

# Registrar System for Graduates Programs

## Requirements Specification and Analysis

1.0

19-03-2018

Muhammed Ali ALTINEL

Prepared for  
CSE490 -Project



IŞIK UNIVERSITY  
COMPUTER  
SCIENCE AND  
ENGINEERING

## Table of Contents

1.	Introduction.....	1
1.1.	Purpose of the System .....	1
1.2.	Scope of the System .....	1
1.3.	Objectives and Success Criteria of the Project .....	1
1.4.	Definitions, Acronyms, and Abbreviations .....	1
2.	Current System.....	1
3.	Proposed System .....	2
3.1.	Overview.....	2
3.2.	Functional Requirements .....	2
3.3.	Nonfunctional Requirements .....	2
	Usability.....	2
	Reliability .....	2
	Performance .....	2
	Supportability.....	2
	Implementation .....	2
	Interface .....	3
	Packaging.....	3
	Legal.....	3
3.4.	System Models.....	3
	Scenarios.....	3
	Use case model .....	15
	Object model.....	16
	Dynamic model.....	<b>Error! Bookmark not defined.</b>
	User interface—navigational paths and screen mock-ups.....	20
4.	Glossary .....	20
5.	References.....	20

## **REQUIREMENTS ANALYSIS DOCUMENT [1]**

### **Introduction**

#### **1.1. Purpose of the System**

The System is registration, information and overall control system. Students would add/delete their personal schedule in a semester and can track their situations. Lecturer will have information their classes, courses and students. Admin would have control on overall system basically.

#### **1.2. Scope of the System**

This systems scope will be large because of its functionality. Admin would set courses at every semester and can manipulate on students and lecturers daily. Students, lecturers and admin can use messaging system on daily. Showing material will be daily function. Searching and filtering data is major for all user's type.

#### **1.3. Objectives and Success Criteria of the Project**

- Accessibility is the must for lecturer and admin.
- Data lose is not wanted
- Deadline is fixed
- Rate of usage is stable for max
- HTTP protocols are used

#### **1.4. Definitions, Acronyms, and Abbreviations**

- DAL is data access layer
- MVC is model view controller
- HTTP is hypertext transfer protocol
- Domain is main model manipulation

### **2. Current System**

Campus online system([campus.isikun.edu.tr](http://campus.isikun.edu.tr)) is current system that we will build it. Actors of system has large range of user. We are going to decrease the number of users in order to get efficient system.

Registrar

### **3. Proposed System**

Our users are academic stuff and graduate students. This will be cause low traffic and maximum performance. Efficiency is our key.

#### **3.1. Overview**

Systems actors are students, lecturers as we mention before.

#### **3.2. Functional Requirements**

The System provides registrar functions to various users. Lecturers and students will have common platform in order to getting in touch, making course arrangements and getting information about what they need on School System. Membership, admin panel, student panel and lecturer panel are the basic packages that provide basic functions as described on use case model.

#### **Nonfunctional Requirements**

##### **Usability**

The System have different interface for each unique user and their role. This provide easy accessibility on one system.

##### **Reliability**

The system has authentication in its nature of membership and this is a secure process in order to provide safe platform and it will have a SSL certification for more secure platform.

##### **Performance**

The System is going to run overall devices. It has responsive user interface that is why every user will have same user experience. Data access layer will be independent about platform it means every Operating System and browsers will accept.

##### **Supportability**

The system will be running on Linux machine for distiribution.HTML 5 and CSS 3 are acceptable all new versions of browsers.

##### **Implementation**

We are going to use Python as a programming language and its framework which is Django will be using to build our platform. Its design methodology is Model-View-Controller. It has not got any hardware dependencies.HTML5, CSS3 and Bootstrap will be using for basic.

Registrar

## **Interface**

The system will not interact with any existing system. Data provider will be job for next team.

## **Packaging**

There is no installation process.

## **Legal**

It will be concern for next team.

### **3.3. System Models**

### **3.4. Scenarios**

**Scenario Name:** Add New Course

**Participating Actor Instances:** Taner: System Admin

**Flow of Events:**

1. Taner is a person who defined by the system administrator at the beginning of the project. So, he has own panel to manage the system. Now, the new item should be added in to the system. Therefore, he can login to system as an admin.
2. Taner goes to admin panel, then admin functions will be open in the crud operation page. He clicks to “+”(plus) icon
3. “Add New Course” page has own attributes: name, class, lecturer, code of class
4. Taner adds “System Programming” as a new class and its attributes are name: “System Programming”, “DK-201”, Lecturer: “Armağan Elibol”, code: “SE-322
5. Taner clicks to the “Save” button to add the item to the list.

**Scenario Name:** Delete Course

**Participating Actor Instances:** Taner: System Admin

**Flow of Events:**

1. Taner is a person who defined by the system administrator at the beginning of the project. So, he has own panel to manage the system. Now the item should be deleted from the system. Therefore, he can login to system as an admin.
2. Taner goes to admin panel, then admin functions are appeared in here. He clicks to “X”(delete) icon
3. Taner clicks “YES” the pop up alert that written “Are you delete the course?”

Registrar

**Scenario Name:** Assign Role

**Participating Actor Instances:** Taner: System Admin

**Flow of Events:**

1. Taner is an admin. He logs in to the system as admin. Then, he is redirected to admin panel where there are functions of the admin to manage the system.
2. He clicks to “Assign Role” button.
3. Next page has own attributes. These are e-mail list and role list on the select box to match
4. Taner clicks to “Assign” button to finish the operation.

**Scenario Name:** Change Personal Information

**Participating Actor Instances:** Taner: Admin, Lecturer, Student

**Flow of Events:**

1. Taner is a person who defined a system administrator at the beginning of the project. His task (or role) is to manage the system, and also he has own profile. He wants to change his profile information. Then, he logs in to the system. Then, he goes to “My Profile” button.
2. Incoming page has own attributes. These are first name, last name, e-mail, password, phone and profile picture. Each attribute has “Change” button. “Taner” for first name, “Eskil” for last name, “[tanereskil@isikun.edu.tr](mailto:tanereskil@isikun.edu.tr)” for e-mail, “123456” for password, “05333333333” for phone and “...uploads” for profile picture.
3. Then, he clicks to “Save” button to finish the operation.

**Scenario Name:** Search

**Participating Actor Instances:** Taner: Lecturer, Student, System Administrator

**Flow of Events:**

1. Taner is a person who defined a system administrator at the beginning of the project. His task (or role) is to manage the system, and also he has own profile. He wants to change his profile information. Then, he logs in to the system.
2. Taner goes to “Search” field. Type students name or lecturer or courses name
3. Taner sees the page of list according to his criteria

**Scenario Name:** Crud Process

**Participating Actor Instances:** Taner: Admin

**Flow of Events:**

1. Taner is a person who defined a system administrator at the beginning of the project. His task (or role) is to manage the system, and also he has own profile. He wants to make some CRUD operations on courses and staffs. Then, he logs in to the system.
2. When Taner is redirected to related page, he clicks to “ADD/DELETE/UPDATE” buttons

Registrar

3. There are some components which relevant input text-fields. These are first name, last name, phone, e-mail address, name of course and lecturer Then, he clicks to “Save” button to finish the operation.

