research and publish regularly in peer-reviewed journals. Along with conventional classroom based teaching, students are engaged in research relatively early in their studies. IUB has academic research collaborations with various universities including Harvard University, Stanford University, University of Colorado at Boulder, Brown University, McMaster University, University of Heidelberg. IUB also participate in various national level inter-university sports, robotics, debates and similar competitions.

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 theme of this project is to find the systemic problems and limitation we have in our current system in few areas and how can we improve it. The aim of our project is to design, build and deliver a developed software that we believe will help universities everywhere to promote a more productive and effective way of evaluating students. Also there need to be some functional changes in the system and department. We also analyze individual processes that take place under the current system of monitoring student performance and the concerns and problems with those process from start to finish.

Objective of the project:

We want to develop the existing software iras in such way that can be more user friendly and helpful .it will help the institution to improve the quality of education. where the students and the faculty can use the system and find information more easily .in a short passage of time they can find all the information related to student enrollment, student grades, students CGPA and also CO and PLO.it will also benefit all the departments of the institution. this development will boost the work rate of everyone. it will be more productive and effective. not only the iras but also in different aspect few things need to be changed where we worked on. Monitoring semester wise student performance report by an Instructor and also analyze how to Department head submit grades of the students instead of faculty.

Scope of the project:

Project scope is a prerequisite to guarantee the success of a project. We have to make sure that the new system can be more successful than the present one when we are modifying an existing system.

We build an interface for faculties to able to see grades of another courses of a Student. Department can also access the systems for uploading grades instead of Instructor. If for some reason the instructor cannot upload the grade, then the Department can do it. On the other hand, Department head will be able to view different activities according to the different courses and sections of the instructor like Instructor’s Attendance,

Course wise Student performance etc.

Data will also, be protected and each user will be shown only that data which is relevant to them.

**CHAPTER 2**

**REQUIREMENT ANALYSIS**

RICH PICTURE AS-IS

SIX ELEMENTS AS-IS

PROCESS DIAGRAM AS-IS

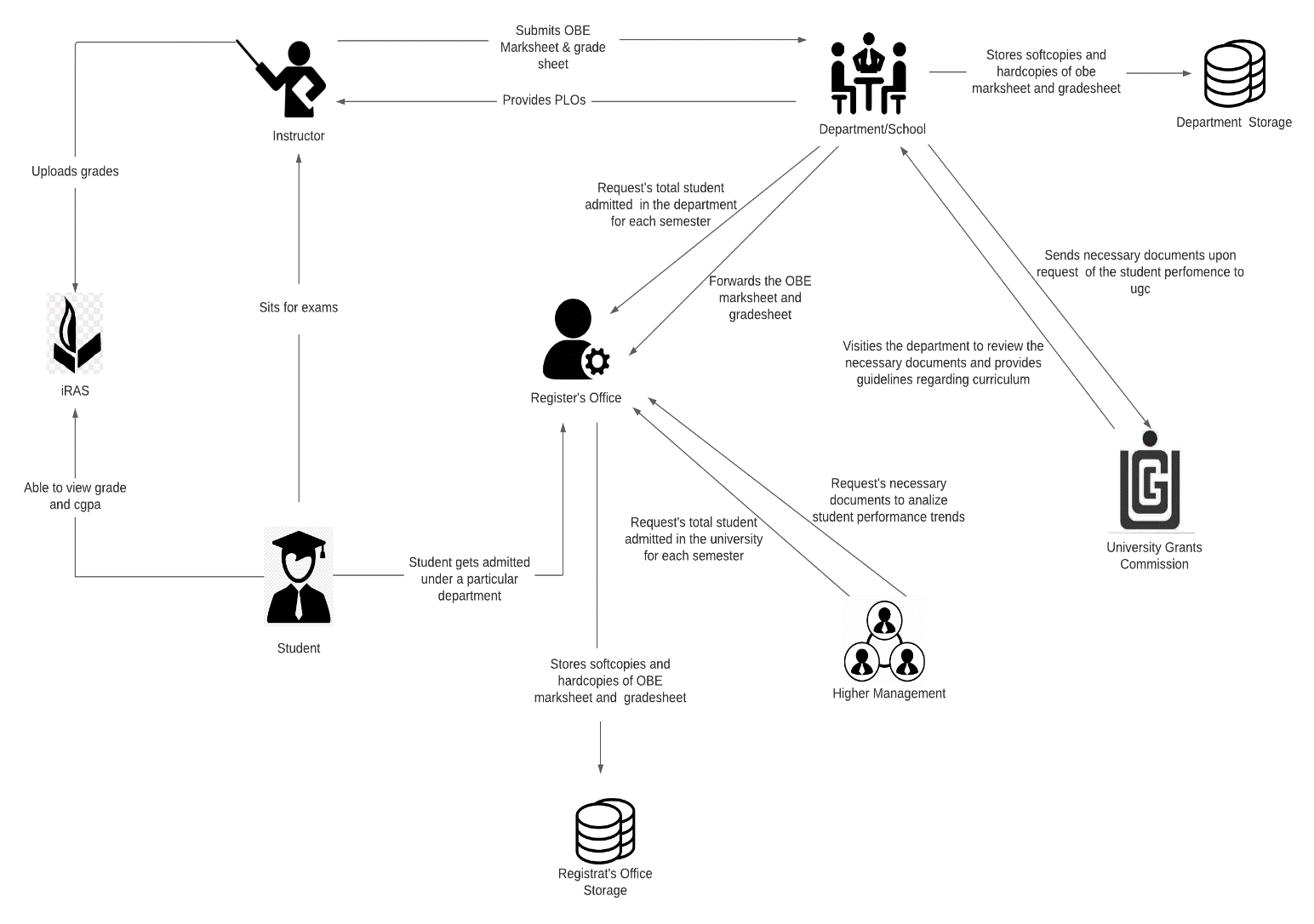
PROBLEM ANALYSIS

RICH PICTURE TO-BE

SIX ELEMENTS TO-BE

PROCESS DIAGRAM TO-BE

**RICH PICTURE(AS-IS)**



**SIX ELEMENT(AS-IS)**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| Student sits for exam | **Instructors**  1) Prepare question according to the mapped COs.  2) Give a particular time and date for the exam  3) Prepare SODs and invigilators  **Students**  1) Attempt the examination | **Stationery**  1) Pen and paper for writing.  2) Compass, ruler and other stationery for drawing diagrams  **Chairs and Table**  1) For using during exam.  **Classroom**  1) A space for conducting the exams  **Stapler**  1) For attaching all the extra paper, rough work and answers | **Computer/**  **Laptop**  1) Some courses require a computer for coding or open book exam.  **Calculators**  1) Some exams require the use of calculators  **Printers & photocopy machine**  1) Instructors use it for printing question papers | **Microsoft Word**  1) Typing the question and generating a printable pdf.  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS.  **Adobe Acrobat Reader**  1) For viewing the question paper in pdf format | **Microsoft Excel**  1) Used for storing exam marks and calculating final grade | **Internet**  1) Used by students during open book exam |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| Student are able to view grades, cgpa and download transcript | **Student**  1) Students have to login to iras by entering the student id and password  2) Select a specific semester  3) View grades for specific semester  4) Click on the transcript button to download a copy of transcript | **Paper**  1) Used for printing and keeping a hardcopy of transcript | **Computer/**  **Smart Phone**  1) Used for accessing iras.  **Printer**  1) For printing the transcript | **iRAS**  1) Provides user interface for view grades and download transcript.  **Browser**  1) Any browser an be used to access iras. e.g. edge, chrome, Firefox  **Adobe Acrobat Reader**  1) For viewing the transcript which is in pdf format.  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS. | **iRAS database server**  1) iras database server is used for storing and receiving student grade information in iras | **Internet**  1) Internet is required for accessing iras |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| Instructors uploads grades to iras | **Instructors**  1) Instructors types in user id and password for logging into the system  2) The instructor clicks to the submit grade section and is taken into the grade submission page  3) The instructor selects grade for each of the student  4) Clicks on the submit button to submit the grades |  | **Computer/**  **Smart Phone**  1) Used for accessing iras and submitting the grade | **iRAS**  1) Provides user interface for submitting the grades  **Browser**  1) Any browser an be used to access iras. e.g. edge, chrome, firefox  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS | **iRAS database server**  1) iras database server stores all the grades | **Internet**  1) Internet is required for accessing iras and submitting the grades |

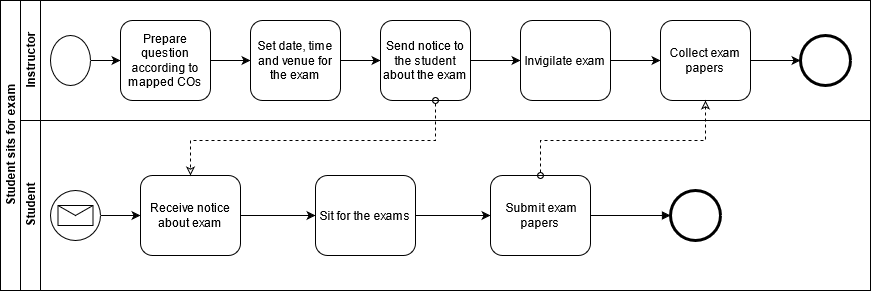
|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Process** | | **System Roles** | | | | | | | | | | |
| **Human** | | **Non-Comp**  **Hardware** | | **Computing**  **Hardware** | **Software** | | **Database** | | **Network &**  **Commination** | |
| Instructors produce OBE marksheet and grades sheet and submits it to the department | | **Instructors**  1) Instructor takes quizzes and exam  2)Checks the exam script  3) Records the mark for each exam in an excel sheet  4) Calculates the final grades and    5) Calculate total marks received for each CO  6) Declare if a student has achieved a specific CO  7) Declare if a student has received a PLO for a related CO  8) Make a verdict and analysis of how many students were able to receive a certain CO and PLO  9) Sends the final version of OBE marksheet to department office  **Department**  1) Receives a copy of the OBE marksheet and grade sheet from the instructors  2) Stores a copy of the OBE marksheet and grade sheet in department storage  3) Sends a copy of the OBE marksheet to the register’s office  **Register’s Office**  1) Receives the OBE marksheet from department  2) Store the OBE marksheet in register’s office storage | | **Paper**  1) Used for storing hardcopies of OBE marksheet | | **Computer**  1) Computer is used for making softcopies of OBE marksheets  **Printer**  1) To print the hardcopies of the OBE marksheet and grade sheet | **Microsoft Excel**  1) Used by instructors to calculate the PLO and CO achievement | | **Department Storage**  1) A hardcopy of OBE marksheet and grade sheet is stored in the department storage  **Register’s Office Storage**  1) A hardcopy of OBE marksheet and grade sheet is stored in the register’s office storage | | **Internet**  1) Online platform such as- google sheets may be used for producing OBE marksheet | |
| **Process** | | **System Roles** | | | | | | | | | | | | |
| **Human** | | **Non-Comp**  **Hardware** | | **Computing**  **Hardware** | | | **Software** | | **Database** | | **Network &**  **Commination** | |
| Map Course Outcomes  (COs) to Program Learning Outcomes  (PLOs) | | **UGC**  1) Provides guide line to the department about the curriculum  **Department**  1) Comes with the PLOs  2) Sends the PLOs to the instructor  **Instructor**  1) List the course content and course outcome  2) Maps the course content to the COs  3)Maps the PLOs  4)Prepares question paper according to the COs | | **Pen and Paper**  1) Used for brainstorming and rough works | | **Computer/Smart devices**  1) Course coordinators use computers to make softcopies of course outcomes (COs)  **Printers**  1) Used for print hardcopies of course outcomes (COs) | | | **Microsoft Word**  1) Course coordinators use MS word for making course outline and course assessment report with COs mapping to the PLOs | |  | | **Internet**  1) Internet is used to communicate with ugc and other stakeholders to discuss topics related mapping COs and PLOs | |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| Student gets admitted under a particular department | **Student**  1) Fills up the admission form for taking admission under a particular department  2) Receive an email regarding successful admission form submission  **Register’s Office**  1) Receives the admission form  2) Analyze the admission  3) Check if the student fulfills all the requirements for getting admitted  4) If the student fulfills all the requirements then admit the student under the requested department.  6) Generate a student id number  5) Sends the total number of students enrolled in a semester under a particular department to the department.  6) Send the total number of students enrolled in the university to the higher management.  **Department**  1) Request total student enrolled in the department  2) Receive information about total student enrolled in department  **Higher Management**  1) Request total student enrolled in the university  2) Receive information about total student enrolled in department. | **Paper**  1) Register’s office keeps a hardcopy of student information. e.g. student blood group, emergence contact number, address | **Computer**  1) Used for accessing iras and filling admission form  **Printers**  1) For printing hardcopies of student information | **iRAS**  1) Provides user interface for filling the admission form  **Browser**  1) Any browser an be used to access iras. e.g. edge, chrome, Firefox  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS. | **iRAS database server**  1) iras database server is used for storing all the admission information. | **Internet**  1) Internet is required for accessing the online admission form. |

