WUHAN UNIVERSITY
International School of Software

PROJECT REPORT - OBJECT ORIENTED ANALYSIS AND DESIGN

Analysis and Design Document (First Iteration) : University Registration System

Author:
MD ALAMGIR KABIR[a]
MOCHALOV ILIA[b]

[a]ID: 2015272160014- sagar.whu@outlook.com.
[b]ID: 2015272160016- 2111311792@qq.com.

Supervisor:
Professor SIMON XU

November 8, 2015

WUHAN UNIVERSITY
International School Of Software

Contents

List of Figures

List of Tables

For/Dedicated to/To my…

Chapter 1

Vision

1.1 Introduction

Our company deliver a next generation of University Registration Software tools allow you to easily create, modify, and deploy forms including schedule, course description, students and professors forms.
This allows you to reap the benefit of a massive cost reduction in your annual enrollment and registration process. No more paper to lose, no more trying to figure out what is the name written in that document. Instead you end up with clear validated data that is ready for your University Registration System.
In short, You are in control. Your forms, Your process, Your way.

1.2 Positioning

University Registration System is a System which goes above a beyond other similar Software. Over the years, these modules have been developed to ensure that you get the best package for your district's needs.

- Student's registration – straight forward, intuitively easy.
- Teacher's registration – saves precious time and efforts.
- Classes and schedule systematization.
- Implement changes easy as one, two, three.
- 100 percent work-ability and safety.

Chapter 2

Use case Modeling

2.1 Use case diagram

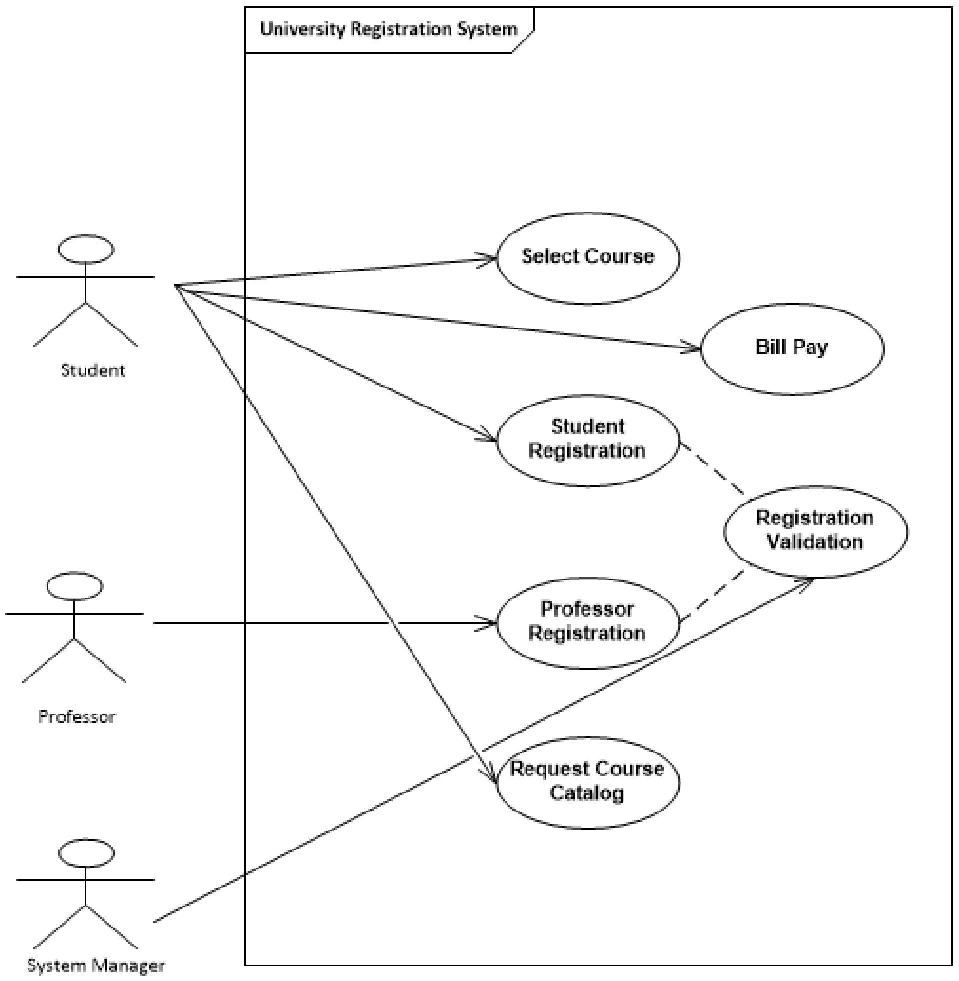

FIGURE 2.1: Use case diagram for first iteration

2.2 Use case description

For use case 1 , check table then follow Main success scenario then follow alternative flow-

TABLE 2.1: Use Case 1: Process selecting courses

Use Case 1: Process selecting courses
Scope: University Registration system **Level:** user goal **Primary Actor:** Student **Stakeholders and Interests:** - Student: Wants accurate, fast system that provides with all necessary information about available courses (including detailed description). - Professor: Wants students to be updated about all the available courses. - System manager: Wants to be able to quickly update all the necessary information in system. **Preconditions:** Student has already registered in the system and System manager verified student's account. Student is identified and authenticated. System is updated and shows all the courses offerings for the coming semester. **Success Guarantee (or Postconditions):** System successfully received 4 courses of that student. In addition, each student will indicate two alternative choices in case a course offering becomes filled or cancelled.

Main Success Scenario (or Basic Flow) for use case 1:
1. Students successfully log in to the system using his authentication information.
2. Students request courses offerings for the coming semester
3. Students selects 4 courses and 2 alternatives in case a course offering becomes filled or cancelled
4. System records student's choice and completes process.
5. System sends information to the billing system so the student can be billed for the semester.
6. Students pays the bill and receives electronic receipt.
7. System sends verification e-mail to the student (e-mail consists personal schedule and all necessary information for upcoming semester)
8. Student log out from the system.

Extensions (or Alternative Flows) for use case 1:

(a.) After students choses 4 courses and 2 alternatives, System informs the student that courses offering becomes filled or cancelled:

1. System notify student that one or more chosen courses are already filled or cancelled.
2. Students approves his alternative course to replace filled/cancelled courses.
3. System goes to the billing stage part.

b. At any time, System fails:

1. Student restarts System, logs in, and requests recovery of prior choice.
2. System reconstructs prior state.
3. Students continues working with the system starting at the point where system failed previously.

Special Requirements for use case 1:
- System availability 24/7.
- Networking system capable to operate such system.
- Language internationalization on the text displayed.

TABLE 2.2: Use Case 2: Process Student Registration in the system

Use Case 2: Process Student Registration in the system
Scope: University Registration system **Level:** user goal **Primary Actor:** Student, System manager **Stakeholders and Interests:** - Student: Wants user friendly, fast interface that allows to register in system. - System manager: Wants to be able to approve registration accounts of students. **Preconditions:** Student will register in the system by entering their department and information. **Success Guarantee (or Postconditions):** After registration, system manager will approve their account.After approval their account, students will get their login information and also can login their account. After login, students can view their course catalogs.

Main Success Scenario (or Basic Flow) for use case 2:
1. Student requests system for online registration form;
2. System offers template for registration;
3. Student fill in the form with some information such as Student ID,
4. Student Name, Email Address Semester, Department, Pasword and submit it;
5. System sends request to system manager for authorization new account (student)
6. System manager approves request for new account;
7. System creates new account and send notification to the user (student)

Extensions (or Alternative Flows) for use case 2:
a. Login ID already exists in this System:
1. System notifies student that he Login ID already exists and suggest him to choose another one.
2. Student inputs changed user ID and try to submit his account again.
3. In case new Login ID also already exists, the system goes to step a.1. In case new Login ID is unique system sends request for authorization to System manager.

b. At any time, System fails for use case 2:
1. Student restarts System, logs in, and requests recovery of prior choice.
2. System reconstructs prior state.
3. Students continues working with the system starting at the point where system failed previously.

Special Requirements for use case 2:
- System availability 24/7.
- Networking system capable to operate such system.
- Language internationalization on the text displayed.

Main Success Scenario (or Basic Flow) for use case 3:
1. Professor requests system for online registration form;
2. System offers template for registration;
3. Professor fill in the form and form information are Professor Name, Email Address, Department, Teaching Subject, Password and after that professor will submit it;

TABLE 2.3: Use Case 3: Process Professor Registration in the system

Use Case 3: Process Professor Registration in the system
Scope: University Registration system **Level:** user goal **Primary Actor:** Professor, System manager **Stakeholders and Interests:** - Professor: Wants user friendly, fast interface that allows to register in system. - System manager:Wants to be able to approve registration accounts of professor. **Preconditions:** Professor will register in the system by entering their information. Such as courses which they will teach. **Success Guarantee (or Postconditions):** After registration, system manager will approve their account. After approval their account, professor will get their login information and also can login their account. After login, professor can view their students list of their courses that they will teach.

4. System sends request to system manager for authorization new account (professor)
5. System manager approves request for new account;
6. System creates new account and send notification to the user (professor)

Extensions (or Alternative Flows) for use case 3:
a. Login name already exists in this System:
1. System notifies professor that Login name already exists and suggest him to choose another one.
2. Professor inputs changed user name and try to submit his account again.
3. In case new Login name also already exists, the system goes to step a.1. In case new Login name is unique system sends request for authorization to System manager.

b. At any time, System fails for use case 3:
1. Professor restarts System, logs in, and requests recovery of prior choice.
2. System reconstructs prior state.
3. Professor continues working with the system starting at the point where system failed previously.

