Learning Management System

I-Learn

I-Learn System is Supervised By

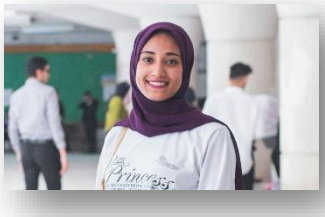
Dr. Laila Abdel-Hamid

Team Members

Miirna Mourad

Mohamed Shawky

Mayar Mostafa



Rahma Ashraf

Aliaa Yasser

Yara Riffat

Submitted as Partial Fulfillment of the Requirements of the Bachelor's Degree in Software Engineering.

Abstract

Introduction

This program solves a daily problem in all societies; it's the issue of finding the good system to use to contact with students, instructors or admins in any Learning community, also without being worried to lost any of the the important documents, mainly it's purpose is to reduce the gap between all learning participatns and facilitate the management.

Problem Statement

With the advent in technology and with the perpetual increase in the strength of the students and the number of departments in the educational institutions, it is laborious to exchange the study materials between the students and the faculties. So, the main objective of the E-Learning is to help the students get over the traditional methods of learning and make them accustomed to the internet where the notes for their respective subjects are easily available. It provides an automation procedure of studying the notes online.

Scope

Initiation "In this phase we're defining the problems and the goals that we wish to reach"

Planning "Here we're planning the entire project, main functions and non-functions, decide the tools used"

Execution "Where the actual work start"

Termination "It is the testing and delivery for the project"

Summary of the Project Outcomes and Findings

Findings

We've found that we're all pass by this issue one day, so we decided to choose that idea to works on.

Also After analysis some common application, we've found their weakness and problems that we will work to cover it in ours.

Outcomes

I-Learn system will outcome to the user the best LMS system with the reasonable price and all the important functions needed for any college/school system.

Acknowledgements:

First and Foremost, we would express our deepest gratitude and appreciation to our advisor **Dr. Laila Abdel-Hamid** for her continuous support, outstanding guidance and encouragement throughout our graduation project as she was always there for us providing more than the needed time and effort since the very beginning of our project.

Finally, we would also thank our faculty for allowing us to improve our soft skills, technical and how to think well, And for providing the suitable environment that leaded us to represent the best image that computer science graduates of Helwan University are supposed to represent.

Table of Contents

List of Figures

List of Tables

Chapter One

In this chapter, we are going to discuss and go deeper in the overview of the project and know more about its scope and limitations, and explain some terminologies we will find throughout the document.

Chapter Headlines:

1. Overview
2. Motivation
3. Purpose
4. Problem Statement
5. Objectives
6. Scope
7. General Constraints

1.1 *Overview*

- E-learning can be as a network enabled transfer of skills and knowledge, and the delivery of education is made to a large number of recipients at the same or different times.
- The E-Learning facility is an online system that allows students and staff to communicate effectively.
- It also allows them to submit assignments online and receive their grades for tasks and assignments as well.
- It provides all the necessary documents for efficient study and it supports independent learning.
- System will facilitate announce for any important thing by ensuring that every student will be informed with it.

1.2 Motivation

Today's learners want relevant, mobile, self-paced, and personalized content. This need is fulfilled with the online mode of learning it is an inexpensive, efficient and comfortable way for students to easily access notes and an easier alternative to study for exams.

1.3 Purpose

Provide education to many beneficiaries at the same time or at different times and provide an inexpensive, efficient and convenient way for students and staff to communicate.