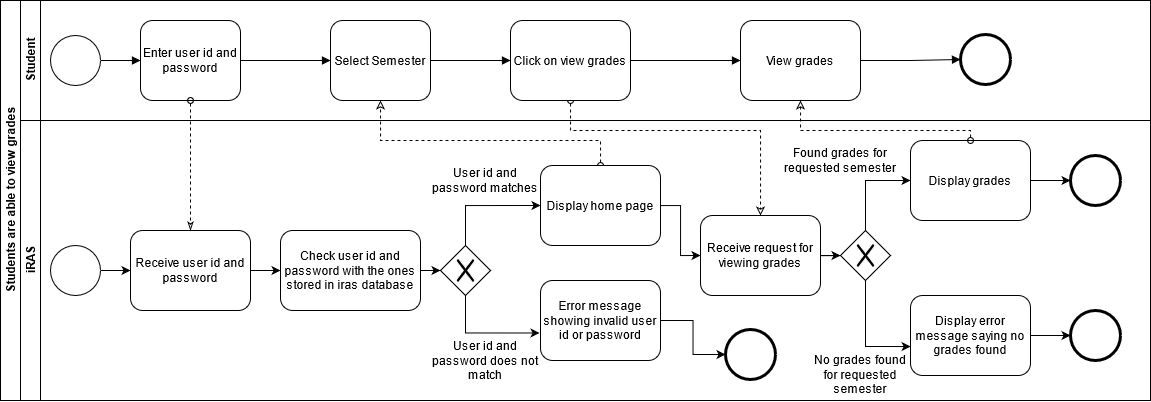
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| Request for review and change of grades | **Student**  1)Request an Instructor for grade change by sending an application via email.  **Instructor**  1)Receive a grade change mail from the student.  2)Check exam  Papers and other assessment upon request.  3)If change needs to be made, then the instructor informs the department.  4) If not, end the process. Mail the student that his request has been denied.  **Department**  1) Receives information regarding grade change of a specific student in a course.  2) Sends a request to the register’s office for grade change  3)Updates the OBE marksheet and grade sheet with the new grade and stores it in the department storage  **Register’s office**  1)Receive a request from the department for the changing the grade of a student in a specific course.  2)Changes the grade of the particular student in the requested course.  3)Updates the register’s office storage with the new grade | **Pen and Paper**  1)used to note down key points or marks on the students’ answer sheets. | **Computer/**  **Laptop**  1) Used for sending email to the instructor | **iRAS**  1)Used by the Register office for changing the grade  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS. | **iRAS database server**  1) Update student grade data.  **Department Storage**  1)Update student grade data**.**  **Register office’s Storage**  1)Update student grade data. | **Internet**  1) Internet is needed to the mail a grade change request. |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process** | **System Roles** | | | | | |
| **Human** | **Non-Comp**  **Hardware** | **Computing**  **Hardware** | **Software** | **Database** | **Network &**  **Commination** |
| View Records OBE Marksheets and Course  Assessment Reports | **UGC**  1. Inform the university head of a deadline within which OBE Marksheets, Course Assessment Reports and other documents are needed for quality inspection to make necessary improvements to degree programs.  2. Inform the university head if an UGC personnel will visit the campus or softcopies will suffice.  3. Visit university heads and relevant schools to receive the necessary documents and reports if that is what was informed.  **Department**  1) Request to view records of OBE Marksheets, Course Assessment Reports to analyze students’ performance trends.  2) Direct Department Staff to gather necessary documents, OBE Marksheets and Assessment report for a given time-period specified by UGC.  3) Receive the necessary documents gathered by the Department  4) Evaluate the need to change/ improve the department’s educational resources based on students’ performance trends.  5) Send necessary documents to ugc.  **Higher Management**  1) Requests the register’s office to send records of OBE Marksheets, Course Assessment Reports to analyze students’ performance trends.  **Register’s Office**  1) Receive a request from higher management for sending OBE marksheet and grade sheets.  2) Sends the requested OBE marksheets and grade sheets to the register’s office. | **Paper and Pen**  1)Used for noting/marking down key points of the report. | **Computer**  1) Used for viewing softcopies of OBE marksheet and grade sheet.  2) Used for send softcopies of OBE marksheet to the ugc officials. | **Microsoft Excel**  1) Used for viewing softcopies of marksheet  **Operating System**  1) Any OS may be used. e.g. Windows, MacOS. | **Department Storage**  1) Used for retrieval of OBE marksheet and grade sheet when needed  2) Stores hardcopies and softcopies of OBE marksheet and grade sheet | **Internet**  1) Softcopies of OBE marksheet and grade sheet may be mailed to the ugc officials.  2) Online platforms such as google sheet may be use for displaying softcopies of marksheet. |

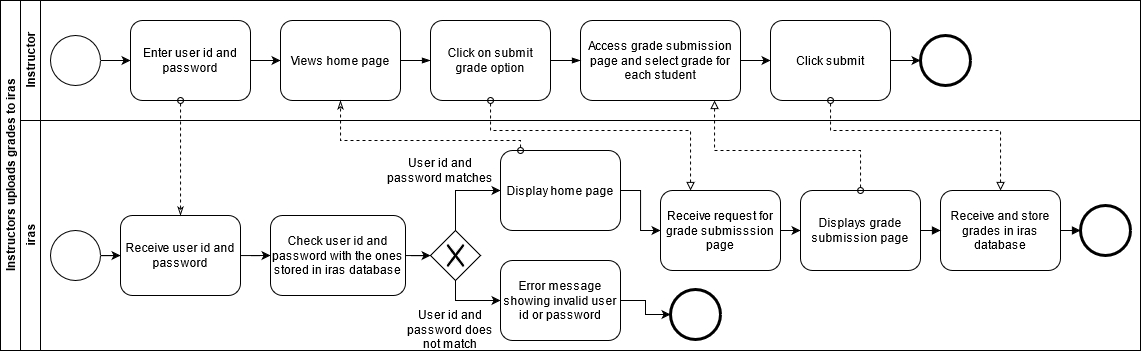
**PROCESS DIAGRAM(AS-IS)**



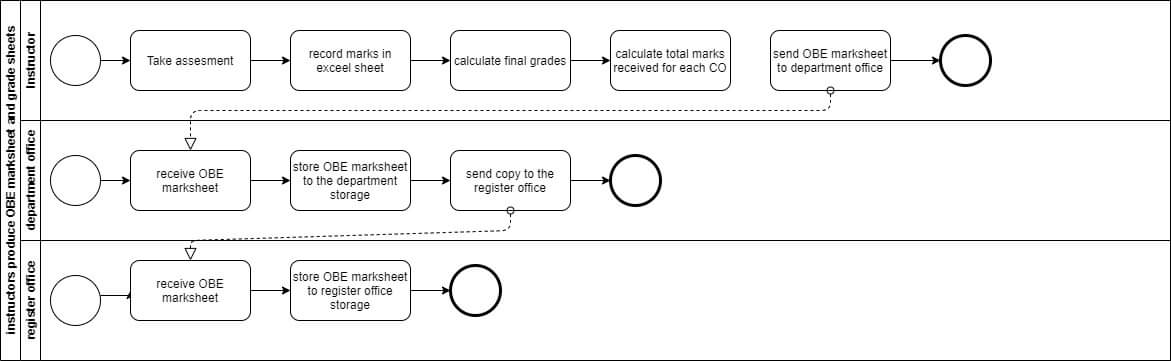
**FIGURE 2.1 Process Diagram for Student Sits for exam**

****

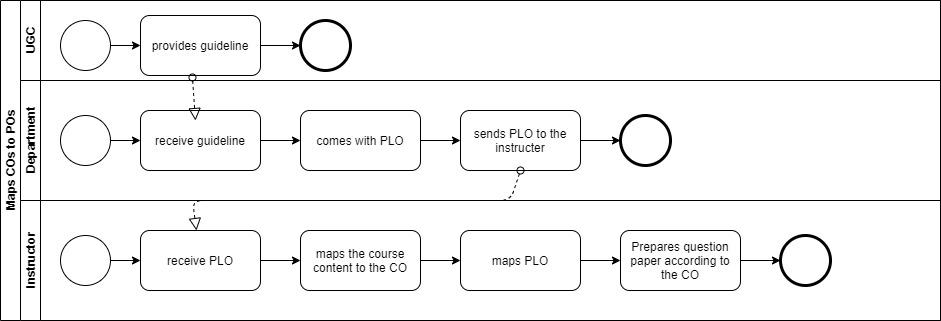
**FIGURE 2.2 Process Diagram for Student are able to view grades and CGPA**

****

**FIGURE 2.3 Process Diagram for Instructor uploading grade to iras**

****

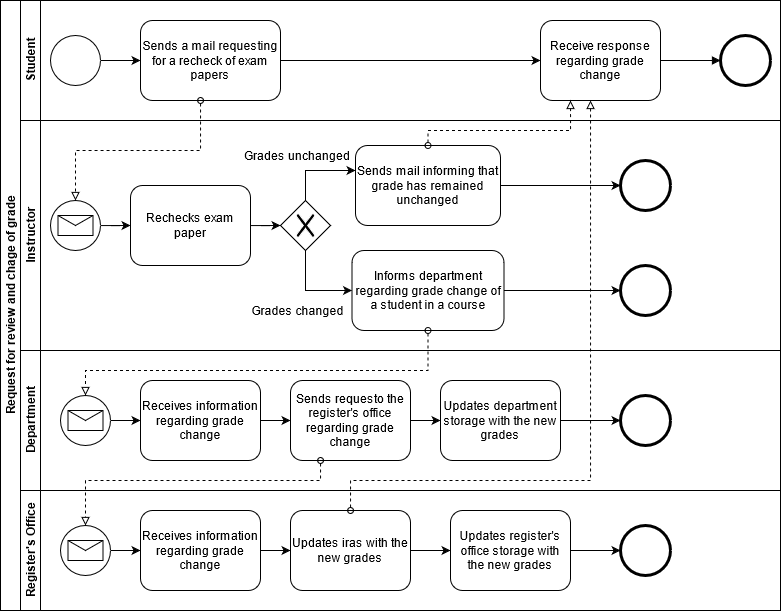
**FIGURE 2.4 Process Diagram for Instructor produces OBE marksheet**

****

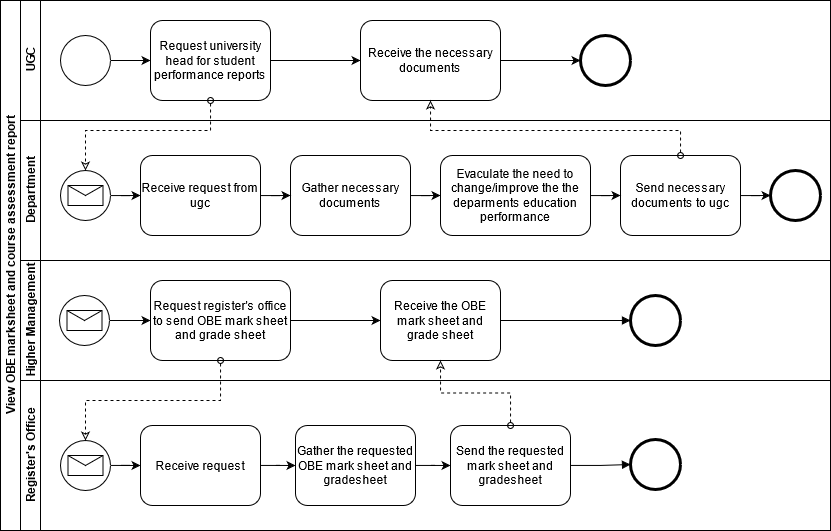
**FIGURE 2.5 Process Diagram for Map COs and Pos**