Special Requirements for use case 3:
- System availability 24/7.
- Networking system capable to operate such system.
- Language internationalization on the text displayed.

Main Success Scenario (or Basic Flow) for use case 4:
1. Student requests system for login;
2. After login, student request for course catalog;
3. Student press menu to see for all courses;
4. Student logout for the system

Extensions (or Alternative Flows) for use case 4:
a. Login name already exists in this System:
1. System notifies student that he Login ID already exists and suggest him to choose another one.
2. Student inputs changed user ID and try to submit his account again.
3. In case new Login name also already exists, the system goes to step a.1. In case new

TABLE 2.4: Use Case 4: Process Request the course catalog

Use Case 4: Process Request the course catalog
Scope: University Registration system **Level:** user goal **Primary Actor:** Student **Stakeholders and Interests:** - Student: Wants detailed and updated information about available courses. **Preconditions:** Student has already received students ID and personal number. **Success Guarantee (or Postconditions):** After loading course catalog, student will see the course list of their current semester. Student will also see the information of each course such as professor of the specific course and prerequisites of the course.

Login name is unique system sends request for authorization to System manager.
b. At any time, System fails for use case 4:
1. Student restarts System, logs in, and requests recovery of prior choice.
2. System reconstructs prior state.
3. Students continues working with the system starting at the point where system failed previously.

Special Requirements for use case 4:
- System availability 24/7.
- Networking system capable to operate such system.
- Language internationalization on the text displayed.

2.3 System Sequence Diagram

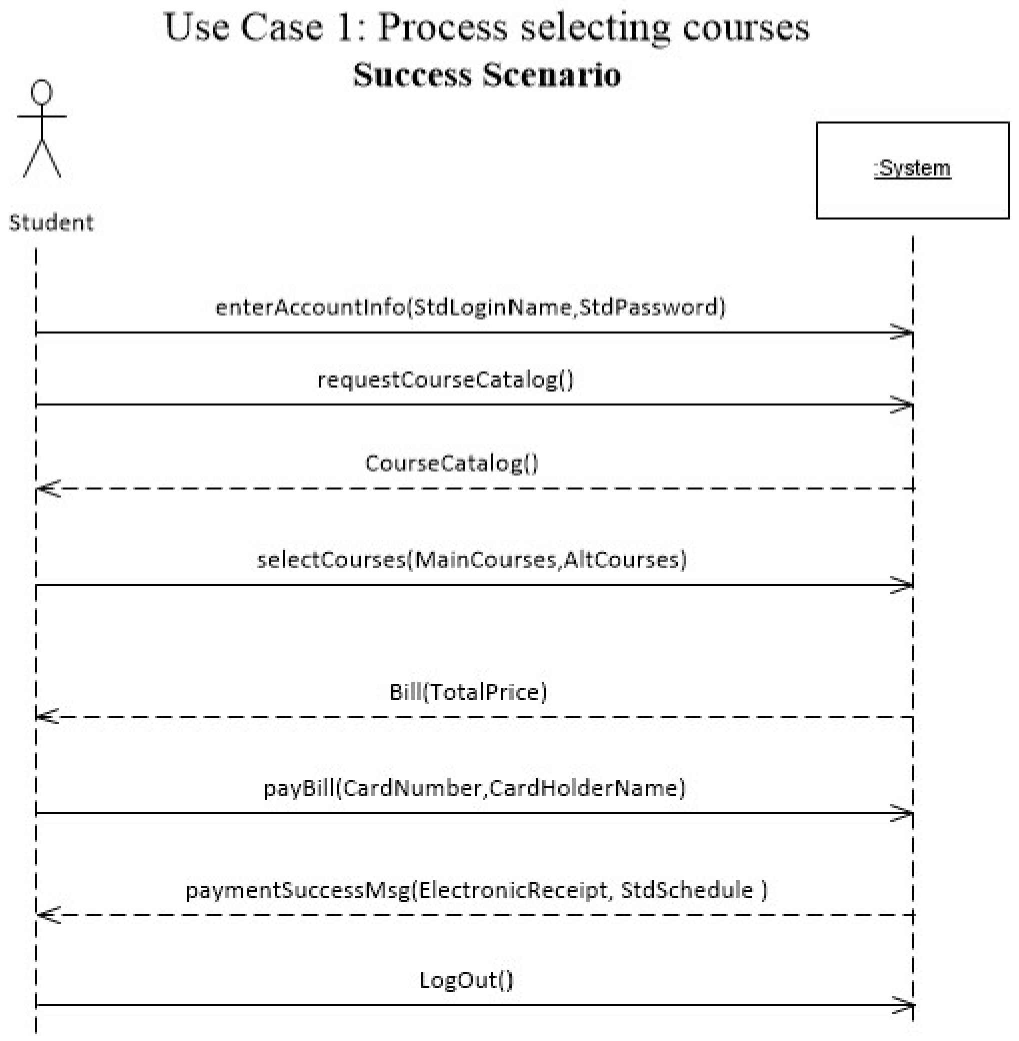
Use Case 1: Process selecting courses
Success Scenario
Student
:System
enterAccountInfo(StdLoginName,StdPassword)
requestCourseCatalog()
CourseCatalog()
selectCourses(MainCourses,AltCourses)
Bill(TotalPrice)
payBill(CardNumber,CardHolderName)
paymentSuccessMsg(ElectronicReceipt, StdSchedule)
LogOut()

FIGURE 2.2: System Sequence Diagram: Use case 1 (Success Scenario)

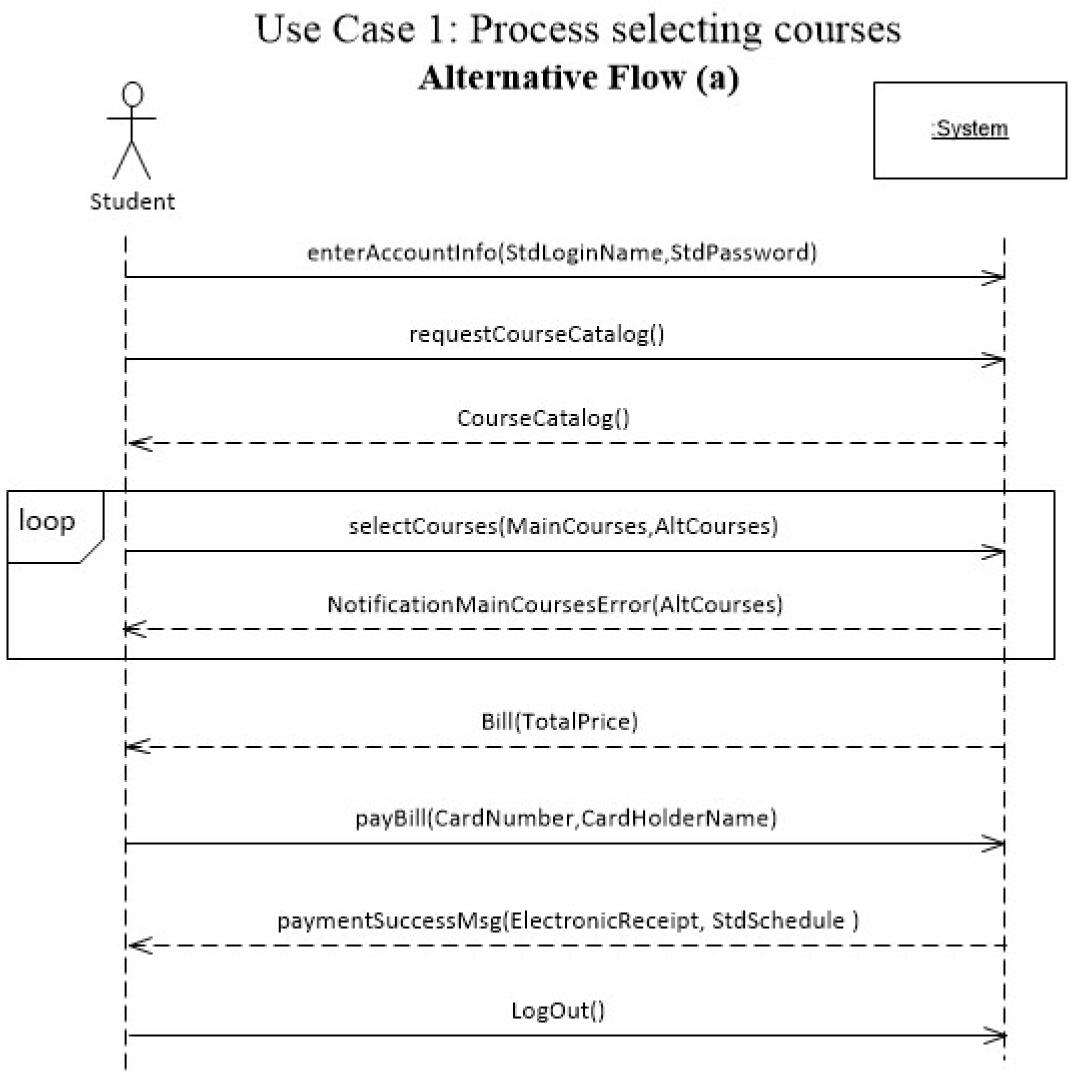
Use Case 1: Process selecting courses
Alternative Flow (a)
Student
:System
enterAccountInfo(StdLoginName,StdPassword)
requestCourseCatalog()
CourseCatalog()
loop
selectCourses(MainCourses,AltCourses)
NotificationMainCoursesError(AltCourses)
Bill(TotalPrice)
payBill(CardNumber,CardHolderName)
paymentSuccessMsg(ElectronicReceipt, StdSchedule)
LogOut()

