Oussy Sylla
CS 6359.001
10/15/2020

Module8_Activity_Question1

1. Functional Requirements

Requirement	Description	Priority
R1	The user must be able to login to SAMS System	5
R2	The students must be able to contact University advisors and counselors through the SAMS	4
R3	The students must be able to search universities	5
R4	Students shall be able to send their application through the system SAMS	4
R5	The Admins shall be able to manage all users.	3

2. Use Cases
   a. Use Case 1
      Abstract Use Case: UC1. login
      High level Use Case:

TUCBW	The User input their username and password then click on the login button.
TUCEW	The user will get directed to the welcome page when the input credential is correct.

Expanded Use Case:

UC1. login	
Precondition: the user must already have an active account	
Actor: User	System: SAM/Access
1. TUCBW The User needs to input their username and password then click on the login button.	2. The system authenticate the user credential and route to the welcome page
3. TUCEW The user will get directed to the welcome page when the input	

credential is correct.	

b. Use Case 2
Abstract Use Case: UC2. contact advisor/counselor
High level Use Case:

TUCBW	The student contacts University advisors and counselors through the system.
TUCEW	The student receives a success message after the message is sent

Expanded Use Case:

UC2. contact	
Precondition: the user must login first in the system	
Actor: Student	System: SAMS/Contact
1. TUCBW The student contacts University advisors and counselors through the system.	2. The system provides the student with the form
3. Student fills the form and sends it.	4. The system sends the email and the response to the student
5. TUCEW The student receives a success message after the message is sent	

c. Use Case 3
Abstract Use Case: UC3. search universities programs
High level Use Case:

TUCBW	The student search universities programs
TUCEW	The student is shown a list of Universities programs meeting the criteria.

Expanded Use Case:

UC3. Search	
Precondition: the user must login first in the system	
Actor: Student	System: SAMS/Program
1. TUCBW The student search universities programs	2. The system shows the submenu containing criteria for the search

3. The Student fills the search criteria and submit it	4. The system collects all programs meeting the search criteria
5. TUCEW The student is shown a list of Universities programs meeting the criteria.	

d. Use Case 4
Abstract Use Case: UC4. Application
High level Use Case:

TUCBW	The student sends the application through the system
TUCEW	The student get a response that the application is submitted

Expanded Use Case:

UC4. Application	
Precondition: the user must login first in the system	
Actor: Student	System: SAMS/Program
1. The students shall be able to send their application through the system	2. The system shows the submenu containing criteria for the search
3. The Student fills the application form and submit it.	4.The system forward the application to the chosen university and the response to the student
5. TUCEW The student get a response that the application is submitted	

e. Use Case 5
Abstract Use Case: UC5. Management
High level Use Case:

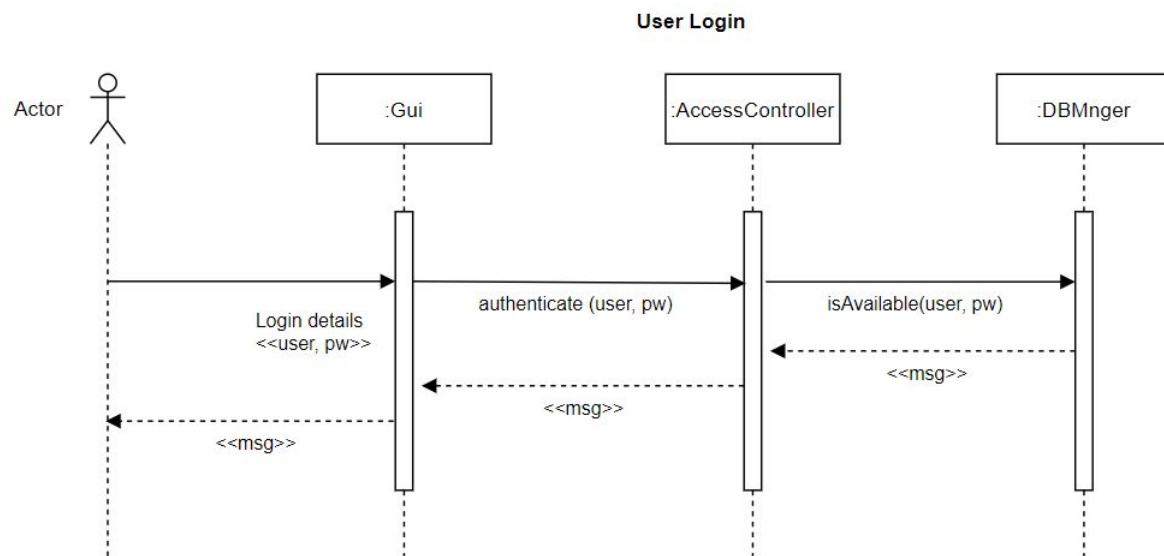| TUCBW	The Admin manage all users.
TUCEW	The Admins shall see a success message after the operation