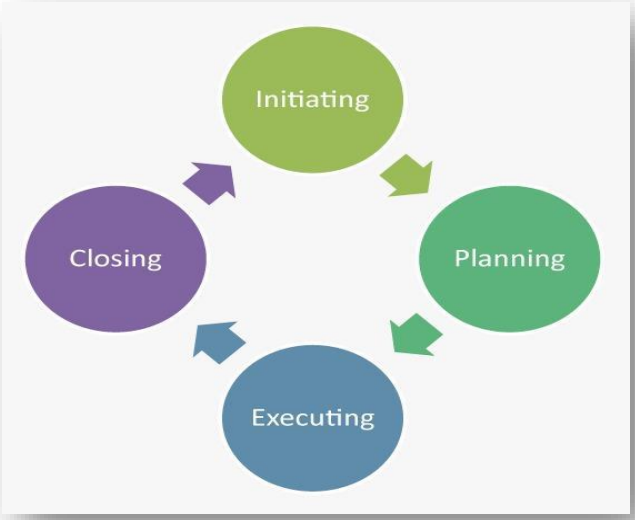
1.4 Problem Statement

With the advent in technology and with the perpetual increase in the strength of the students and the number of departments in the educational institutions, it is laborious to exchange the study materials between the students and the faculties. So, the main objective of the E-Learning is to help the students get over the traditional methods of learning and make them accustomed to the internet where the notes for their respective subjects are easily available. It provides an automation procedure of studying the notes online.

1.5 Objectives

- Help define your E-Learning course
- Help keep your planning E-Learning strategies on track
- Be in line with educational philosophy
- Help define your E-Learning assessment
- Tell the learner exactly what to expect
- Enhance the quality of learning and teaching
- Meet the learning style or needs of students
- Improve the efficiency and effectiveness
- Improve user-accessibility and time flexibility to engage learners in the learning process

1.6 Scope



Initiation

Develop a Business Case

Undertake a Feasibility Study

Establish the Project Charter

Appoint the Project Team

Planning

Create a Project Plan

Create a Resource Plan

Create a Quality Plan

Create a Risk Plan

Execution

Monitor and Control

Perform Time Management

Perform Quality Management

Perform Risk Management

Closure

Perform Project Closure

Review Project Completion

1.7 General Constraints

- Scope: project outcome as defined in the contract.
- Budget: funding limits imposed by project team/sponsors.
- Quality: agreed quality metrics with internal quality standards.
- Resources: availability of skilled human resources.
- Risks: uncertainties associated with the project.
- Time: deadline for delivering the output.
- Learning new technologies may take much time.
- Time management.
- Indiscipline “Human factor” like being late in delivering tasks or attending meetings.

Chapter TWO

In this chapter, we are going to discuss and go deeper in how we plan the project and show the steps and the instructions that we have followed to plan the website.

Chapter Headlines:

1. Project planning
2. Analysis and Limitation of existing system
3. Need for the new system
4. Analysis of the new system
5. Advantages of the new system
6. Risk and Risk Managements

2.1 Project Planning

2.1.1 Feasibility Study

1. Financial Feasibility:

Being a Web-based application will have an associated hosting cost. As the system consists of multimedia data transfer, bandwidth required for the operation of this application is almost medium. The system will follow the freeware software standards. No cost will be charged from the potential students. Bug fixes and maintaining tasks will have an associated cost. At the initial stage the potential market space will be all Colleges and universities. Beside the associated cost, there will be many benefits for the students. From these it's clear that the project is financially feasible.

2. Technical Feasibility Project

Our E-learning system is a Web-based application. The main technologies and tools that are associated with are: -

- NodeJS & Express
- MongoDB ATLAS
- Visual Studio Code
- Diagram drawing tools: -
  - Visual Paradigm
  - Online Tool for diagram
  - Microsoft Project

Each of the technologies is freely available and the technical skills required are manageable. Time limitations of the product development and the ease of implementing using these technologies are synchronized.

Initially the application will be hosted in a free hosting space, but for later implementations it will be hosted in a paid hosting space with a sufficient bandwidth. Bandwidth required in this application is almost medium, as it incorporates multimedia aspect. From these it's clear that the project is technically feasible.

3. Operational Feasibility:

Our system tries to help staff of collage from students till doctors and mangers.

Users don't need training for using the system.