FIGURE 2.3: System Sequence Diagram: Use case 1 (Alternative Flow(a))

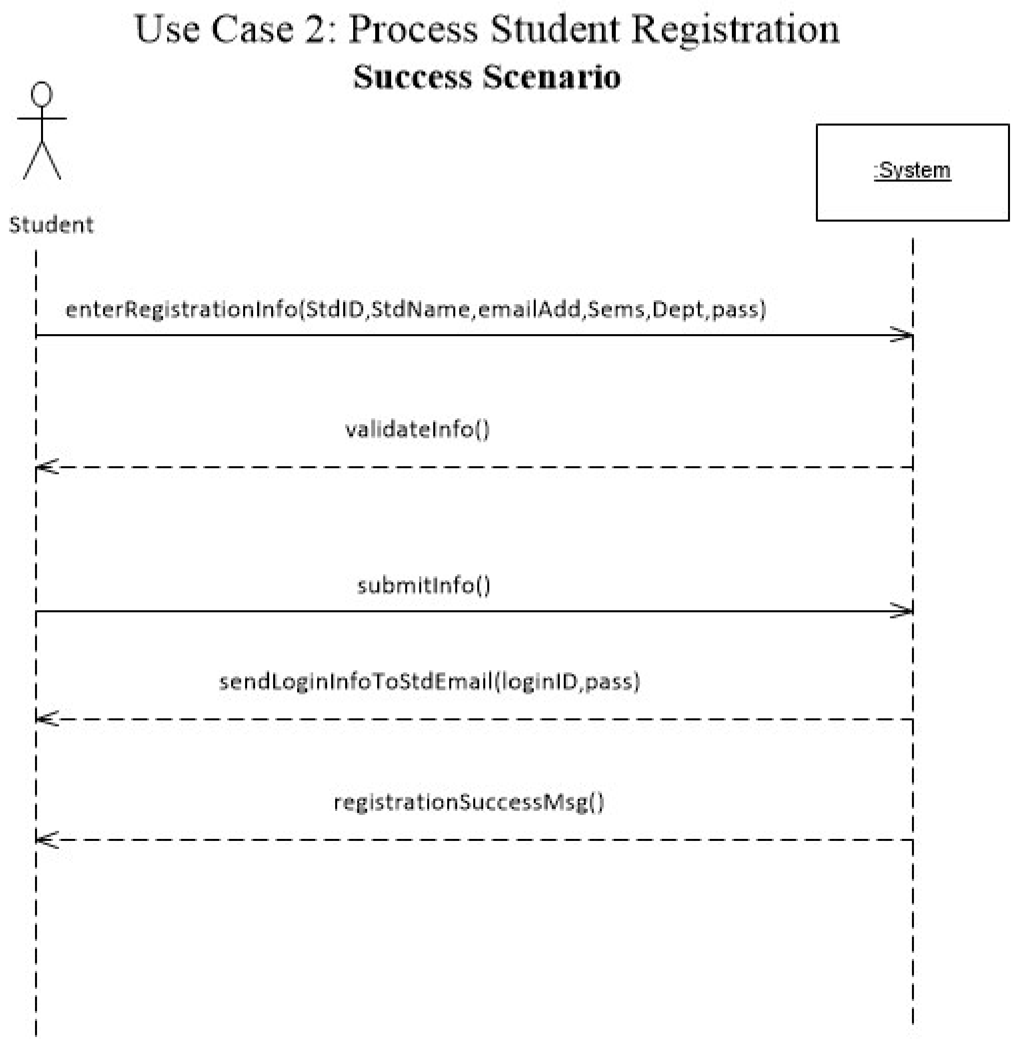
Use Case 2: Process Student Registration
Success Scenario
Student
:System
enterRegistrationInfo(StdID,StdName,emailAdd,Sems,Dept,pass)
validateInfo()
submitInfo()
sendLoginInfoToStdEmail(loginID,pass)
registrationSuccessMsg()

FIGURE 2.4: System Sequence Diagram: Use case 2 (Success Scenario)

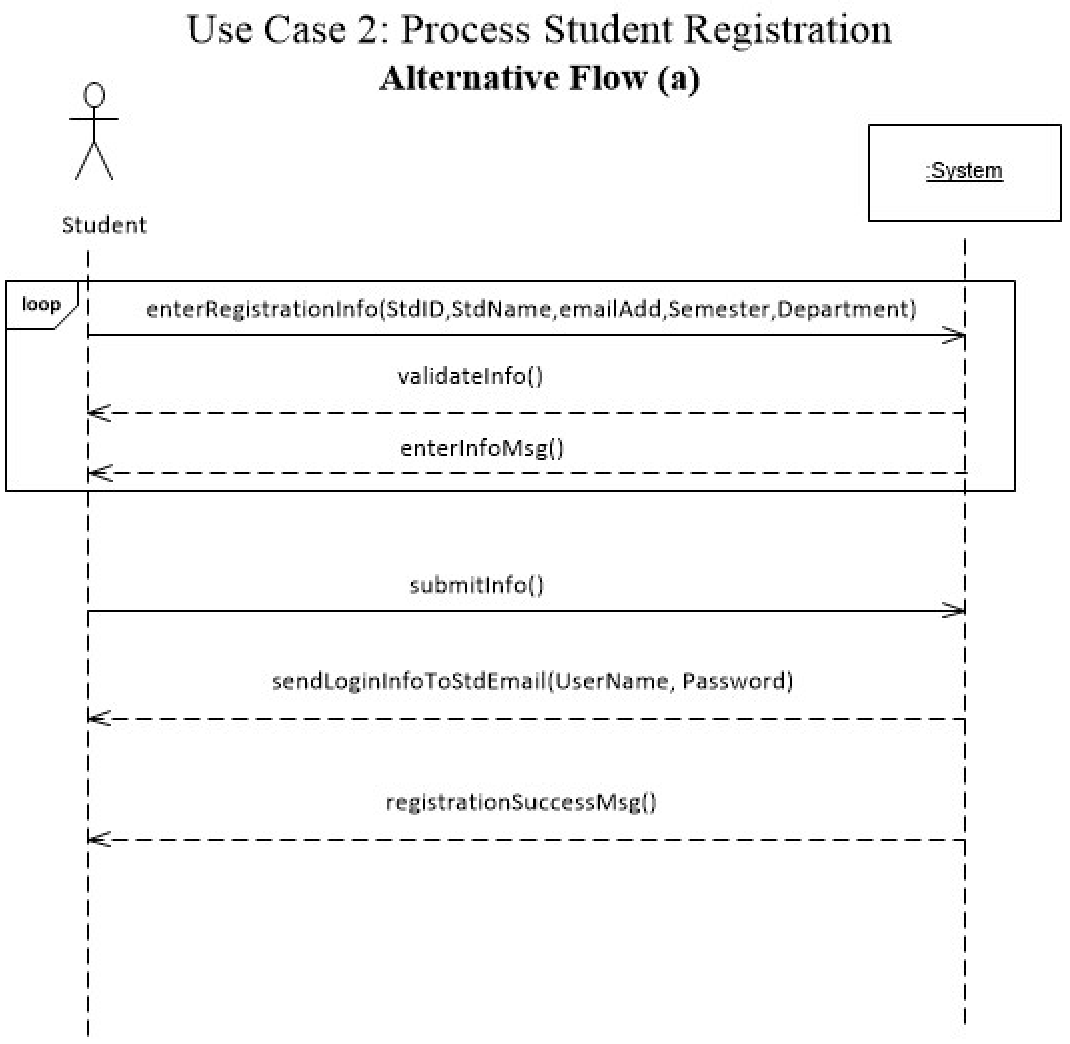
Use Case 2: Process Student Registration
Alternative Flow (a)
Student
:System
loop
enterRegistrationInfo(StdID,StdName,emailAdd,Semester,Department)
validateInfo()
enterInfoMsg()
submitInfo()
sendLoginInfoToStdEmail(UserName, Password)
registrationSuccessMsg()

FIGURE 2.5: System Sequence Diagram: Use case 2 (Alternative Flow(a))

