

**University of The Punjab FCIT**

**Department of Software Engineering**

**Software Design and Architecture**

# Group Assignment-01

11/23/2022

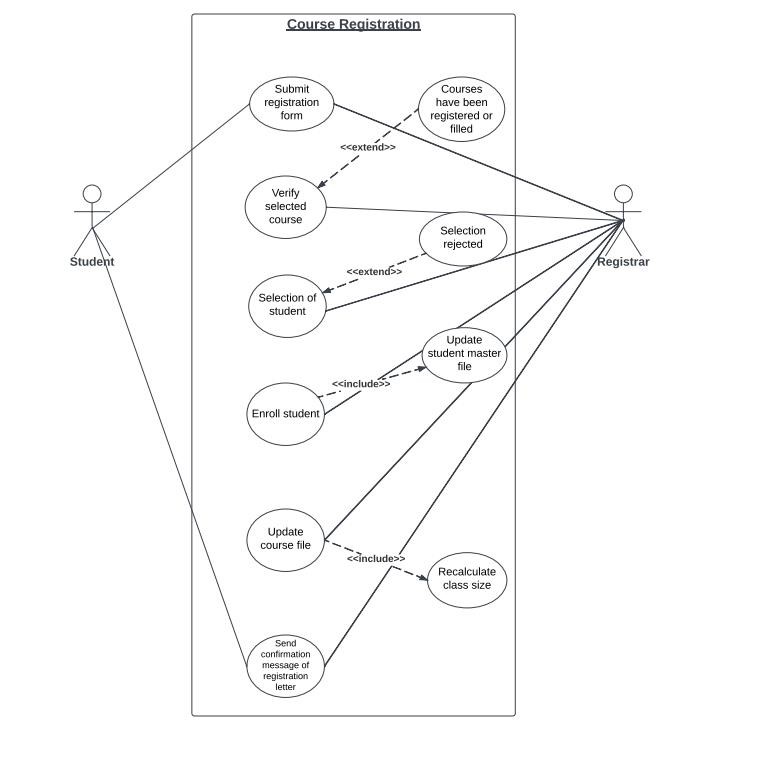
**Task # 01**

**Actors and goal list**

|  |  |
| --- | --- |
| **Actors** | **Goals** |
| **Student** | 1. Submit registration form(with his name, ID number and number of courses he/she wish to take |
| **Registrar** | 1. Verify registration(if courses are open or not) 2. Determines acceptance or rejection of students selections 3. Enroll student(if their selection is accepted) 4. Update university’s course file 5. Recalculate class size in case of enrollments 6. Update student master file 7. Send a confirmation-ofregistration letter to each student applicant. |

## Task # 02

**Use case diagram.**



## Task # 03

### Brief Format of Use Case

**UC-4: Enroll Student**

The registrar enrolls the student in the courses in which he/she wishes to take admission. The registrar also updates the student’s master file with information about the new student, and then the registrar updates the course file with re-calculation of class size.

## Task # 04

**Fully Dressed Format of UC-4:**

**Description:**

The registrar enrolls the students in a specific course and updates the information of the master file.

**Pre-conditions:**

Before enrolling the students, the registrar accepts or rejects the student’s selection. On the basis of this selection the student is enrolled in the courses he/she wishes to take.

**Post-conditions:**

After enrollment of students in selected courses, the registrar updates the university course file with the student name and identification number. Then the registrar will re-calculate the size of class.

**Success Scenario:**

The success scenarios of use case are given below:

**4a**. On successful enrollment of student

**4a.1** The registrar updates the student’s master file with the information about new student.

**4b.** After successful enrollment and updating a student's master file, the registrar will update the university course file.

**4b-1** The registrar re-calculates the class size after a new student is enrolled. **4b-1.1** If maximum enrollment is reached, the course will be closed.

