Simran Gupta(002124423)
Unnati Kolhe(002133343)
Shubhangi Srivastava(002197298)

10/22/2021

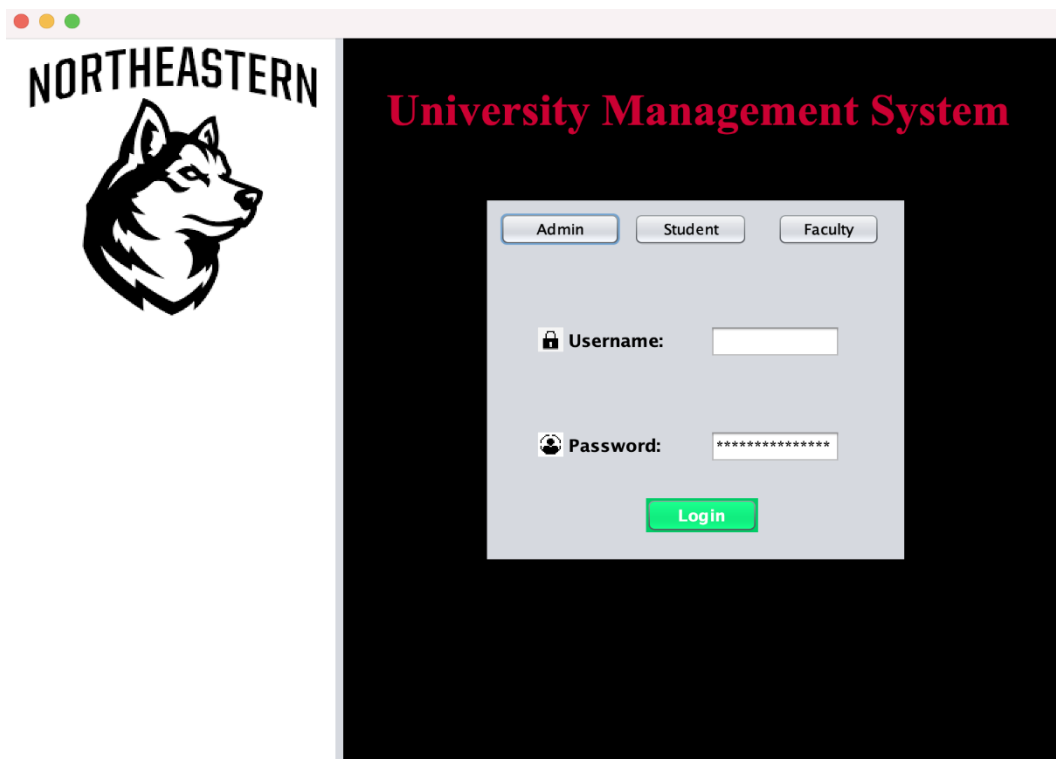# University Management System

Aim and Objective:

The aim of this system is to create a performance measurement solution to enable universities to measure the quality of the education they deliver to their students and the revenue generated by the university every year. We have designed the system to figure out ways to track the jobs graduates get and assign rankings accordingly. We have also analyzed the way courses taken by the students, contribute towards their career growth.

Implementation:

We have created login pages for students, admins, and the faculty members, where they fill in their details and their information will be recorded. In our system the university administrators will be able to compare the performance of the various academic units, also, this ranking system will help the students to understand which department they want to study in and which courses to take.

User Interface:

Login Page:

- Student, administrators, and the faculty members can login with the help of their username and password.
- Administrator has the rights to login to all the other portals, i.e., student, faculty dashboards.

Course Registration:

The students can register for courses by selecting the department and the semester. The available courses can be selected from the drop-down menu for the chosen department and semester. The Add Courses button will help to keep the record of the courses selected by the student.

Department Ranking on the basis of Students Hired:

- Created a Dashboard displaying the Ranking of all the three Departments in both colleges i.e.:
  a) Khoury College under Northeastern University has department Computer Science.
  b) College Of Engineering under Northeastern University has departments:
  - Information Systems
  - Data Architecture Engineering

- The Dashboard attached above, shows a basic view of the total number of Students who are in each department and the students who have been hired from the corresponding department.
- Upon clicking on "Submit", based on the entered data, a Graph is created showing the highest ranked department based on the maximum number of Students who have been hired from the particular department.