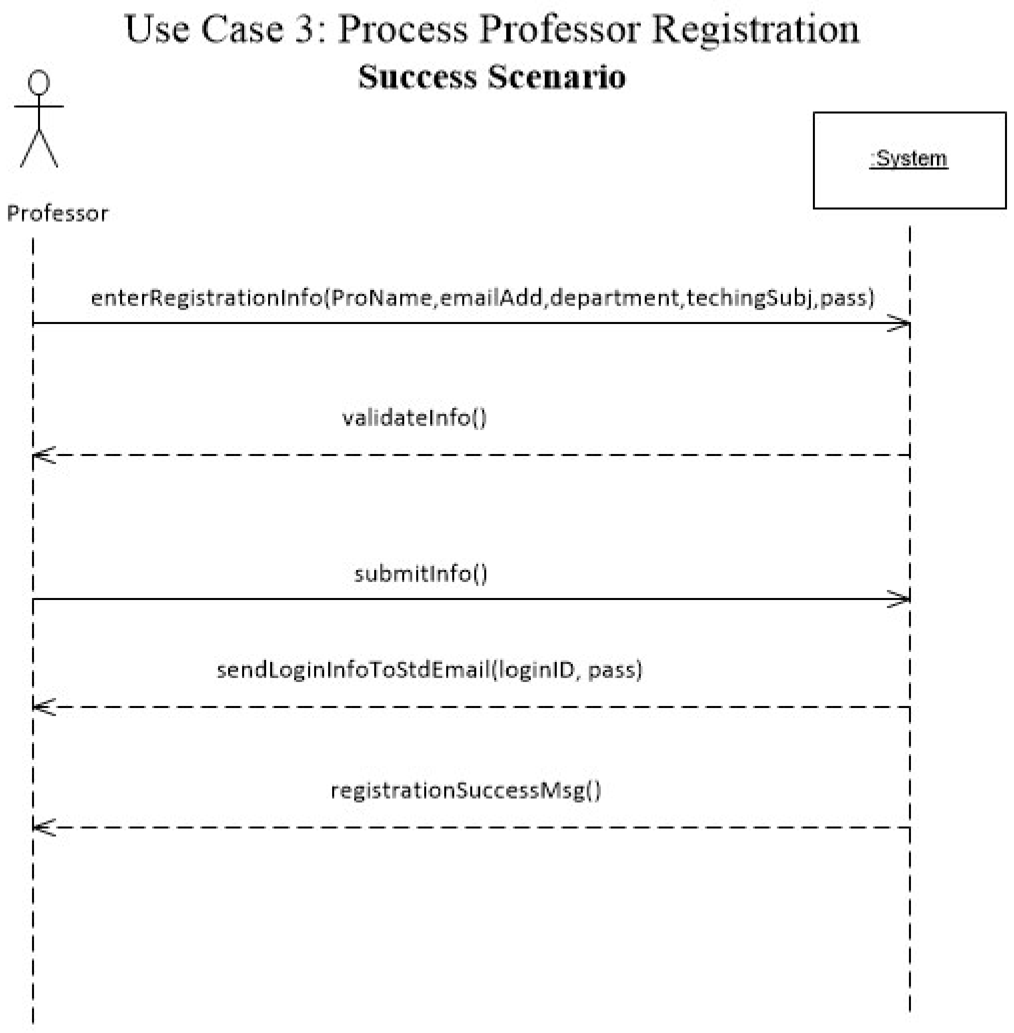
Use Case 3: Process Professor Registration
Success Scenario
Professor
:System
enterRegistrationInfo(ProName,emailAdd,department,techingSubj,pass)
validateInfo()
submitInfo()
sendLoginInfoToStdEmail(loginID, pass)
registrationSuccessMsg()

FIGURE 2.6: System Sequence Diagram: Use case 3 (Success Scenario)

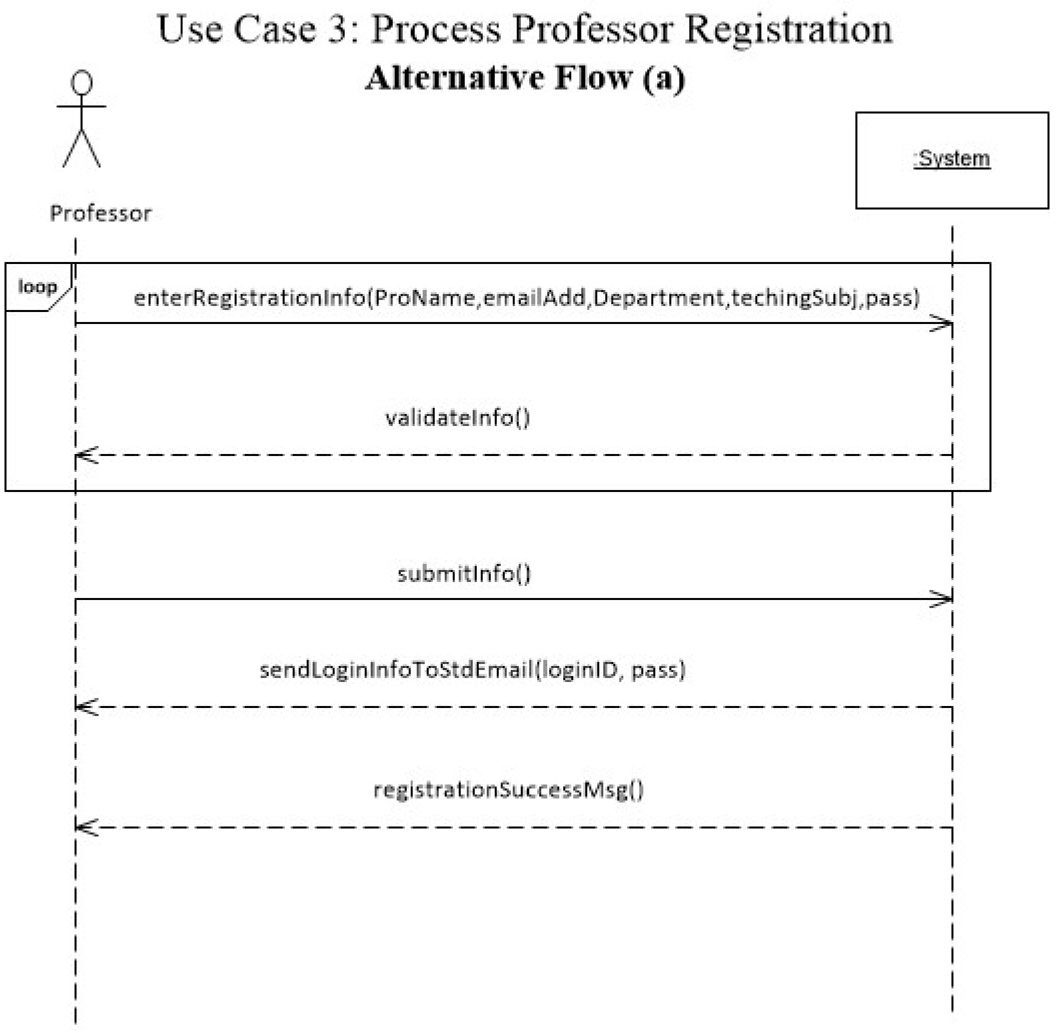
Use Case 3: Process Professor Registration
Alternative Flow (a)
Professor
:System
loop
enterRegistrationInfo(ProName,emailAdd,Department,techingSubj,pass)
validateInfo()
submitInfo()
sendLoginInfoToStdEmail(loginID, pass)
registrationSuccessMsg()

FIGURE 2.7: System Sequence Diagram: Use case 3 (Alternative Flow(a))

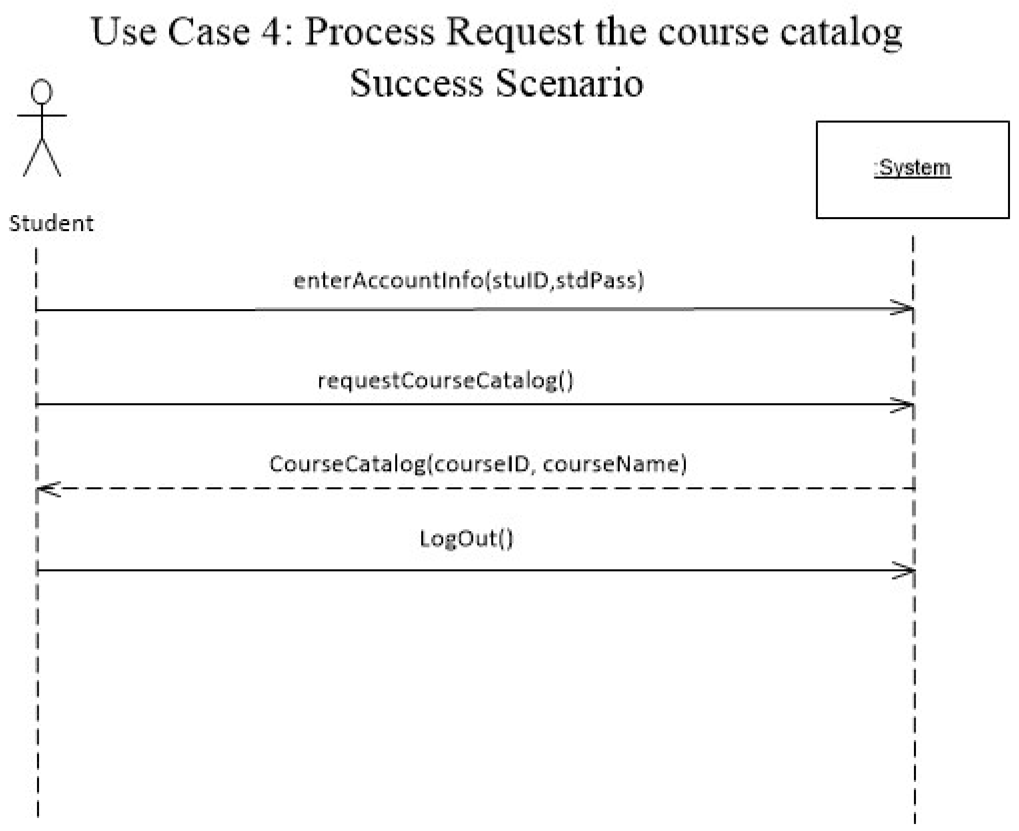
Use Case 4: Process Request the course catalog
Success Scenario
Student
:System
enterAccountInfo(stuID,stdPass)
requestCourseCatalog()
CourseCatalog(courseID, courseName)
LogOut()

FIGURE 2.8: System Sequence Diagram: Use case 4 (Success Scenario)