****

**FIGURE 2.6 Process Diagram for Student gets admitted under particular department**

****

**FIGURE 2.7 Process Diagram for request for review and change of grades**

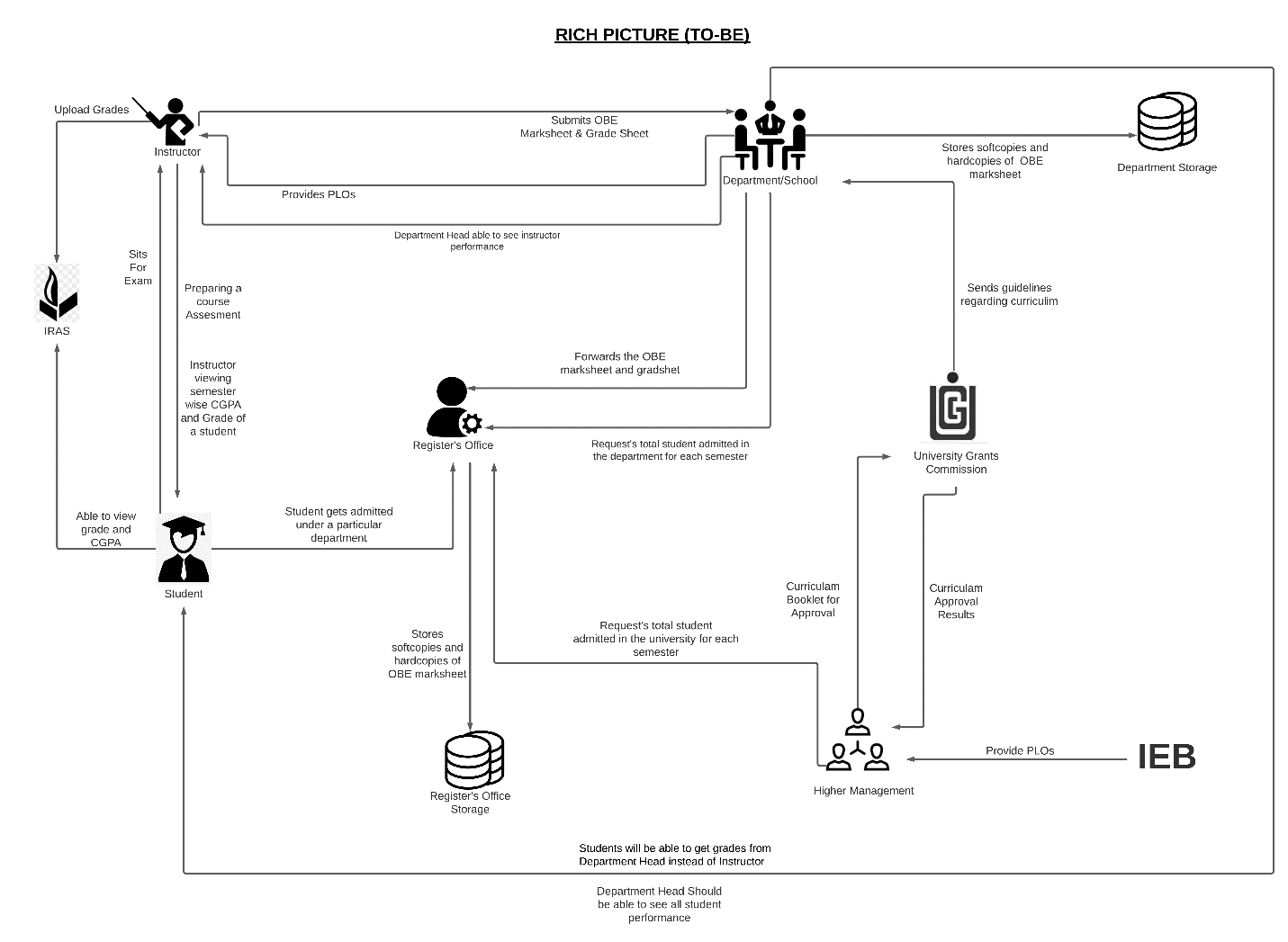
****

**FIGURE 2.8 Process Diagram for view obe marksheet and course assessment report**

**Problem Analysis**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Process Name** | **Stakeholders** | **Concern (Problems)** | **Analysis (reason of the problem)** | **Proposed Solutions** |
| Preparing a Course  Assessment | 1.Instructor 2.Student | Sending hardcopy  And softcopy  Students examination marks  And course  Assessment report  to the register office store the info  Time consumption  And delay is prime limitation. Even after storing data in the register office store, if there is any need to see the information of any student or any course performance or a particular section of high management then It is very difficult to find these. | In our existing system higher management store assessment data manually  As sending hardcopy and softcopy to the register office involve multiple  persons and different  processes, it could  easily led to confusion, loss of  important student report card. It also wastes unnecessary resources such as paper and printer. | We will create a system where Higher management will no longer have to wait for the registered office for searching particular student data. If higher management wants to find student data, specific course data, or find specific section-wise student data they can enter only student ID, Course ID, or Section ID in our new system. They will be able to see student performances in the graph shows. And they can download student information. |
| Higher Management Viewing Individual Instructor Performance | 1.Department Head  2.Dean  3.Instructor | In our existing system higher management can’t see their instructor performance digitally. Higher management see only Instructor performance send by the hardcopy of the course wise student performance report. Higher Management can’t see how many quizzes and assignment they are taken, whether he is taking regular classes, whether he is giving exam papers properly, what is the result of the student in his section, what was the result of the last semester even under that faculty and what kind of project they are maintains for specific course and prepare a projects  specification based on their course | In our exacting system higher management can only see Hardcopy for an individual instructor performance, but it's difficult for measuring a performance instructor by instructor, and it's also difficult comparing with previous semester performance because its hardworking and time consumption matter. It also wastes unnecessary resources such as paper and printer. | We will create a new system where Higher Management can see their Instructor Performance department wise, section wise, and course wise. Higher Management can download instructor performance data with graphs or charts. Then they can easily compare to each other and also compare with previous semester result in the same course. After download data Higher Management can see their performance like how many quizzes and assignment they are taking, whether instructor attend the class regularly, also see instructor class performance and class performance feedback by the student after faculty evaluation |
| |  | | --- | | Instructor viewing the CGPA and change the grade | | 1.Instructor  2.Student | In our existing system without the Higher Management, the faculty cannot see any student's CGPA and grade sheet. They only  know about the courses they have taken. Even once they upload the grade to the system, they cannot change it later. If a student's grade changes or applies for a change, the instructor has to help the Register Office and Department Head. And it takes the permission of the obsessed department head to change the grade | Now, instructor can’t see any student CGPA and grade sheet and also  If a student feels that his or her grade  has not been returned or correct, the student will apply along with the instructor. After Application Instructor Contact Department Head Than They Can Check the Script Again. If change is another grade then department head request to Register Office for Change The Grade, It's a Long Term and Hard Process Also its Time Consumption process. | We will create a new system where Higher Management and instructor can see the student CGPA and Grade sheet using student ID in this case instructors and  students should be in the same department. And also we will create a system where higher management and instructor can change the grade easily getting application from student after checking script with department head and controller of examination.  After. And instructor get permission to resubmits the grades easily using our new system. |
| Higher management and Instructor viewing OBE mark sheet and grade sheet | 1.Higher management (HM)  2.Instructor  3.Department  4.Dean/Vc | The current process of requesting the head of the department to view records for analysis and inspection can result in delays due to various problems in communication. Since the OBE Marks sheets course assessment reports and other necessary documents are only saved in softcopies (Without database management) and hard copies, it can get tedious and time-consuming to retrieve them when needed. | Due to being a hardcopy, when the Higher management wants to see each course, section, and department wise OBE mark sheet and course assessment then a lot of trouble to maintain this kind of documents, and it is also very difficult to analyze by looking at the hard copy so that the data is likely to be wrong and lost and when these data are compared with any previous data it becomes more difficult. It also wastes unnecessary resources such as paper and printer. | We will create a new system where Higher Management and instructor can see the OBE Mark Sheet, Course Assessment using their ID (Only those to whom Higher management will give permission will be able to see)  The system that we will build be there the mark sheet and course assessments will be according to the section, course, and department, and they can download them as needed. |
| Students will be able to get grades from Department instead of Instructor | 1. Department 2. Instructor 3. Student | We don't have the option to grade someone else instead of the Instructor in our system. If for some reason an instructor cannot give a grade If there is an instructor leave or something tragic happens then there is no option to continue the semester and submit a grade, unless the department manages it. | If necessary, if an instructor is on leave, then the whole matter has to be handled by the department Instead, another instructor has to be appointed and he has to explain the whole process again, it's difficult to manage in a short period of a semester. | We will create a new system where the Department Head can see the performance of the students and give them a grade for Emergency Situations. Based on their PLO & co achievement and OBE mark sheet in the Previous semester |
| Higher Management & Instructor Uploading & Viewing PLOs/CO | Higher Management (HM)  Department | In our existing system Higher Management (HM), Department Head, Dean/VC and instructor see only hardcopy PLOs and Co achievement, but its time consuming when they want to check it manually. There are many students in one section and every course has many sections and each department has many students, so a lot of student information is not possible to check manually. In this case, there is a possibility to lose data. | The current system does not support  Viewing PLOs and CO achievement. Due to which no one Instructor, Higher Management cannot see the POL & Co Achievement and student performance | We will create a new system where  Where instructors can upload Plo & Co reports, all of the higher management and instructors can see and download the data. They will be able to view this data using input Student id to the system and see Plo & co achievement of any specific student, course-wise, and section-wise. |
| Student viewing PLO & CO | 1.Student | In our existing system Student cannot see our PLOs and Co achievement. They cannot even see the hardcopy. | It is important for every student to see their Plo and co-Achievement, what course they are doing, it is important to know what did they achieved and what issues need to be improved. But it is not seen in our existing system now. | We will create a new system where Students will be able to see and download the file and they will be able to view their Plo & Co achievement and compare with the other Course. |
| UGC approves  curriculum  based on PLO and CO | 1. Higher  Management (HM)  2. UGC | HM needs to send the curriculum  booklet manually. HM needs to send the updated  Curriculum to the  Department every time. | It will take time for the UGC to receive the Curriculum booklet and process the information.  It is a hassle to send manually every time the curriculum is updated | We can transfer the  curriculum in our new system by which it could be accessed  easily by the members and it also could be edited real time by the HM and updated instantly whenever changes are required by the UGC. |

