**PERFORMANCE MEASUREMENT SOLUTION TO**

**MEASURE THE QUALITY OF EDUCATION**

Created By :-

TEAM NAME :- **PERPETUAL MOTION SQUAD**

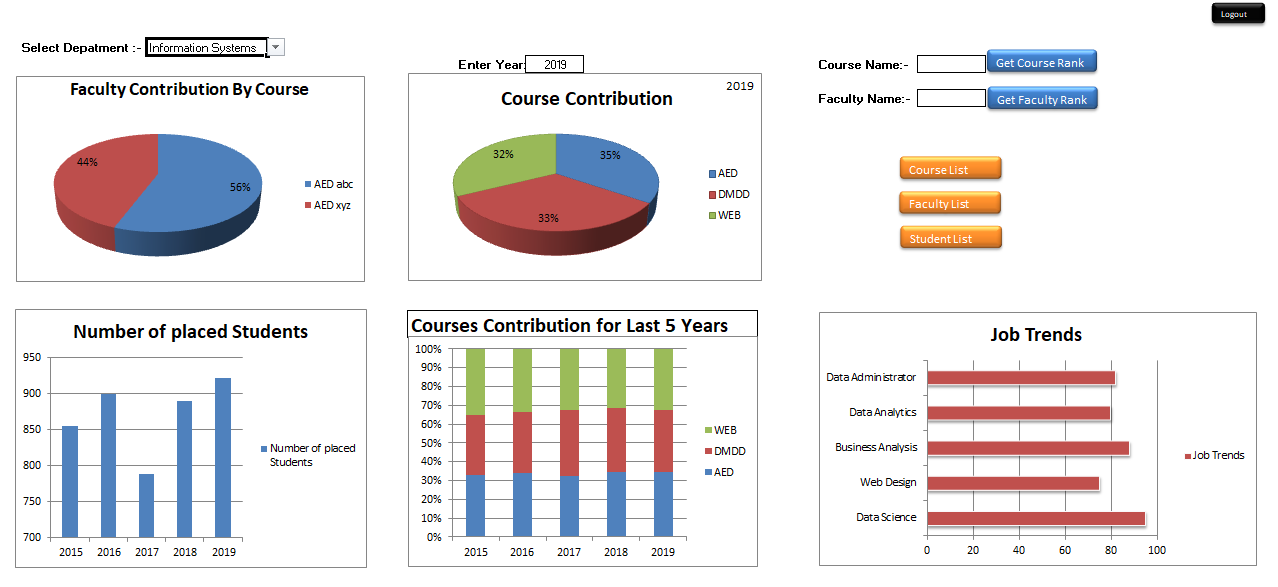
TEAM MEMBERS : -

* SHUBHAM MAHAJAN
* ANURAG RACHCHA
* VIDHI PATEL

**Abstract**

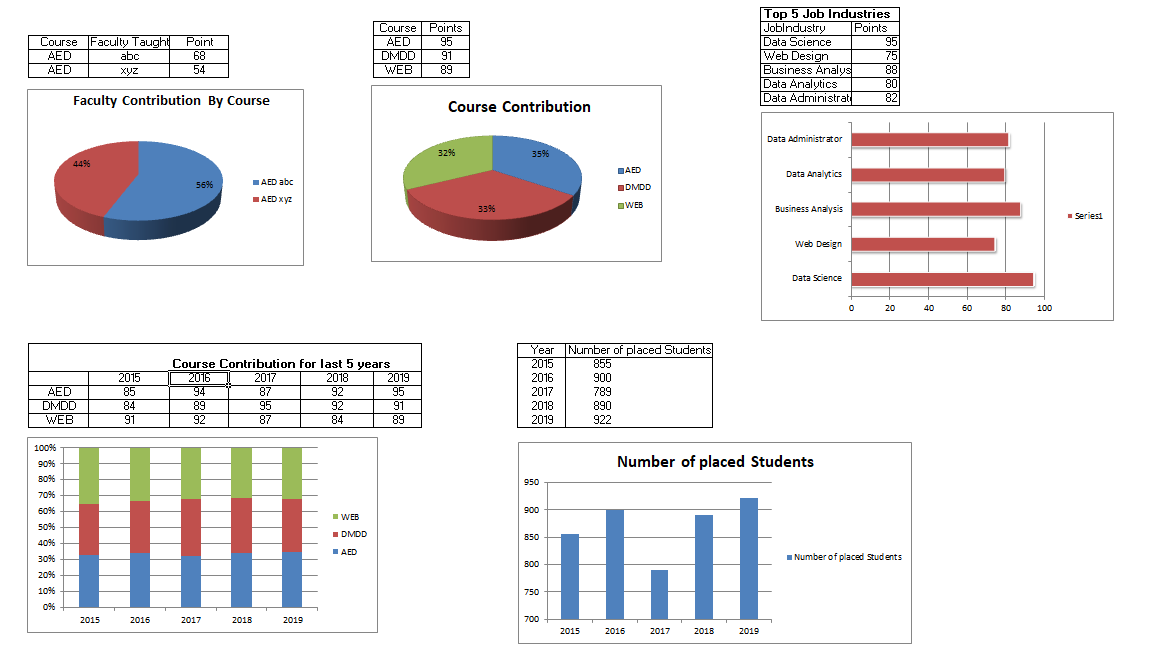
Improvement in the quality of the education is the critical requirement of the society. To improve the quality of education different software engineering techniques required. Here comes the need of object model. This report provides performance measurement solution to enable universities to measure the quality of education they deliver to their students. Object model provides a way by which educational system can track contribution of faculty and course to the growth of their students over 5 year period. It also provides way to track jobs and promotions graduates students get over time. The report also provides dashboard by which academic administrators can compare performance of their academic units.