Expanded Use Case:

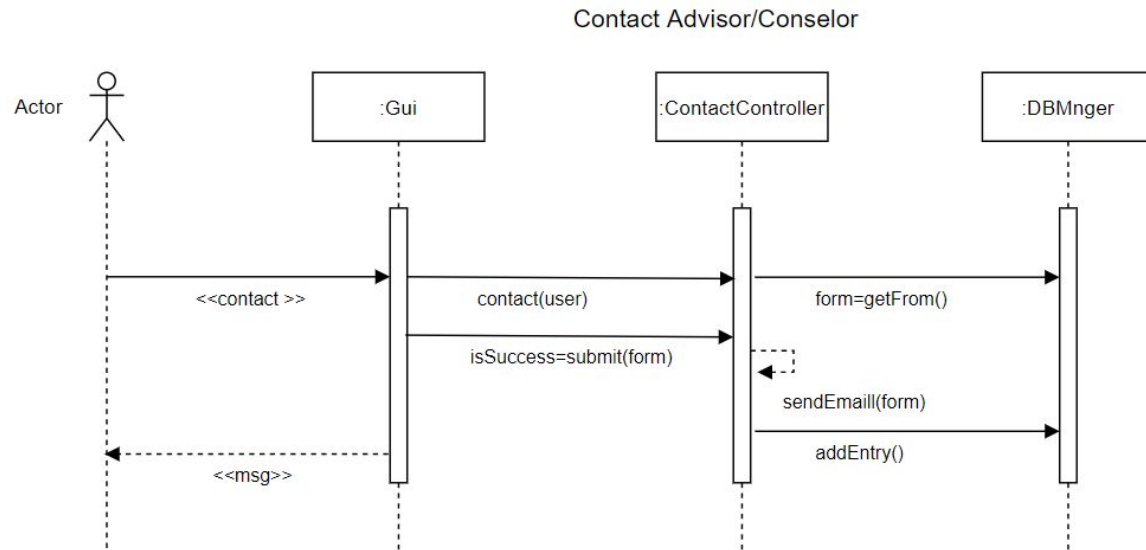| UC4. Application |
|---|
| Precondition: the user must login first in the system and be an Admin |

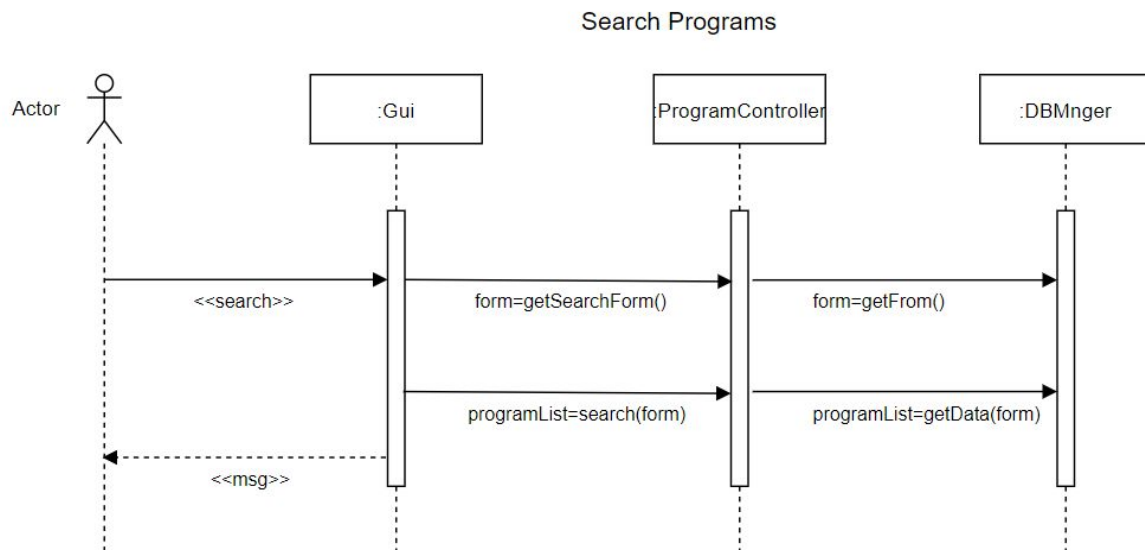| Actor: Admin	System: SAMS/Management
1. The Admins shall be able to manage all users.	2. The system shows the submenu containing criteria for the operation
3. The Admin fills the submenu for the operation and submit it	4.The system perform the operation and return the response to the Admin
5. TUCEW The Admin shall see a success message after the operation	

