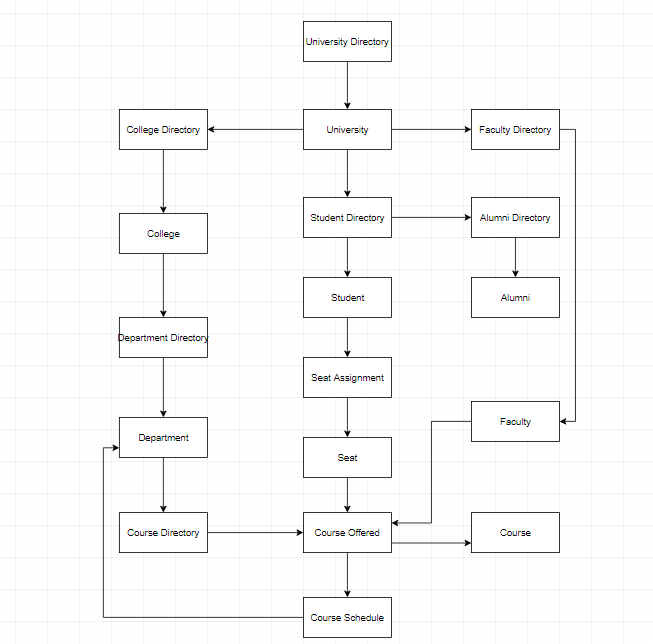
**REPORT**

**Abstract:**

This report contains explanation of a University model and its functionalities which are described by the means of different diagrams such model diagrams, class diagrams, use case diagrams. To determine what factors are to be taken into consideration by the University to measure the quality of education that is being provided to its students by the means of feedback and surveys. Also, it is essential to keep a track of the growth of the graduates over a period of 5 years, the progress that the university alumni are making by tracking the jobs and promotions they attain so that will play a major role in the university statistics and ranking. It is also important to determine who the relevance of the courses will impact the growth of the graduate so that the performance of the units can be accounted for.

**Object Model :**

Object-oriented design is built upon a sound engineering foundation, whose elements we collectively call the object model. This model encompasses the principles of abstraction, encapsulation, polymorphism, inheritance. The below object model depicts the general working of how an admin can control the working of a list of universities and depict factors like quality of education, alumni jobs and promotions, their university ranking. The university directory gives a list of all the universities which can be added or removed by the administrative officers. Every university have various departments which consists of a list of colleges like COE, CCIS, etc from where a student can be part of any one college. Many departments may be present in each college for example IS under COE, etc. The student chooses courses from their respective departments when the faculty opens the course for a particular term.