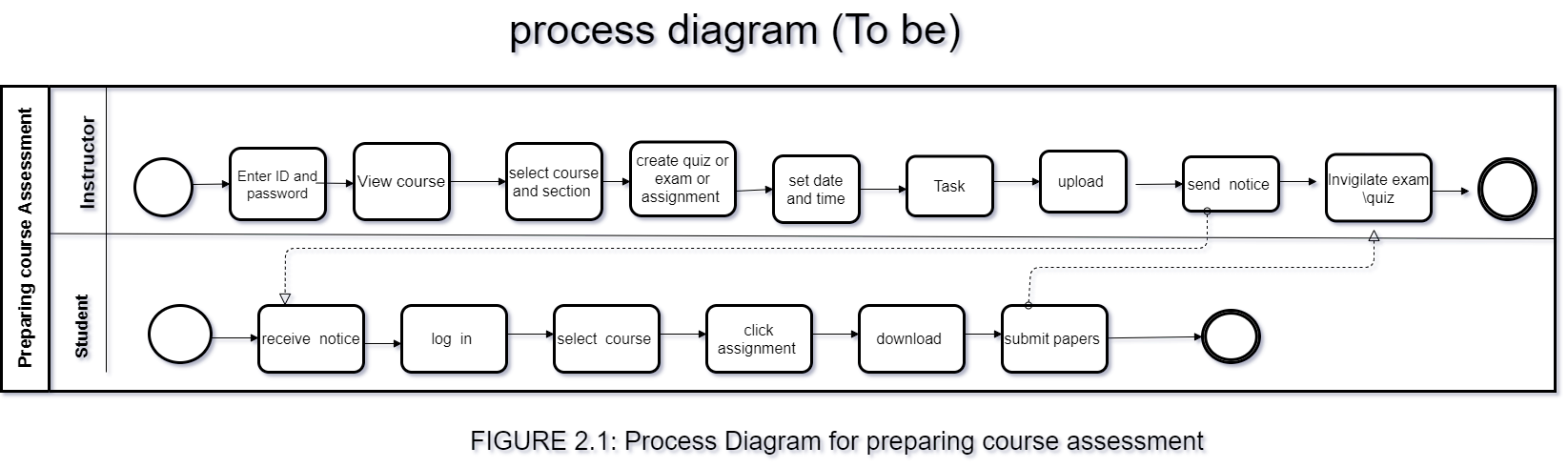
**RICH PICTURE (TO-BE)**

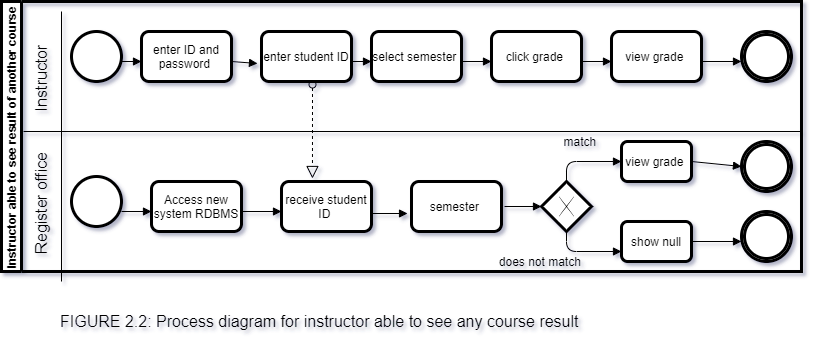
****

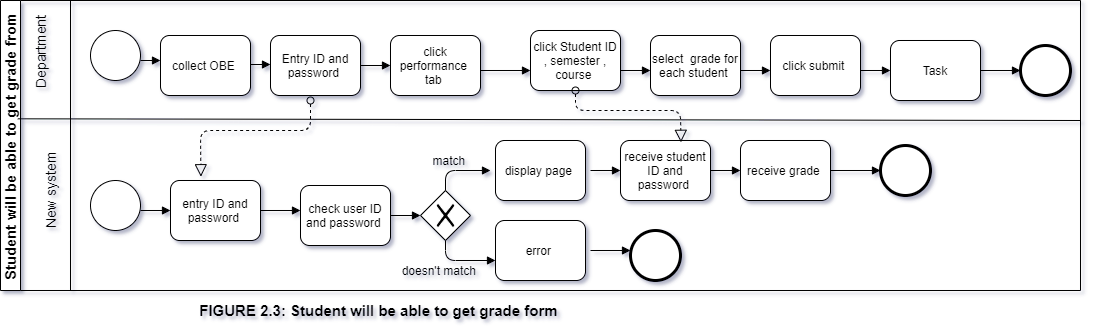
**SIX ELEMENT (TO-BE)**

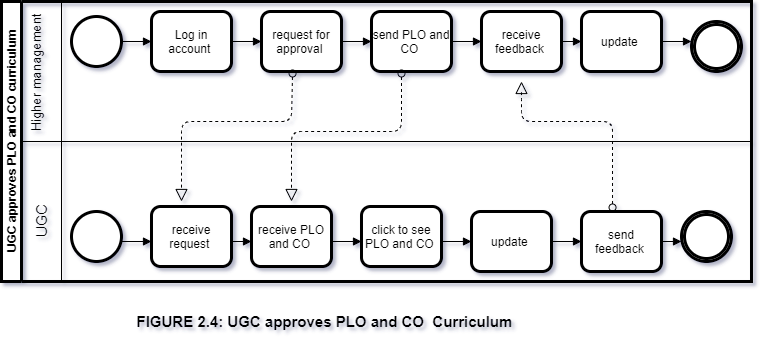
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Process**  **System Process** | **Human** | **Non-comb Hardware** | **Computing Hardware** | **Software** | **Database** | **Network and Communication** |
| Preparing Course Assessment of Instructor | **Instructor:**  1)Log in to a “New **System”.**  2) Instructor will be shown the courses they have/had for every semester under “Semester” Tab.  3)Select course (section and thereof).  4)Create  (quiz/ exam/ project)  5)For each student, each student’s score for each question.  6) Upload the Assessment report for the students.  Student:  1)Login to the “New System”.  2)Goes to desired course.  3)Click on “Course Assessment’  4) Download it. | **Google**  **Forms:**  1)Used for recording a student’s remote response to the questions. | **Computer**:  1)Used for accessing the “New System”.  **Printer:**  1)Printout the softcopy of Assessment report. | **New System**  **Faculty frontend:**  1)Provides user interface for the faculty to enter student assessment data | **Google Classroom:**  1)Import assessment data from google forms(or classroom, depending on their API), manually or automatically | **Internet:**  1)New System is a fully online web application: all preparing and requests thereof are sent through the internet.  **Email:**  1)Email is the primary method of notifying the students about major assessment |
| **Instructor Able to see the result of another courses of a Student** | **Instructor**:  1.Login to **New System.**  2. Search that specific student's id.  3. See the grades of other courses for intended semester but only his/her(Instructor) Department.  **Register Office:**  1.Access **New System.**  2.View Students grades of other courses if and when it’s necessary | **Pen and Paper:**  Note down the grade if needed. | **Computer/Phone:**  1.Used for accessing **New System.**  2.Used Computer to make softcopies.  **Printer**:  Printout the softcopies. | **New System**  **Instructor frontend: 1.** Provides the online user interface for viewing grades. | **Networking devices (Router, Switch Bridge, Hub):**  Used by Instructor and students to access the Internet.  **Database Server**:  Instructor receive the student information in **New System.** | **Internet:**  All related data searched through internet. |
| Students will be able to get grades from  Department instead | **Department**:  1.Collect the student’s OBE mark sheet & grade sheet.  2.Log in to  **New System.**  3.Click on “Performance Monitoring” tab.  4.Search Student I’d to upload his/her grade.  3.Select a particular course & section according to the Department.  4. Submit the grade next to the student’s name  based on their PLO & co achievement and OBE mark sheet | **Calculator**:  Marks are calculated with a calculator. | **Computer:**  Used for accessing IRAS.  **Printer:**  Printout the softcopy of the mark sheet. | **Excel sheet:**  Marks-sheet can be created using Excel sheet, Google sheet  **Email Software:**  Used for communication between Department head and Instructor. | **New System RDBMS:**  1. This Database management used to store and maintain student grades’ information | **Internet and Gmail:**  The marks sheet can be taken through emails or any other internet messaging platforms. |
| **UGC approves curriculum based on PLO and CO** | **Higher management:**  1)Log in to  **New System**.  2)Requests for Program approval to UGC based on Plo & CO.  **UGC:**  1)Receive the request from Higher Management.  2)Feedback the higher management. | **Paper:**  1)Use to print book of curriculum.  2)Use for signature. | **Printer:**  1)Use for print.  **Computer**:  1)Save the file. | **Microsoft Word:**  1)Use for save book.  **Excel sheet:**  Necessary data store. | **Gmail:**  Using for mail send.  **Web Server:**  1)Update information.  **Microsoft Excel Database:**  Instructor excess CO’s form. | **Internet:**  Using send mail  UGC and update and upload new  Version |
| **Department Head able to see all instructor Performance** | **Department Head** :  1.Login to **New System**.  2.Click on “Performance Monitoring” tab.  3.Select course and section, according to Department. | **Paper:**  Instructor send the hardcopy of the semester wise student performance report to the | **Computer/Phone:**  1.Used for accessing **New System**.  2.Create softcopies of record of all assessment date.  **Printer:**  2.If needed Printout the softcopies. | **Excel sheet:**  Record necessary assessment data in Excel sheet.  **Department frontend:**  Update activity of Instructor.  **Printing Software:**  Used for printing Software doc.  **PDF Viewer:**  To view the transcript in PDF-form. | **New System server:**  Store update activity.  **Department Storage:**  Record of instructor assessment. | **Internet:**  Need to connect **New System.** |
| **Higher Management and Instructor viewing OBE mark sheets and grade sheet** | **Department Head/ Dean/ VC/ Board of Trustees:**  1)Log into New System Department Head dashboard.  2)View department  Assessment report  .  3)View Course  Assessment Reports  &  OBE Mark sheets, searchable by year, according to the Department & Course.  4)View individual student reports.  **Instructor:**  1)Log into New System Instructor dashboard.  2)Using ID & Password.  3)Click on “Performance Monitoring” tab.  4)View Course  Assessment Reports  &  OBE Mark sheets according to the Department, Course & Section.  5)Download them if they want or need. | **Pen and paper:**  **1**. May be used for high-level notetaking. | **Cloud Server:**  1. Receive and process incoming requests  Computer/ mobile:  1. View reports & mark sheet, grade sheet. | **New System Instructor frontend:**  **1**. Provide user interface for online Instructor navigation.  2. Show specific reports on request.  3. Sort report data in customizable ways (by PLO, by CO, by semester, by course, by time)  **Excel sheet**:  Record necessary report in Excel sheet. | **System RDBMS:**  **1**. For a specific course and student(s), retrieve PLO/ CO achievement data from RDBMS and tabulate them.  2. From tabulated data, derive outcome analysis and verdict | **Internet:**  1. New System is a fully online web application: all packets and requests thereof are sent through the internet. |
| **Instructor viewing CGPA and change the grade** | **Student:**  1.Log into **New System** Student Dashboard  2. Goes to desired course  2.Click on “Request Grade Change”  3.Fills form e.g. with reason for grade change  4.Submits the grade change request  **Instructor:**  1.Logs into Instructor dashboard  2.Reviews grade change request  3.Check exam  Papers and other assessment upon request.  4.If change needs to be made, then the instructor changes the grade and inform or Submit the grade to the Department. 5.If not, end the process. Mail the student that his request has been denied.  **Department**  1.Receives  information regarding grade change of a specific student in a course.  2. Updates the OBE mark sheet and grade sheet with the new grade and stores it in  the department storage.  3.Inform to the Register’s office for changing the grade.  **Register’s Office:**  1)Receive a request from the department for updating new grade of a student in a specific course.  3)Updates the register’s office storage with the new grade | **Pen and paper:**  1. May be used for high-level notetaking.  2. Hard copies of student test papers used for review | **Computer/Phone:**  1.Used for viewing and making changes to grades | **New System** **Student frontend**:  1. Provide user interface making grade change requests  2.Show “Request Grade Change” interface  3.Provide field to input reason for grade  4.Show submit button interface  **New System Instructor frontend:**  1.Provide user interface for instructor to make grade changes  2.Show requested grade change details  3.Show approve or disapprove button  4.If approved, provide field for new grade input | **New System RDBMS**:  1. Changed grade data are stored here | **Internet:**  1. This **New System**is a fully online web application: all packets and requests therefore are sent through the internet. |
| **Student viewing PLO & CO** | **Student:**  **1.**Log into **New System** Student Dashboard  2. Click on “Performance Monitoring” tab  3. Select course and time period  4. Click on “Plo & CO's report”  5. View OBE mark sheet in browser.  7. Obtain information about their performance for the selected semester.  8. Download report in PDF form | **Pen & Paper:**  Note down the grade if needed.  **Calculator:**  Marks are calculated with a calculator. | **Computer/Phone:**  1.Used for accessing **New System.**  **Printer**:  1.If needed Printout the softcopies | **System Student frontend:**  1.Provide user interface for online Student navigation  2. Show specific reports  3. Sort report data in customizable ways (by PLO, by CO, by semester, by course, by time) | **New System RDBMS**:  1. A Database Management Service is used to store, maintain, edit and receive the list of COs and PLOs of each student, student’s grade information and transcript. | **Internet:**  All related data searched through internet. |

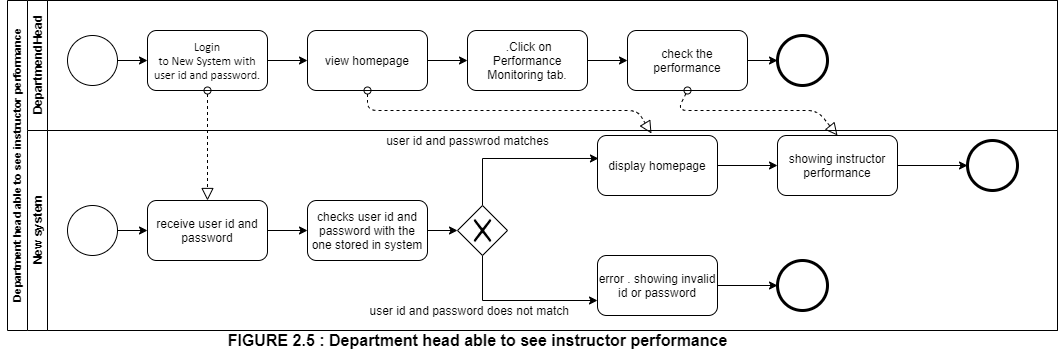
**PROCESS DIAGRAM (TO-BE)**

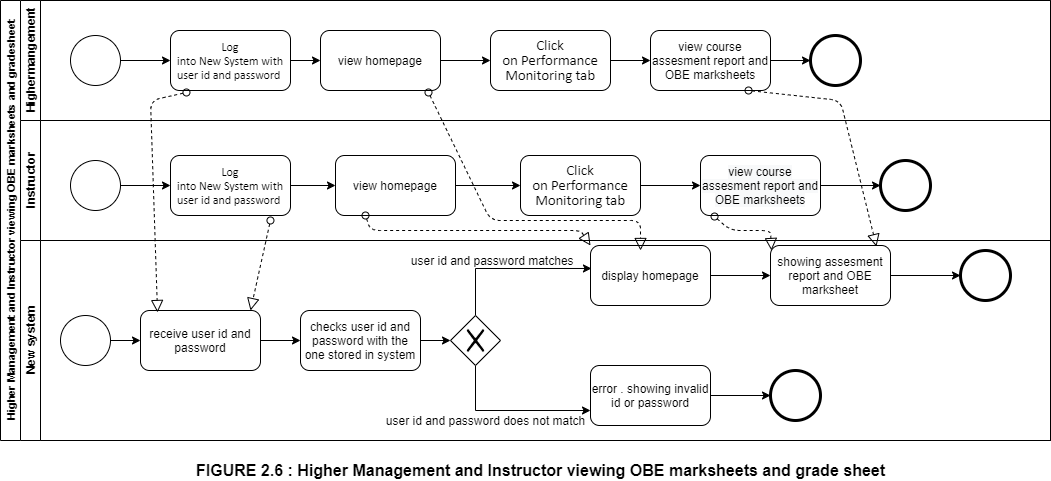
****

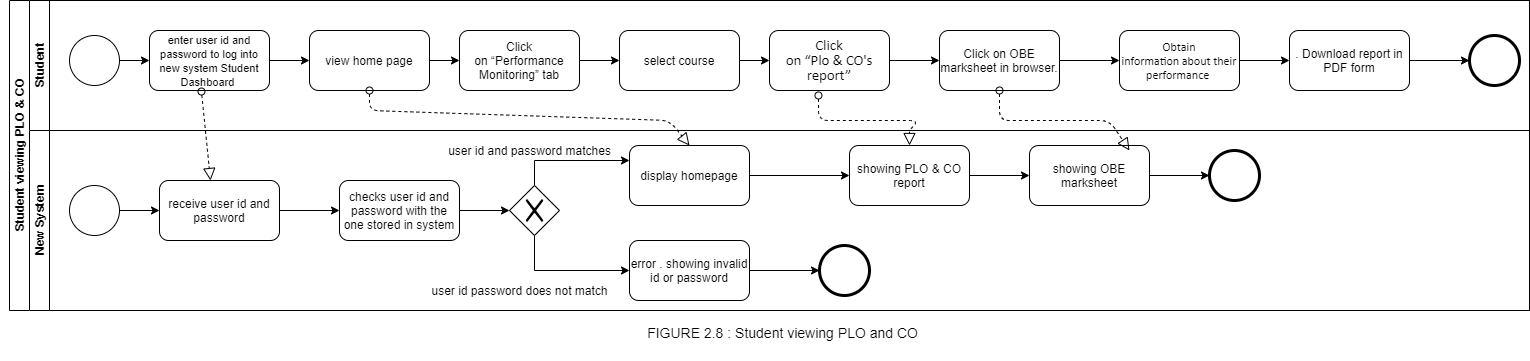
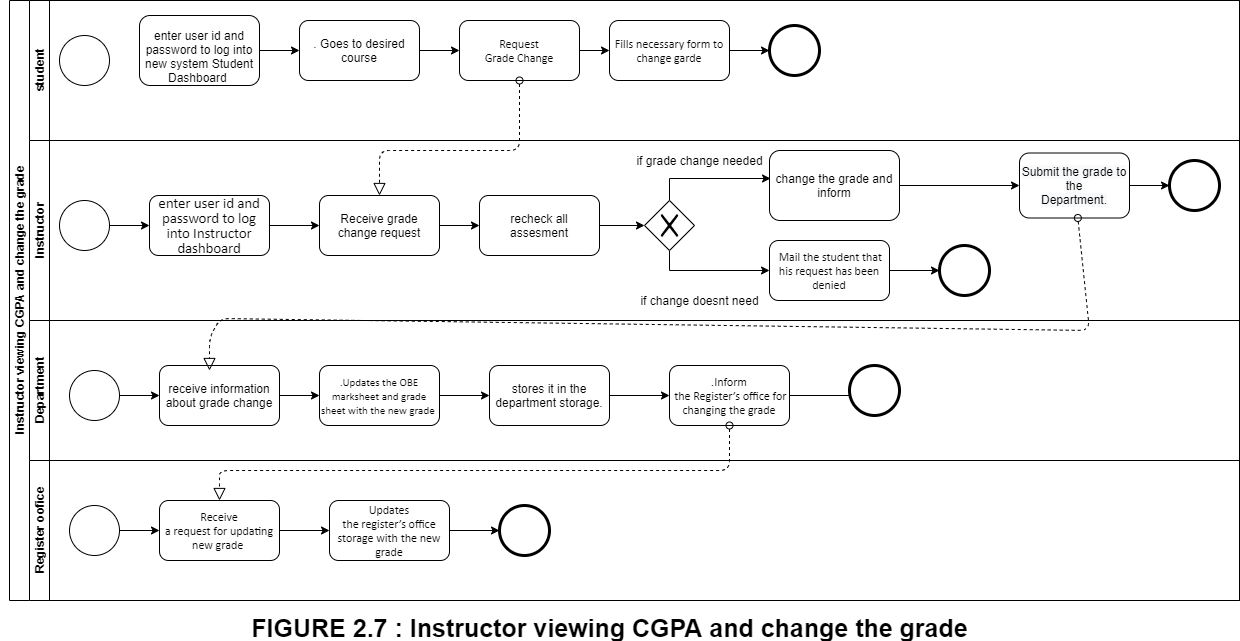
****

****

****

****

****



**CHAPTER 3**

**LOGICAL SYSTEM DESIGN**

BUSINESS RULE

ENTITY RELATIONSHIP DIAGRAM

ENTITY RELATIONSHIP DIAGRAM TO RELATIONAL SCHEMA

NORMALIZATION

DATA DICTONARY

**BUSINESS RULE**

1) A student may register under one or more programs. A program many have multiple students.

2) A department may have multiple programs. A program must be exactly under one department.

3) A school may have multiple departments. A department must be exactly under one school.

4) A department may have multiple instructors. An instructor must be exactly under one department.

5) A department must have exactly one head.

6) A school must have exactly one dean.

7) A program may have multiple PLOs. A PLO many be under multiple programs.