Course Ranking based on Students Hired from the highest ranked Department:

- Created a Dashboard displaying the Course Ranking based on the highest ranked Department.
- The Dashboard attached above, shows a basic view of the total number of Students who were hired and the number of students who have registered for the corresponding course.
- Upon clicking on "Submit", based on the entered data, a Pie Chart is created showing the highest ranked course based on the maximum number of hired students who have chosen the course.

Revenue generated by the University based on the Department Ranking :
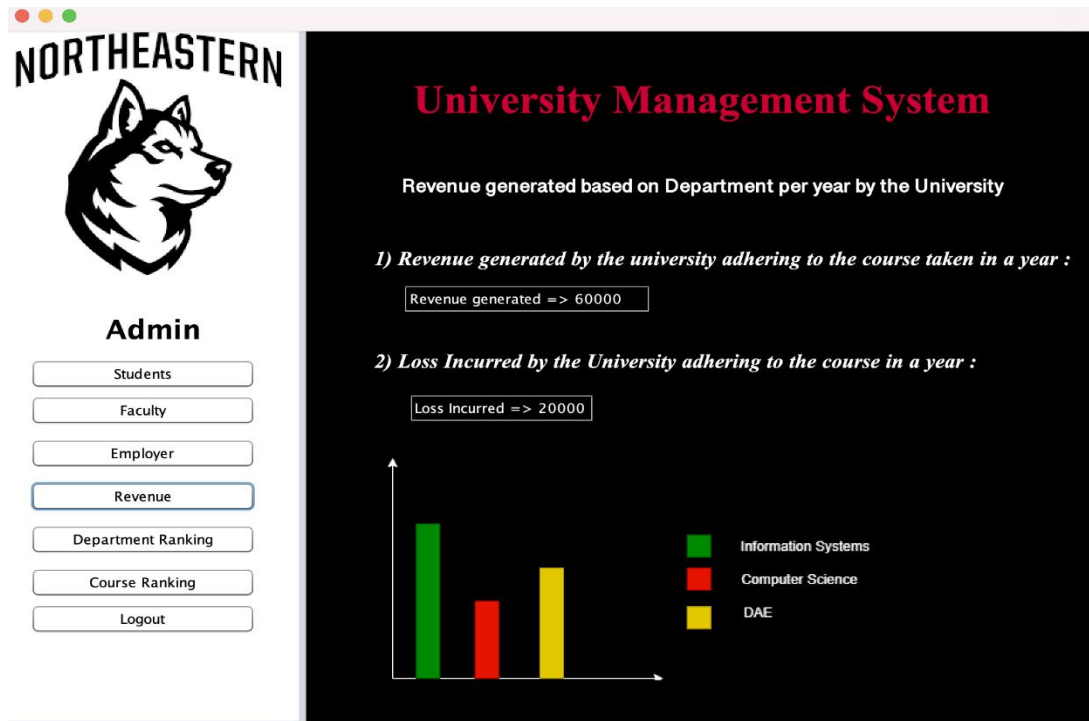
Created a Dashboard to display the revenue generated by the university based on the department ranking every year. This dashboard also displays the loss incurred by the university each year based on the courses.

Ranking Criteria:

- **Popularity of Department based on Hiring:**
  Fetching the records of the students who were hired the most in a particular department and analyzed the most popular department based on it.
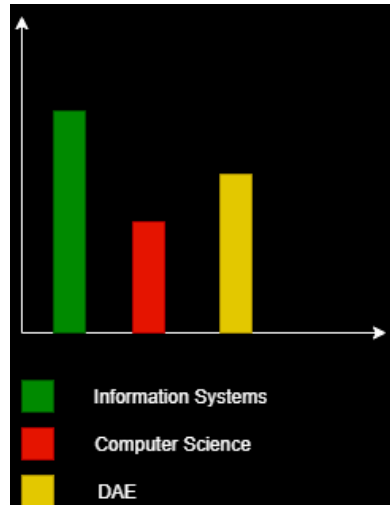
- **Popularity of Course:**
  It is calculated based on courses opted by students in a semester who were hired in each department.