**Scenario Name:** Delete Member

**Participating Actor Instances:** Taner: System Administrator

**Flow of Events:**

1. Taner is a person who was defined as admin at beginning of the project. He can log in to the system with his authentication information (*i.e. e-mail address and password*) over login panel. He has a standard menu with “Admin Panel” linked text-label.
2. Taner goes to “Admin Panel” page on menu-navigation bar from above the header section. When the “Admin Panel” page is loaded, there are functions which are managed by Admin.
3. Then, he clicks to “Delete Member” button. When the “Delete Member” page is loaded, there is a list that contains some attributes that are ID of member, first name, last name, e-mail address, phone. Each row has a ratio button at the beginning of the line.
4. Taner selects an instance to delete from the system. “73” for ID of member, “Taner” for first name, “Eskil” for last name, “[tanereskil@isikun.edu.tr](mailto:tanereskil@isikun.edu.tr)” for e-mail address, “0533333333” for phone.
5. After he selected an instance, he clicks to “Save” button to finish the operation.

**Scenario Name:** Register-for-Program

**Participating Actor Instances:** Ali: Visitor

**Flow of Events:**

1. Ali is a person who is not defined as member (pre-register). Therefore, he does not have authentication information , so he cannot log in to the system over login panel. As we have mentioned earlier, admin is a member who was approved by system administrator. Thus, admin has authentication to register to program.
2. Ali clicks to “Register” button at main page. Then, he goes to the page.
3. When he has gone to the “Pre-Register” page, there is a form that runs with POST HTTP message. The form contains some components which are first name, last name, phone, e-mail address, password and profile picture.
4. Ali types her information to get authentication from the system.
5. She enters “Ali” for first name, “ALTINEL” for last name, “05333333333” for phone, “[malialtinel@gmail.com](mailto:malialtinel@gmail.com)” for e-mail address, “\*\*\*\*\*” for password and “...Documents/avatar.jpeg” for profile picture.
6. Then, there is ratio button. It says “Are you going to be an member?”. The answers are “Yes” and “No”. Ali selects “Yes” option to become an author on the system.
7. Finally, she clicks to “Save” button to finish the operation.
8. He waits for approving

**Scenario Name:** Login

**Participating Actor Instances:** Ali:Student,Lecturer,Admin

**Flow of Events:**

1. Ali is a person who is normal user. Therefore, he can be authenticated on the system. He can log in to the system with authentication information
2. Ali clicks to “Login” button on main page. Then, he goes to the page. When she has gone to the “Login” page, there is a login form that runs with POST HTTP message. The form contains e-mail address and password components.
3. He types her authentication information. “malialtinel@gmail.com” for e-mail address and “\*\*\*\*\*” for password.
4. Then, he clicks “Let me go inside” button. There is also clean button to refresh the form.

**Scenario Name:** Logout

**Participating Actor Instances:** Ali: Lecturer,Student, Admin

**Flow of Events:**

1. Ali is a member whose role is author. Therefore, he can be authenticated on the system through her role. He can log in to the system with authentication information .
2. He logs out from the system. He clicks to the “Logout” button in header section.
3. Finally, he can be logged out.

**Scenario Name:** Approve Student

**Participating Actor Instances:** Ali:Student,Ali:Admin

**Flow of Events:**

1. Ali is a member that wants to become a student, therefore he has to select “Yes” option for author question at registration phase. After he has typed variables on to the relevant fields and has selected option, clicks to “Save” button to finish the operation. When he has clicked, back end of the system runs, gets information and a query begins to run.
2. While execution is completing, a notification goes to relevant page in Admin Panel.
3. Taner is a person who was defined as system administrator at beginning of the project. He can log in to the system with his authentication information over admin panel .
4. Then, he clicks to “Admin Panel” linked text-label on menu navigation bar.
5. When the page is loaded, there are administrative functions in here. He selects “Approve Author” button.



## Registrar

6. When the page is loaded, there is a list that contains people who waiting for approval. Each instance has some components which are first name, last name, phone, e-mail address. Also, each instance selection menu. The list contains “0” and “2” options.
7. 0 is for normal user membership. 2 is for approved membership that is called author.
8. Taner selects an instance. “Ali” for first name, “ALTINEL” for last name, “[malialtinel@gmail.com](mailto:malialtinel@gmail.com)” for e-mail address, “053834567890” for phone number. Then, he observes her e-mail address, does it belong to a company or search on internet. If so, he selects “2” and clicks to “Save” button to finish the operation.

## *Use case scenarios*

**Use-Case Name:** Add New Course

**Participating Actor Instances:** Admin

**Flow of Events:**

1. The system administrator logs in to the system over login panel.
2. The method that is in back-end, checks the administrator. The method searches the person in Admin table in database, if it exists. If so, redirects to home page, as Admin.
3. Admin has a standard navigation menu with Admin Panel button. Admin goes to admin panel.
4. System opens the Admin Panel, there is going to be functions who are managed by Admin.
5. Admin clicks to the “Add New Course”.
6. System opens the “Add New Course” page, it has own form with attributes: name, course name, lecturer, class, code of course,
7. Admin sees attributes of Add New Course page. Then, he types to some variable to the input-fields. After this entering operation, he clicks to the “Save” button to finish the operation.
8. System checks input fields. Controllers in the back-end runs with these variables. If there is no problem (i.e. null input point), the related method is called by controller. Then, the calling method runs query to add the variable to the related table in the database. The parse operation runs, and finishes. Statement is finished, and variables adds to the table in database.

## **Entry Conditions:**

- Admin must be logged in to the system.

## **Exit Conditions:**

- Admin can add new course successfully.

## **Exceptional Cases:**

- If the person who try to access to system over admin panel, is not included on Admin table in database, the system can detect any invalid input.
- When admin adds new course to system, the system can detect any invalid input.

Registrar

**Use-Case Name:** Delete Course

**Participating Actor Instances:** Admin

**Flow of Events:**

1. The system administrator logs in to the system over login panel.
2. The method that is in back-end, checks the administrator. The method searches the person in Admin table in database, if it exists. If so, redirects to home page, as Admin.
3. Admin has a standard navigation menu with Admin Panel button. Admin goes to admin panel.
4. System opens the Admin Panel, there is going to be functions who are managed by Admin.
5. Admin clicks to the “Delete Course”.
6. System opens the “Delete Course” page, it has own list with attributes: name, lecturer name, class, code of course.
7. Admin selects and clicks to the ratio button, whatever he wants to remove. Then, he clicks to the “Save” button to finish the operation.
8. The relevant controller in back-end checks input (coming from ratio button that is selected), if there is no problem, it calls related method in back-end. The calling method begins to run a query for removing selected instance from related table in the database. It executes the statement and is finished the operation.

**Entry Conditions:**

- Admin must be logged in to the system.

**Exit Conditions:**

- Admin can delete course successfully.

**Exceptional Cases:**

- If the person who try to access to system over admin panel, is not included on Admin table in database, the system can detect any invalid input.
- When admin deletes course from system, the system can detect any invalid input.