**Dashboard:-**

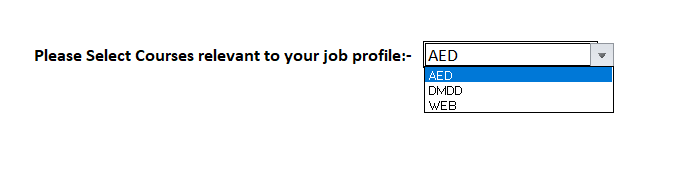
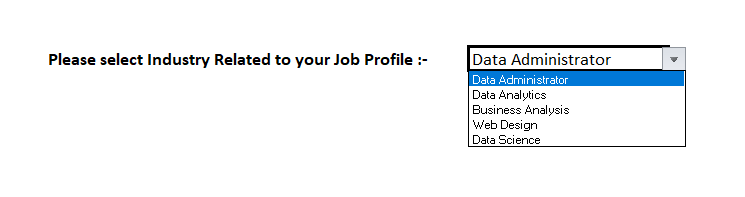


This dashboard enables college and Department administrators to compare the performance of their academic units.

Dashboard consists of different pie chart to provide visual representation of performance , contribution of different academic units. For example, for particular department this dashboard provides faculty contribution by course, Number of placed students, Course contribution over 5 years, Trending Jobs, etc. The department administrator can also check rank of particular course or faculty.