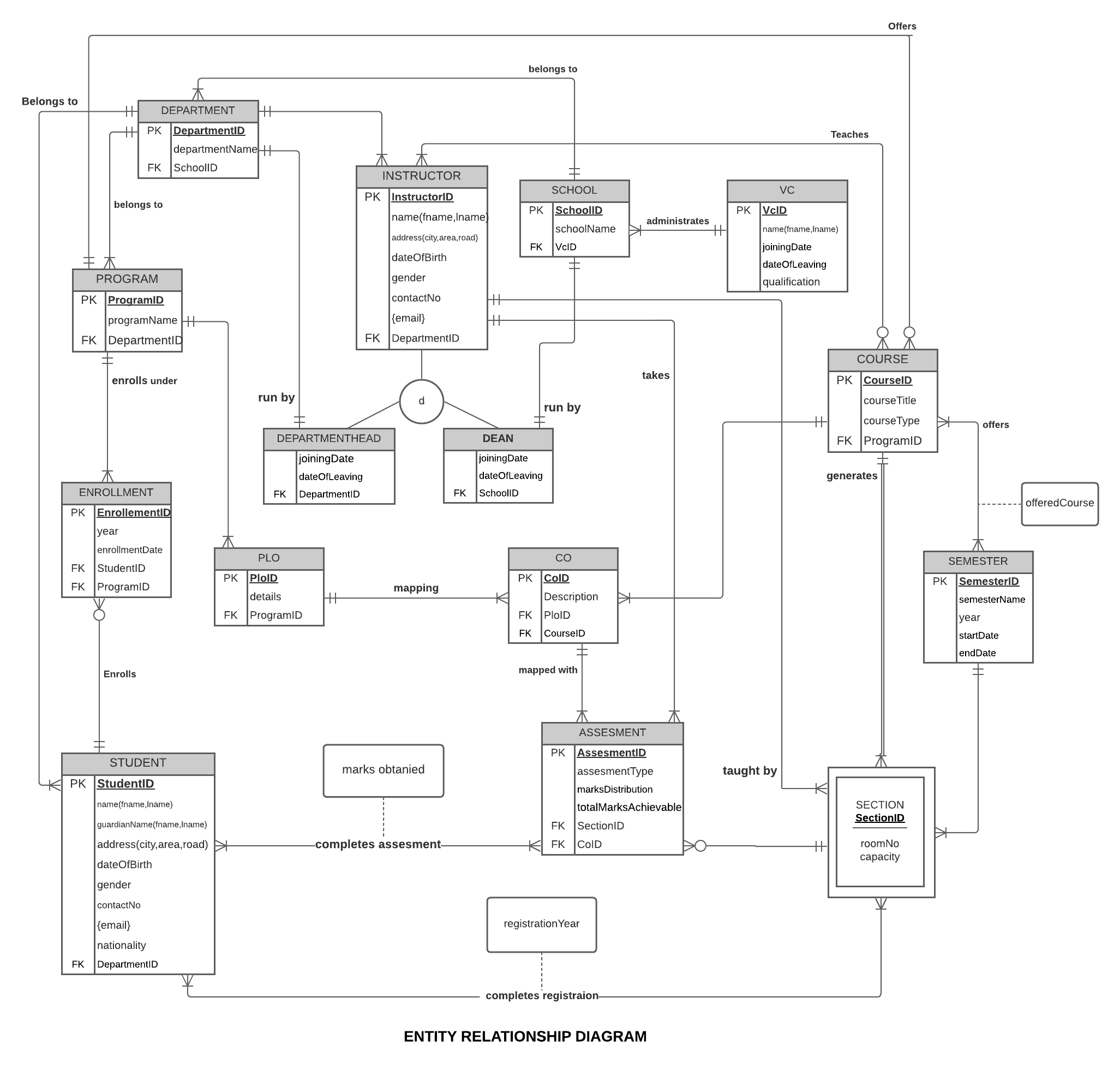
8) An instructor may teach multiple courses. A course must have exactly one instructor.

9) A course may have multiple sections. A section must be under exactly one course.

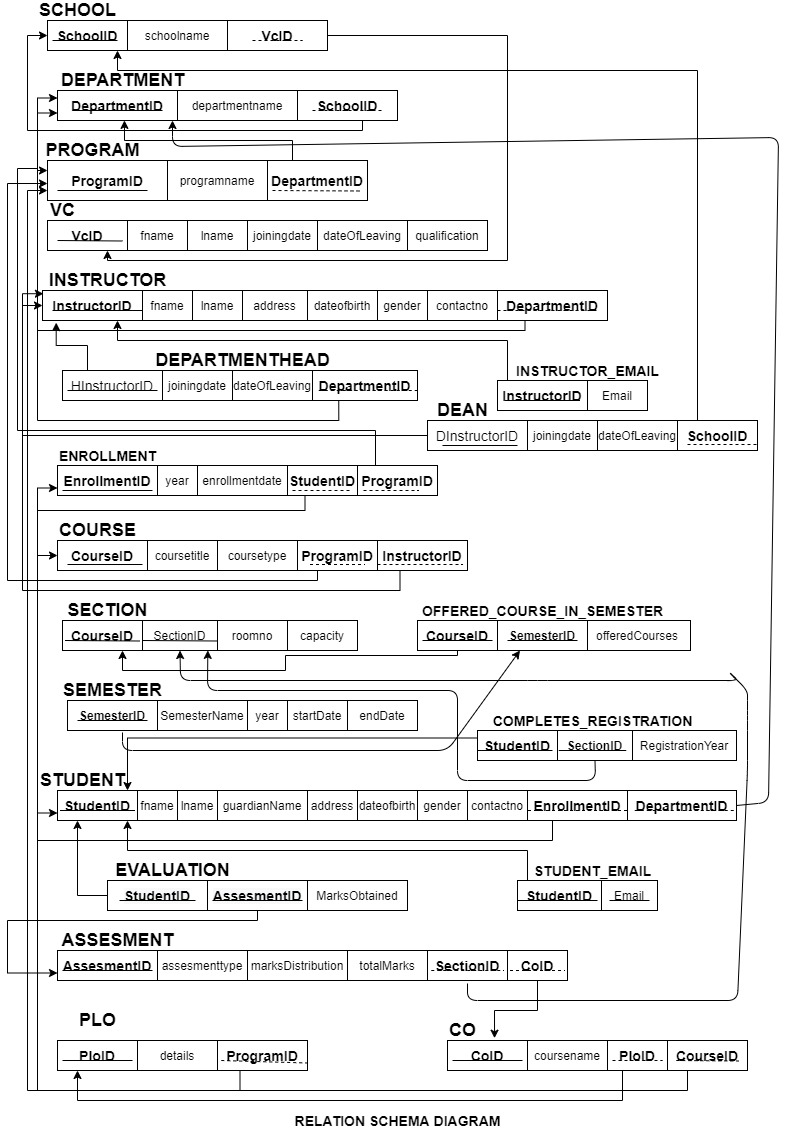
10) A student may tale multiple assessments. A particular assessment must be taken exactly by one student.

11) A section may have multiple assessments. An assessment must have one exact section.

**ENTITY RELATIONSHIP DIAGRAM**



**ENTITY RELATIONSHIP DIAGRAM TO RELATIONAL SCHEMA**

****

**NORMALIZATION**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| School | SchoolID | s1 | Enrollment | enrollemntID | n1 |
| year | n2 |
| School name | s2 | Enrollment date | n3 |
| VCID | v1 | studentID | t1 |
| ProgramID | p1 |
| VC | vcID | v1 | student | studentID | t1 |
| Fname | v2 | fname | t2 |
| Iname | v3 | iname | t3 |
| Joining datew | v4 | City | t4 |
| Leaving date | v5 | Road | t5 |
| Qualification | v6 | Area | t6 |
| Department | departmentID | d1 | Date of birth | t7 |
| Departmentname | d2 | Gender | t8 |
| Contact no | t9 |
| schoolID | S1 | Nationality | t10 |
| program | programID | p1 | enrollmentID | n1 |
| Program name | p2 | departmentID | d1 |
| departmentID | d1 | Assessment | Assessment ID | a1 |
| Instructor | InstructorID | i1 | Assessment type | a2 |
| fname | i2 | Marks distribution | a3 |
| Iname | i3 | sectionNO | e1 |
| City | i4 | studentID | t1 |
| Area | i5 | COID | o1 |
| Road | i6 | PLOID | l1 |
| Date of birth | i7 | Student complete assessment | a4 |
| Gender | i8 | Student marks obtained | a5 |
| Contact no (gmail) | i9 | Course | courseID | c1 |
| DepartmentID | d1 | Course title | c2 |
| Department  Head | departrmentheadID | h1 | Course type | c3 |
| qualification | h2 | programID | p1 |
| Joining date | h3 | InstructorID | i1 |
| Date of leaving | h4 | semesterID | r1 |
| Dean | deanID | x1 | Section | sectionNO | e1 |
| Annual salary | x2 | courseID | c1 |
| Joining date | x3 | Room no | e2 |
| Date of leaving | x4 | capacity | e3 |
| Start time | e4 |
| End time | e5 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| PLO | PLOID | l1 | Semester | SemesterID | r1 |
| Details | l2 | year | r2 |
| programID | p1 | Start date | r3 |
| CO | COID | o1 | End date | r4 |
| Course name | o2 |  |  |  |
| PLOID | l1 |

|  |  |
| --- | --- |
| s1-> | s 2, v1 |
| v1-> | v2,v3,v4,v5,v6 |
| d1-> | d2,s1 |
| p1-> | p2,d1 |
| i1-> | i2,i3,i4,i5,i6,i7,i8,i9,d1 |
| i1,h1-> | h2,h3,h4 |
| i1,x1-> | x2,x3,x4 |
| n1-> | n2,n3,t1,p1 |
| t1-> | t2,t3,t4,t5,t6,t7,t8,t9,t10,n1,d1 |
| a1-> | a2,a2,a3,a4,a4,e1,t1,o1,l1 |
| c1-> | c2,c3,p1,i1,r1 |
| e1-> | e2,e3,e4,c1 |
| l1-> | l2,p1 |
| o1-> | o2,l1 |
| r1-> | r2,r3,r4 |

|  |  |
| --- | --- |
| SchoolID-> | School name, VCID |
| vcID -> | Fname,iname, Joining date, Leaving date, Qualification |
| departmentID  -> | Department name, schoolID |
| ProgramID-> | Program name , departmentID |
| Instructor ID-> | Fname, iname, city, area, road, date of birth, gender , contact no (gmail), departmentID |
| Instructor Deartment HeadID-> | Qualification , joining date , date of leaving |
| Instructor DeanID-> | Annual salary, joining date ,date of leaving |
| enrollmentID-> | Year, enrollment date, studentID, programID |
| studentID-> | Fname,iname,city,road,area,date of birth,gender , contact no(gmail),nationality , enrollmentID, departmentID |
| Assessment > | Assessment type, marks distribution, sectionNO,, studentID, COID ,PLOID, student complete assessment , student marks distribution |
| courseID-> | Course title, course type, programID, intructorID, semesterID |
| sectionNO-> | courseID, room no, capacity , start time |
| PLOID-> | Details, programID |
| CO-> | Course name , PLOID |
| SemesterID-> | Year, start time, end date |

**1NF**

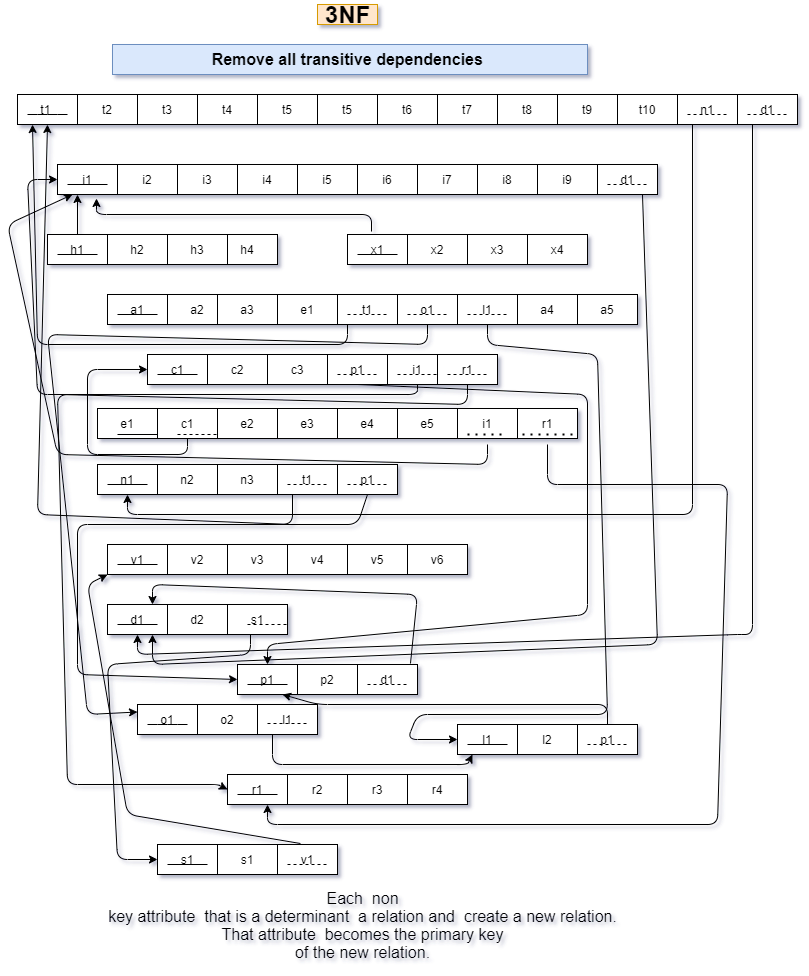
If a relation that has a primary key and in which there are no repeating groups will be 1nf.

But our functional dependency table and relation have repeating groups and a primary key has not defined so the relation not will be 1nf.

**2NF**

If a relation in first normal form in which every non key attribute is fully functionally dependent on the primary key and a functional dependency in which one or more non key attributes are functionally dependent on part of the primary key that’s time relation will be 2NF.

But our relation not fully dependent on primary key and functional dependency have not any non-key attributes are not functionally dependent on primary key. This relation not also 1NF. So the relation not will be 2NF.

**3NF **

**BCNF**

Relation which does not have multiple overlapping candidate keys and every determinant is a candidate key is said to be in BCNF.

Each and every functional dependency relation have candidate key and s1, v1, d1, p1, i1, x1, n1, t1, a1, c1, e1, l1, o1, r1 are candidate key identify non key attribute. This relation has not multiple overlapping and no non-key attribute can identify primary key. So all relation in a BCNF.