4. Design Sequence Diagram
a. Use Case 1 (login)
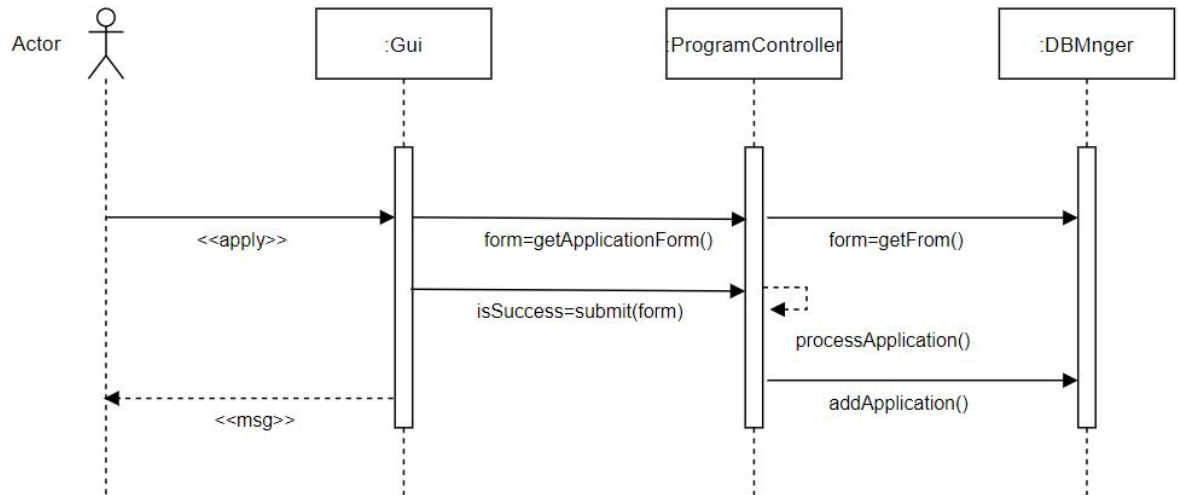

b. Use Case 2 (contact advisor/counselor)

Contact Advisor/Conselor

Actor → :Gui : <<contact >>
:Gui → :ContactController : contact(user)
:Gui → :ContactController : isSuccess=submit(form)
:ContactController → :DBMnger : form=getFrom()
:ContactController → :ContactController : sendEmaill(form)
:ContactController → :DBMnger : addEntry()
:Gui → Actor : <<msg>>

c. Use Case 3 (search universities programs)

Search Programs

Actor → :Gui : <<search>>
:Gui → :ProgramController : form=getSearchForm()
:ProgramController → :DBMnger : form=getFrom()
:Gui → :ProgramController : programList=search(form)
:ProgramController → :DBMnger : programList=getData(form)
:Gui → Actor : <<msg>>