4. Resource Feasibility:

Resources that is required for building project includes,

- Programming device (Laptop).
- Hosting space (freely available).
- Programming tools (freely available).
- Programming individuals.

So, it's clear that the project has the required resource feasibility.

5. Risk Feasibility

Risk feasibility can be discussed under several contexts.

I. **Risk associated with size Estimated size of the product in line of codes:**

Being a web-based system with many numbers of stakeholders, project will contain significant amount of code lines. As the system contains multimedia aspect, the file sizes and the complete project size will not exceed 200MB.

II. **Estimated size of product in number of programs:**

Though the system supports many stakeholders, it will be constructed as a single website with a single login page and so on.

III. **Size of database created or used by the product:**

Database size will not exceed the values supported by MongoDB. Number of relations and entities are minimized by using best practices of normalization theories.

IV. **Users of the product:**

a. Students
b. Doctors and teach assistants.
c. System admin.
d. Super admin.

V. **Changes to the requirements for the product:**

The requirements are clearly identified before the implementation phase. Being a general product, the requirements will be changed only if new functionalities are added to the system.

VI. Sophistication of end users:

Project is designed while maintaining the complexity at a very low level. Usability is highly improved by providing help documents and making interface easy to use.

VII. Development environment risks:

Is a software project management tool available?

Microsoft Project will be used as the main project management tool.

Are tools for analysis and design available?

Project will require several designing software

- Draw.IO (database design)
- NCLASS (class diagram)
- Visio (Software related diagrams)

Are compilers or code generators available and appropriate for the product to be built?

Visual Studio Code will be used as the main development tool. All the Packages will be freely available.

Are software configuration management tools available?

Configuration management will be done using GIT that is freely available.

Does the environment make use of a database or repository?

No SQL/SQL database system that will use MongoDB.

VIII. Social/Legal Feasibility:

Project freely available development tools and provide the system as an open source system.

NodeJS Software Packages that are used in this system are free open sources.

2.1.2 Estimated Cost

A cost estimate is the approximate of a program, project or operation, and our estimated cost for this project comes as following:

1. **Meetings effort and time we spent to get the best services and functionality for our site:**

   Time: every week.
   Place1: Co-working Space (from August to February).
   Place2: Home using Zoom online meeting (from March to July)
   Duration: 11 months (August To July)
   Paying: for Place1 → 100 L.E every meeting * 28 week = 2800 L.E
   for Place2 → 20 L.E every zoom meeting for the internet * 20 weeks = 600 L.E each.
   So, total paying is 3400 L.E for each member.