**Use-Case Name:** Assign Role

**Participating Actor Instances:** System Admin

**Flow of Events:**

1. The system administrator logs in to the system over login panel.
2. The method that is in back-end, checks the administrator. The method searches the person in Admin table in database, if it exists. If so, redirects to home page, as Admin.
3. Admin has a standard navigation menu with Admin Panel button. Admin goes to admin panel.
4. System opens the Admin Panel, there is going to be functions who are managed by Admin.
5. Admin clicks to “Assign Role” button.
6. System opens the “Create Admin” page, it has own form with attributes: e-mail and role

Registrar

7. He clicks to the “Save” button to finish the operation.
8. The relevant controller in back-end checks input, if there is no problem, it calls related method in back-end. The calling method begins to run a query for creating admin. Therefore, added information to input fields adds to the related table in database. If there is a problem, the error message is appeared.

**Entry Conditions:**

- Admin must be logged in to the system.

**Exit Conditions:**

- Admin can create admin instance successfully.

**Exceptional Cases:**

- If the person who try to access to system over admin panel, is not included on Admin table in database, the system can detect any invalid input.
- When admin adds new item to system, the system can detect any invalid input.

**Use-Case Name:** Personal Information Queries

**Participating Actor Instances:** Student

**Flow of Events:**

9. The student logs in to the system over login panel.
10. The method that is in back-end, checks the administrator. The method searches the person in student table in database, if it exists. If so, redirects to home page, as Student.
11. Student has a standard navigation menu with Student Panel button. Admin goes to Student panel.
12. System opens the Student Panel, there is going to be functions who are managed by Admin.
13. Admin clicks to the “CCR”, ”Transcript”, ”Schedule”.
14. System opens the related page; it has own list with attributes
15. The relevant controller in back-end checks input (coming from ratio button that is selected), if there is no problem, it calls related method in back-end. The calling method begins to run a query for removing selected instance from related table in the database. It executes the statement and is finished the operation.

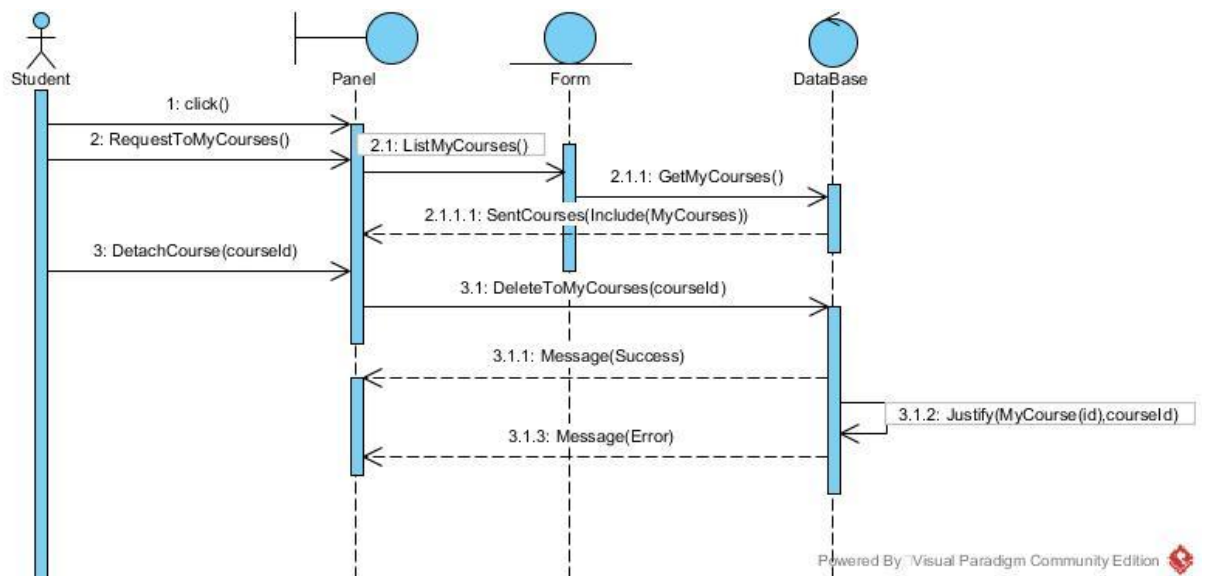
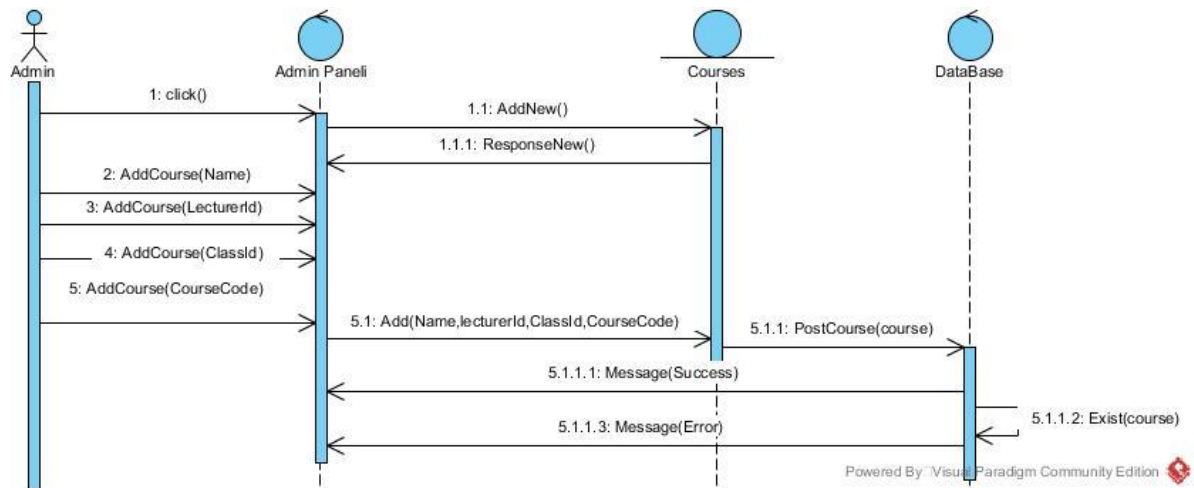
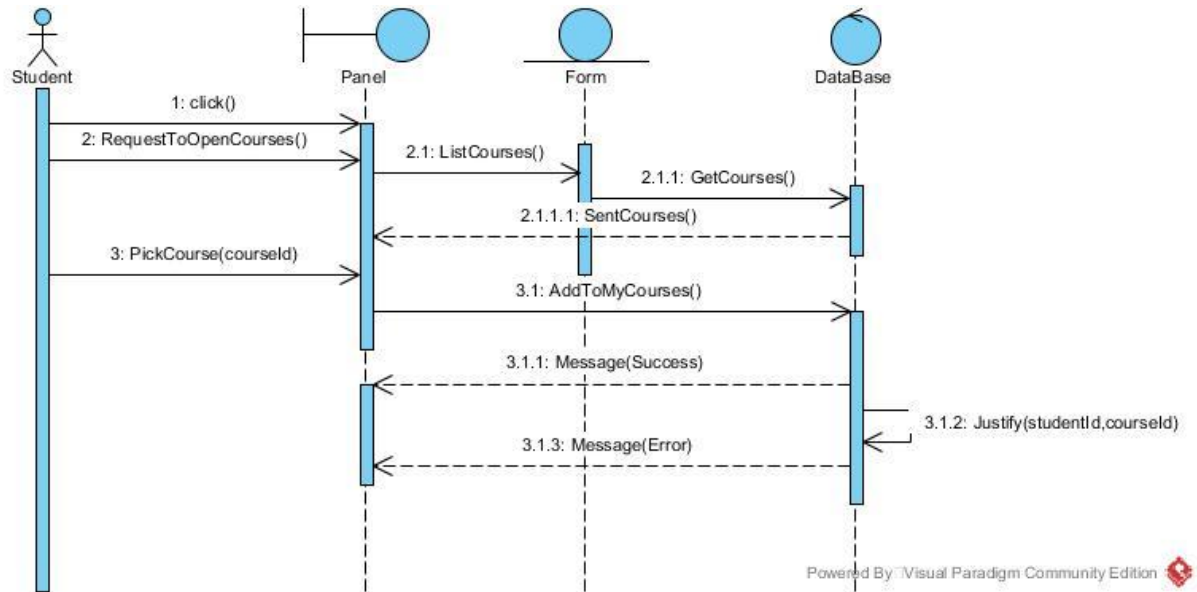
**Entry Conditions:**