**DATA DICTIONARY**

VC\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nvcid | Number | 7 | This is the Primary Key for VC.  Example: “19\*\*\*\*\*” |
| cname | Text |  | This is the name of vc Example: “md khan” |
| djoiningDate | DateTime |  | This contains the date when vc took charge of his role.  Example : “01.01.2016” |
| dleavingDate | DateTime |  | This contains the date when vc discharged from his role.  Example : “01.01.2020” |
| cqualification | Text |  | This contains the qualification of vc Example “ PHD , BSC” |

School\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| cschoolid | Text | 5 | This is the Primary Key of SchoolExample: “SETS” |
| Cschoolname | Text |  | This is the name of the School.Example: “School of Engineering, Technologyand Science” |
| nvcid | Number |  | This is the foreign key from the VC table. Example: “19\*\*\*\*\*. “ |

Department\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| cdepartmentid | Text |  | This is the Primary Key of the Department.Example: “EEE” |
| cdepartmentname | Text |  | This is the name of the Department. Example: “Computer Science and Engineering” |
| cschoolid | Text |  | This is the Foreign Key of the table School.Example: “SETS” |

Student\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nstudentid | Number |  | This is the Primary Key for the Student. Example: “1800001” |
| cname | Text |  | This is the name of the Student. Example: “Muhammad Akib” |
| cguardianname | Text |  | This is the name of the guardian. Example: “Muhammad karim” |
| caddress | Text |  | This is the address of the Student. Example: “House 270, Road 6, Block C, Bashundhara, Dhaka, Bangladesh |
| ddateofbirth | Datetime | “dd/mm/yy” | This the Date of Birth of the Student. Example: “01-01-2000” |
| cgender | Text |  | This is the gender of the Student. Example: “M” |
| ncontactno | Number |  | This is the phone number of the Student. Example: “0191211141” |
| cemail | Text |  | This is the email address of the Faculty.Example: “mahady@iub.edu.bd” |
| cnationality | Text |  | This contains nationality of the student  Example: “Bangladeshi” |
| cdepartmentid | Text |  | This is the Foreign Key from the Department table. Example: “CSE |

Instructor\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| ninstructorid | Number |  | This is the Primary Key for Faculty. Example: “1501001“ |
| cname | Text |  | This is the first name of the instructor.  Example : “ Abdur Rahim” |
| caddress | Text |  | This is the address of the instructor. Example: “House 1, Road 1, Sector 1, Uttara, Dhaka, Bangladesh |
| ddateofbirth | DateTime | DD-MM-Y YYY | This the Date of Birth of the instructor. Example: “01-01-1993” |
| cgender | Text |  | This is the gender of the instructor . Example: “F” |
| ncontactno | Number |  | This is the phone number of the instructor. Example: “01910101010” |
| cemail | Text |  | This is the email address of the **instructor**.Example: “rakib@iub.edu.bd” |
| cdepartmendid | Text |  | This is the Foreign Key from the Department table. Example: “CSE” |

Departmenthead\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| djoiningdate | DateTime |  | This contains the date when a department head took charge of his role  Example : “01.01.16” |
| dleavingdate | DateTime |  | This contains the date when a department head discharged from his role  Example : “01.01.2020” |
| cdepartmentid | Text |  | This is the Foreign Key from the Departmenttable.Example: “CSE |

Dean\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| djoiningdate | DateTime |  | This contains the date when a dean took charge of his role  Example : “01.01.2016” |
| dleavingdate | DateTime |  | This contains the date when a dean discharged from his role  Example : “01.01.2020” |
| cschoolid | Text |  | This is the Foreign Key from the schooltable.Example: “SETS” |

PLO\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| cploid | Text | 5 | This is the primary key for Program LearningOutcome.Example: “PLO1” |
| cdetails | Text |  | This is the details of the Program LearningOutcome.Example: “An ability to select and apply theknowledge, techniques, skills, and modern tools of the computer science andengineering discipline” |
| cporgramid | Text |  | This is the foreign key from Program tableExample: ”B.Sc”. |

CO\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| ccoid | Text |  | This is the Primary Key for Course Outcome.  Example: “CO1” |
| cdescription | Text |  | This is the description of the course |
| cploid | Text |  | This is the foreign key from the Program Learning Outcome table. Example: “PLO1” |

Enrollement\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nenrollmentid | Number |  | This is the Primary Key for Enrollment  Example : “ 1” |
| dyear | Datetime |  | This is the year of Enrollment Example: “2017” |
| denrollmntdate | DateTime | DD-MM-Y YYY | This contains the date of the enrollment.  Example : 30/01/2021 |
| nstudentid | Number |  | This is the Foreign key from the Student Table. Example: “1800001” |
| cprogramid | Text |  | This is the Foreign Key from Program table Example: ”B.Sc”. |

Program\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| cprogramid | Text |  | This is the Primary Key for program.  Example : “BSC” |
| cprogramname | Text |  | This is the name of the program .  Example : “Bachelor of Science” |
| cdepartmentid | Text |  | This is the Foreign Key from the Department table. Example: “CSE” |

Course\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| ccourseid | Text |  | This is the Primary Key for the Course. Example: “CSE203” |
| ccoursetitle | Text |  | This is the name of the Course. Example: ”Data Structure” |
| ccoursetype | Text |  | This is the type of the Course.Example: “Core” |
| cprogramid | Text |  | This is the Foreign Key from Program table Example: ”B.Sc”. |

Section\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nsectionid | Number |  | This is the Primary Key for Section  Example :”2" |
| croomno | Text |  | This is the room number .  Example : “B7107” |
| ncapacity | Number |  | This contains the total capacity of a room  Example : “50” |

Assesment\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nassesmentid | Number |  | This is the Primary Key for assessment .  Example : “1” |
| cassesmenttype | Text |  | This is the type of assessment .  Example : “Assignment ,Viva” |
| cmarksdistribution | Text |  | This contains the marks distribution |
| Ctotalmarksachievable | Text |  | This contains how much mark a student can achieve in total .  Example : “100” |
| nsectionid | Number |  | This is the foreign key from section table  Example” 1001” |
| ccoid | Text |  | This is the foreign key from the ProgramLearning Outcome table.Example: “CO1” |

Semester\_T

|  |  |  |  |
| --- | --- | --- | --- |
| Name | DataType | Size | Remark |
| nsemesterid | Number |  | This is the Primary Key for semester  Example:1 |
| csemesterName | Text |  | This is the name of the semester  Example: “Fall” |
| dyear | DateTime |  | This contains the year of that semester .  Example :” 2021” |
| dstartdate | DateTime |  | This is the starting date of the semester .  Example : “15.02.21” |
| denddate | DateTime |  | This is the ending date of the semester .  Example : “10.05.21” |

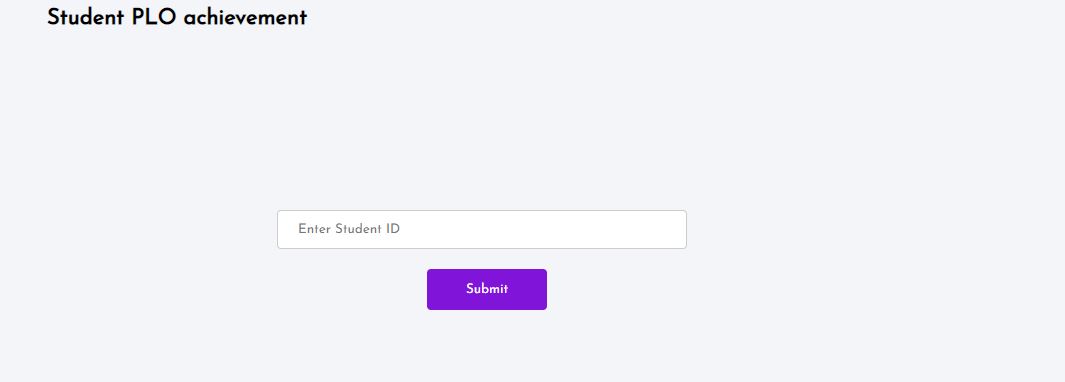
**CHAPTER 4**

**PHYSICAL SYSTEM DESIGN**

INPUT FORMS

OUTPUT GRAPHS

**INPUT FORMS**

****

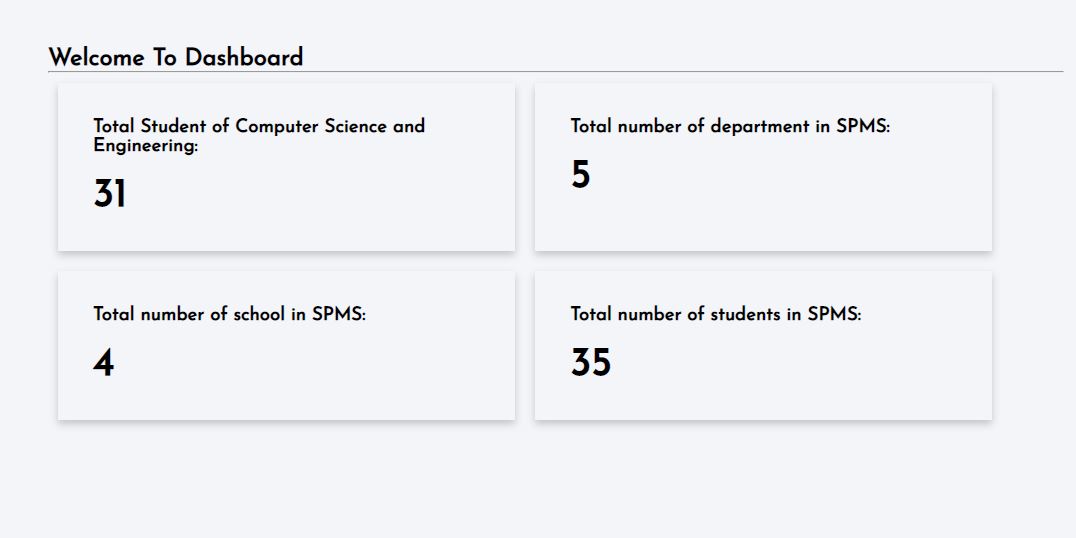
     <form name="register" action="table.php" method="post">

           <input type="text" name="first"   placeholder="Enter Student ID"/>

           <input type="submit" name="submit" value="Submit" />

     </form>

**OUTPUT TABLES & GRAPHS**



**Dashboard**

**<?php**

**$con = mysqli\_connect("localhost", "root", "", "spms");**

**$sql = '**

**SELECT**

**COUNT(student.StudentID) AS total**

**FROM student**

**INNER JOIN**

**department ON department.DepartmentID = student.DepartmentID**

**INNER JOIN**

**school ON school.SchoolID = department.SchoolID**

**WHERE school.SchoolName = "School of Computer Science and Engineering"';**

**$fire = mysqli\_query($con, $sql);**

**$result = mysqli\_fetch\_assoc($fire);**

**echo $result['total'];**

**?>**

**<?php**

**$con = mysqli\_connect("localhost", "root", "", "spms");**

**$sql = '**

**SELECT**

**COUNT(department.DepartmentID) AS total**

**FROM department**

**WHERE 1';**

**$fire = mysqli\_query($con, $sql);**

**$result = mysqli\_fetch\_assoc($fire);**

**echo $result['total'];**

**?>**

**<?php**

**$con = mysqli\_connect("localhost", "root", "", "spms");**

**$sql = '**

**SELECT**

**COUNT(school.SchoolID) AS total**

**FROM school**

**WHERE 1';**

**$fire = mysqli\_query($con, $sql);**

**$result = mysqli\_fetch\_assoc($fire);**

**echo $result['total'];**

**?>**

**<?php**

**$con = mysqli\_connect("localhost", "root", "", "spms");**

**$sql = '**

**SELECT**

**COUNT(student.StudentID) AS total**

**FROM student**

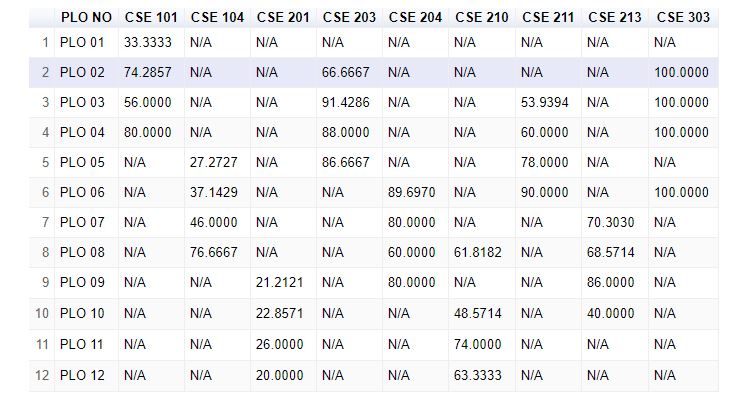
**WHERE 1';**

**$fire = mysqli\_query($con, $sql);**

**$result = mysqli\_fetch\_assoc($fire);**

**echo $result['total'];**

**?>**

****

**Course Wise PLO Achievement**

<?php

$conn = mysqli\_connect("localhost", "root","","spms");

$sql = "SELECT co.CourseID, co.CoNo, plo.PloNO,(PLO/TotalCoMark \* 100) AS PLOpercentage

FROM co,plo, (

SELECT co.CourseID, co.CoNo, co.PloID, evaluation.Obtainedmarks AS PLO,

assessment.Marks AS TotalCoMark

FROM co,evaluation,assessment,plo,registration

WHERE registration.StudentID='1722006'

AND registration.RegistrationID=evaluation.RegistrationID

AND evaluation.AssessmentID=assessment.AssessmentNo

AND assessment.CoID=co.CoID

AND co.PloID=plo.PloNO

GROUP BY registration.SectionID, co.PloID) coursewisePLO

WHERE co.CoNo=coursewisePLO.CoNo

AND plo.PloNO=coursewisePLO.PloID

AND co.CourseID=coursewisePLO.CourseID

ORDER BY coursewisePLO.PloID, coursewisePLO.CourseID";

$result = mysqli\_query($conn, $sql);

$output = mysqli\_fetch\_all($result);