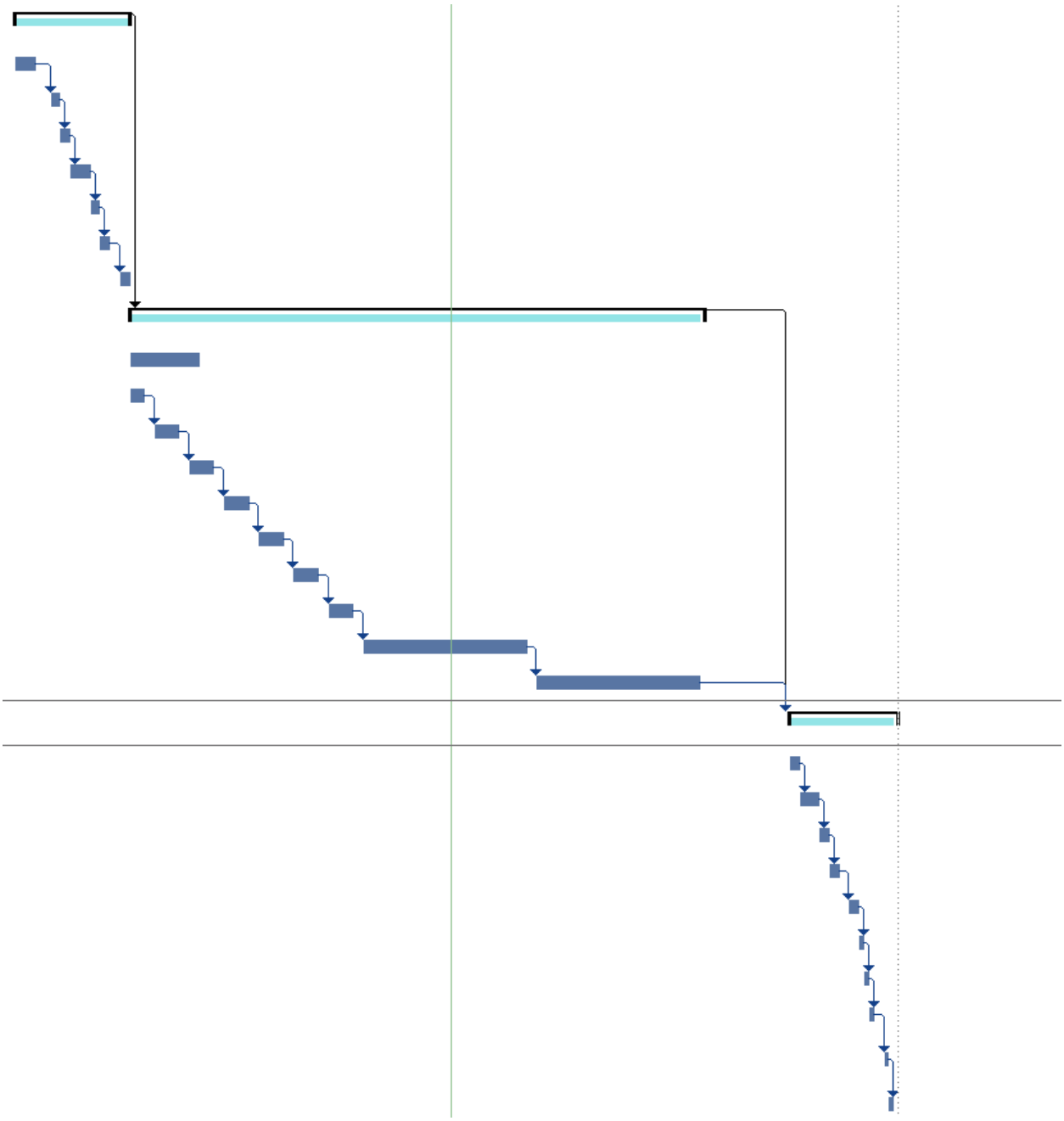
2. **Purchasing programs and resources we use like:**
   - Visual paradigm
   - Visual studio code
   - Adobe illustrator
   - Adobe After effect
3. **Stationery and printings:**
   - Stationary for the staff.
   - Printed Document for each doctor at presentation of Graduation Project.
4. **Salaries paid to employees:**

   Like (Admin, Tester, developer, database developer, system analysts, etc..).
5. **Software Courses:**

   Take some courses to improve our skills like NodeJS and MongoDB.

