Group 5

Contents

1. Introduction

1.1 Purpose of the system

The aim of the project is to develop an online Graduate Recruitment (GradRec) web portal for the university. The online website will allow the departments to advertise upcoming graduation recruitment (Masters/PhD) programs with research details for the prospective students.

The system will allow students to submit their profiles with the information such as education background, previous research experience and interests, and other relevant details. They will be able to select a program based on their research interests from an available list, and provide their motives and justifications for selecting that specific program.

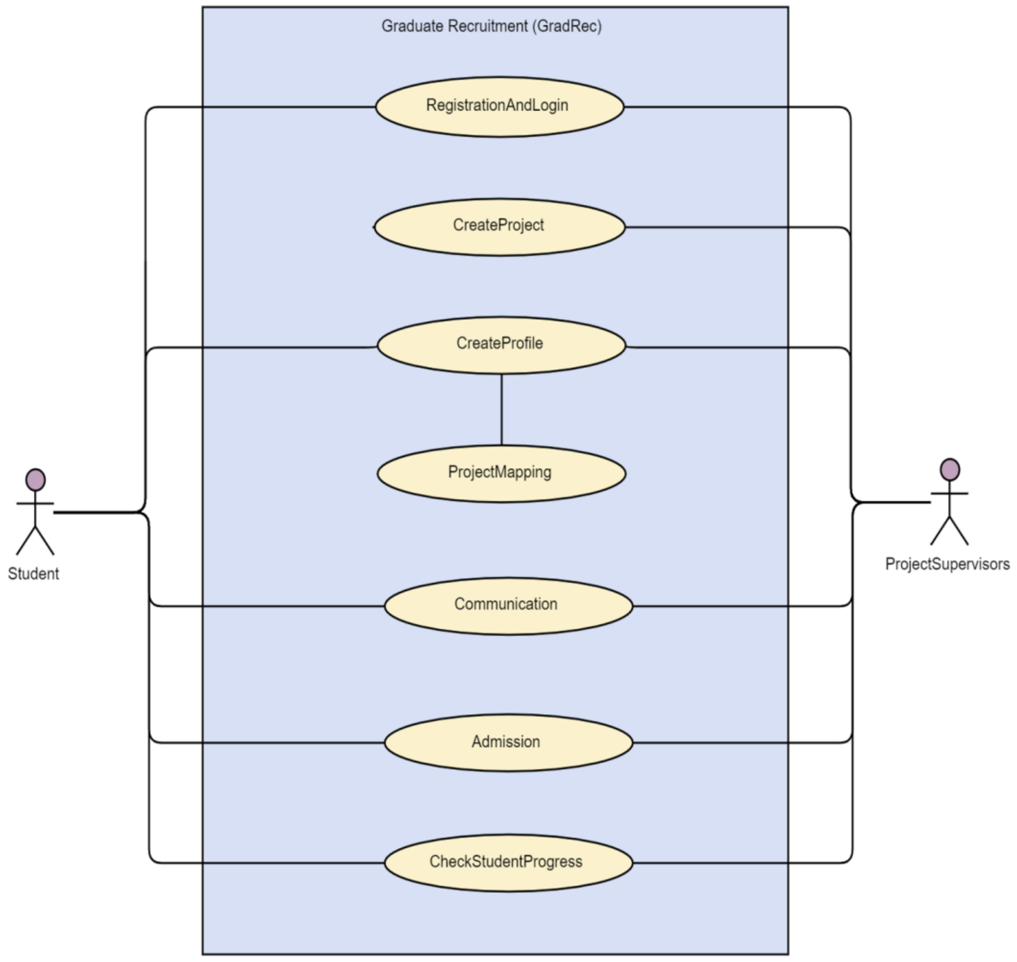
The system is also going to be responsible for handling all future communications between the project management team and the students, and handling functionalities such as offering the formal accept or decline letter to the students and monitoring the progress of the student throughout the life of his/her program.

The GradRec system will use a student - project mapping algorithm to analyze the student submitted profile data and criteria to be selected for a particular project and then notify the project management team or supervisors about the possible student project mapping.

2. System Model

2.1 Use case model

Graduate Recruitment (GradRec) system use case diagram



Use case description:

RegistrationAndLogin:
This use case is for the system security. ***project supervisors*** and ***students*** can use online Graduate Recruitment (GradRec) web portal with proper credential and have access to the various functionality of the system based on their role.

CreateProject:
This use case will allow ***project supervisors*** across all the university departments to register upcoming graduate program and to advertise on online Graduate Recruitment (GradRec) web portal. ***Students*** have access to there matched program and access to more detailed information.

Createprofile And ProjectMapping:
This use case is for ***student*** to submit their profiles with the information such as education background, previous research experience and interests, and any other relevant details. They will also be able to select a program based on their research interests from an available list. ***project supervisors*** able design project criteria which is use by ***student-project mapping algorithm*** to create student-project mapping based on analysis of student profile data and project criteria and in case of possible mapping informed to the project supervisors. ***project supervisors*** are also able to create his/her profile.