Course Ranking

Cloud Computing: 30
Web Developement: 40
Database Management: 30
Application Engineering and Developement: 350

Application Engineering and Developement   Database Management   Web Developement   Cloud Computing

Class diagram — Info5001UniversityExample

Info5001UniversityExample
- main(args: String [1..*])

Person
- Id : String
- Person(i : String) : Person
- getPersonId() : String
- isMatch(id) : boolean

Degree
- title : String
- courseList: ArrayList <...>
- Degree(name: String) : Degree
- addCoreCourse(c : Course)
- addElectiveCourse(c : Course)
- isStudentReadyToGraduate(sp : StudentProfile) : boolean
- areElectivesSatisfied(sa : SeatAssignment) : boolean
- isValidableCoreClasses(sas : ArrayList) : boolean
- getCoreSatisfied(sas : ArrayList , c : Course) : boolean
- getTotalElectiveCoursesTaken (sas : ArrayList) : int

PersonDirectory
- personList: ArrayList <...>
- PersonDirectory() : PersonDirectory
- newPerson (id : String) : Person
- findPerson (id : String) : Person

CourseCatalog
- lastUpdated : String
- courseList: ArrayList <...>
- getCourseList() : ArrayList <...>
- newCourse (n : String, nm : String , cr : int) : Course
- getCourseList() : ArrayList
- getCourseByNumber (n : String) : Course
- getCourseCatalog (d : Department) : CourseCatalog

CourseSchedule
- courseList : ArrayList <...>
- semester : String
- newCourseOffer (c : String) : CourseOffer
- getCourseOfferByNumber (n : String) : CourseOffer
- calculateTotalRevenues() : int
- getCourseSchedule (s : String, cc : CourseCatalog) : CourseSchedule

WorkRequest
- work : String
- WorkRequest(sng : Person , pm : Person , t : String) : WorkRequest

Department
- name : String
- teamMemberownership: HashMap <...>
- getDepartment (s : String) : Department
- getPersonDirectory () : PersonDirectory
- getStudentDirectory () : StudentDirectory
- newCourseSchedule (semester : String) : CourseSchedule
- getCourseSchedule(semester : String) : CourseSchedule
- getCourseCatalog () : CourseCatalog
- newCourse (n : String , nm : String , cr : int) : Course
- calculateRevenueBySemester (semester : String) : int
- RegisterForAClass (studentId : String , cc : String, semester : String)
- addCoreCourse(c : Course)
- addElectiveCourse(c : Course)

Ranking
- Department : String
- Student_Applied : int
- Student_Selected : int
- Course_Name : String
- Student_Hired : int
- Student_Registered : int
- getDepartment() : String
- setDepartment(Department : String)
- getStudent_Applied() : int
- setStudent_Applied(Student_Applied : int)
- getStudent_Selected() : int
- setStudent_Selected(Student_Selected : int)
- getCourse_Name() : String
- setCourse_Name(Course_Name : String)
- getStudent_Hired() : int
- setStudent_Hired(Student_Hired : int)
- getStudent_Registered() : int
- setStudent_Registered(Student_Registered : int)
- getRank(dept : String, stu_app : int , stu_selec : int , cou_name : String , hired : int , registed : int) : int

Course
- number : String
- name : String
- credits : int
- price : int = 1500
- Course (n : String , numb : String , cn : int) : Course
- getCourseNumber() : String
- getCoursePrice() : int
- getCredits()

Admin
- employeeList : ArrayList <...>
- studentList : ArrayList <...>
- teacherList : ArrayList <...>
- getemployee : ArrayList <...>
- teacherownership : HashMap <...>

CourseOffer
- facultyProfile : FacultyProfile
- getCourseNumber() : String
- generateSeats(n : int)
- getEmptySeat() : Seat
- getTotalCourseRevenues() : int
- getSubjectCourse() : Course
- getCodeHours() : int
- getCourseOffer(c : Course) : CourseOffer
- AssignAsTeacher (fp : FacultyProfile)
- assignEmptySeat(s2 : CourseLoad) : SeatAssignment

FacultyDirectory
- faculyList: ArrayList <...>
- getTopProfessor() : FacultyProfile
- findTeachingFaculty(id : String) : FacultyProfile
- facultyDirectory (d : Department) : FacultyDirectory
- newFacultyProfile (p : Person) : FacultyProfile

StudentDirectory
- studentList: ArrayList <...>
- findStudent(id : String) : StudentProfile
- StudentDirectory (d : Department) : StudentDirectory
- newStudentProfile (p : Person) : StudentProfile