- Admin must be logged in to the system.

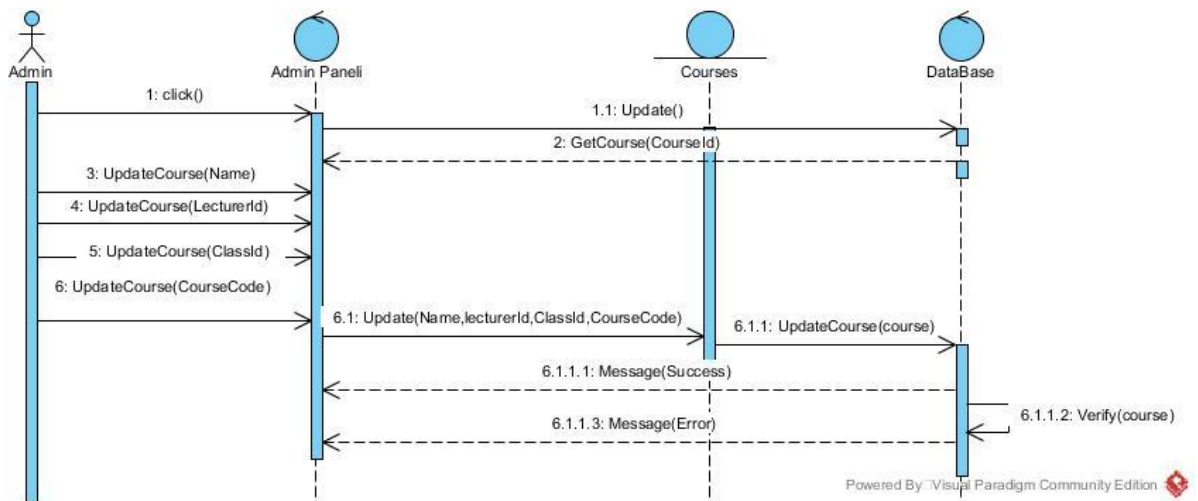
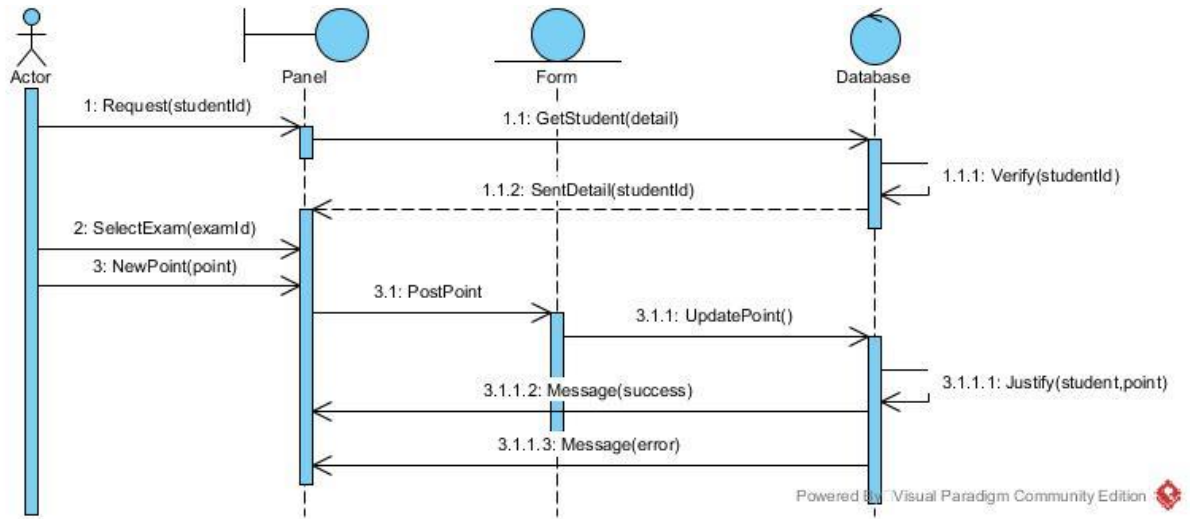
**Exit Conditions:**

- Admin can delete course successfully.