$table = array(

    array('PLO 01', $output[0][3], 'N/A', 'N/A','N/A', 'N/A', 'N/A', 'N/A', 'N/A', 'N/A'),

    array('PLO 02', $output[1][3], 'N/A', 'N/A', $output[2][3], 'N/A', 'N/A', 'N/A', 'N/A', $output[3][3]),

    array('PLO 03', $output[4][3], 'N/A', 'N/A', $output[5][3], 'N/A', 'N/A', $output[6][3], 'N/A', $output[7][3]),

    array('PLO 04', $output[8][3], 'N/A', 'N/A', $output[9][3], 'N/A', 'N/A', $output[10][3], 'N/A', $output[11][3]),

    array('PLO 05', 'N/A', $output[12][3], 'N/A', $output[13][3], 'N/A', 'N/A', $output[14][3], 'N/A', 'N/A'),

    array('PLO 06', 'N/A', $output[15][3], 'N/A', 'N/A', $output[16][3], 'N/A', $output[17][3], 'N/A', $output[18][3]),

    array('PLO 07', 'N/A', $output[19][3], 'N/A', 'N/A', $output[20][3], 'N/A', 'N/A', $output[21][3], 'N/A'),

    array('PLO 08', 'N/A', $output[22][3], 'N/A', 'N/A', $output[23][3], $output[24][3], 'N/A', $output[25][3], 'N/A'),

    array('PLO 09', 'N/A', 'N/A', $output[26][3], 'N/A', $output[27][3], 'N/A', 'N/A', $output[28][3], 'N/A'),

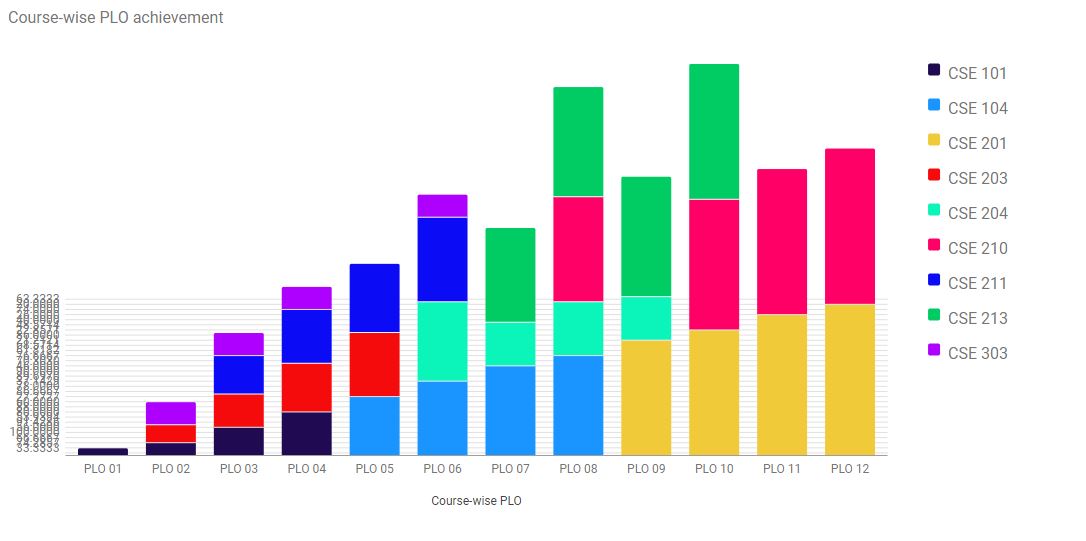
    array('PLO 10', 'N/A', 'N/A', $output[29][3], 'N/A', 'N/A', $output[30][3], 'N/A', $output[31][3], 'N/A'),

    array('PLO 11', 'N/A', 'N/A', $output[32][3], 'N/A', 'N/A', $output[33][3], 'N/A', 'N/A', 'N/A'),

    array('PLO 12', 'N/A', 'N/A', $output[34][3], 'N/A', 'N/A', $output[35][3], 'N/A', 'N/A', 'N/A'),

);

?>

****

**Course wise PLO achievement**

          <?php

          $con = mysqli\_connect("localhost", "root", "", "spms");

          $sql = '

              SELECT

                COUNT(student.StudentID) AS total

              FROM student

              INNER JOIN

                department ON department.DepartmentID = student.DepartmentID

              INNER JOIN

                school ON school.SchoolID = department.SchoolID

              WHERE school.SchoolName = "School of Computer Science and Engineering"';

          $fire = mysqli\_query($con, $sql);

          $result = mysqli\_fetch\_assoc($fire);

          echo $result['total'];

          ?>

          <?php

          $con = mysqli\_connect("localhost", "root", "", "spms");

          $sql = '

              SELECT

                COUNT(department.DepartmentID) AS total

              FROM department

              WHERE 1';

          $fire = mysqli\_query($con, $sql);

          $result = mysqli\_fetch\_assoc($fire);

          echo $result['total'];

          ?>

        <?php

          $con = mysqli\_connect("localhost", "root", "", "spms");

          $sql = '

              SELECT

                COUNT(school.SchoolID) AS total

              FROM school

              WHERE 1';

          $fire = mysqli\_query($con, $sql);

          $result = mysqli\_fetch\_assoc($fire);

          echo $result['total'];

          ?>

          <?php

          $con = mysqli\_connect("localhost", "root", "", "spms");

          $sql = '

              SELECT

                COUNT(student.StudentID) AS total

              FROM student

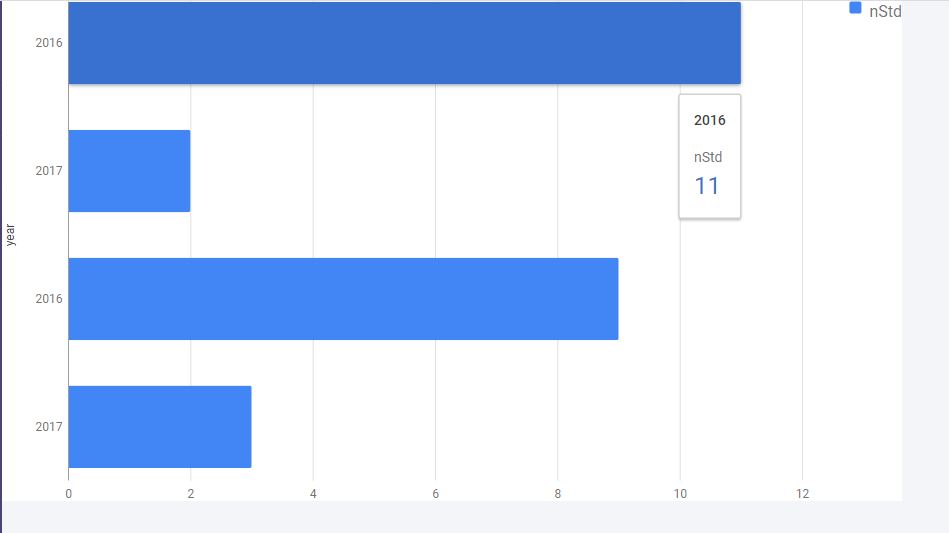
              WHERE 1';

          $fire = mysqli\_query($con, $sql);

          $result = mysqli\_fetch\_assoc($fire);

          echo $result['total'];

          ?>

****

**Program wise student Enrollment**

          <?php

          $sql = "SELECT count(student.studentId) as nStd,program\_enrollment.enrollmentYear, semester.semesterId, program.programId, program.programName

          from student

          inner join program\_enrollment on student.studentId=program\_enrollment.studentId

          inner join program on program\_enrollment.programId=program.programId

          inner join semester on program\_enrollment.semesterId=semester.semesterId

          where program\_enrollment.enrollmentYear in ('2016','2017') and program.programId in ('1','2')

          group by program.programId,program\_enrollment.enrollmentYear";

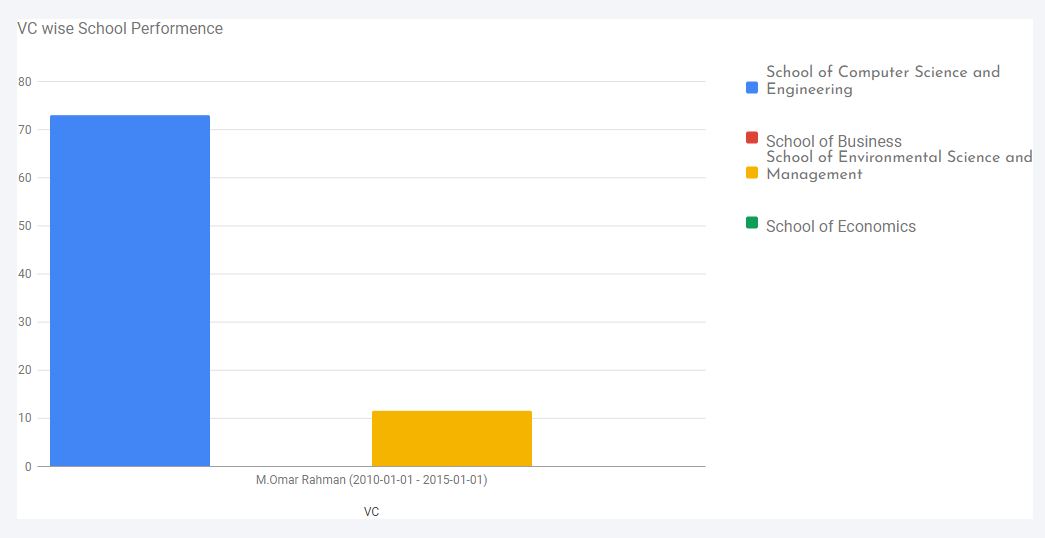
          $fire = mysqli\_query($con, $sql);

          while ($result = mysqli\_fetch\_assoc($fire)) {

            echo "['" . $result['enrollmentYear'] . "'," . $result['nStd'] . "],";

          }

          ?>



**VC wise School Performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$cgpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_school= "SELECT \* FROM `school`";

$vc\_id = 1;

$barchartdata .="[ 'M.Omar Rahman (2010-01-01 - 2015-01-01)' ,";

$fire\_school = mysqli\_query($con, $sql\_school);

while ($result\_school = mysqli\_fetch\_array($fire\_school)) {

$SchoolName = '"'.$result\_school['SchoolName'].'"';

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$sql = '

SELECT

student.Name,student.StudentID, (evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNo = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

INNER JOIN

department ON department.DepartmentID = student.DepartmentID

INNER JOIN

school ON school.SchoolID = department.SchoolID

INNER JOIN

vc ON vc.VcID = school.VcID

WHERE student.StudentID = '.$StudentID." AND school.SchoolName = ".$SchoolName."AND vc.VcID = ".$vc\_id;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$cgpa += $gpa;

$sum = 0;

$gpa = 0;

}

}

$barchartdata .= $cgpa.",";

$cgpa = 0;

}

$barchartdata .="],";

?>

<?php



**VC wise school performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$cgpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_school= "SELECT \* FROM `school`";

$vc\_id = 2;

$barchartdata .="[ 'Milan Pagon (2016-01-01 - 2020-01-01)' ,";

$fire\_school = mysqli\_query($con, $sql\_school);

while ($result\_school = mysqli\_fetch\_array($fire\_school)) {

$SchoolName = '"'.$result\_school['SchoolName'].'"';

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$sql = '

SELECT

student.Name,student.StudentID, (evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNo = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

INNER JOIN

department ON department.DepartmentID = student.DepartmentID

INNER JOIN

school ON school.SchoolID = department.SchoolID

INNER JOIN

vc ON vc.VcID = school.VcID

WHERE student.StudentID = '.$StudentID." AND school.SchoolName = ".$SchoolName."AND vc.VcID = ".$vc\_id;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$cgpa += $gpa;

$sum = 0;

$gpa = 0;

}

}

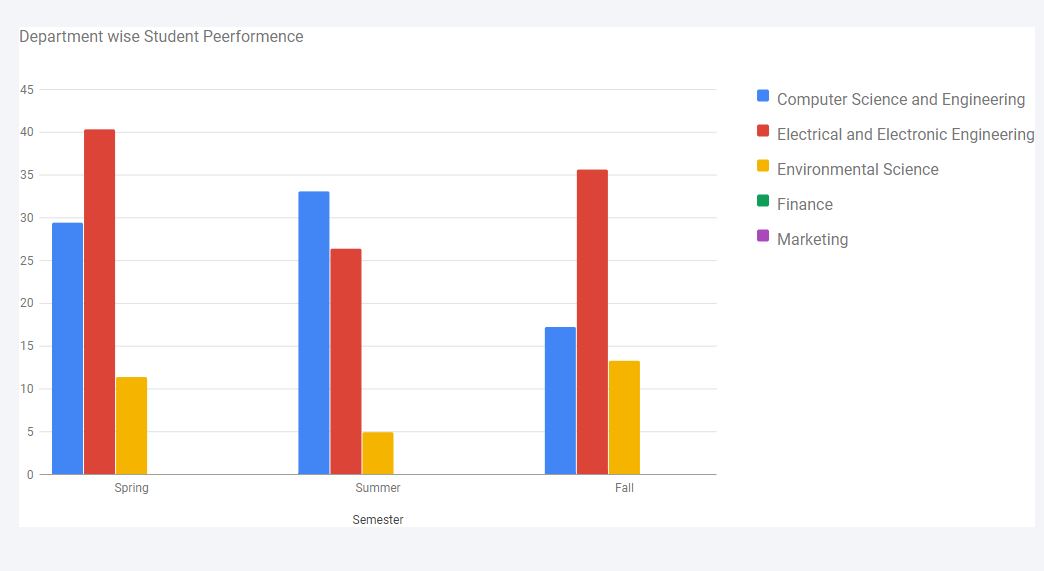
$barchartdata .= $cgpa.",";

$cgpa = 0;

}

$barchartdata .="],";

?>



**Department Wise Student Performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$cgpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_semester= "SELECT \* FROM `semester`";

$sql\_department= "SELECT \* FROM `department`";

$fire\_semester = mysqli\_query($con, $sql\_semester);

while ($result\_semester = mysqli\_fetch\_array($fire\_semester)) {

$SemesterName = '"'.$result\_semester['SemesterName'].'"';

$barchartdata .="[".$SemesterName.",";

$fire\_department = mysqli\_query($con, $sql\_department);

while ($result\_department = mysqli\_fetch\_array($fire\_department)) {

$DepartmentName = '"'.$result\_department['DepartmentName'].'"';

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$sql = '

SELECT

student.Name,student.StudentID, (evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNo = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