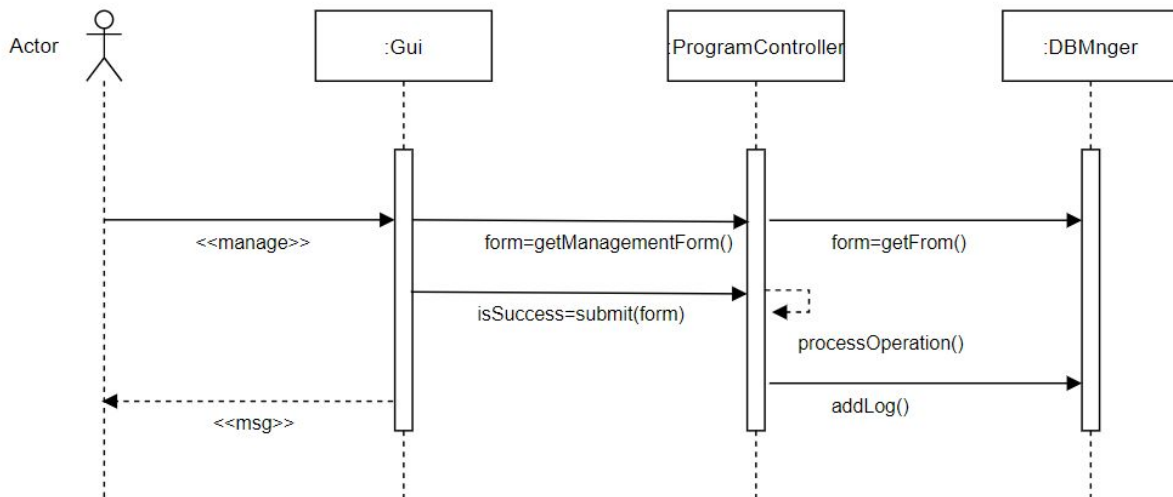### d. Use Case 4 (Application)
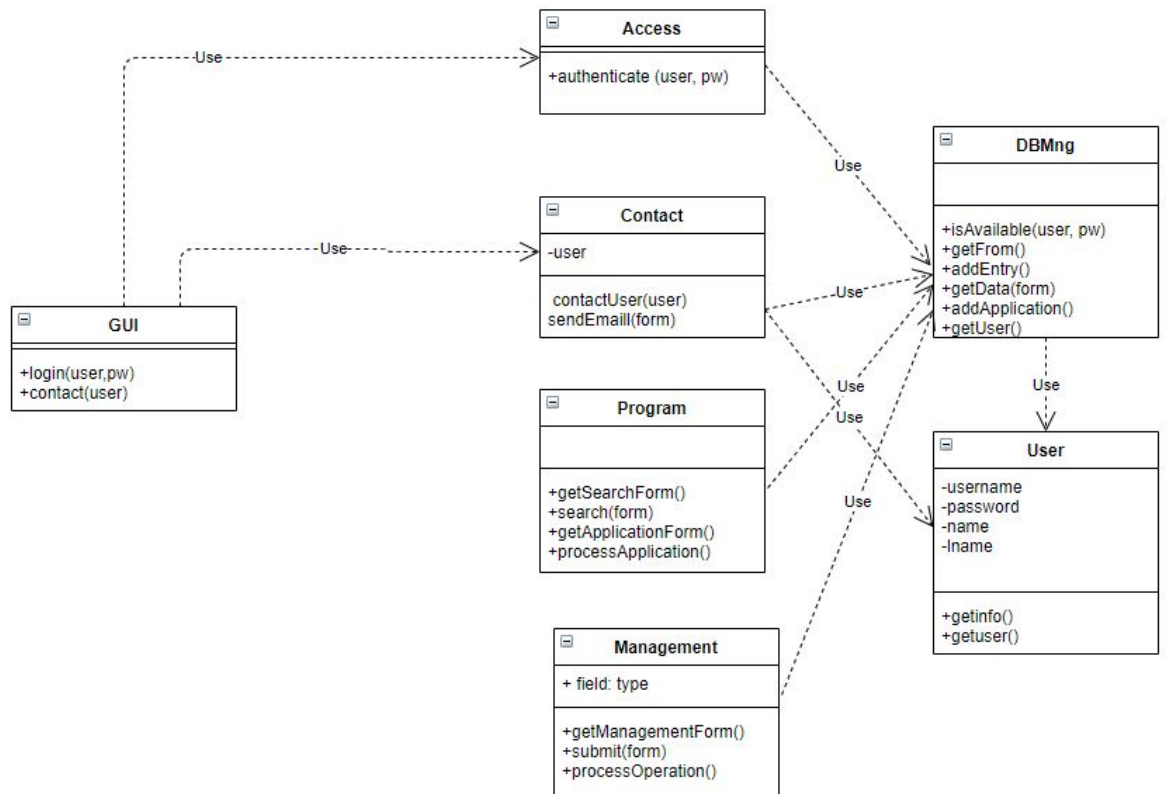
Programs Application

e. Use Case 5 (Management)

Management

5. Design Class Diagram

6. Improvements from GRASP patterns
- Access , Contact, Progream, Management are controllers which use the Controller Design Pattern. This allows the separation of concerns and helps the design to be high cohesion and low couplings.
- The Controllers and DBMgr are indirection and Pure Fabrication, this allows coupling, reusability, and allows to collect responsibilities.