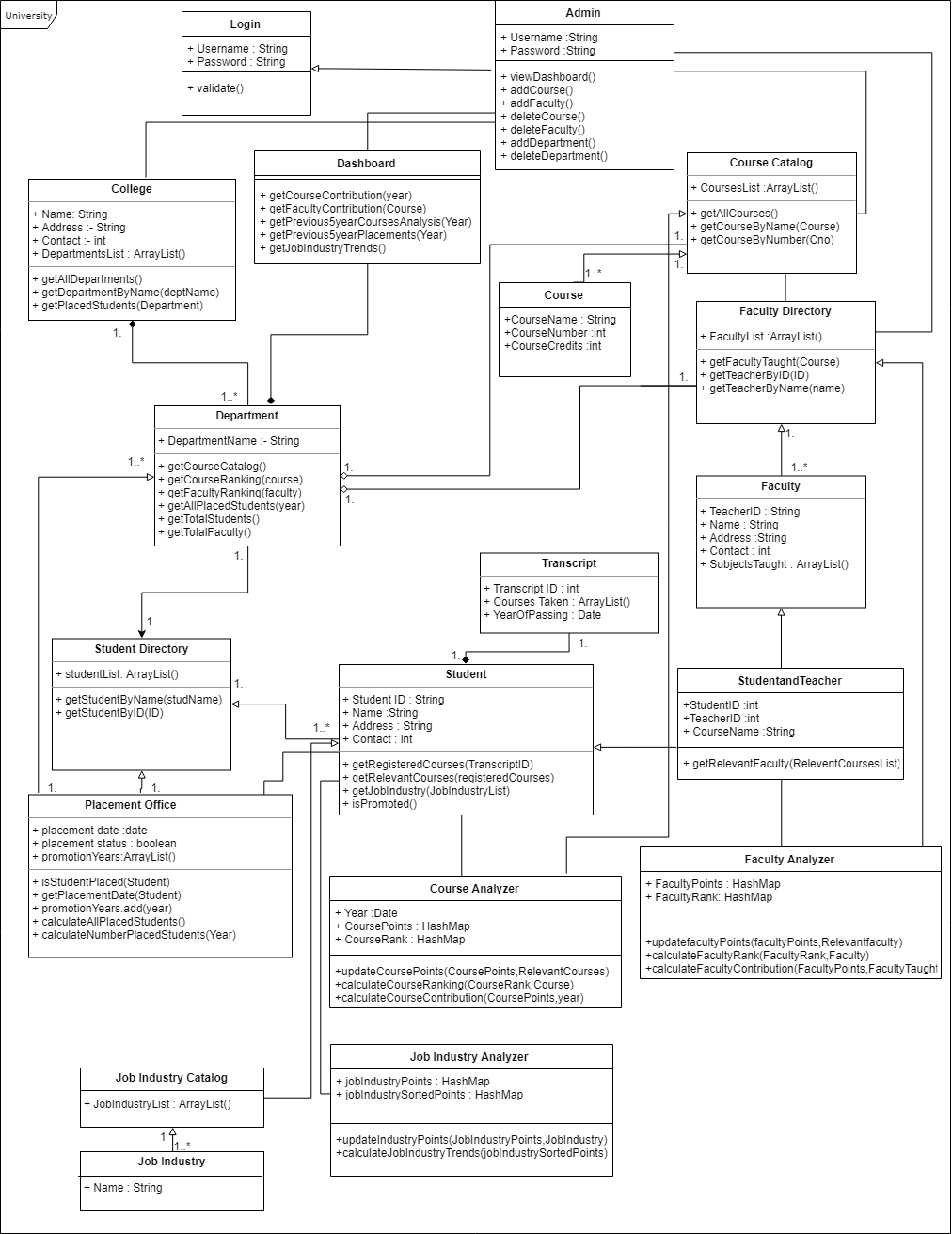


**Student Feeback :-**

This are the parts of Student Information Panel, We are deciding ranks of the Courses based on student feedback. This panel enables student to provide information about the courses he feel relevant to his job. We are also taking Industry in which student is working to get current market trends. Student will also provide us information regarding his job promotions over 5 years. The Idea is, each year student will receive mail asking him information about his job, like is he promoted ? Which industry is he currently working on ? ,etc. The information regarding relevant subjects will be taken only once. Students promotion will also be taken under consideration while ranking the courses which that particular student mareked as relevant for his job.

**Class Diagram :-**



**Classes and Methods:-**

* **Login**

**Validate() :- validate entered Credentials**

* **Admin   
  viewDashboard() :- to view Dashboard  
  addCourse()/removeCourse() :- admin can add or remove courses.  
  addFaculty()/removeFaculty() :- admin can add or remove faculty.**

**addDepartment()/removeDepartment() :- admin can add or remove department.**

* **College   
  getAllDepartments():- return list of all departments**

**getDepartmentByName(deptName):- Search for particular department  
getPlacedStudents(deptName):- returns number of placed students of particular department.**

* **Dashboard  
  getCourseContribution(Year): generates PieChart Course Contribution  
  getFacultyContribution(Course): generates PieChart Faculty Contribution by Course**

**getPrevious5YearCourseAnalysis(Year) :- generates stackedColumn chart course contribution over 5 years  
getPrevious5YearPlacements(Year) : generates column chart Number of placements for last 5 years  
getJobIndustryTrends(): generates Series Chart Trends**

* **Department  
  getCourseCatalog() :- return course list of particular department.**

**getCourseRanking(Course) :- returns rank of entered course.  
getFacultyRanking(Faculty) :- returns rank of entered faculty.**

**getAllPlacedStudents(year) :- returns numbers of students placed for entered year.**

**getTotalStudents() :- returns total number of students in department.**

**getTotalFaculty():- returns number of faculty members in the department.**

* **Course Catalog**

**getAllCourses() :- returns course list.**

**getCourseByName(name) :- Search for course by course name.**

**getCourseByNumber(courseNumber) :- Search for course by course number.**

* **Course**
* **Faculty Directory**

**getFacultyTaught(course):- returns faculty members who teaches entered course.**

**getTeacherByID(ID) :- search for faculty by ID.**

**getTeacherByName(Name) :- Search for faculty by Name.**

* **Faculty**
* **Student Directory  
  getStudentByName(name):- search for student by name.**