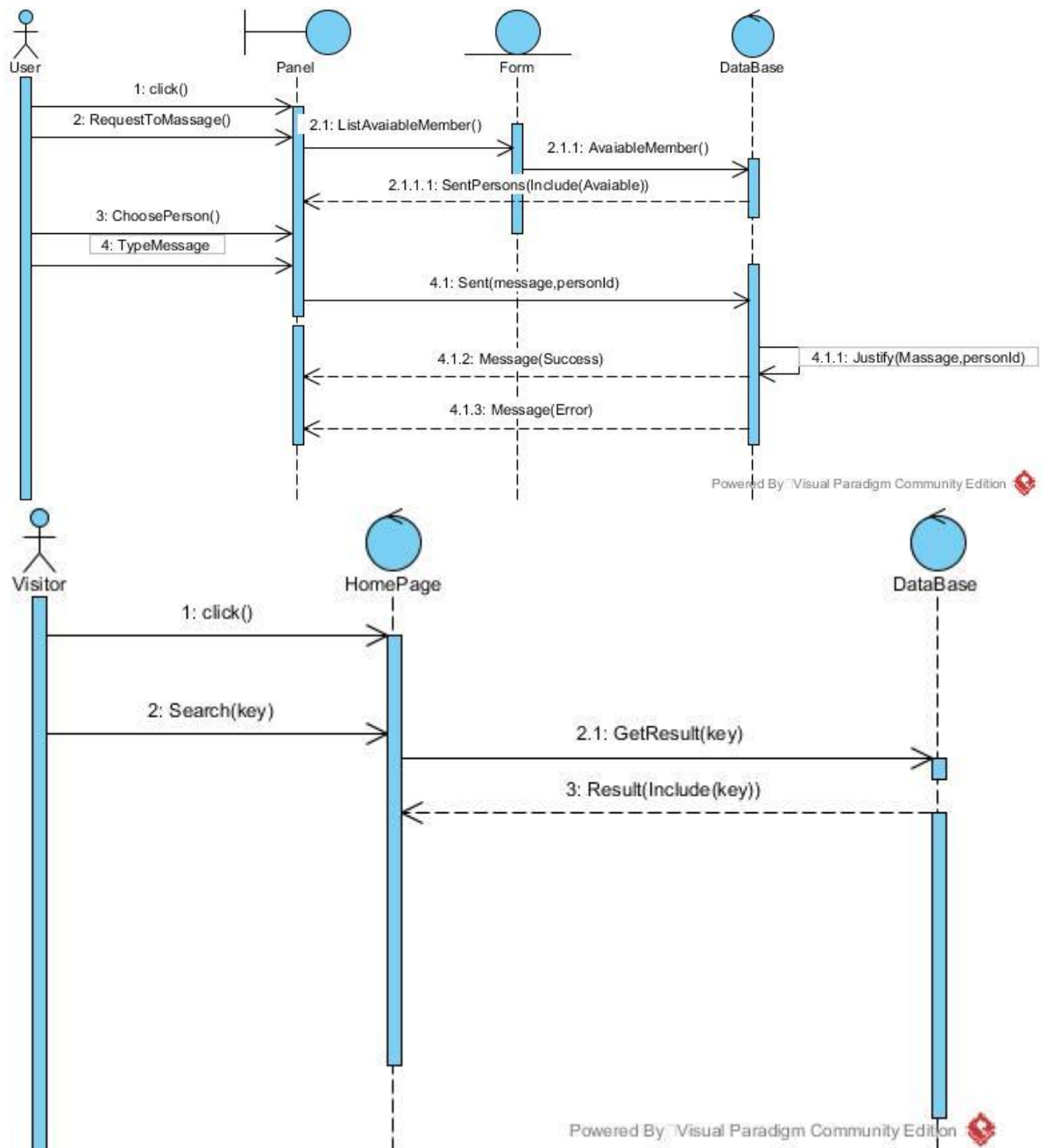
## Sequence Diagrams



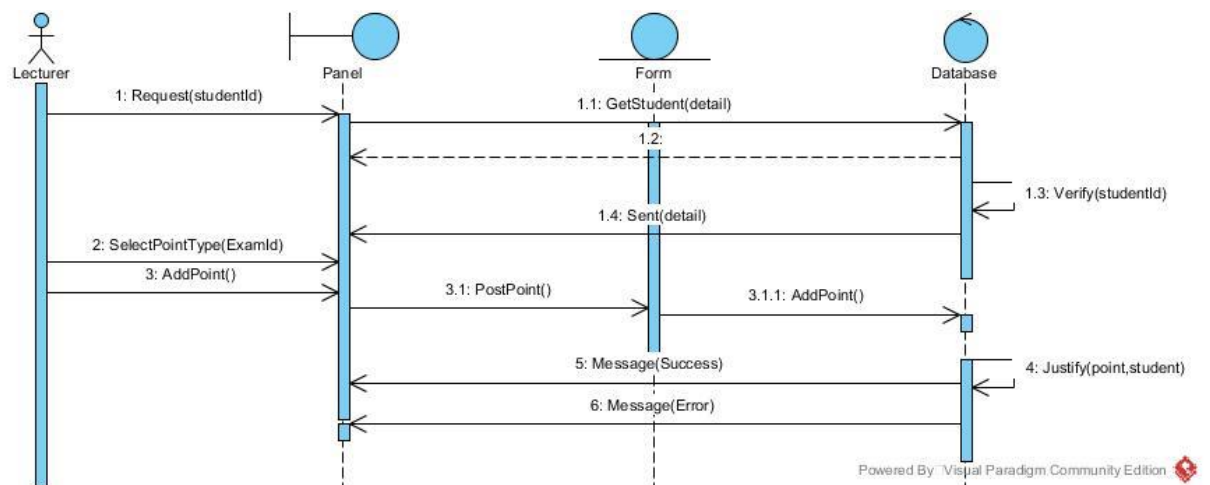
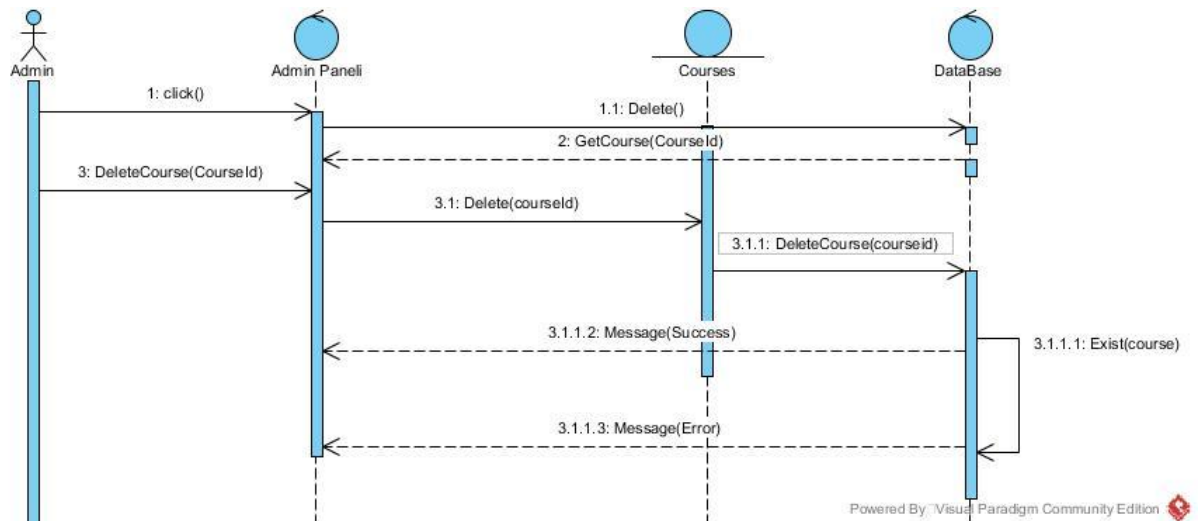
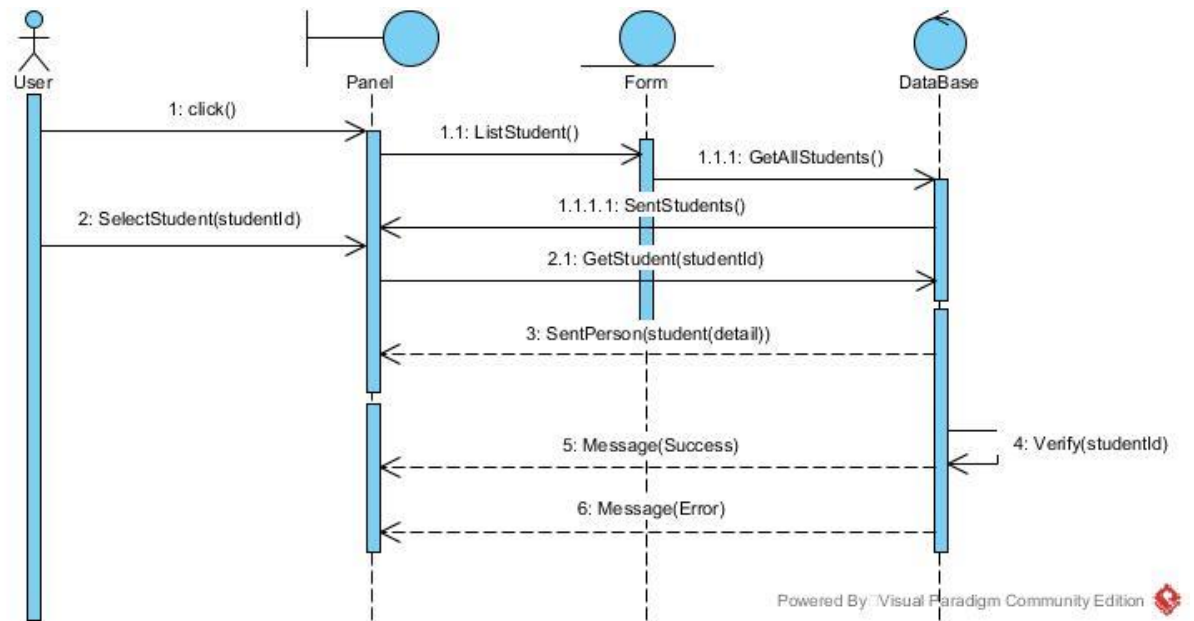
## Registrar