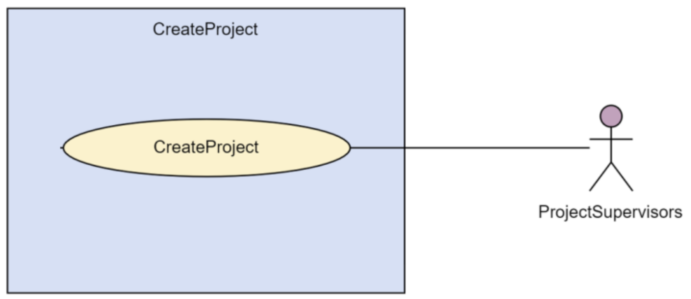
Communication:
This use case is used to create communication channel between ***project supervisors*** and ***students.***

Admission:
This use case is used by ***project supervisors*** to offer formal admission letter and by the ***student*** to accept or decline admission offer and to register.

CheckStudentProgress:
This use case is used by ***project supervisors*** to update student progress details such as grades, to comment on his/her research project and to evaluate student's performance, and ***Student*** can use this use case to keep track of his/her progress.

2.2 Use case textual description

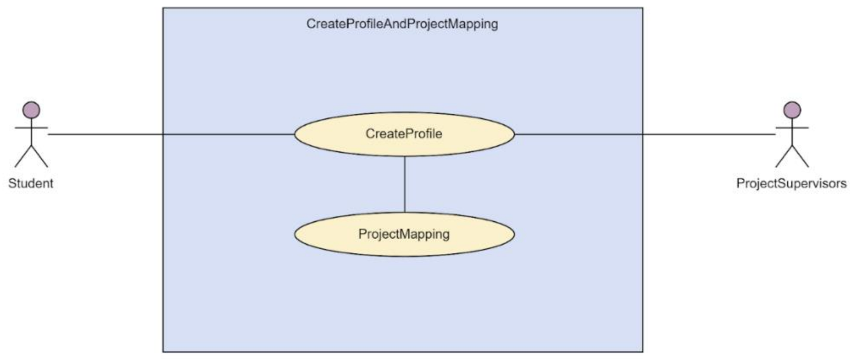
1. CreateProject



Textual Description :

Use case name	***CreateProject***
Participating actors	Initiated by ***ProjectSupervisors*** Participant: *Students* and *ProjectSupervisors*
Flow of events	1. The ***ProjectSupervisors*** initiate the program registration form. 2. The ***ProjectSupervisors*** fills out the form • Save the form as a draft **OR** • Final submission to the G***radRec*** system and it will respond with acknowledgment message. 3. On final submission GradRec system will add program to the catalog for the advertisement and send acknowledgment to ProjectSupervisors.
Entry condition	• The ***ProjectSupervisors*** and Students are identified by **GradRec**
Exit condition	• The ***ProjectSupervisors*** receive confirmation message in case of: 1. Draft submission of the new program, 2. Successful registration of the new program and advertisement.
Quality requirements	• The ***ProjectSupervisors*** must receive an acknowledgment either in case of failed or successful submission of forms.

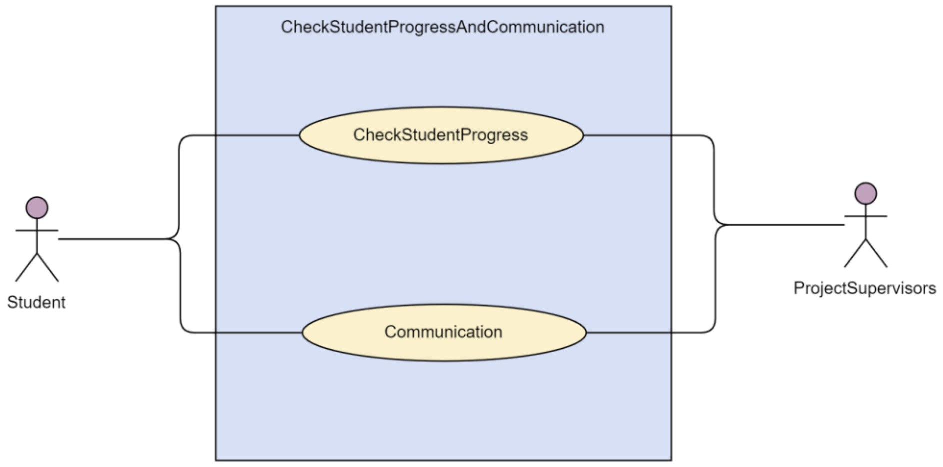
2. CreateProfileAndProjectMapping



Textual description:

Use case name	***CreateProfileAndProjectMapping***
Participating actors	Initiated by ***Student*** Participants: ***Student, ProjectSupervisors***
Flow of events	1. *GradRec* system responds with new empty profile registration form. 2. The ***Student*** fills out the profile form and he/she can select a program based on their research interests from an available list and then submit profile as draft or final submission. 3. The ***Student*** fills out the form • Save the form as a draft **OR** • Make their Final submission to ***GradRec*** system which uses *StudentProjectMapping* algorithm to map student with appropriate project based on analysis of student profile and project criteria. 4. ***GradRec*** system send notification to the *ProjectSupervisors* with profile mapping details.
Entry condition	• The *Students* and *ProjectSupervisors* identified by ***GradRec***
Exit condition	• The ***Student*** receive confirmation message in case of draft or final submission of his/her profile.
Quality requirements	• The ***Student*** must receive an acknowledgment either in case of failed or successful submission of profile forms.