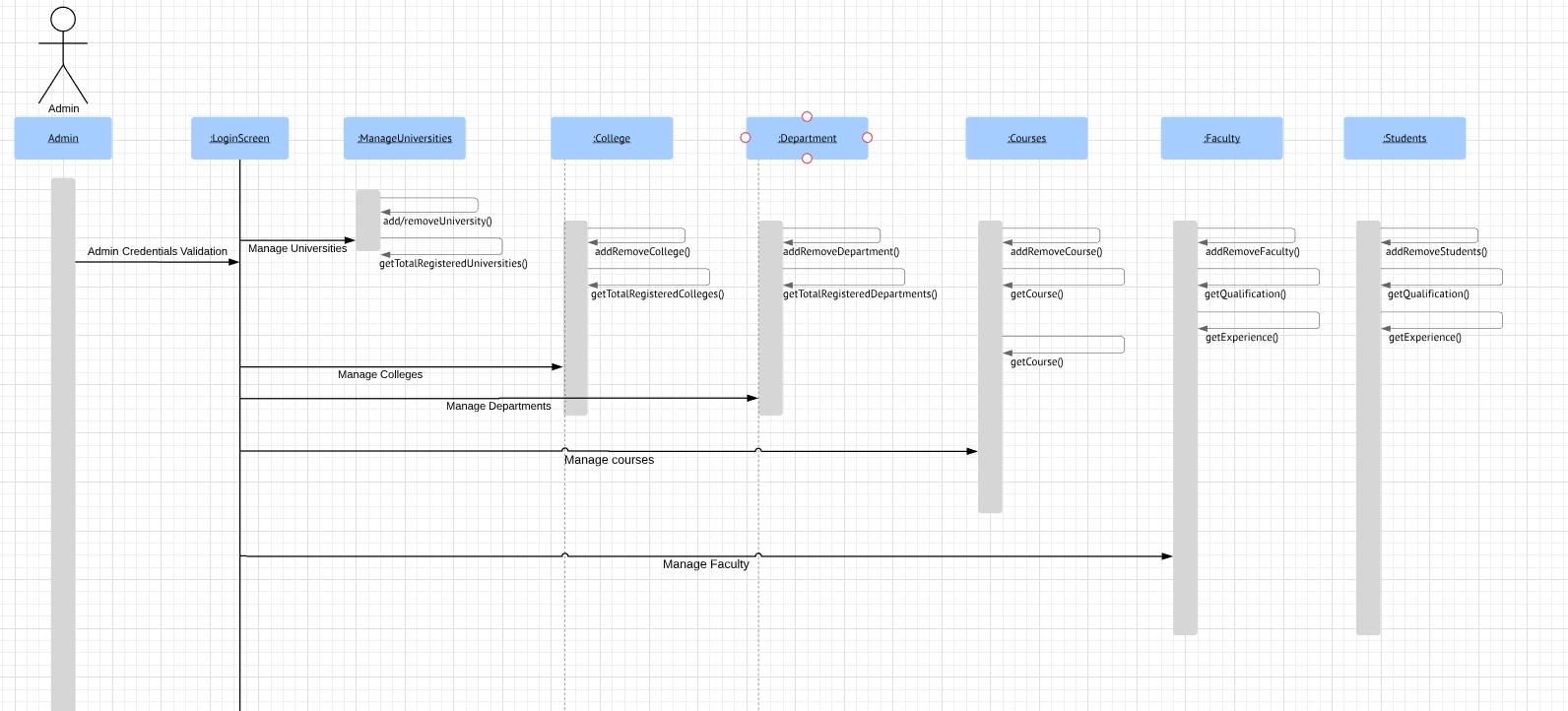


**Object Model of Universities:**

This object model of Universities contains essential entitites which are defined as objects in the object model diagram below. The Object model has the following entitites.

* University Directory
* University
* College Directory
* College
* Department Directory
* Department
* Course Directory
* Course Offered
* Course Scheduled
* Course
* Faculty Directory
* Faculty
* Student Directory
* Student
* Alumni Directory
* Alumni

**Sequence Diagram :**



**Operations Example :**

University Directory

Manages all the university details pertaining to all the schools offered to students. Ex: Northeastern University, Harvard University, MIT

* UniversityDirectory.getTotalNumberOfRegisteredUniversities() - returns the total number of universities which are registered
* UniversityDirectory.add/removeUniversity() – adds or removes university from university list

College Directory

Manages all the college’s details pertaining to that university.

Ex: COE, CCIS, CPS in Northeastern University

* CollegeDirectory.getTotalNumberOfColleges() - returns the total number of universities which are registered
* CollegeDirectory.add/removeCollege() – adds or removes university from college list

Department Directory

Details of all the departments in a particular college of a university.

Ex: Information Systems (Department) in COE(College)

* DepartmentDirectory.getTotalNumberOfDepartments() - returns the total number of universities which are registered
* DepartmentDirectory.getTotalNumberOfRegisteredStudents() – returns the total number of students registered under the department
* DepartmentDirectory.add/removeDepartment() – adds or removes department from college list
* DepartmentDirectory.getCoursesWithEmptySeats() – returns if the courses have empty seats. If a course has no empty seats, then the course is in demand and it is in-line with the industry standards and also, student’s interests. Hence showing the quality of the courses being offered.

Course Directory

Represents all the courses being offered to students for a particular term. E.g.: AED, DMDD, Web Design

* CourseDirectory.getTotalNumberOfCourses() - returns the total number of universities which are registered
* CourseDirectory.getCourse() – returns the course details
* CourseDirectory.add/removeCourse() – adds or removes university from college list
* CourseDirectory.getCourseByName(name) – returns the course name based on the names
* CourseDirectory.getCourseRating() – returns the course rating which is students feedback

Faculty Directory

Manages all the details of all the faculty in the university.