2.1.3 Gantt chart

	Task Name	Duration	Start	End
1	Preparation Meeting	5 Days	5/8/2019	10/8/2019
2	Brain Storming	4 Days	5/8/2019	9/8/2019
3	Choosing Appropriate Idea	1 Day	9/8/2019	10/8/2019
4	Analysis Meeting	2 Days	10/8/2019	12/8/2019
5	Gathering Information	2 Days	10/8/2019	12/8/2019
6	Documentation Chapter One	10 Days	15/8/2019	25/8/2019
7	Overview	2 Days	15/8/2019	17/8/2019
8	Objectives	2 Days	17/8/2019	19/8/2019
9	Purpose	2 Days	19/8/2019	21/8/2019
10	Scope	2 Days	21/8/2019	23/8/2019
11	Constrains	2 Days	23/8/2019	25/8/2019
12	Documentation Chapter Two	25 Days	1/9/2019	26/9/2019
13	Project Plannig	10 Days	1/9/2019	11/9/2019
14	Analysis and Limitation of existing system	1 Day	11/9/2019	12/9/2019
15	Need for the new system	3 Days	12/9/2019	15/9/2019
16	Analysis of the new system	5 Days	15/9/2019	20/9/2019
17	Advantages of the new system	1 Day	20/9/2019	21/9/2019
18	Risk and Risk Managements	5 Days	21/9/2019	26/9/2019
19	Documentation Chapter Three	20 Days	1/10/2019	21/10/2019
20	Class Diagram	5 Days	1/10/2019	6/10/2019
21	Use Case Diagram	5 Days	6/10/2019	11/10/2019
22	Sequence Diagram	5 Days	11/10/2019	16/10/2019
23	Activity Diagram	5 Days	16/10/2019	21/10/2019
24	Study Tutorials	50 Days	25/10/2019	15/12/2019
25	Node JS Tutorial	25 Days	25/10/2019	20/11/2019
26	Angular 8 Tutorial	25 Days	20/11/2019	15/12/2019
27	Implementation	60 Days	20/1/2020	20/3/2020
28	Back End	45 Days	20/1/2020	5/3/2020
29	Front End	15 Days	5/3/2020	20/3/2020
30	Documentation Chapter Four	4 Days	25/3/2020	29/3/2020
31	Softw are architecture	2 Days	25/3/2020	27/3/2020
32	Pseudocode, Flowchart	2 Days	27/3/2020	29/3/2020
33	Documentation Chapter Five	6 Days	30/3/2020	6/4/2020
34	Unit Testing	2 Days	30/3/2020	2/4/2020
35	Integrated Testing	2 Days	2/4/200	4/4/2020
36	Additional Testing	2 Days	4/4/2020	6/4/2020
37	Documentation Chapter Six	2 Days	6/4/2020	8/4/2020
38	Result	1 Day	6/4/2020	7/4/2020
39	Discussion	1 Day	7/4/2020	8/4/2020
38	Documentation Chapter Seven	1 Day	8/4/2020	9/4/2020
39	Conclusion	1 Day	8/4/2020	9/4/2020
40	Documentation Chapter Eight	1 Day	9/4/2020	10/4/2020
41	Future Work	1 Day	9/4/2020	10/4/2020
42	Final Meeting	1 Day	11/4/2020	12/4/2020
43	Preparing Presentation	1 Day	11/4/2020	12/4/2020



2.2 Analysis and Limitation of existing system

Limitation of existing system:

- The high permissions to the super Admin so he can handle the complete data.
- The supervisor enters all students and users.
- You cannot remove the super admin.

2.3 Need for new system

We done some analysis for some websites, and Here's the reason that lead us to our project, and will try to solve it.

FCIH system weakness:

- Very old design that didn't match nowadays.
- It's only for courses register and final results.
- Students/instructors and moderators must use an extra website or Facebook to communicate to each other.
- Can't share courses syllabus on it.
- The more students that access the platform, the slower the system becomes.

Moodle weakness:

- Not well designed.
- Very complex to be used.
- The system might not work efficiently with larger colleges or serve as a great way to conduct all classes in a city.
- the more students that access the platform, the slower the system becomes
- There are parts of Moodle that can feel like they were designed by programmers.

So, here's the need of our system:

- Users need a modern & easy system to use.
- Users need an efficient and effective system that allows large number of users' access.
- Students need to get their course syllabus form their collage system.
- Students need to get all the important announcement through fixed system.
- Students need to communicate with their instructors & colleges.
- Students need to share their Feedbacks through an official system.
- Students need to get all their grades, in less time to try improving themselves.
- Instructors need an official place to share with students all syllabuses, assignments and quizzes.
- Instructors need an easy way to assign students grades.
- Instructors need to communicate with their students & TAs.
- Moderators need a fixed and official way to announce for all important news/schedules to students/instructors.
- Moderators need a fixed way to get all Feedbacks for better performance.
- Students/instructors need an official way to communicate for graduation projects.