3. CheckStudentProgressAndCommunication



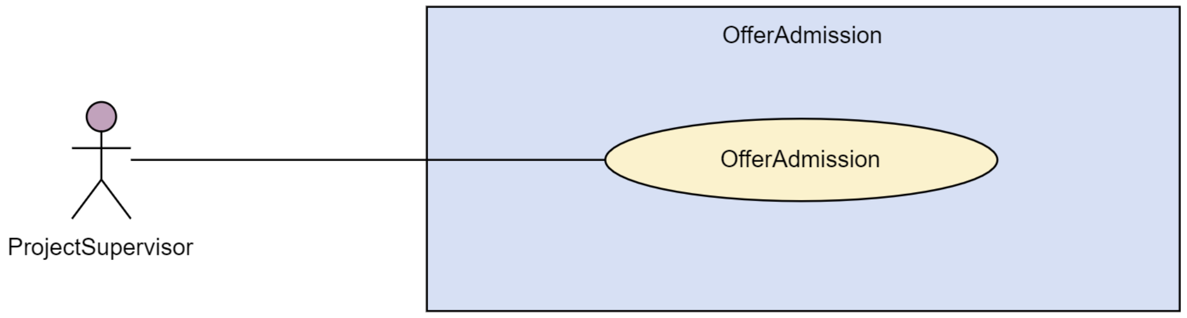
Textual description:

Use case name	***CheckStudentProgressAndCommunication***
Participating actors	Initiated by ***Student or ProjectSupervisors*** Participants: ***Student, ProjectSupervisors***
Flow of events	1. The ***Student*** and ***ProjectSupervisor*** uses *CheckStudentProgress* use case to monitor his/her progress. 2. ***ProjectSupervisor*** update new details regarding programs or student grades 3. The ***Student and ProjectSupervisor*** uses *Communication* use case to communicate with each other.
Entry condition	• The ***Students*** and ***ProjectSupervisors*** identified by ***GradRec***
Exit condition	• The ***Student*** and ***ProjectSupervisor*** receive notification in case of new changes in the profile or on confirmation of the program.
Quality requirements	• The ***Student*** have access to their profile and his/her progress. The *ProjectSupervisor* and *Student* can access communication channel to communicate with each other.

Textual Description:

Use case name	***RegistrationAndLogin***
Participating actors	User
Flow of events	1. Register new user a) Fill register form with details like firstName, Email, Password b) Successful registration allow user to access internal part of website. 2. Existing user a) Login with user id and password b) Successful login allow user to access internal part of website.
Entry condition	None
Exit condition	None
Quality requirements	None

4. OfferAdmission

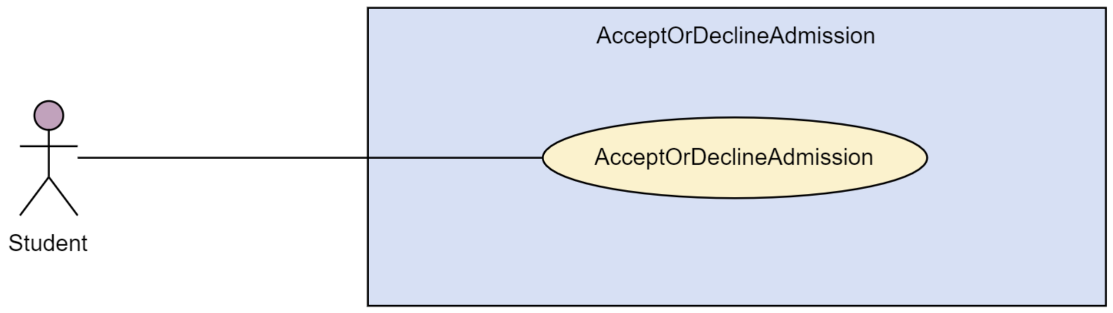


Textual description:

Use case name	***OfferAdmission***
Participating actors	Participants: ***ProjectSupervisor***
Flow of events	1) The ***ProjectSupervisor*** evaluate student profile. 2) Filter the list of selected students. 3) The ***ProjectSupervisor*** offer admission letter
Entry condition	• The ***User*** identified by ***GradRec***

Exit condition	-
Quality requirements	• The ***Users*** must receive an acknowledgment either in case of failed or successful action.