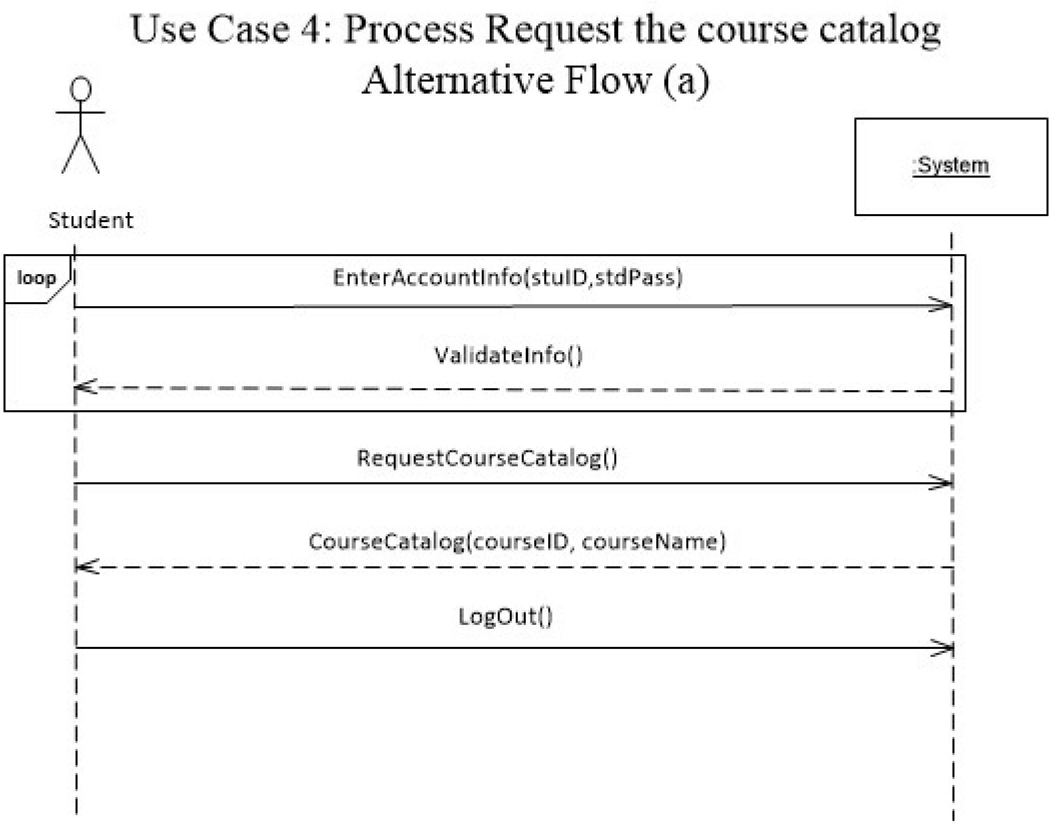
Use Case 4: Process Request the course catalog
Alternative Flow (a)
Student
:System
loop
EnterAccountInfo(stuID,stdPass)
ValidateInfo()
RequestCourseCatalog()
CourseCatalog(courseID, courseName)
LogOut()

FIGURE 2.9: System Sequence Diagram: Use case 4 (Alternative Flow(a))

2.4 Operation Contracts

Operation Contracts: enterAccountInfo
Operation: enterAccountInfo(StdLoginName,StdPassword)
Cross References: Use Case 1: Process selecting courses (Success Scenario)
Preconditions: Student has stable internet access to the system
Post conditions:
- A new enterAccountInfo instance eAI was created
- eAI was associated with current request to log into the system
- eAI. StdLoginName becomes variable StdLoginName
- eAI. StdPassword becomes variable StdPassword
- eAI was associated with a StdLoginName,StdPassword

Operation Contracts: selectCourses
Operation: selectCourses(MainCourses,AltCourses)
Cross References: Use Case 1: Process selecting courses (Alternative Flow (a))
Preconditions: Student successfully received course catalog
Post conditions:
- A new selectCourses instance sC was created
- sC was associated with current student logged into the system
- sC.MainCourses list becomes MainCourses list
- sC.AltCourses list becomes AlternativeCourses list
- sC was associated with a MainCourses,AltCourses

Operation Contracts: enterRegistrationInfo
Operation: enterRegistrationInfo(StdID,StdName,emailAdd,Semester,Department)
Cross References: Use Case 2: Process Student Registration (Success Scenario)
Preconditions: Student registration system
Post conditions:
- A new Registration instance enReInfo was created
- enReInfo was associated with current student registration
- enReInfo was associated with Student ID, Student Name,emailAdd, Semester and Department

Operation Contracts: enterRegistrationInfo
Operation: enterRegistrationInfo(StdID,StdName,emailAdd,Semester,Department)
Cross References: Use Case 2: Process Student Registration (Alternative Flow (a))
Preconditions: Student registration system
Post conditions:
- A new Registration instance enReInfo was created
- enReInfo was associated with current student registration
- enReInfo was associated with StudentID, Pass
- reReInfo was associated with message if StudentID, Pass wrong

Operation Contracts: enterRegistrationInfo
Operation: enterRegistrationInfo(ProName,emailAdd,department,techingSubj,pass)
Cross References: Use Case 3: Process Professor Registration (Success Scenario)
Preconditions: Professor registration system
- A new Registration instance enReInfoP was created
- enReInfoP was associated with current professor registration

- enReInfoP was associated with ProName,emailAdd,department,teachingSub,pass

Operation Contracts: enterRegistrationInfo
Operation: enterRegistrationInfo(ProName,emailAdd,department,techingSubj,pass)
Cross References: Use Case 3: Process Professor Registration (Alternative Flow (a))
Preconditions: Professor registration system
Post conditions:
- A new Registration instance enReInfoP was created
- enReInfoP was associated with current professor registration
- enReInfoP was associated with ProName,emailAdd,department,teachingSub,pass
- reReInfoP was associated with message if Pass wrong

Operation Contracts: requestCourseCatalog
Operation: requestCourseCatalog()
Cross References: Use Case 4: Process Request the course catalog (Success Scenario)
Preconditions: Student login in the system
Post conditions:
- A new CourseCatalog instance cC was created
- cC was associated with current Student ID
- cC was associated with StdName,emailAdd,department,pass
- cC was associated allCourse

Operation Contracts: requestCourseCatalog
Operation: requestCourseCatalog()
Cross References: Use Case 4: Process Request the course catalog (Alternative Flow (a))
Preconditions: Student login in the system
Post conditions:
- A new CourseCatalog instance cC was created
- cC was associated with current Student ID
- cC was associated with StdName,emailAdd,department,pass
- cC was associated allCourse
- cC was associated with message if pass wrong