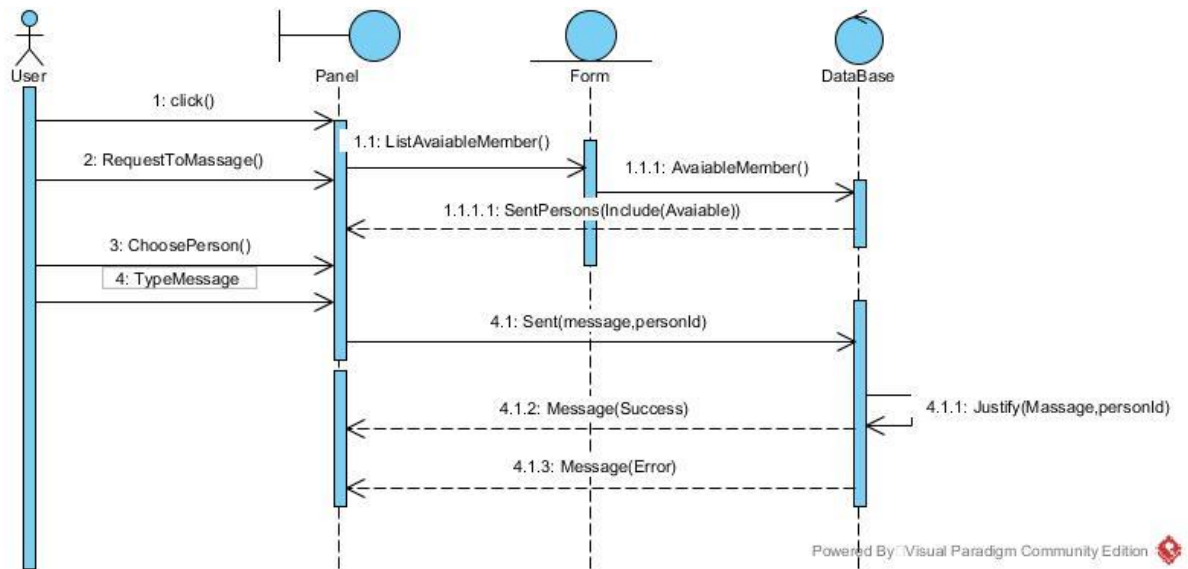
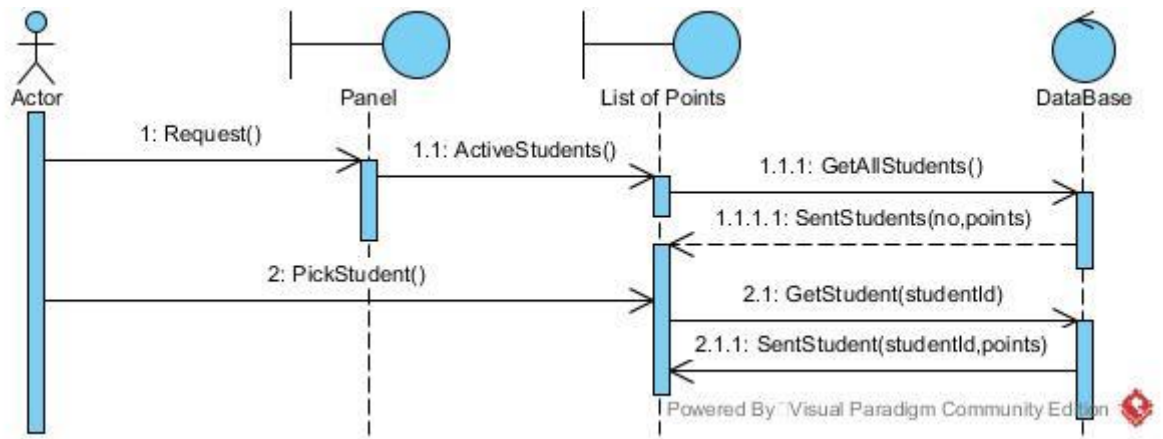
## Registrar