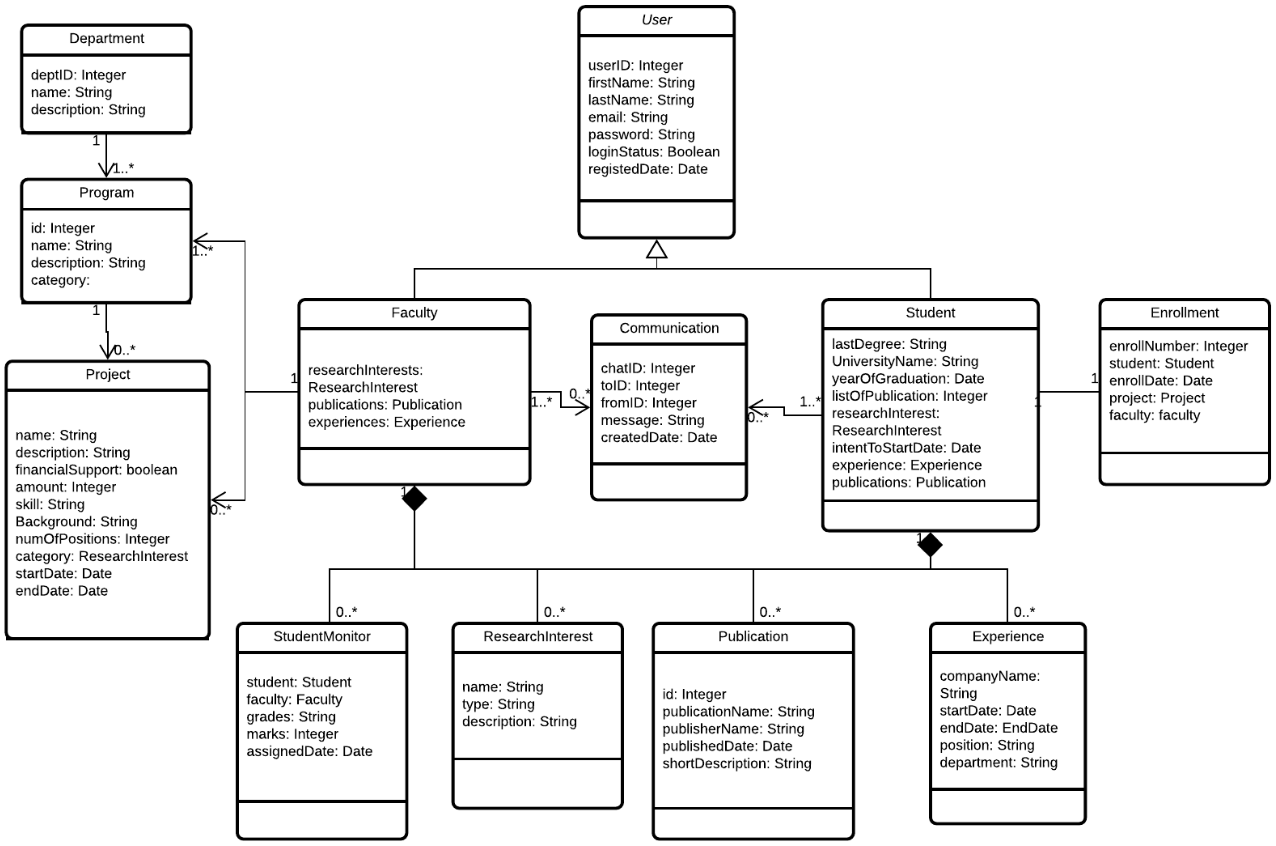
5. AcceptOrDeclineAdmission



Textual description:

Use case name	***AcceptOrDeclineAdmission***
Participating actors	Participants: ***Student***
Flow of events	1. The ***Student*** evaluate admission letter. 2. Accept admission a. In case of multiple offers ***Student*** accept one and decline others. b. Student enroll course and pay fees. 3. Decline admission c. Decline all admission letters. 4. In both case confirmation sends to ***ProjectSupervisor***
Entry condition	• The ***User*** identified by ***GradRec***
Exit condition	-
Quality requirements	• The ***Users*** must receive an acknowledgment either in case of failed or successful action.

2.3 Conceptual classes, attributes and associations



2.4 Entity, boundary and control objects

Entity Object:

- Student
- Faculty/ProjectSupervisor
- Program

Boundary Object:

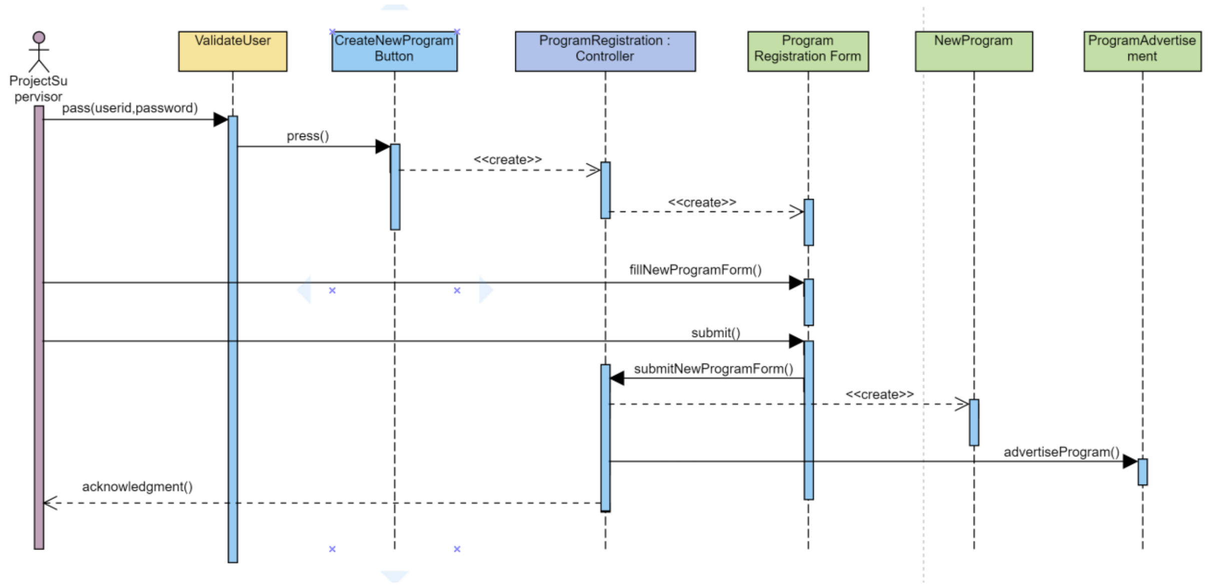
- HTMLForm

Control Object:

- RegistrationAndLogin
- CreateProfile
- Admission

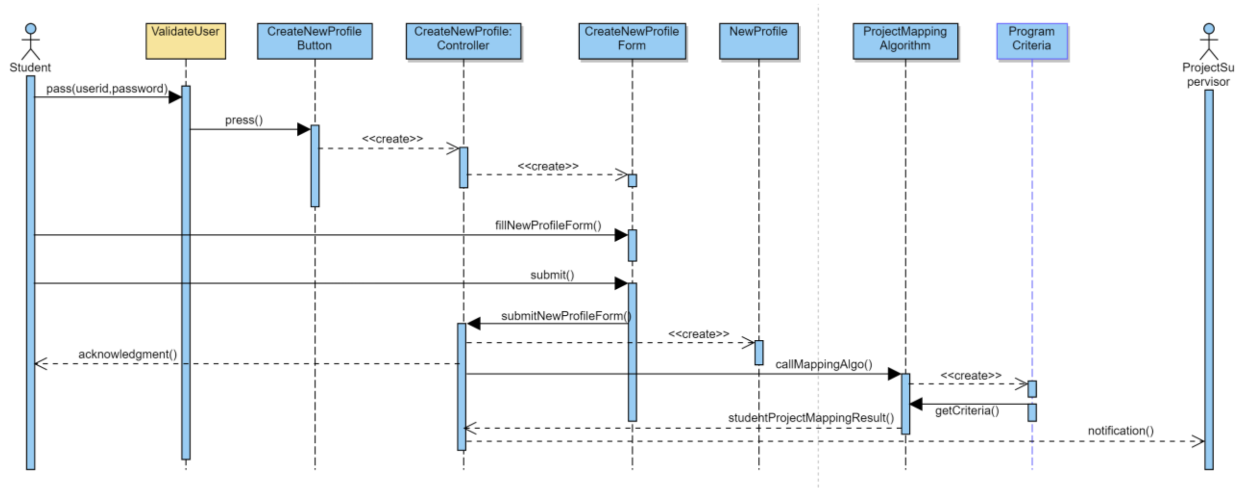
2.5 Sequences diagrams

1. ProgramRegistrationAndAdvertisement



Sequence diagram for the ***ProgramRegistrationAndAdvertisement*** use case

2. CreateProfileAndProjectMappingNotification



Sequence diagram for the ***CreateProfileAndProjectMappingNotification*** use case

3. Design Goals:

Security: Users of GredRec must be authenticated by the system before accessing functionality such as creating research project or creating student profile. Information like passwords will be stored in encrypted format. The system will provide a secure channel for the communication and ensure the privacy of the user's information.