**getStudentByID(ID) :- search for student by ID.**

* **Student**

**getRegisterdCourses(TranscriptID):- returns list of courses particular student registered for.  
getRelevantCourse(RegisteredCourses) :- returns list of courses student feel relevant.  
getJobIndustry(JobIndustryList) :- returns job industry of the student.**

**isPromoted():- returns promotion status of the student.**

* **StudentandTeacher  
  getRelevantFaculty(relevantCourses):- returns faculty for the relevant courses entered by student**
* **Job Industry Catalog**
* **Job Industry**
* **Placement office  
  isStudentPlaced(Student) :- checks if entered student is placed or not.**

**getPlacementDate(Student) :- returns date on which entered student got placed.**

**promotionYears.add(Year,Student) :- adds currentyear in promotionYears of the particular student when isPromoted() returns true.**

**calculateNumberPlacedStudents(year) :- Calculated Number of Students got placed in entered year.**

* **Transcript**
* **Course Analyzer**

**updateCoursePoints(coursePoints,RelevantCourses) :-**

**When Student enters his relevant courses, the list of those relevant courses is generated. This list will then pass to updateCoursePoints method of class Course Analyzer. This method has two parameters, coursePoints and relevant courses list. coursePoints is the HashMap in which keys are list of all courses in the department and value is points of that particular course. Whenever updateCoursePoints method called, the points of courses from the relevant course list gets incremented by 1.**

**calculateCourseRanking(courseRank,course) :-**

**This method will return rank of the course user has requested for. The parameter courseRank is HashMap generated by performing descending sort on values of coursePoints HashMap. The first entry in the courseRank is the course with first Rank. As index of HashMap Starts with 0. To get a rank of particular course we wil do index+1.**

**calculateCourseContribution(coursePoints,Year) :-**

**This method will return coursePoints all key , value pairs for requested Year. Using this user can check different years Course Contributions.**

* **Faculty Analyzer**

**updateFacultyPoints(facultyPoints , RelevantFaculty) :-**

**When Student enters his relevant courses, the list of those relevant courses is generated. The faculty who taught that relevant course to that particular student is returned by getRelevantFaculty(RelevantCoursesList) method. The result will then pass to updateFacultyPoints method of class Faculty Analyzer. This method has two parameters, facultyPoints and relevantFaculty(result). facultyPoints is the HashMap in which keys are list of all faculties in the department and value is points acquired by particular faculty. Whenever updateFacultyPoints method called, the points of faculty from the relevant faculty list gets incremented by 1.**

**calculateFacultyRank(facultyRank,course) :-**

**This method will return rank of the faculty user has requested for. The parameter facultyRank is HashMap generated by performing descending sort on values of facultyPoints HashMap. The first entry in the facultyRank is the faculty with first Rank. As index of HashMap Starts with 0. To get a rank of particular faculty we wil do index+1.**

**calculatefacultyContribution(facultyPoints,facultyTaught) :-**

**When user wants faculty contribution of particular course. First list of faculties taught that particular course will be generated from getFacultyTaught(Course) method of Faculty Directory class. This result is then send as a facultyTaught to parameter of calculatefacultyContribution methond. This method will return facultyPoints all faculties received from facultyTaught parameter. Using this user can check different faculties Contributions for particular course.**

* **Job Industry Analyzer**

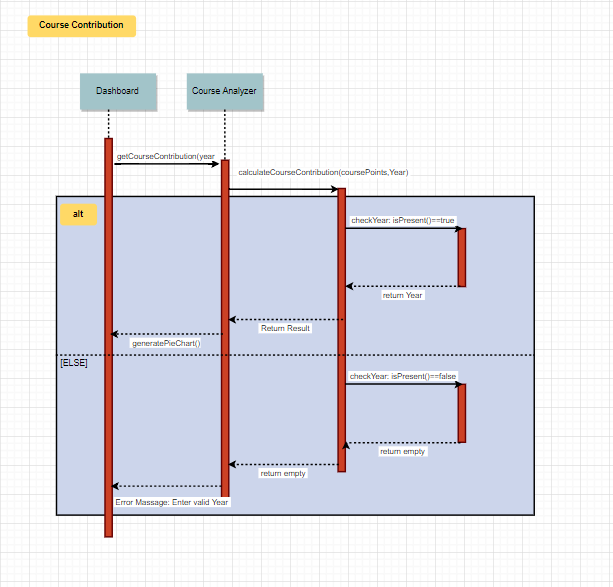
**updateIndustryPoints(JobIndustryPoints , JobIndustry) :-**