* FacultyDirectory.getTotalNumberOfFaculty() - returns the total number of faculty associated with the specific university.
* FacultyDirectory.getDepartment() - returns the department name the faculty belongs to.
* UniversityDirectory.add/removeFaculty() – adds or removes university from university list.
* FacultyDirectory.getQualification() - returns the level of qualification of the faculty.
* FacultyDirectory.getExperience() - returns the number of years of experience of the faculty.
* FacultyDirectory.getFacultyCalculation() – This method calculates the rating of the faculty based of few parameters

Student Directory

Manages all the student details who are admitted into the university.

* StudentDirectory.getTotalNumberOfRegisteredStudents() - returns the total number of universities which are registered
* StudentDirectory.getRegisteredCourses() – will give the details of the courses that the student has registered.
* UniversityDirectory.add/removeStudent() – adds or removes students from student list.
* StudentDirectory.isAlumni() – will determine if the student has graduated or currently studying at the university.
* StudentDirectory.getGPA() – will fetch the GPA of the student.
* StudentDirectory.facultyFeedback()- provides the faculty feedback provided by the students.

Alumni Directory

Manages all the university alumni details and keeps a track of their job details and promotions.

* AlumniDirectory.getTotalNumberOfAlumni() - returns the total number of universities which are registered.
* UniversityDirectory.add/removeAlumni() – adds or removes alumni from alumni list.
* AlumniDirectory.alumniPromotion() – returns the number of times the alumni has been promoted.
* AlumniDirectory.alumniPlacement()- returns percentage of the alumni placed and which will help .
* AlumniDirectory.isPlaced() – sets to true if the Alumni is placed into a company.

Represents that a specific seat in the course is assigned to a student. The class is responsible for the student status in the class as well as the student grades (midterm, final, homework solutions, etc

Seat Assignment

* SeatAssignemnt.getCourse() - returns the courses taken by the student
* SeatAssignemnt.getStudent() - returns the student who has taken the course
* SeatAssignemnt .getStudentGrade() - returns the total student grades

Represents that a specific seat in the course is assigned to a student. The class is responsible for the student status in the class as well as the student grades (midterm, final, homework solutions, etc

Course offered

* CourseOffered.getCoursesWithEmptySeats() -returns the courses with empty seats for the course
* CourseOffered.getTotalNumberOfSeats() - returns the total number of seats available for the course
* CourseOffered.getTotalNumberOfFilledSeats() - returns the total number of filled seat by the students for the course

**Operation Details**

**getFacultyCalculation() –**

facultyPoints = 0;

For each faculty associated with the department:

Nested If to calculate experiencePoints for Faculty:

Nested If to calculate qualificationPoints based on the qualification of the faculty:

Nested If to calculate feedbackPoints from students for the faculty:

facultyPoints = experiencePoints + qualificationPoints + feedbackPoints

return facultyPoints;

**getCourseCalculation() –**

totalCoursePoints = 0;

For each course associated with the department:

NestedIf to calculate coursePoints to get based on getCoursesWithEmptySeats():

Nested If to calculate feedbackPoints from students for the Course:

totalCoursePoints = coursePoints +feedbackPoints

return totalCoursePoints;

**isAlumni()-**

For each student:

If graduateDate < currentDate:

Student is Alumni;

**alumniPlacement() –**

employedCount =count(isPlaced = True)

employedPercentage = employedCount/Total Alumni \* 100

**alumniPromotion() –**

for each alumni:

Nested If to calculate alumniPoints based on jobLevel:

Nested If to calculate alumniPromotionPoint based on experience:

**courseRelevance() –**

for each alumni :

If courseUndertaken is same as job business field:

set courseRelevant to true;

relevancePercentage = (count(courseRelevant)/total alumni)\*100

Nested If to set the rating for relevancePercentage:

**universityRanking()-**

calculated based on getFacultyCalculation() , getCourseCalculation(), alumniPlacement().

Citation