2.4 Analysis of the new system

2.4.1 User Requirements:

Our website will have more than one user types, but it mainly focuses on the students as it provides most of their needs in college, they will be able to:

- Register to a course / View resources
- Receive announcements and notifications.
- Course post (start discussion with students and instructor)
- View course plan and track the progress.
- Communicate with supervisors for Graduation projects.

Knowing that that instructors are responsible for adding such data & mange GP.

2.4.2 System Requirements

The website should run on any browser at any platform

The system also should…

- Control admins and members
- Collect data

2.4.3 Domain Requirements

- Multiple users must be able to use the website simultaneously without corrupting the database (whatever form it may be).
- The necessary software required to run the application.
- The database should be backed up every once in a while in case the original becomes corrupt.
- The application must have update capabilities for future models and accessories.
- System must have authorization module to prevent unauthorized access

2.4.4 Functional Requirements

1) **Admin/ Super Admin:**
    - Login
    - Add/update/delete Admin (Only if super admin)
    - Add New Instructor / Student
    - Update Instructors / Students
    - Delete Instructors / Students
    - Add New Course
    - Update Courses
    - Delete Courses
    - Assign Course to Instructor
    - Accept instructors requests
    - View Courses Rating
    - Add Graduation project team
    - Manage system settings
    - Add/view announcements
    - Upgrade students level
    - Add/update/delete departments
    - Enroll students (Bulk enroll)
    - Enroll individual student
    - Accept their enrollment requests
    - Bulk add (Courses, Users)
    - Logout
2) **Instructor**:
    - Login
    - Send Request to Admin to be assigned to a course
    - Add card in Storyboard
    - Delete card in Storyboard
    - Update card status in Storyboard
    - Create Project Form

- Update Project Form Data
- View Project Form Data
- Add Material of Course
- Delete Material from Course
- Add Post at course
- Add Comment in Post
- Approve the Graduation Project team request to be supervisor
- Reject the Graduation Project team request to be supervisor
- Start chatting with Graduation Project team
- Add/view Announcements
- List students in course
- Send email to student
- View and Grade students answer
- Logout

3) **Student:**

- Login
- Register to a course
- Register to a Project Form
- View Material of Course
- Download Material of Course
- Add Post at course
- Add Comment in Post
- Send request to doctor to be supervisor
- Chatting with GP supervisor
- Drop course
- Get notifications
- View Announcements
- Upload Tutorial
- View Tutorial Grade
- Rate Course

1) Login Process

Use case name	Login
Actors	User
Description	This process is responsible for login to the system as a certain type of its users like: Super admin, Admin, Instructor and student and use its functionalities.
Goal	To make sure that user see the specific dashboard according to his user role.
Pre-condition	User is at home page and writing his valid e-mail and password.
Post condition	User is logged in as a specific user and having his all factions that came with his logged in role.

2) Create admin process

Use case name	Create admin
Actors	Super admin
Description	This process is responsible for creating admins users.
Goal	To have more than admin can access if one has trouble.
Pre-condition	Super admin is logged in
Post condition	A new admin user is added to list of admins.

3) Update admin process

Use case name	Update admin
Actors	Super admin
Description	This process is responsible for updating information of admins users.
Goal	To get in touch with any changes permanently.
Pre-condition	Super admin is logged in, the admin user is determined and valid.
Post condition	Admin information is updated.

4) Delete admin process

Use case name	Delete admin
Actors	Super admin
Description	This process is responsible for deleting admins users.
Goal	To get rash of any admin that have trouble and can't solve, or admin is not in the community any more.
Pre-condition	Super admin is logged in, the admin user is determined and valid.
Post condition	An admin user is deleted from list of admins.