Chapter 3

Domain Modeling

3.1 Domain Modeling

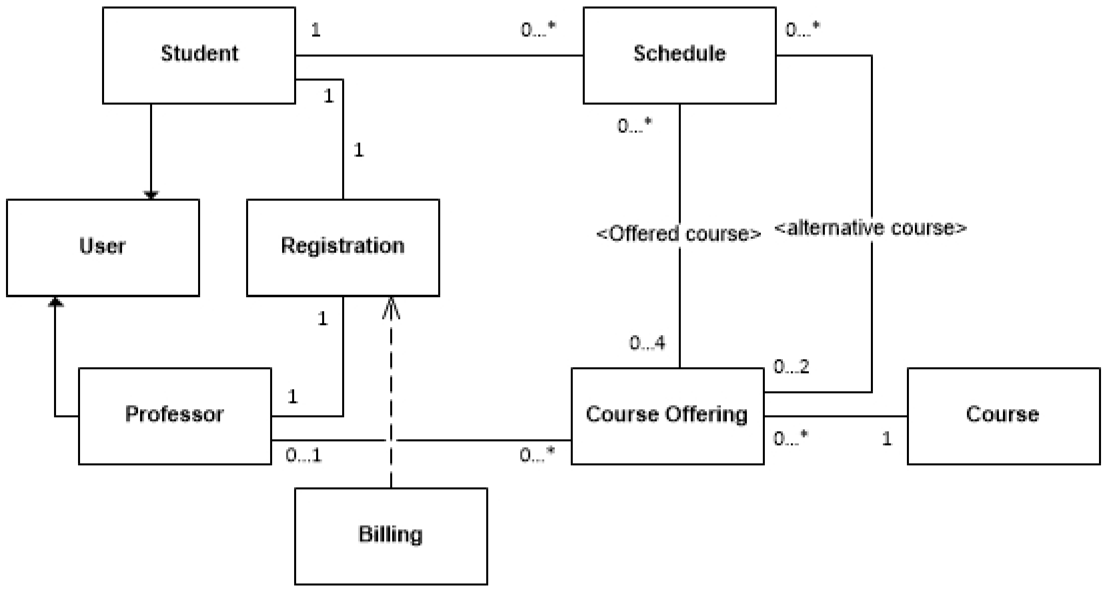

FIGURE 3.1: Domain Model for first iteration

Chapter 4

Class Modeling and Dynamic Modeling

4.1 Class Diagram

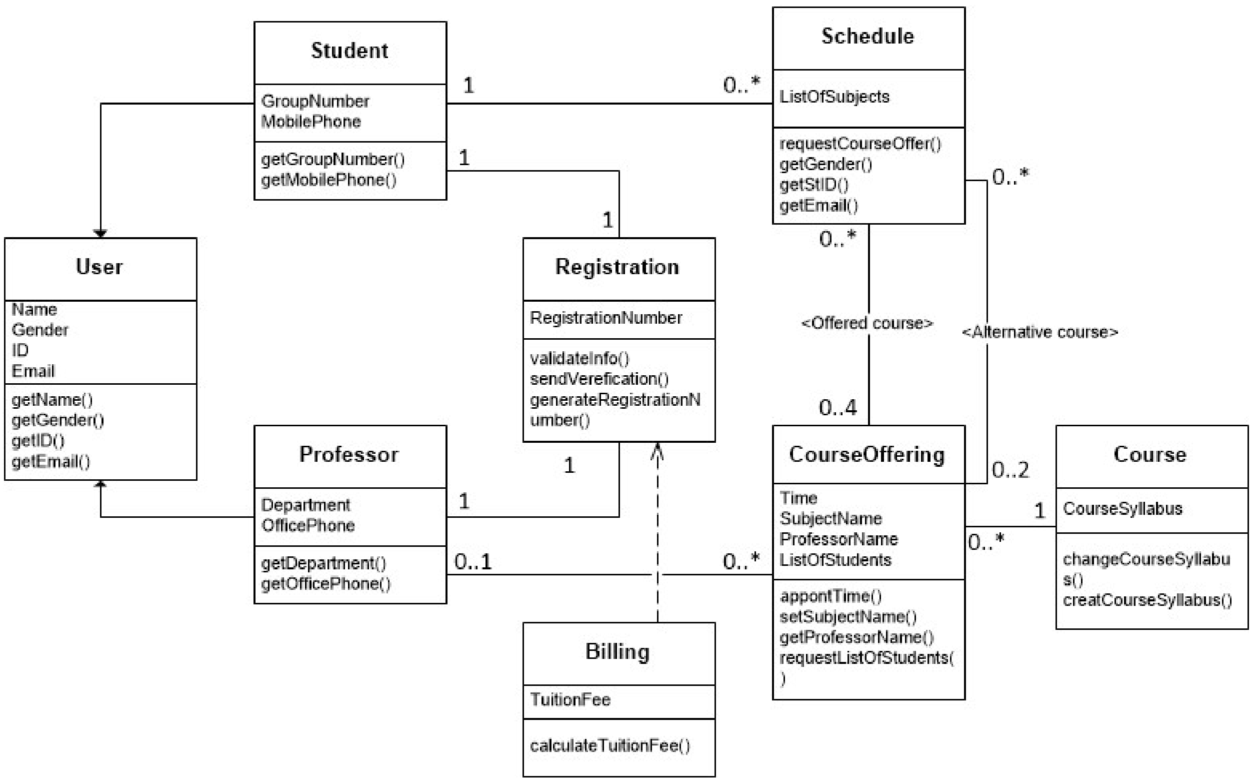

FIGURE 4.1: Class diagram for first iteration

4.2 Sequence Diagram

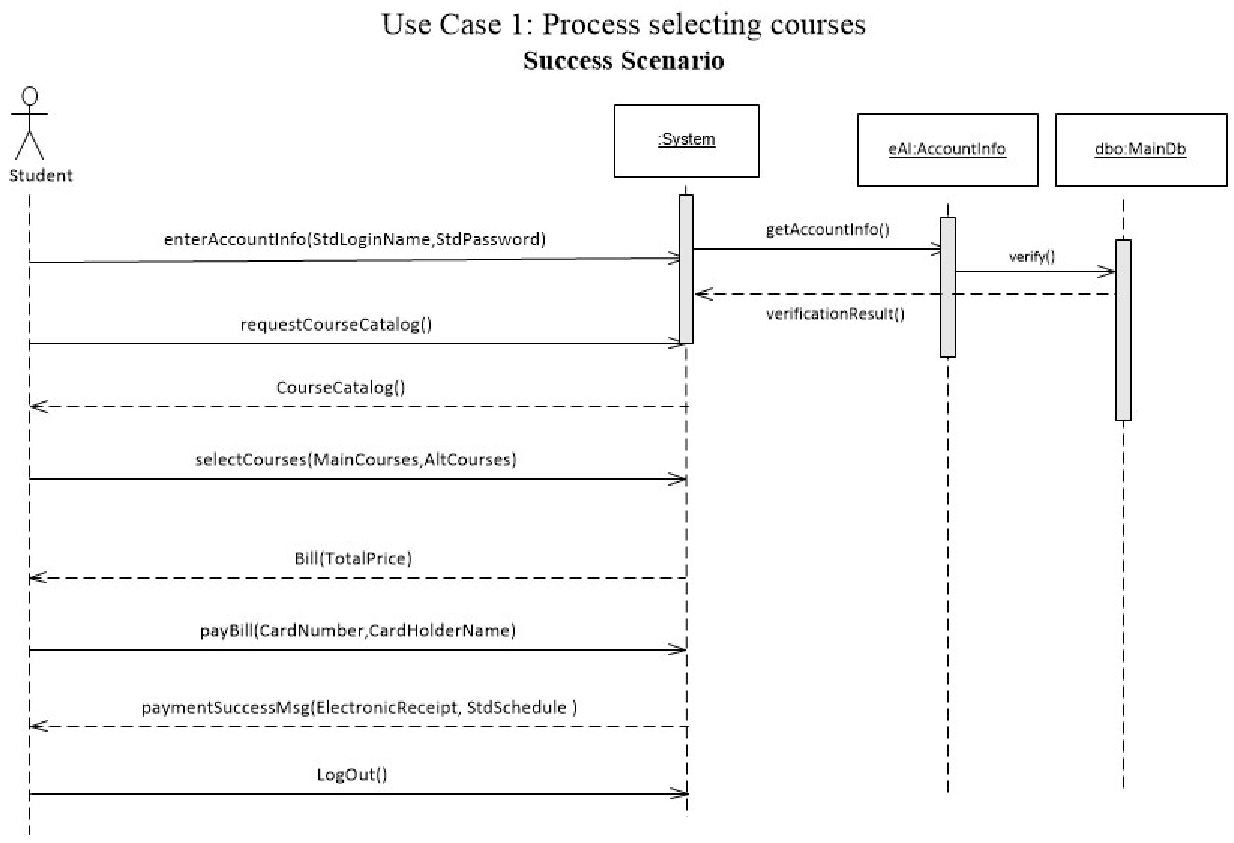
Use Case 1: Process selecting courses
Success Scenario
Student
:System
eAI:AccountInfo
dbo:MainDb
enterAccountInfo(StdLoginName,StdPassword)
getAccountInfo()
verify()
verificationResult()
requestCourseCatalog()
CourseCatalog()
selectCourses(MainCourses,AltCourses)
Bill(TotalPrice)
payBill(CardNumber,CardHolderName)
paymentSuccessMsg(ElectronicReceipt, StdSchedule)
LogOut()

FIGURE 4.2: Sequence diagram for use case 1: Success Scenario

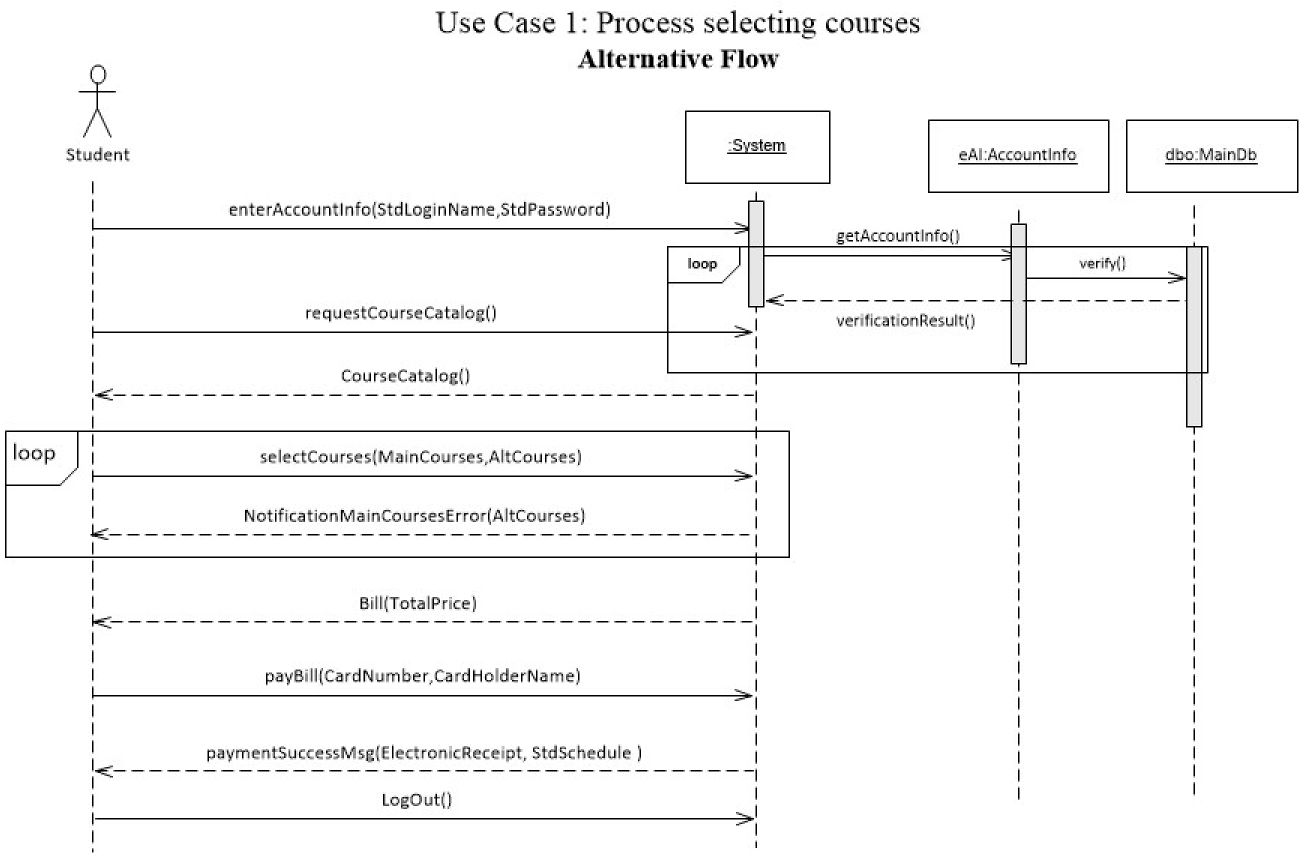
Use Case 1: Process selecting courses
Alternative Flow
Student
:System
eAI:AccountInfo
dbo:MainDb
enterAccountInfo(StdLoginName,StdPassword)
getAccountInfo()
loop
verify()
verificationResult()
requestCourseCatalog()
CourseCatalog()
loop
selectCourses(MainCourses,AltCourses)
NotificationMainCoursesError(AltCourses)
Bill(TotalPrice)
payBill(CardNumber,CardHolderName)
paymentSuccessMsg(ElectronicReceipt, StdSchedule)
LogOut()