**Failure Scenario:**

**4a.** If student is not successfully enrolled

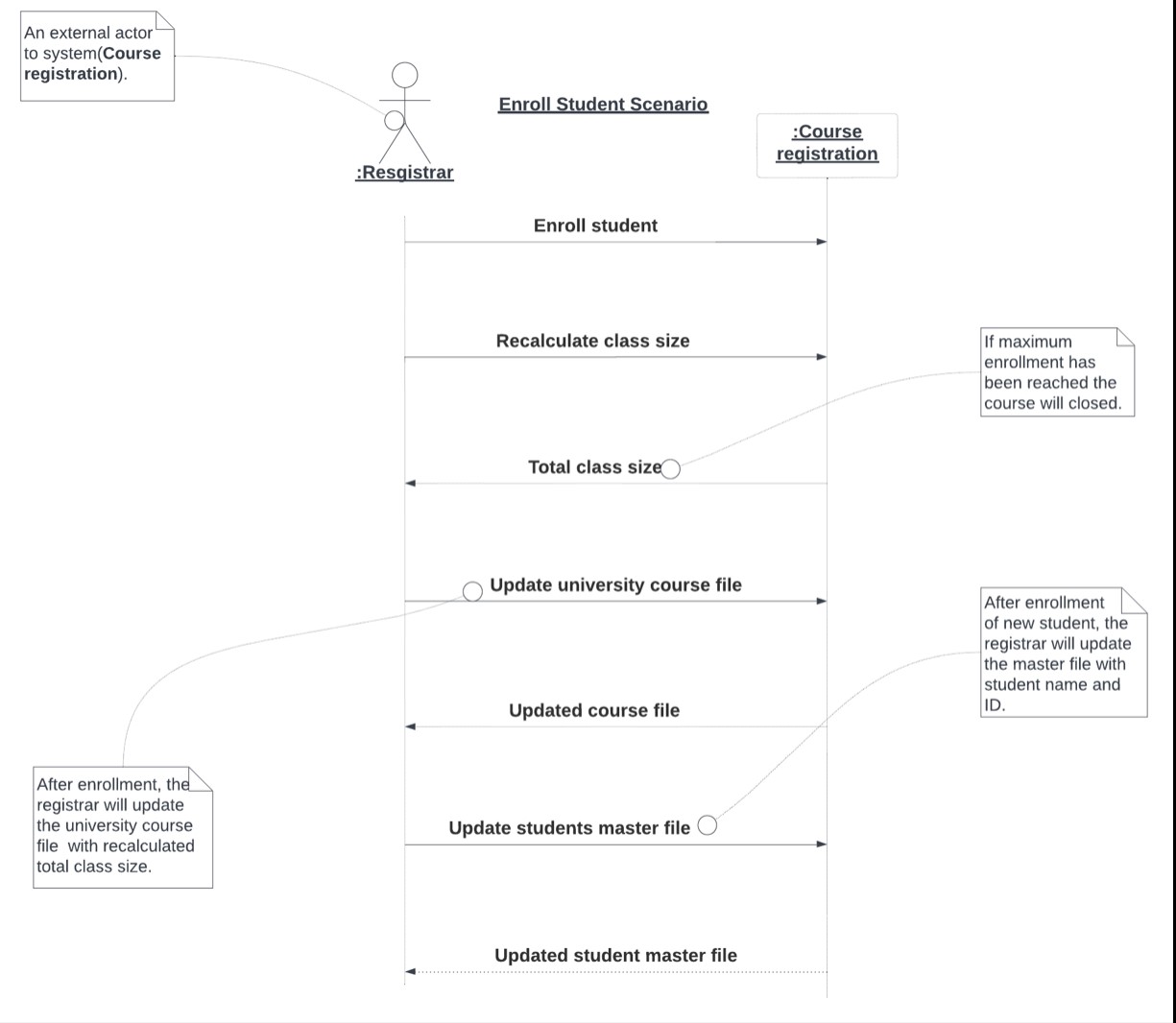
**4a.1** The registrar will not update student’s master file.