5) Create instructors\students process

Use case name	Create instructors\students
Actors	Super admin and admin
Description	This process is responsible for creating instructors and students users.
Goal	To add any new instructor to complete the staff permanently.
Pre-condition	Super admin or admin are logged in
Post condition	A new (instructor\student) user is added to list of admins.

6) Update instructor\student process

Use case name	Update instructor\student
Actors	Super admin and admin
Description	This process is responsible for updating information of instructor\student users.
Goal	To get in touch with any changes permanently.
Pre-condition	Super admin or admin are logged in, the instructor\student user is determined and valid.
Post condition	The instructor\student information is updated.

7) Delete instructor\student process

Use case name	Delete instructor\student
Actors	Super admin and admins
Description	This process is responsible for deleting instructor\student users.
Goal	To get rash of any instructor\student that have trouble and can't solve, or instructor\student is not in the community any more.
Pre-condition	Super admin or admin are logged in, the instructor\student user is determined and valid.
Post condition	An instructor or a student user is deleted from system's database.

8) Create course process

Use case name	Create course
Actors	Super admin and admins
Description	This process is responsible for creating new courses.
Goal	To add any new course to category permanently.
Pre-condition	Super admin or admin are logged in.
Post condition	A new course is added to list of courses.

9) Update course process

Use case name	Update course
Actors	Super admin and admins
Description	This process is responsible for updating information of courses on the system.
Goal	To get in touch with any changes permanently.
Pre-condition	Super admin or admin are logged in, and the course is available at the system.
Post condition	Course data is updated.

10) Delete course process

Use case name	Delete course
Actors	Super admin and admins
Description	This process is responsible for deleting courses from the system.
Goal	To get rash of any course that have trouble and can't solve, or course is not in the category any more.
Pre-condition	Super admin or admin are logged in, and the course is available at the system.
Post condition	A course is deleted from list of courses.

11) Assign course process

Use case name	Assign course
Actors	Super admin, admins and instructors.
Description	This process is responsible for assigning a specific course to an instructor.
Goal	To complete the course information's and know who will start the course.
Pre-condition	Super admin or admin are logged in, the course is available at the system and the instructor who assigned to the course is available at the system.
Post condition	The course assigned to the instructor.

12) Course availability process

Use case name	Course availability
Actors	Student, admin and super admin.
Description	This process is responsible for make course available to students and their levels to be able to register it.
Goal	To make the student register in the course they should study.
Pre-condition	Admin and super admin make course available at the system and the instructor is available at the system.
Post condition	Student is registered at the course.

13) View courses rating process

Use case name	View courses rating
Actors	Super admin and admins
Description	This process is responsible for viewing information of students' rates of the courses on the system.
Goal	To know the final rating of each course to improve them permanently.
Pre-condition	Super admin or admin are logged in, the course is available at the system and the students put their rates of the course.
Post condition	Course rates are viewed.

14) Course request process

Use case name	Course request
Actors	Instructor
Description	This process is responsible for requesting to be assigned to a specific course on the system.
Goal	To make instructors request for teaching that course then choose who will teach that course to the students.
Pre-condition	Instructor is logged in, and the course is available at the system.
Post condition	Request is sent to admins.

15) Notifications process

Use case name	Update course
Actors	Super admin, admins, instructors and students.
Description	This process is responsible for send notifications of any news on the system.
Goal	To make all user in touch.
Pre-condition	Super admin admins, instructors and students are logged in.
Post condition	Notifications is sent to them.

16) Adding a story board card

Use case name	Adding story board\calendar
Actors	Instructor
Description	This process is responsible for adding a story board cards to the course storyboard for students to know the process of that course.
Goal	To make plan for the course and know every process will done.
Pre-condition	Instructor is logged in, and the course is available at the system.
Post condition	A course is related to the story board.

17) Create project process

Use case name	Create project
Actors	Instructor
Description	This process is responsible for creating project and attaches it to its course.
Goal	To add project schedule to the required course.
Pre-condition	Instructor is logged in, course is available at the system and the instructor is assigned to it.
Post condition	A new project is added to the course requirement.