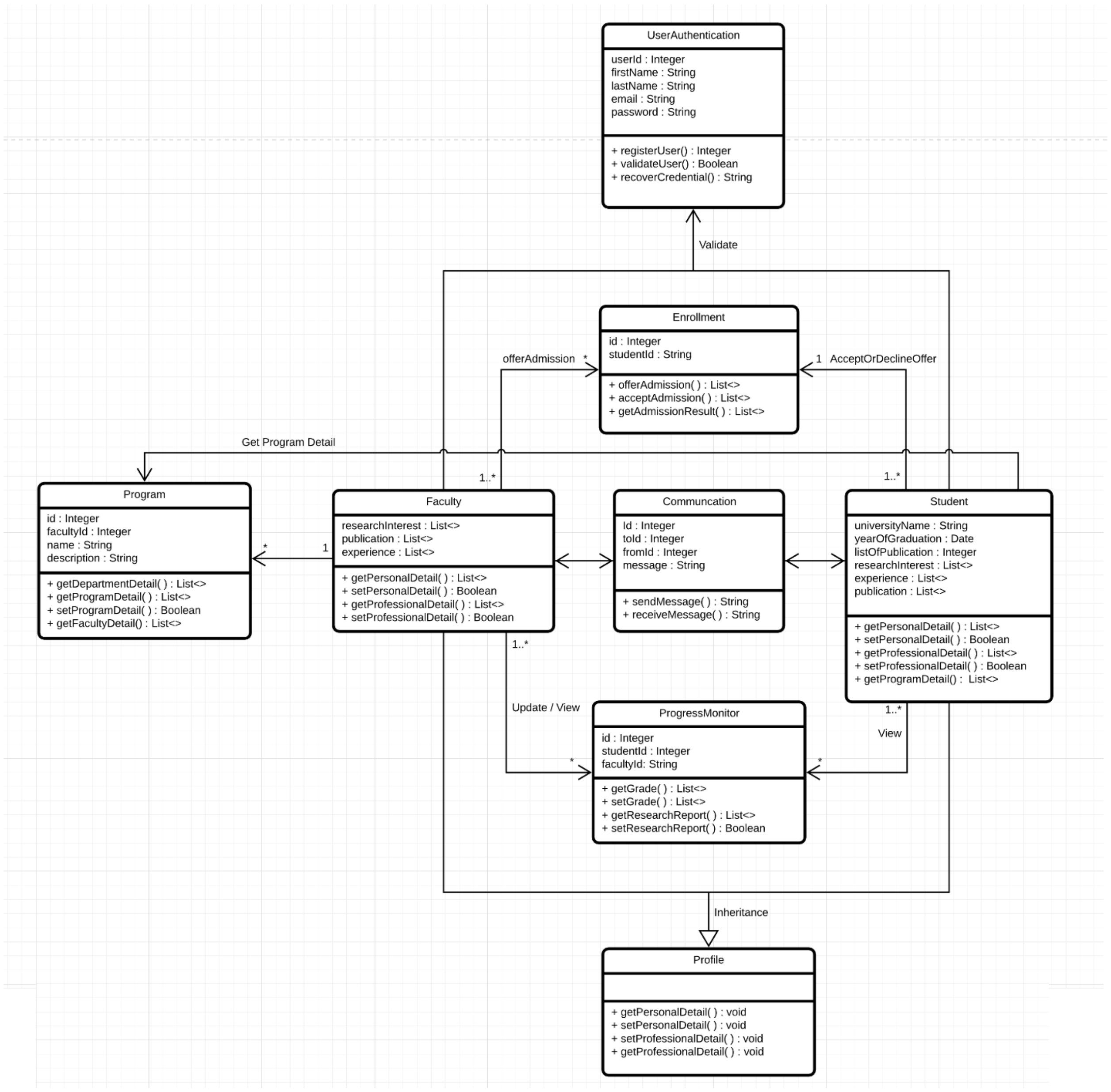
Reliability: GredRec system must be reliable in terms of persisting data on the permanent storage system (like Flat-File, MYSQL or MongoDB etc.) and be able to distinct different kinds of data. The student profile mapping algorithm should be consistent with the logic of creating the mapping between program and student based on the available information.

Usability: GredRec system should have a consistent look and feel across all user's interface. It should also be consistent with user privilege level, for example only the project supervisor can access the create program functionality, and all registered students should only have access to published programs. The system should allow easy start of multiple communication channels either with project supervisors or other students.

Modifiability: GredRec system will allow departments to edit their draft programs until they are published and/or before the last date of submission. Similarly, students will be allowed to edit their draft profile before the final submission and before the program deadline.

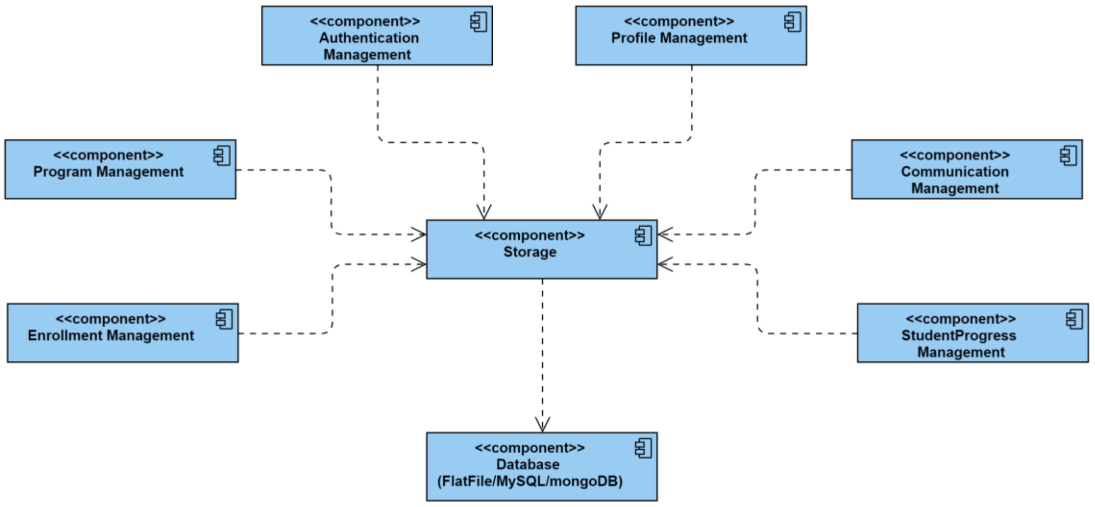
Fault tolerance: In case of error, the system should inform the user about the failure with a proper message without affecting other parts of the working system.