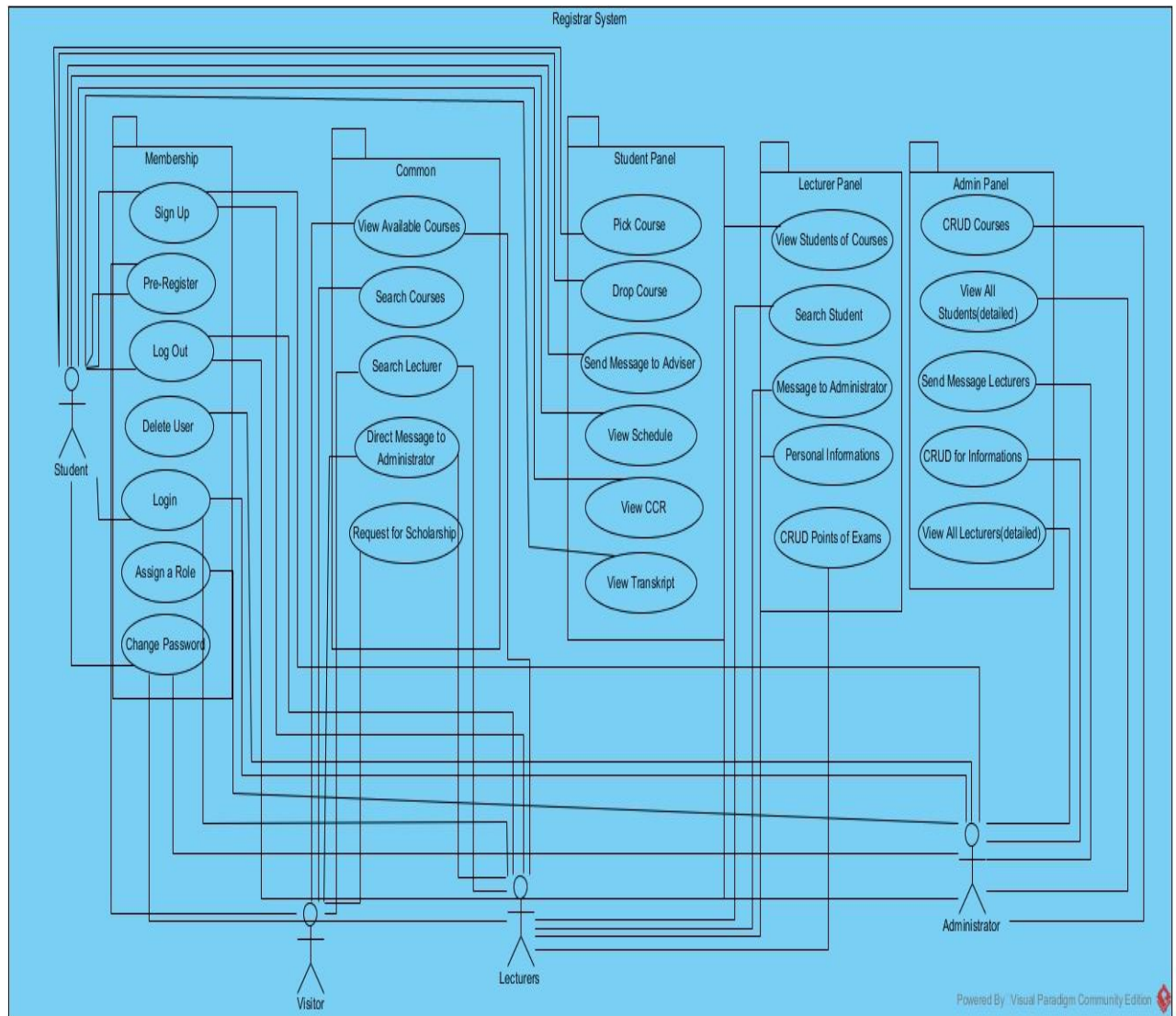
## Registrar



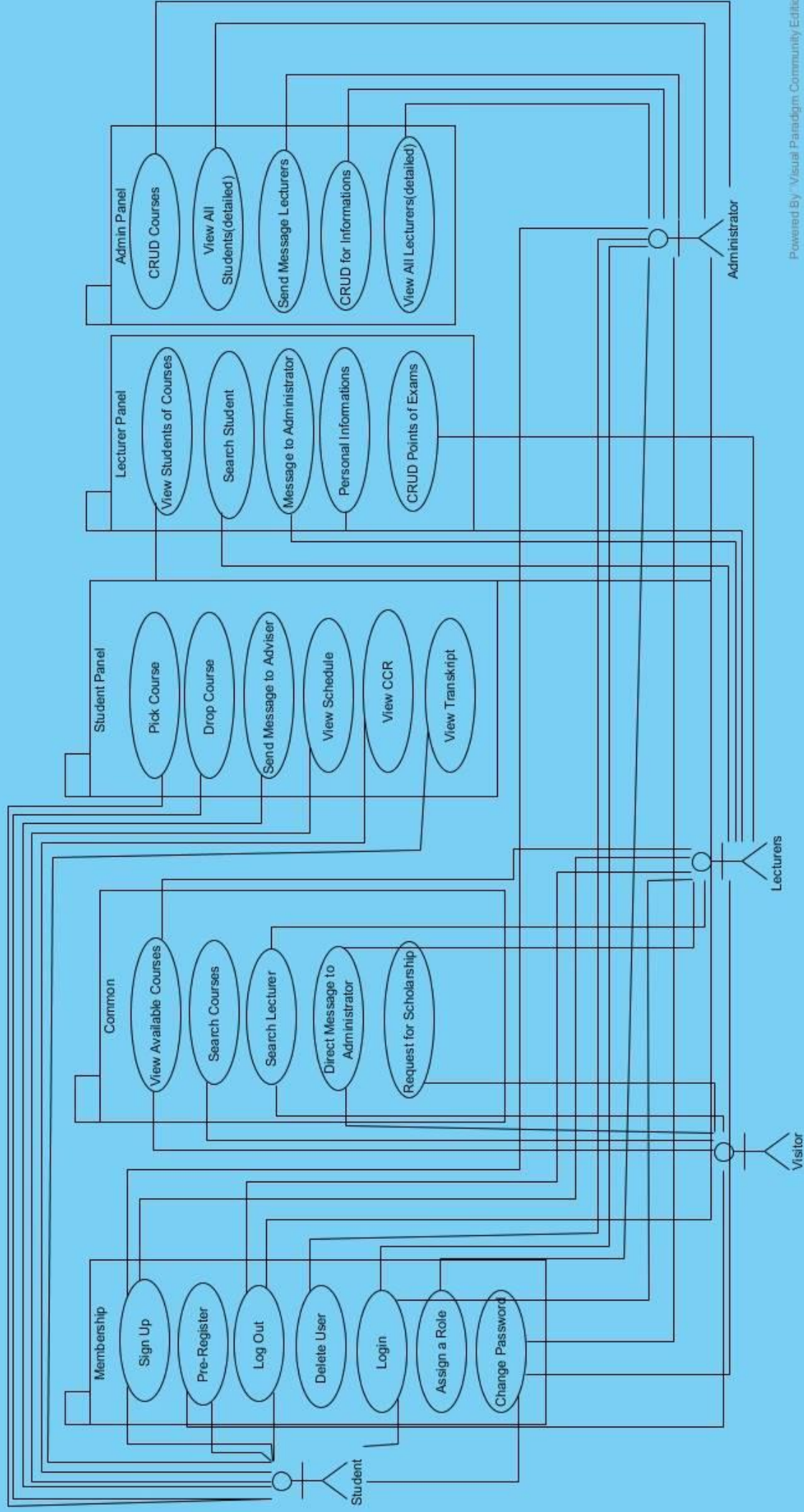
## Registrar







# Registrar System

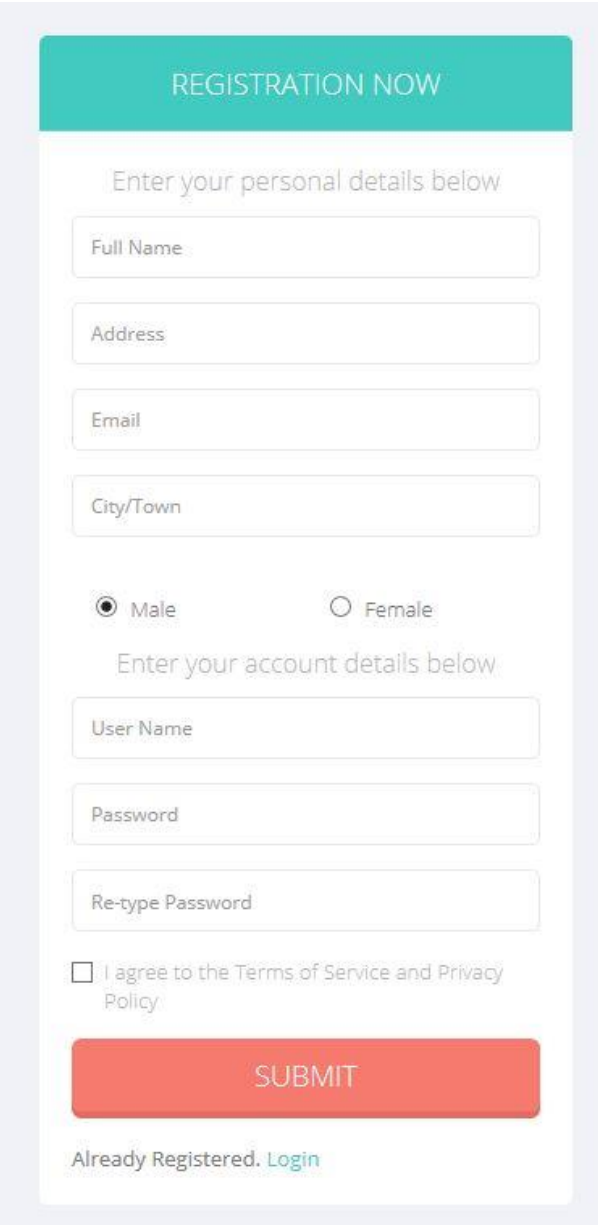


## Registrar

### *Object model*

The analysis object model, depicted with UML class diagrams, includes classes, attributes, and operations. The analysis object model is a visual dictionary of the main concepts visible to the user.

### Mock -Ups



A registration form mock-up with a teal header bar containing the text "REGISTRATION NOW". Below the header, the text "Enter your personal details below" is displayed. The form contains four text input fields: "Full Name", "Address", "Email", and "City/Town". Below these fields are two radio buttons labeled "Male" (selected) and "Female". The text "Enter your account details below" is displayed. Below this text are three text input fields: "User Name", "Password", and "Re-type Password". Below these fields is a checkbox labeled "I agree to the Terms of Service and Privacy Policy". At the bottom of the form is a red "SUBMIT" button. Below the button is the text "Already Registered. [Login](#)".

REGISTRATION NOW

Enter your personal details below

Full Name

Address

Email

City/Town

☒ Male ☐ Female

Enter your account details below

User Name

Password

Re-type Password

☐ I agree to the Terms of Service and Privacy Policy

SUBMIT

Already Registered. [Login](#)

Registrar

SIGN IN NOW

User ID

Password

☐ Remember me

Forgot Password?

SIGN IN

Admin Main Page

ALI

6

5

7

John Doe

Dashboard

Layouts

UI Elements

Components

Form Stuff

Data Tables

Mail

Charts

Shop

Google Maps

Extra

Login Page

Multi Level Menu

495

New Users

947

Sales

328

New Order

10328

Total Profit

Earning Graph

100

80

60

40

20

0

JAN

FEB

MAR

APR

MAY

JUN

JUL

AUG

SEP

OCT

NOV

DEC

Friday

\$ 57.00 | 15%

New Earning

Market | Referral | Online

June

23 Days | 65%

Total Earning

\$, 76,54,678

Anjelina Joli

Senior Architect

New Task Issued

02

Task Pending

14

Inbox

45

New Notification

09

Work Progress

Anjelina Joli

1

Target Sell

75%

2

Product Delivery

43%

3

Payment Collection

67%

4

Work Progress

30%

5












Delivery Pending

15%

## Registrar

### Search Result Page

File Search

File Name & Location	Created	Last Modify	Size	Type