**4a.2** The registrar will not update university course file.

**4a.3** The registrar will send letter to the applicant, that he/she could not enroll in the courses at which he/she want to take admission.

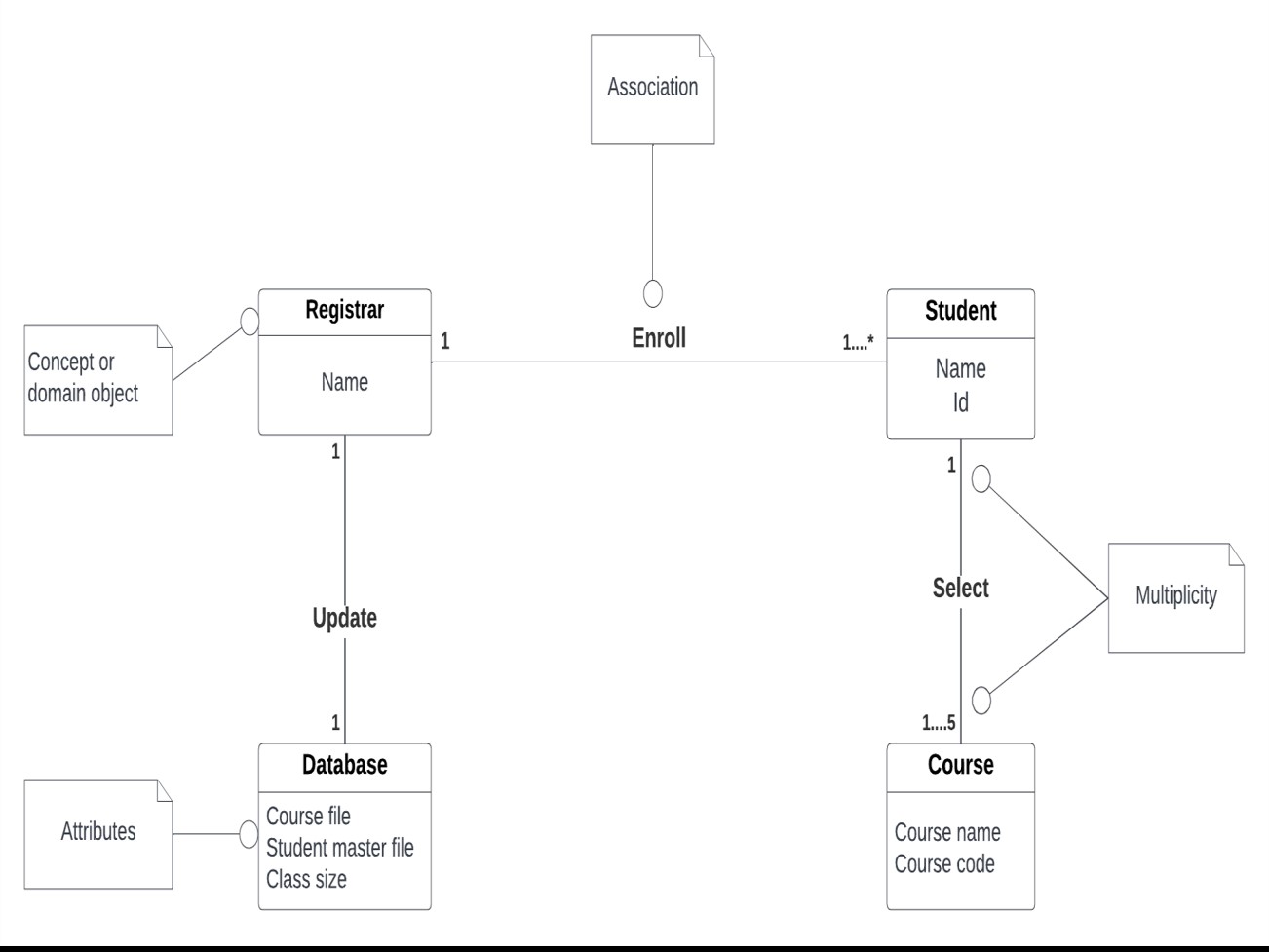