4. Object modelling:

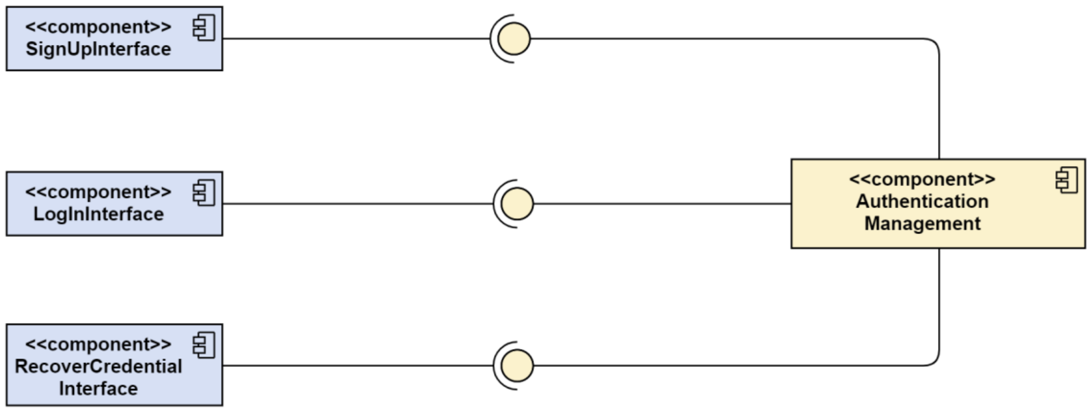

UserAuthentication
userId : Integer
firstName : String
lastName : String
email : String
password : String
+ registerUser() : Integer
+ validateUser() : Boolean
+ recoverCredential() : String
Validate
Enrollment
id : Integer
studentId : String
+ offerAdmission( ) : List<>
+ acceptAdmission( ) : List<>
+ getAdmissionResult( ) : List<>
offerAdmission
*
1
AcceptOrDeclineOffer
Get Program Detail
1..*
1..*
Program
id : Integer
facultyId : Integer
name : String
description : String
+ getDepartmentDetail( ) : List<>
+ getProgramDetail( ) : List<>
+ setProgramDetail( ) : Boolean
+ getFacultyDetail() : List<>
*
1
Faculty
researchInterest : List<>
publication : List<>
experience : List<>
+ getPersonalDetail( ) : List<>
+ setPersonalDetail( ) : Boolean
+ getProfessionalDetail( ) : List<>
+ setProfessionalDetail( ) : Boolean
Communcation
Id : Integer
toId : Integer
fromId : Integer
message : String
+ sendMessage( ) : String
+ receiveMessage( ) : String
Student
universityName : String
yearOfGraduation : Date
listOfPublication : Integer
researchInterest : List<>
experience : List<>
publication : List<>
+ getPersonalDetail( ) : List<>
+ setPersonalDetail( ) : Boolean
+ getProfessionalDetail( ) : List<>
+ setProfessionalDetail( ) : Boolean
+ getProgramDetail() : List<>
1..*
Update / View
1..*
View
ProgressMonitor
id : Integer
studentId : Integer
facultyId: String
+ getGrade( ) : List<>
+ setGrade( ) : List<>
+ getResearchReport( ) : List<>
+ setResearchReport( ) : Boolean
*
*
Inheritance
Profile
+ getPersonalDetail( ) : void
+ setPersonalDetail( ) : void
+ setProfessionalDetail( ) : void
+ getProfessionalDetail( ) : void

5. Decomposing of the system:

System Decomposition:

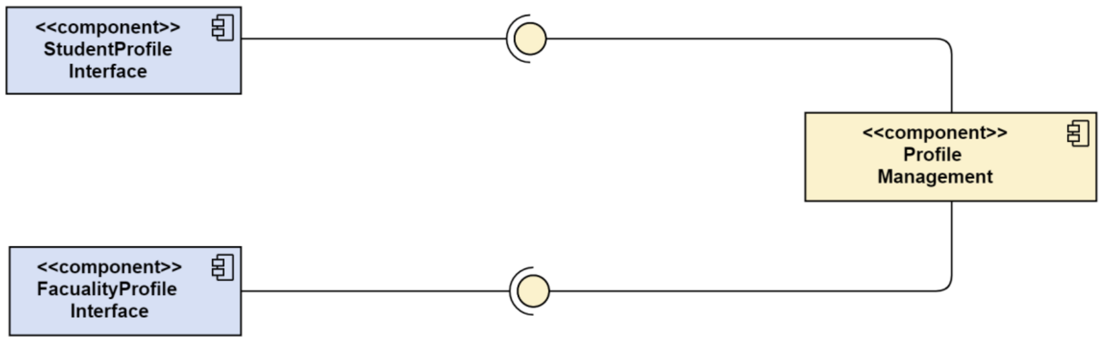


<<component>> Authentication Management
<<component>> Profile Management
<<component>> Program Management
<<component>> Communication Management
<<component>> Storage
<<component>> Enrollment Management
<<component>> StudentProgress Management
<<component>> Database (FlatFile/MySQL/mongoDB)