 Linux Manual for dummies.doc C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 User Documentation.ppt C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Price chart Table.xls C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Linux Wallpaper.jpg C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 All Main files.zip C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Metro Lab User Manual and Help fiile.pdf C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Vector Lab Logo and Other stuff.ai C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Vectorlab wallpaper.psd C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 themeforest feed.rss C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Order and Contact.eml C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File
 Metro Lab.eps C:\Users\Murat\Documents\My Dropbox	01.01.2012	12.05.2013	193 KB	File

<

1

2

3

4

5

>

### Student&Lecturer&Class List View for Admin

Responsive table

Code	Company	Price	Change	Change %	Open	High	Low	Volume
AAC	AUSTRALIAN AGRICULTURAL COMPANY LIMITED.	\$1.38	-0.01	-0.36%	\$1.39	\$1.39	\$1.38	9,395
AAD	ARDENT LEISURE GROUP	\$1.15	+0.02	1.32%	\$1.14	\$1.15	\$1.13	56,431
AAX	AUSENCO LIMITED	\$4.00	-0.04	-0.99%	\$4.01	\$4.05	\$4.00	90,641
ABC	ADELAIDE BRIGHTON LIMITED	\$3.00	+0.06	2.04%	\$2.98	\$3.00	\$2.96	862,518
ABP	ABACUS PROPERTY GROUP	\$1.91	0.00	0.00%	\$1.92	\$1.93	\$1.90	595,701
ABY	ADITYA BIRLA MINERALS LIMITED	\$0.77	+0.02	2.00%	\$0.76	\$0.77	\$0.76	54,567
ACR	ACRUX LIMITED	\$3.71	+0.01	0.14%	\$3.70	\$3.72	\$3.68	191,373
ADU	ADAMUS RESOURCES LIMITED	\$0.72	0.00	0.00%	\$0.73	\$0.74	\$0.72	8,602,291
AGG	ANGLOGOLD ASHANTI LIMITED	\$7.81	-0.22	-2.74%	\$7.82	\$7.82	\$7.81	148
AGK	AGL ENERGY LIMITED	\$13.82	+0.02	0.14%	\$13.83	\$13.83	\$13.67	846,403
AGO	ATLAS IRON LIMITED	\$3.17	-0.02	-0.47%	\$3.11	\$3.22	\$3.10	5,416,303

Flip Scroll

Code	Company	Price	Change	Change %	Open	High	Low	Volume
AAC	AUSTRALIAN AGRICULTURAL COMPANY LIMITED.	\$1.38	-0.01	-0.36%	\$1.39	\$1.39	\$1.38	9,395
AAD	ARDENT LEISURE GROUP	\$1.15	+0.02	1.32%	\$1.14	\$1.15	\$1.13	56,431

l

The dynamic model is depicted with sequence diagrams and with state machines. Sequence diagrams represent the interactions among a set of objects during a single use case. State machines represent the behavior of a single object (or a group of very tightly coupled objects). The dynamic model serves to assign responsibilities to individual classes and, in the process, to identify new classes, associations, and attributes to be added to the analysis object model.

When working with either the analysis object model or the dynamic model, it is essential to remember that these models **represent user-level concepts, not actual software classes or components.**

Registrar

*User interface—navigational paths and screen mock-ups*

#### **4. Glossary**

**ACTOR**=End User

**System**=Registrar

#### **5. References**

1. Bruegge B. & Dutoit A.H.. (2010). *Object-Oriented Software Engineering Using UML, Patterns, and Java*, Prentice Hall, 3<sup>rd</sup> ed.
2. Campus Online System for Isik University