EmployerDirectory
- employeeList : ArrayList <...>
- newEmployerProfile (p : Person) : EmployerProfile
- findTeachingFaculty(id : String) : EmployerProfile
- EmployerDirectory (d : Department) : EmployerDirectory

WorkArea
- WorkQueue : ArrayList <...>
- WorkArea(p : Person) : WorkArea
- addWorkQueueRequest(w : WorkRequest)

DepartmentAccount
- calculateRevenueBySemester (semester) : int
- DepartmentAccount (d : Department) : DepartmentAccount

EmploymentHistory
- employmentList : ArrayList <...>
- EmploymentHistory() : EmploymentHistory
- newEmployment(job : String) : Employment

UserAccount
- UserAccount(p : Person) : UserAccount

Calendar
- courseScheduleList: ArrayList <...>
- Calendar() : Calendar
- addCourseSchedule(semester : String, cs : CourseSchedule)

Address
- street : String
- city : String
- Address() : Address

EmployerProfile
- name : String
- employmentList : ArrayList <...>
- EmployerProfile(p : Person) : EmployerProfile
- isMatch(id : String) : boolean

Transcript
- courseLoadList : HashMap <...>
- newCourseLoad (sem : String) : CourseLoad
- getCurrentCourseLoad() : CourseLoad
- getCourseLoadBySemester (semester : String) : CourseLoad
- getStudentRanking() : float
- getGPA() : float
- getCourseList() : ArrayList <...>
- Transcript(sp : StudentProfile) : Transcript

FacultyAssignment
- rating : double
- getRating() : double
- getProfRating() : double
- getFacultyProfile() : FacultyProfile
- FacultyAssignment(fp : FacultyProfile , co : CourseOffer) : FacultyAssignment

FacultyProfile
- getProfAverageOverallRating() : double
- getCourseOffer(coursed : String) : FacultyProfile
- isMatch(id : String) : boolean
- FacultyProfile(p : Person) : FacultyProfile
- AssignAsTeacher(co : CourseOffer) : FacultyAssignment

CourseLoad
- semester : String
- courseassignments: ArrayList <...>
- CourseLoad(s : String) : CourseLoad
- getSemesterScores() : float
- getSeatAssignments() : ArrayList <...>
- newSeatAssignment (c : CourseOffer) : SeatAssignment
- registerStudent (sa : SeatAssignment)

Seat
- occupied : boolean
- number : int
- isOccupied() : boolean
- getCourseOffer() : CourseOffer
- getSeatCourseCredits() : int
- Seat(c0 : CourseOffer , n : int) : Seat
- newSeatAssignment(cl : CourseLoad) : SeatAssignment

SeatAssignment
- grade : float
- getCreditHours() : int
- getSeat() : Seat
- getCourseOffer() : CourseOffer
- getCourseCreditScores() : float
- SeatAssignment(cl : CourseLoad , s : Seat) : SeatAssignment
- assignSeat(cl : CourseLoad)

Employment
- workareaprocessPlan: ArrayList <...>
- workId : int
- faculty : String
- job : String
- Employment(s : String) : Employment

StudentProfile
- isMatch (id : String) : boolean
- getTranscript() : Transcript
- getCourseLoadBySemester (semester : String) : CourseLoad
- getCurrentCourseLoad () : CourseLoad
- newCourseLoad (s : String) : CourseLoad
- getCourseList() : ArrayList <...>
- StudentProfile(p : Person) : StudentProfile

Sequence diagram with lifelines: Admin, University, College, Department, Faculty, Student, Seat Assignment, Course, Course Offer, Transcript, Employer, Revenue

- Dept List — College → Department
- Student Employment History — Student → Employer
- Faculty Directory — Department → Faculty
- GPA — Student → Transcript
- Course List — Department → Course
- Generates Ranks — Admin → Department
- Assigned — Faculty → Course Offer
- Generates Ranks — University → Department
- Student DIrectory — Department → Student
- Course In — Course → Course Offer
- Course Ranking — Student ⇢ College
- Seat Assigned — Seat Assignment → Course Offer
- Student Feedback — Course ⇢ Faculty
- Employer feedback on Employer feedback on student — Revenue ⇢ College
- Employer List — Student → Employer
- Generates Revenue — College → Revenue