FIGURE 4.3: Sequence diagram for use case 1: Alternative Flow

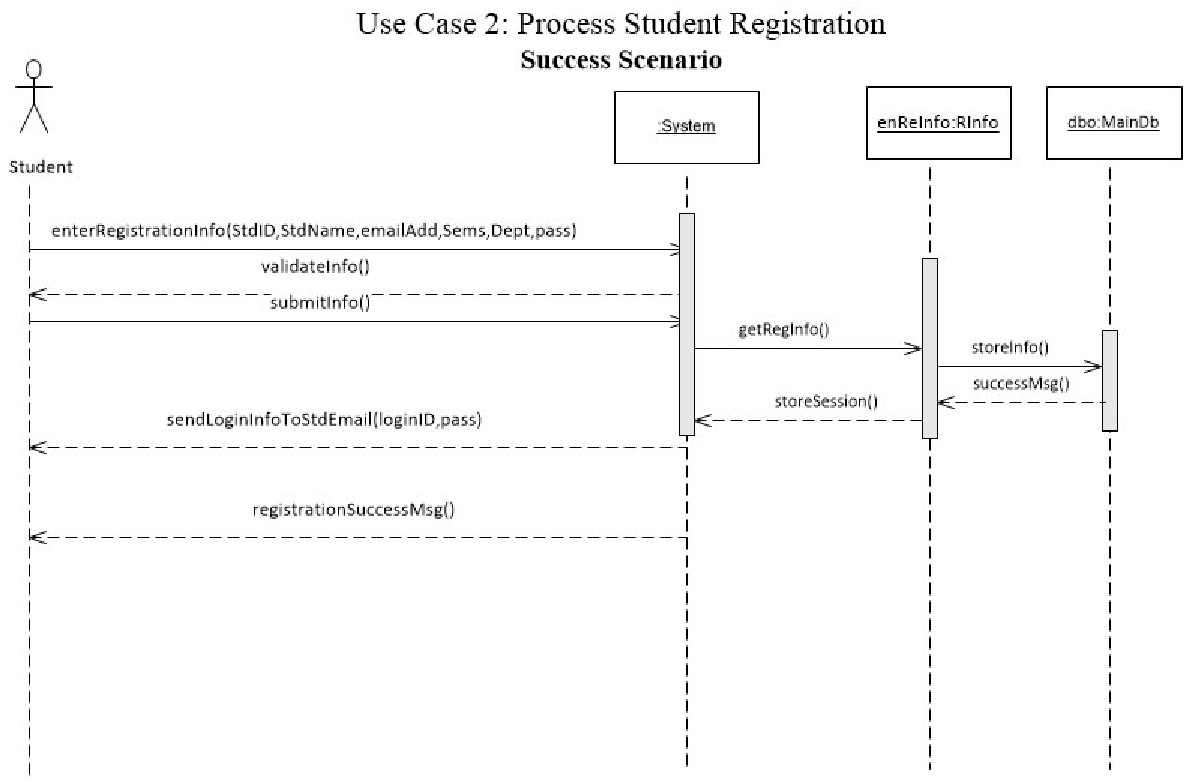

FIGURE 4.4: Sequence diagram for use case 2: Success Scenario

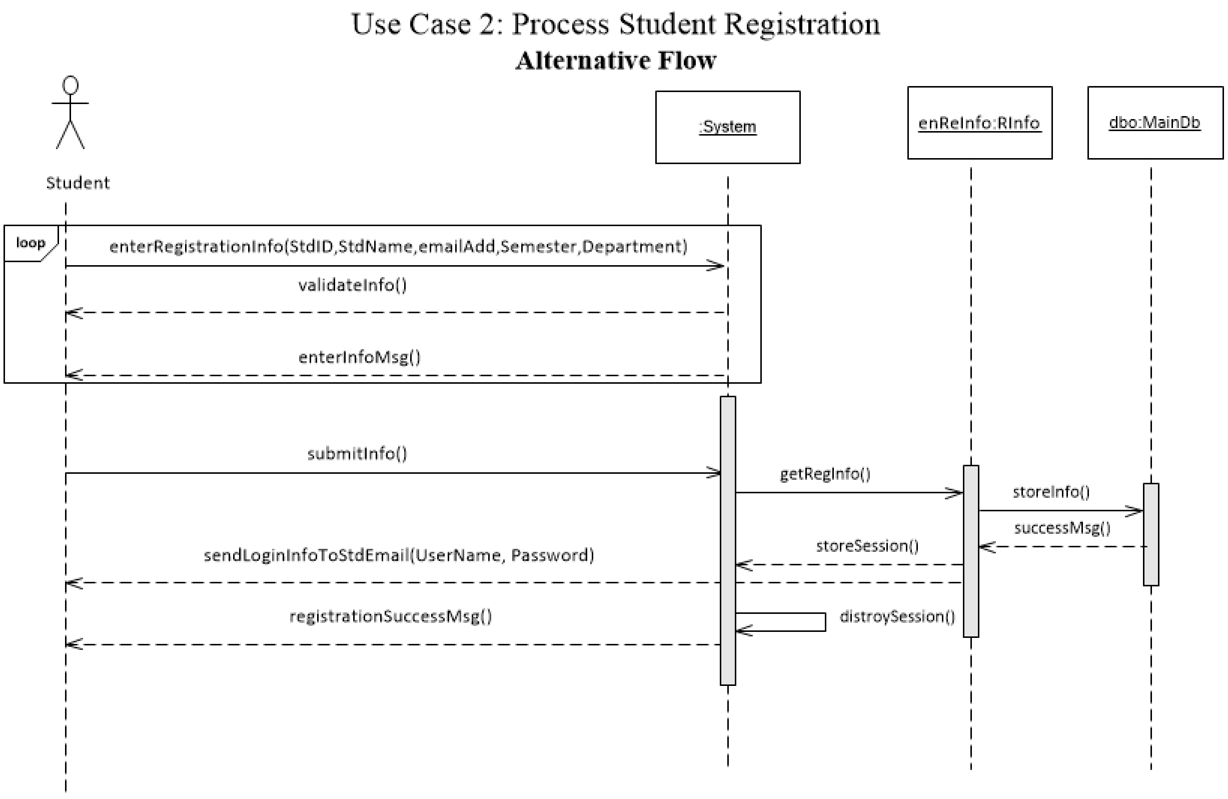

FIGURE 4.5: Sequence diagram for use case 2: Alternative Flow

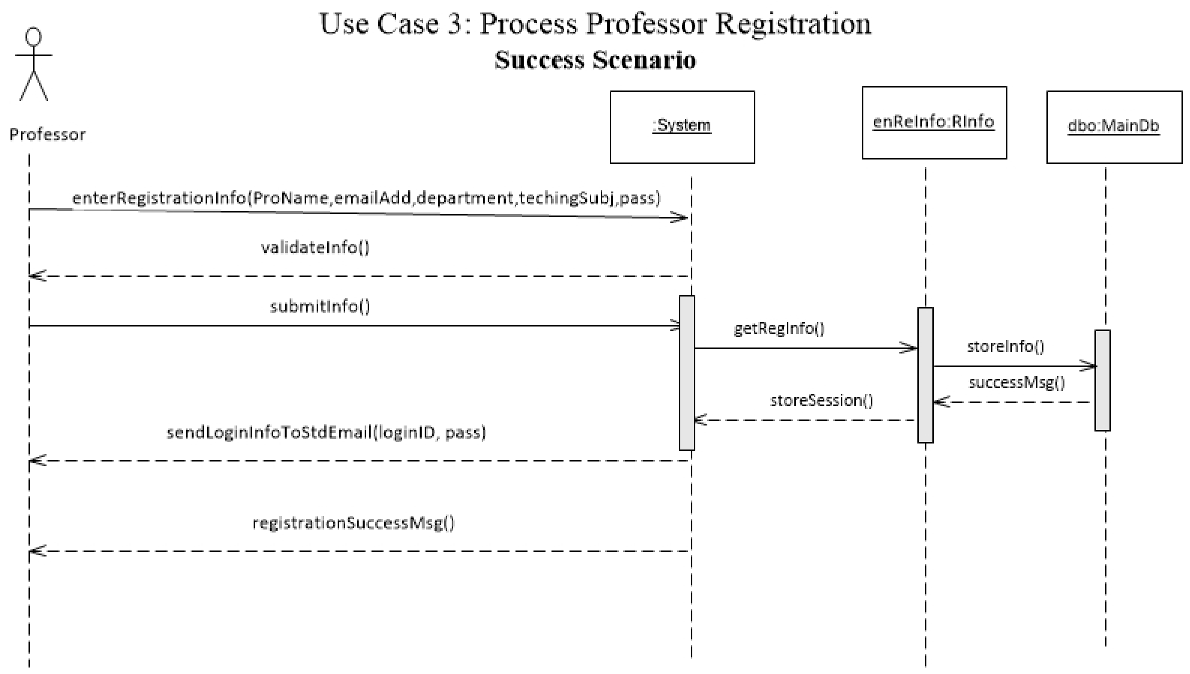

FIGURE 4.6: Sequence diagram for use case 3: Success Scenario

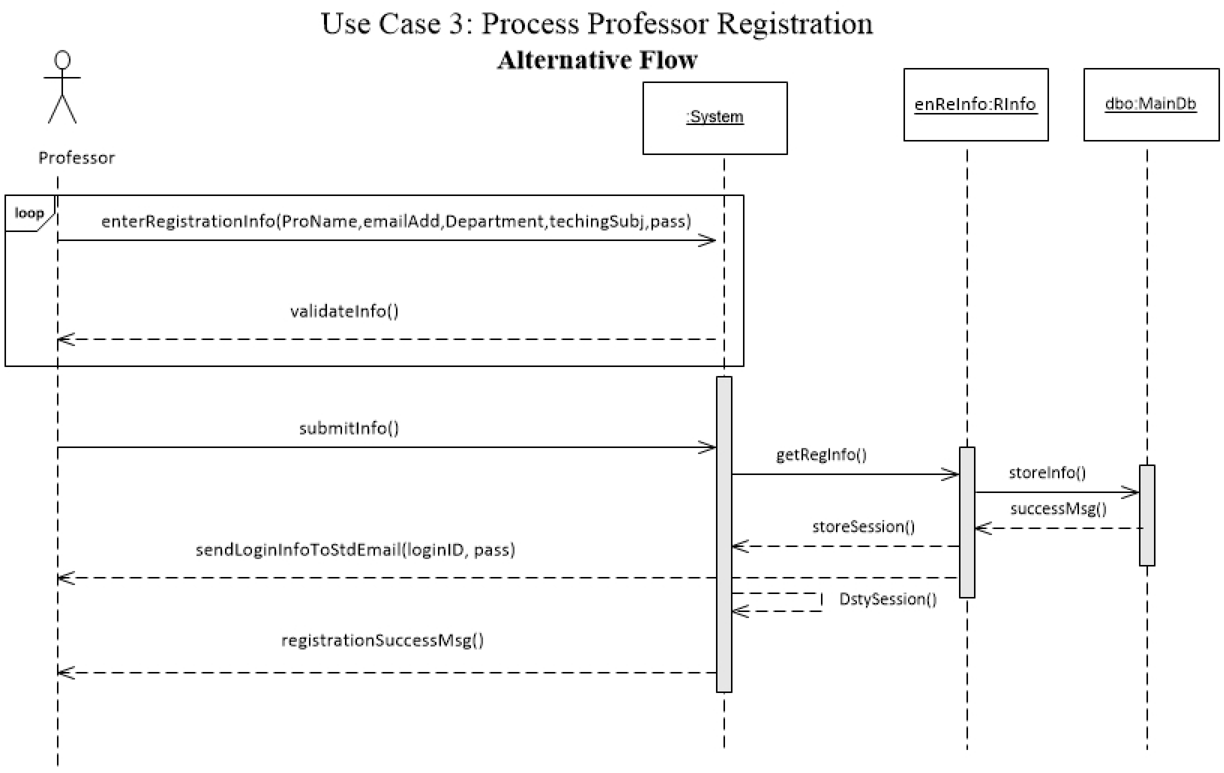

FIGURE 4.7: Sequence diagram for use case 3: Alternative Flow

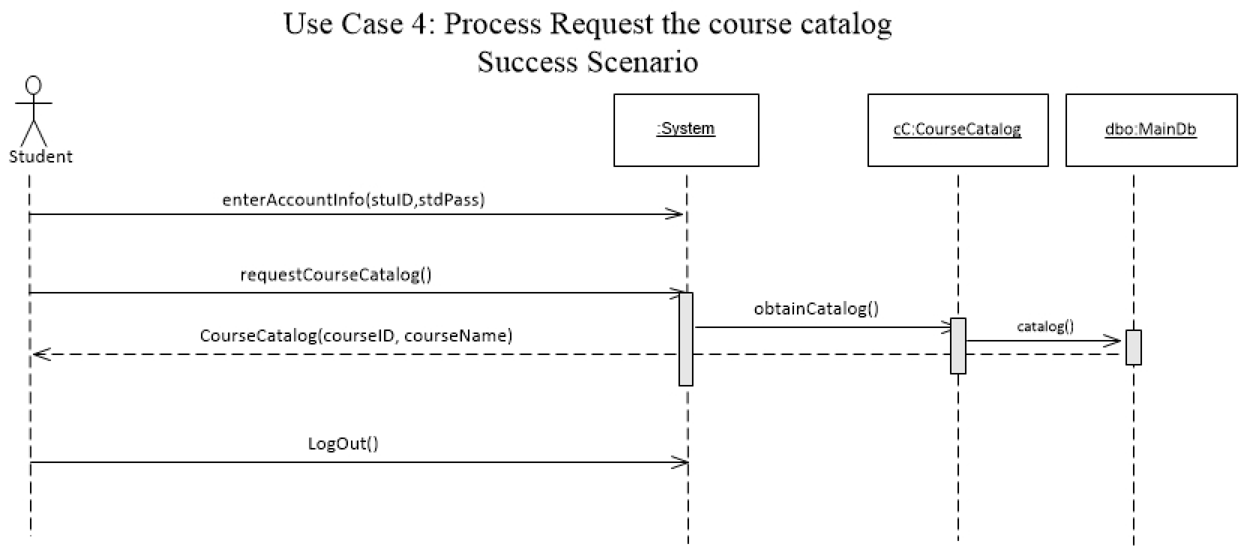
Use Case 4: Process Request the course catalog
Success Scenario
Student
:System
cC:CourseCatalog
dbo:MainDb
enterAccountInfo(stuID,stdPass)
requestCourseCatalog()
obtainCatalog()
catalog()
CourseCatalog(courseID, courseName)
LogOut()

FIGURE 4.8: Sequence diagram for use case 4: Success Scenario

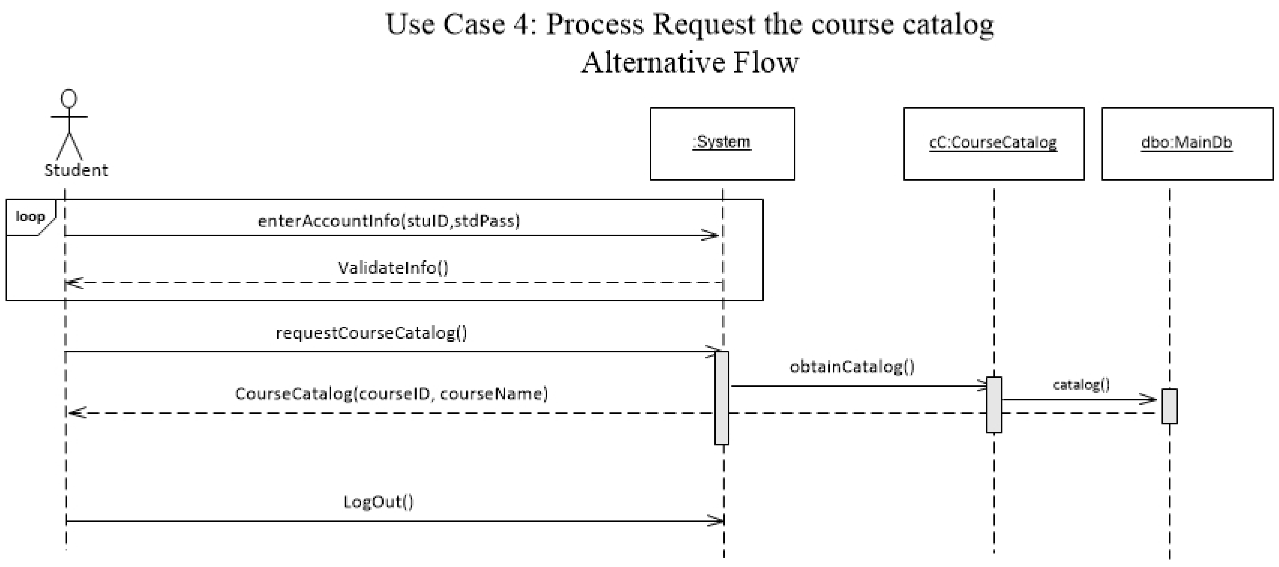

FIGURE 4.9: Sequence diagram for use case 4: Alternative Flow

Appendix A

Supplementary Specification

A.1 Non functional Requirements

Security
All usage requires user authentication.
Usability
Human Factors
The customer will be able to see a large-monitor display of the POS. Therefore:
-Text should be easily visible from 1 meter.
-Avoid colors associated with common forms of color blindness.
Speed, ease, and error-free processing are paramount in sales processing, as the buyer wishes to leave quickly, or they perceive the purchasing experience (and seller) as less positive.
The cashier is often looking at the customer or items, not the computer display. Therefore, signals and warnings should be conveyed with sound rather than only via graphics.
Reliability
Recoverability
If there is failure to use external services (payment authorizer, accounting system, ...) try to solve with a local solution (e.g., store and forward) in order to still complete a sale. Much more analysis is needed here...

A.2 Glossary

application – a major, fully-functional, stand-alone work product that is developed by a development organization for use by a user organization.
architecture – the strategic design that captures the most important, pervasive design decisions and their rationales. An architecture must fulfill (and is therefore validated against) the architecturally significant operational and quality requirements. The architecture drives and constrains the tactical (i.e., detailed) design. See also system architecture and software architecture.
authentication – a security technique used to determine if an actor actually is who he seems to be. Contrast with authorization.
authorization – a security technique used to determine if an actor has the required privileges to use specific system capabilities. Contrast with authentication.
browser – a software tool enabling the display and navigation of web pages on the World Wide Web.
business goal – a goal of the application that is primarily of management interest.