INNER JOIN

department ON department.DepartmentID = student.DepartmentID

WHERE semester.SemesterName = '.$SemesterName." AND student.StudentID = ".$StudentID." AND department.DepartmentName = ".$DepartmentName;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$cgpa += $gpa;

$sum = 0;

$gpa = 0;

}

}

$barchartdata .= $cgpa.",";

$cgpa = 0;

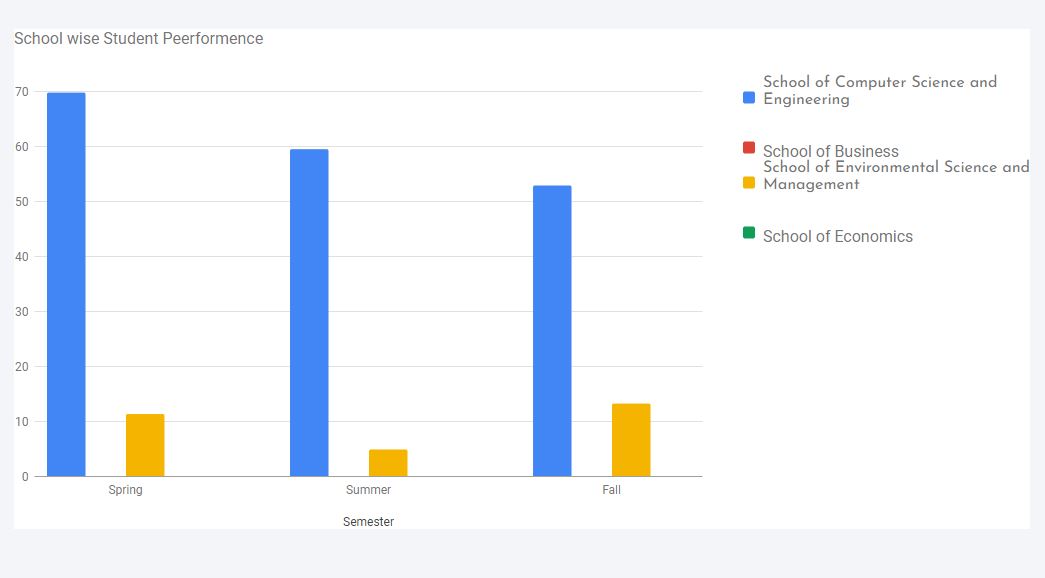
}

$barchartdata .="],";

}

?>

**Dashboard**



**School wise student Performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$cgpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_semester= "SELECT \* FROM `semester`";

$sql\_school= "SELECT \* FROM `school`";

$fire\_semester = mysqli\_query($con, $sql\_semester);

while ($result\_semester = mysqli\_fetch\_array($fire\_semester)) {

$SemesterName = '"'.$result\_semester['SemesterName'].'"';

$barchartdata .="[".$SemesterName.",";

$fire\_school = mysqli\_query($con, $sql\_school);

while ($result\_school = mysqli\_fetch\_array($fire\_school)) {

$SchoolName = '"'.$result\_school['SchoolName'].'"';

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$sql = '

SELECT

student.Name,student.StudentID, (evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNo = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

INNER JOIN

department ON department.DepartmentID = student.DepartmentID

INNER JOIN

school ON school.SchoolID = department.SchoolID

WHERE semester.SemesterName = '.$SemesterName." AND student.StudentID = ".$StudentID." AND school.SchoolName = ".$SchoolName;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$cgpa += $gpa;

$sum = 0;

$gpa = 0;

}

}

$barchartdata .= $cgpa.",";

$cgpa = 0;

}

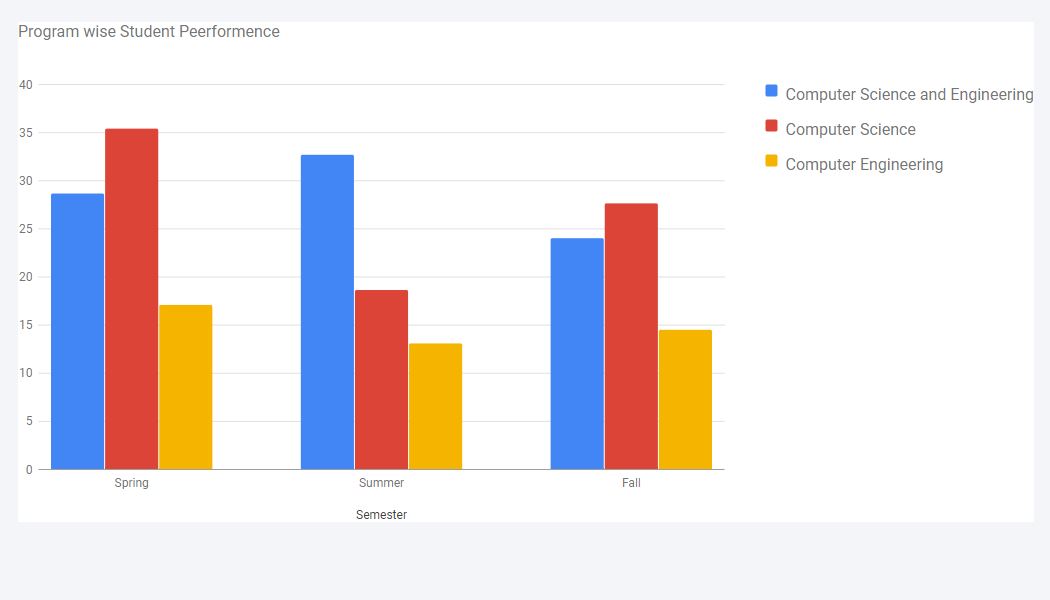
$barchartdata .="],";

}

?>

**Dashboard**

**Dashboard**



**Program wise student Performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$cgpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_semester= "SELECT \* FROM `semester`";

$sql\_program= "SELECT \* FROM `program`";

$fire\_semester = mysqli\_query($con, $sql\_semester);

while ($result\_semester = mysqli\_fetch\_array($fire\_semester)) {

$SemesterName = '"'.$result\_semester['SemesterName'].'"';

$barchartdata .="[".$SemesterName.",";

$fire\_program = mysqli\_query($con, $sql\_program);

while ($result\_program = mysqli\_fetch\_array($fire\_program)) {

$ProgramName = '"'.$result\_program['ProgramName'].'"';

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$sql = '

SELECT

student.Name,student.StudentID, (evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNo = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

INNER JOIN

program\_enrollment ON program\_enrollment.StudentID = student.StudentID

INNER JOIN

program ON program.ProgramID = program\_enrollment.ProgramID

WHERE semester.SemesterName = '.$SemesterName." AND student.StudentID = ".$StudentID." AND program.ProgramName = ".$ProgramName;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$cgpa += $gpa;

$sum = 0;

$gpa = 0;

}

}

$barchartdata .= $cgpa.",";

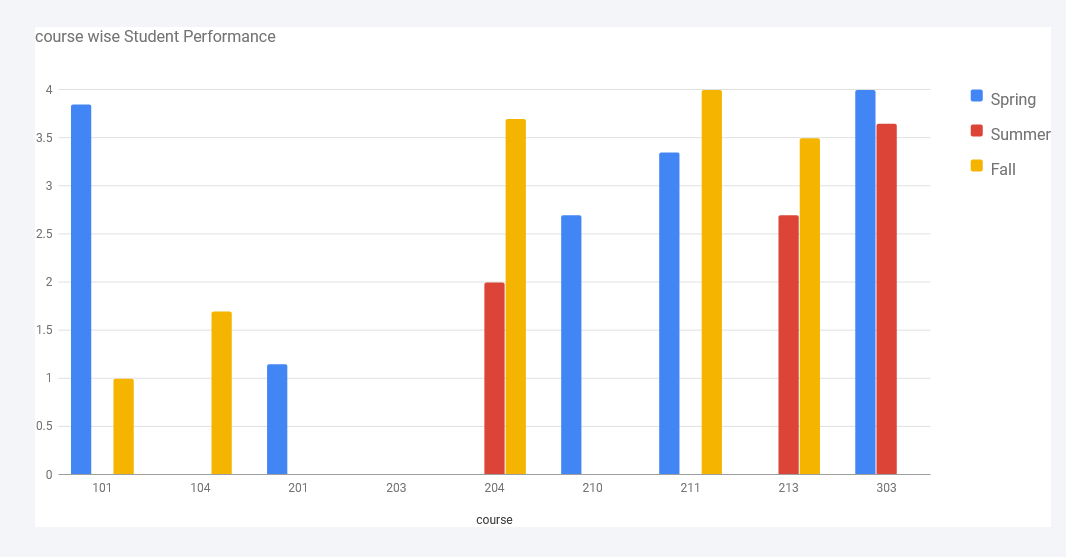
$cgpa = 0;

}

$barchartdata .="],";

}

?>



**Course Wise Student Performance**

<?php

$barchartdata = "";

$count = 0;

$sum = 0;

$gpa = 0;

$sql\_id= "SELECT \* FROM `student`";

$sql\_semester= "SELECT \* FROM `semester`";

$sql\_course= "SELECT \* FROM `course`";

$fire\_course = mysqli\_query($con, $sql\_course);

while ($result\_course = mysqli\_fetch\_array($fire\_course)) {

$CourseID = '"'.$result\_course['CourseID'].'"';

$barchartdata .="[".$CourseID.",";

$fire\_semester = mysqli\_query($con, $sql\_semester);

while ($result\_semester = mysqli\_fetch\_array($fire\_semester)) {

$fire\_id = mysqli\_query($con, $sql\_id);

while ($result\_id = mysqli\_fetch\_array($fire\_id)) {

$StudentID = $result\_id['StudentID'];

$SemesterName = '"'.$result\_semester['SemesterName'].'"';

$sql = '

SELECT

student.StudentID ,course.CourseID,(evaluation.Obtainedmarks / assessment.Marks)\*100 AS FinalMark , semester.SemesterName

FROM

evaluation

INNER JOIN

assessment ON assessment.AssessmentNO = evaluation.AssessmentID

INNER JOIN

student ON student.StudentID = evaluation.StudentID

INNER JOIN

section ON section.SectionID = assessment.SectionID

INNER JOIN

course ON course.CourseID = section.CourseID

INNER JOIN

semester ON semester.SemesterID = section.SemesterID

WHERE semester.SemesterName = '.$SemesterName." AND student.StudentID = ".$StudentID." AND course.CourseID =

".$CourseID;

$fire = mysqli\_query($con, $sql);

$rows = mysqli\_num\_rows($fire);

if($rows == 0){

}

else{

while ($result = mysqli\_fetch\_assoc($fire)) {

$GradePoint = 0;

$Grade = '';

if($result['FinalMark'] >= 85 && $result['FinalMark'] <= 100){

$GradePoint = 4.0;

$Grade = "A";

}

elseif($result['FinalMark'] >= 80 && $result['FinalMark'] <= 85){

$GradePoint = 3.7;

$Grade = "A-";

}

elseif($result['FinalMark'] >= 75 && $result['FinalMark'] <= 80){

$GradePoint = 3.3;

$Grade = "B+";

}

elseif($result['FinalMark'] >= 70 && $result['FinalMark'] <= 75){

$GradePoint = 3.0;

$Grade = "B";

}

elseif($result['FinalMark'] >= 65 && $result['FinalMark'] <= 70){

$GradePoint = 2.7;

$Grade = "B-";

}

elseif($result['FinalMark'] >= 60 && $result['FinalMark'] <= 65){

$GradePoint = 2.3;

$Grade = "C+";

}

elseif($result['FinalMark'] >= 55 && $result['FinalMark'] <= 60){

$GradePoint = 2.0;

$Grade = "C";

}

elseif($result['FinalMark'] >= 50 && $result['FinalMark'] <= 55){

$GradePoint = 1.7;

$Grade = "C-";

}

elseif($result['FinalMark'] >= 45 && $result['FinalMark'] <= 50){

$GradePoint = 1.3;

$Grade = "D+";

}

elseif($result['FinalMark'] >= 40 && $result['FinalMark'] <= 45){

$GradePoint = 1.0;

$Grade = "D";

}

elseif($result['FinalMark'] >= 0 && $result['FinalMark'] <= 40){

$GradePoint = 0.0;

$Grade = "F";

}

$sum += $GradePoint;

}

$gpa = $sum/$rows;

$sum = 0;

}

}

$barchartdata .= $gpa.",";

$gpa = 0;

}

$barchartdata .="],";

}

?>

**CHAPTER 5**

**CONCLUSION**

PROBLEM AND SOLUTION

ADDITIONAL FEATURES & FUTURE DEVELOPMENT

CONCLUTION & RECOMMENDATIONS

Problem and Solution:

At the beginning it was really hard for us to collect all the correct information of the university due to online class about how PLO CO works, how student grading has been done, how the student enrollment works, how do they calculate GPA. which information a faculty can see about a student, OBE mark sheet etc. But with the help of our respected faculty members we tried to collect all the info as much as we can.

Due to online system it was also difficult for us to work as a group. it was difficult for us to share our ideas and work the whole project as a group simultaneously.

The bounded and short timeframe of this semester and also at the end too much rush situation has hindered our ability to achieve the full potential of this software. But we tried our level best to utilize our time to make the best possible software from the limited resources and time provided, and we also hope to come up with improvements with better analysis when allowed more time.

Additional feature and future development:

The addition of Curriculum Page in the SPMS where members of the Higher Management team  
can add and edit any changes to curriculum. Moreover, faculty members and students can check  
these updates to stay informed about the latest changes.

All the employee from the university will be able to check the SPMS using their id. they can get their valuable information from this system.

Conclusion:

As we planned to make a better and user friendly system for the betterment of the university, students and faculties. We tried to build, design and implement the best possible version of the ideawe had for our SPMS. we added lots of new feature in the system that It will enhance the quality of education of the university. This system is much more informative Faculties will be able to improve their teaching method. they will be able to keep track of the students’ performance more easily. Higher authorities will be able to know much more information. This software is **also** beneficial to the students who want to improve themselves as a better one. it will be also being helpful for the university employees to regulate their resources. This will certainly improve the institution work rate much faster and it will be a great boost up for the institution.

Recommendation:

Give some more time to the project so that we can implement our ideas completely and finish the whole project nicely.

Hire some people who can develop this software if there is any necessity