## Task # 05

### SSD of fully dressed Use case



## Task # 06

### Domain Model



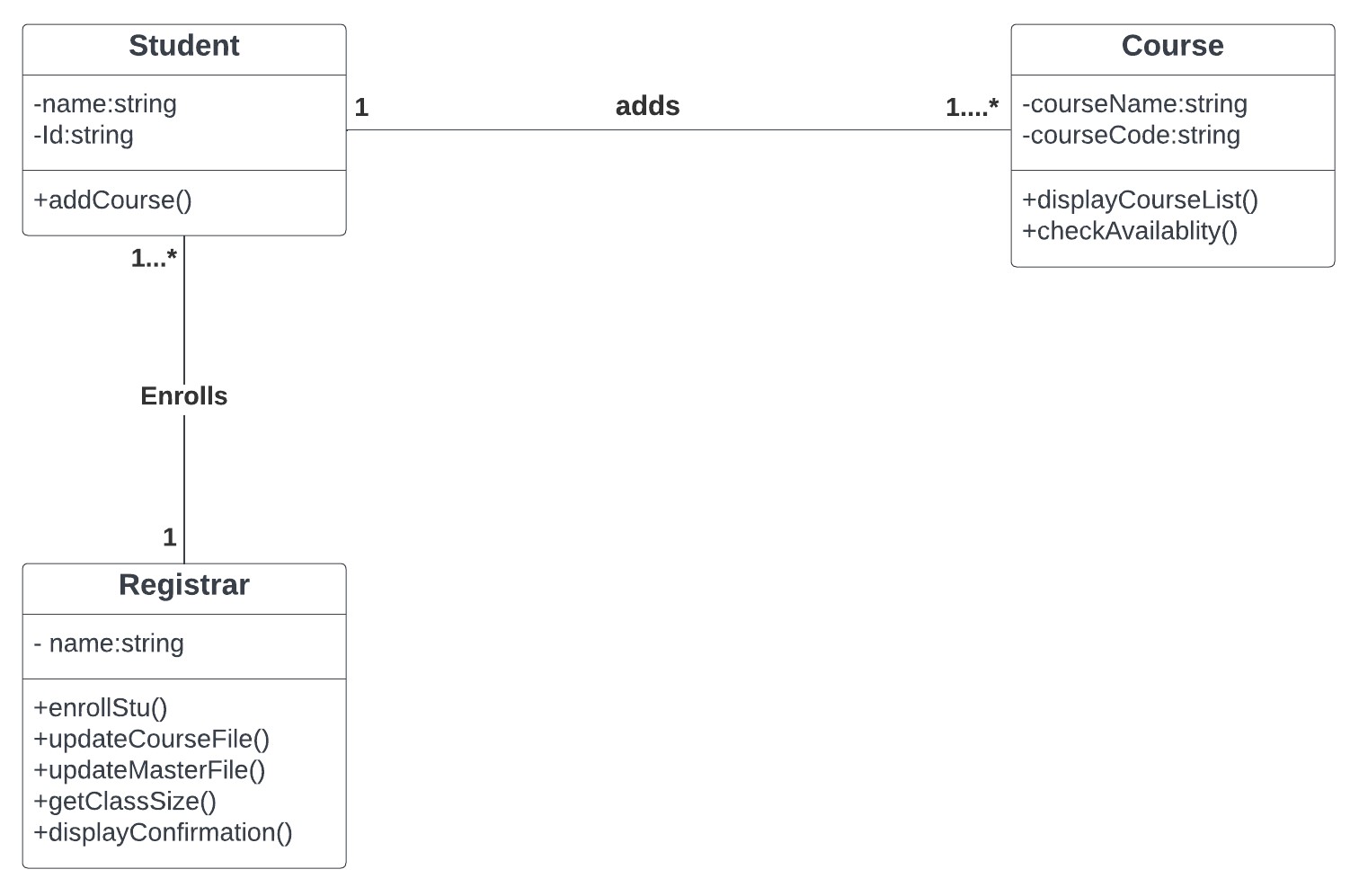
**Association List of Domain model:**

**Category**

* Student selects courses
* Registrar enroll students
* Registrar update university database (course file , student master file)