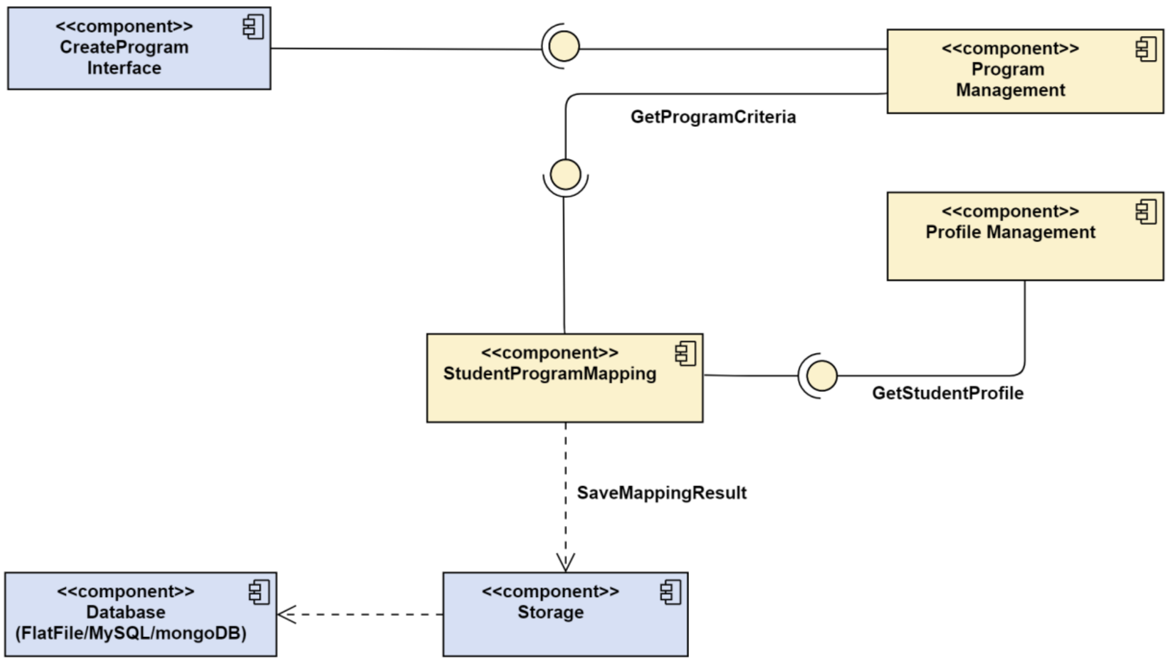


<<component>> SignUpInterface
<<component>> LogInInterface
<<component>> RecoverCredential Interface
<<component>> Authentication Management

SignUpInterface, LoginInInterface and RecoverCredentialInterface require services from Authentication Management to manager user authentication and recover credentials.


<<component>> StudentProfile Interface
<<component>> Profile Management
<<component>> FacualityProfile Interface

StudentProfileInterface and FacualityProfileInterface require services from Profile Management component create and manage student and factuality details.


<<component>> CreateProgram Interface
<<component>> Program Management
GetProgramCriteria
<<component>> Profile Management
<<component>> StudentProgramMapping
GetStudentProfile
SaveMappingResult
<<component>> Database (FlatFile/MySQL/mongoDB)
<<component>> Storage

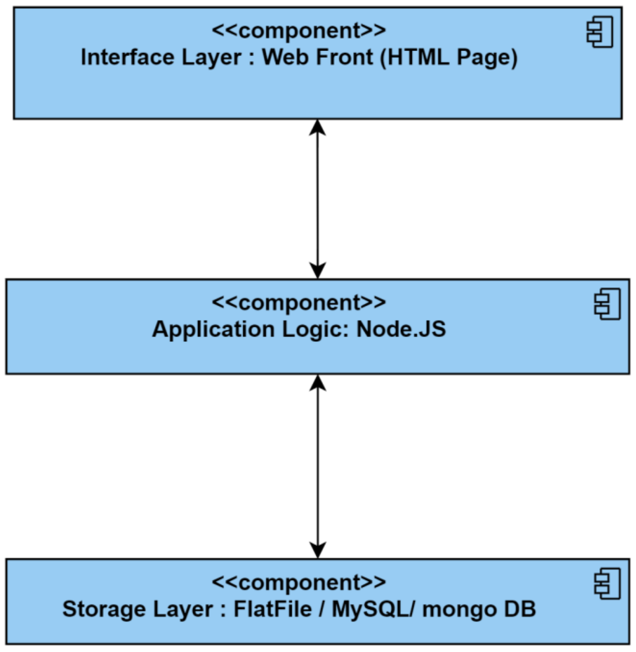
CreateProgramInterface require services from Program Management component to update or create new programs.

StudentProgramMapping component uses Program and Profile Management component to get program criteria and new student profile details to create mapping between student and program then save result in database.

6. Logical architecture:

Gradrec system implemented with three tire architecture.

- The **interface layer** includes all boundary objects that deal with the web pages (HTML page).
- The **application logic** layer includes all control and entity objects, realizing the processing, rule checking, and notification required by the application with the help Node.JS.
- The **storage layer** realizes the storage, retrieval, and query of persistent objects. In of these storage Flat File / MySQL/ mongo DB.


<<component>>
Interface Layer : Web Front (HTML Page)
<<component>>
Application Logic: Node.JS
<<component>>
Storage Layer : FlatFile / MySQL/ mongo DB

7. Git Hub And Website Link:

Web Site Link:

http://sc-5.cs.mun.ca/

Git Hub Link:

https://github.com/kaushlenderk/GraduateRecruitment