**Student’s entered job Industry will be passed to updateIndustryPoints method of class Job Industry Analyzer. This method has two parameters, jobIndustryPoints and JobIndustry. jobIndustryPoints is the HashMap in which keys are list of all Job Industries and value is points of that particular job industry. Whenever updateIndustryPoints method called, the point for entered job industry will be incremented by 1.**

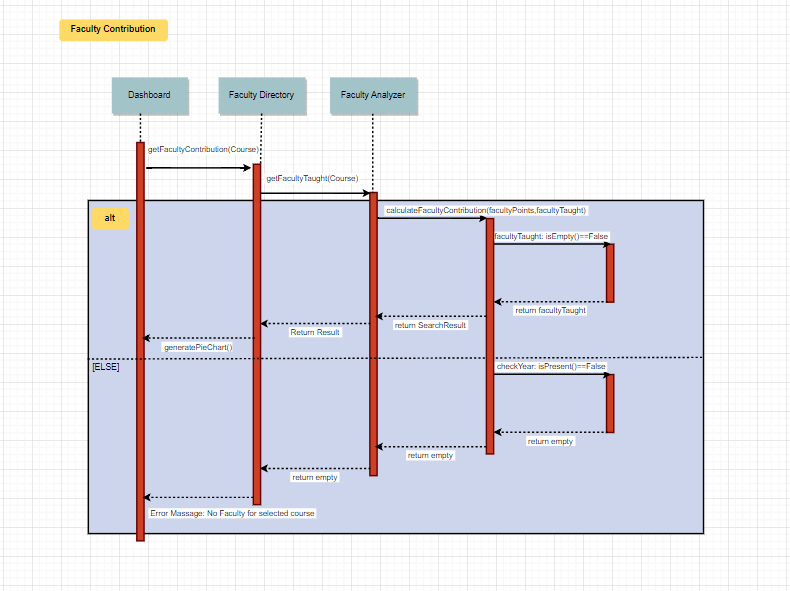
**calculateJobIndustryTrends(JobIndustrySortedPoints) :-**

**This method will return top 5 Job Industries. The parameter JobIndustrySortedPoints is HashMap generated by performing descending sort on values of JobIndustryPoints HashMap. The first entry in the JobIndustrySortedPoints is the industry with first Rank. As index of HashMap Starts with 0. To get top 5 job industries we will return industries from index 0 to index 4.**

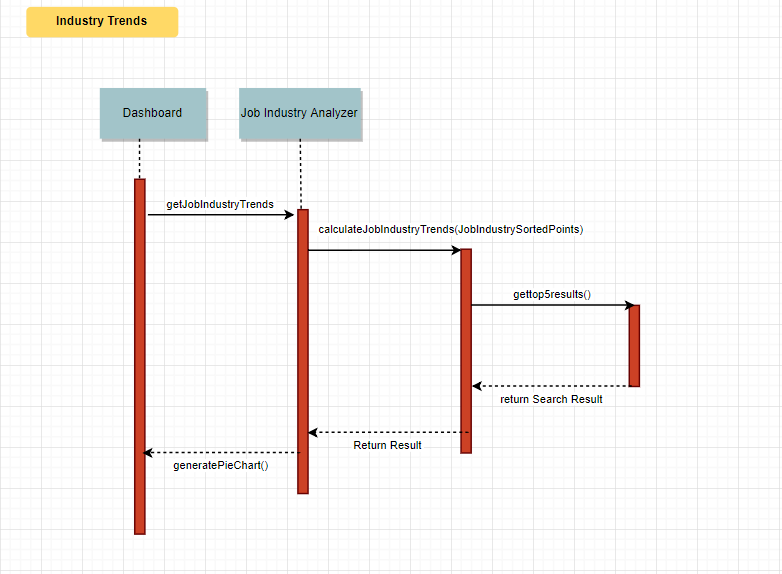
**Sequence Diagram [Course Contribution] :-**



**Sequence Diagram [Faculty Contribution] :-**



**Sequence Diagram [Industry Trends] :-**



**Conclusion :-**

To review, a performance measurement solution to measure quality of education of a university is provided by turning object model into a machine of information gathering and data aggregation.