## Task # 07

### Class Diagram



**Task 8 & 9:**

### Working solution (“Main.cpp”)

#include<iostream>

#include<fstream> #include "string" using namespace std; const string CN\_CODE = "CN10"; const string SDA\_CODE = "SDA11"; const string DSA\_CODE = "DSA12"; const string IS\_CODE = "IS13"; const string WE\_CODE = "WE14"; const int MAX\_LIMIT = 50;

class Registrar

{ private:

string name; public:

void enrollStudent(string c, string name)

{

cout << "The registrar has enrolled the student named " << name << " to the course of " << c << endl;

}

void updateCourseFile(string course)

{

string c; int s;

fstream f;

f.open("Coursefile.txt"); f >> c; f >> s;

while (f)

{

if (c == course)

{

f << s + 1;

}

f >> c; f >> s;

}

f.close();

}

void updateMasterFile(string name, string id, string c)

{

fstream fout;

fout.open("masterFile.txt", ios::app);

fout << id << " " << name << " " << c << endl;

fout.close();

}

void DisplayConfirmation()

{

cout << "Your course has been added successsfully!" << endl;

}

friend class Course; friend class Student;

}; class Course

{ private:

string courseName; string courseCode; public:

void displayCourseList()

{

fstream f;

f.open("Coursefile.txt");

string c; int s; f >> c; f >> s; while (f)

{

cout << "Cousre name: " << c << endl; cout << "Class Size: " << s << endl << endl;

f >> c; f >> s;

}

f.close();

}

void checkAvailability(string course, string name, string id)

{

courseName = course;

if (courseName == "CN")

{

courseCode = CN\_CODE;

}

else if (courseName == "SDA")

{

courseCode = SDA\_CODE;

}

else if (courseName == "DSA")

{

courseCode = DSA\_CODE;

}

else if (courseName == "IS")

{

courseCode = IS\_CODE;

}

else if (courseName == "WE")

{

courseCode = WE\_CODE;

}

string c; int s;

fstream f;

f.open("Coursefile.txt");

f >> c; f >> s; Registrar r; while (f)

{

if (c == courseName)

{

if (s < MAX\_LIMIT)

{

r.DisplayConfirmation();

r.enrollStudent(c, name);

r.updateMasterFile(name, id, c);

r.updateCourseFile(c);

}

}

f >> c; f >> s;

}

f.close();

}

friend class Student;

friend class Registrar;

}; class Student

{ private:

string name; string Id; public:

Student(string n,string id)

{

name = n;

Id = id;

}

void addCourse(string course)

{

Course c;

c.checkAvailability(course,name,Id);

}

friend class Registrar;

friend class Course;

};

int main()

{

string name, Id; cout << "Enter your name: "; cin >> name;

cout << "Enter your ID: "; cin >> Id; Student s1(name,Id); fstream f;

f.open("Coursefile.txt");

string c; int s; f >> c;

f >> s;

cout << "We are offering the folowing courses in this semester:"<<endl; while (f)

{

cout << "Cousre name: "<<c<<endl;

cout << "Class Size: "<<s<<endl<<endl;

f >> c; f >> s;

}

f.close(); string course;

char choice='y';

do

{

cout << "Enter the course u wana to select: ";

cin >> course;

s1.addCourse(course);

cout << "Do u wana to select another course?(y/n)"; cin >> choice;

} while (choice=='y');

return 0;

}

#### Course file .txt

CN

20

SDA

30

DSA

25

IS

35

WE

40

#### Master file.txt

// Esha bsef20a022 CN