18) Update project process

Use case name	Update project
Actors	Instructor
Description	This process is responsible for updating information of projects to specific course.
Goal	To get in touch with any changes permanently.
Pre-condition	Instructors is logged in, the project is in the course that assigned to this instructor.
Post condition	Project data is updated.

19) Delete project process

Use case name	Delete project
Actors	Instructor
Description	This process is responsible for deleting projects of any course at the system.
Goal	To get rash of any project that have trouble and can't solve, or project is not in the course any more.
Pre-condition	Instructors is logged in, the project is in the course that assigned to this instructor.
Post condition	A project is deleted from a course that the instructor assigned to.

20) View project's form process

Use case name	View project's form
Actors	Instructor
Description	This process is responsible for viewing information of students' responses of the form of the project of a course on the system.
Goal	To know the teams and organize the dissection time.
Pre-condition	Instructor is logged in, the course is available at the system, has its own project, the instructor is assigned to this course and the instructor put the project's form at the course.
Post condition	Project's form is viewed.

21) Create resources process

Use case name	Create resource
Actors	Instructor
Description	This process is responsible for adding resources to the course.
Goal	To add resources schedule to the course.
Pre-condition	Instructor is logged in, course is available at the system and the instructor is assigned to it.
Post condition	A resource is added to the course content

22) Delete resources process

Use case name	Delete resource
Actors	Instructor
Description	This process is responsible for deleting resources of any course at the system.
Goal	To get rash of any resources that have trouble and can't solve, or resources is not in the course anymore.
Pre-condition	Instructors is logged in, the resource was added to the course that assigned to this instructor.
Post condition	A resource is deleted from a course that the instructor assigned to.
Post condition	A quiz is added to the course students

23) Post\comment process

Use case name	Post a post\comment
Actors	Instructor and student
Description	This process is responsible for posting a post or a comment about a specific course.
Goal	Increase the communication between the instructor, students and other students.
Pre-condition	Instructor or student are logged in, and the course is available at the system.
Post condition	Course is updated with a new post\comment.

24) Course register process

Use case name	Course register
Actors	Student
Description	This process is responsible for registering to be enrolled at a specific course on the system.
Goal	Organize the student for each course.
Pre-condition	Student is logged in, and the course is available at the system.
Post condition	Request is sent to admins.

25) View resources process

Use case name	View grades/resources rating
Actors	Student
Description	This process is responsible for viewing information of students' course's resources on the system.
Goal	Get for the students the information they need.
Pre-condition	Student is logged in, the course is available at the system and the student enrolled in this course.
Post condition	Course resources are viewed.

26) View/Request doctors for GP

Use case name	View/Request doctors for GP
Actors	Student
Description	This process is responsible for see and request doctors to communicate for graduation project.
Goal	To start graduation project with doctors, know which doctor is still get requests and who not.
Pre-condition	Student is logged in, the project is available at the system and the student enrolled in level 4.
Post condition	Go to the page which would be able to request doctors on it.

27) Approve team GP

Use case name	Approve team GP
Actors	Instructor
Description	This process is responsible for see and approve them if instructor agree.
Goal	To start graduation project with doctors and let student know that.
Pre-condition	Instructor is logged in, the team is available at the system and the student enrolled in level 4.
Post condition	Go to the page which would be able to communicate with team leaders on it.

28) Reject team GP

Use case name	Reject team GP
Actors	Instructor
Description	This process is responsible for see and reject them if instructor agree.
Goal	To let student know that they is rejected.
Pre-condition	Instructor is logged in, the team is available at the system and the student enrolled in level 4.
Post condition	Go to the page which would has the other requests from team leaders to reject.

29) Chatting with team GP

Use case name	Chatting with team GP
Actors	Instructor
Description	This process is responsible for see team and chatting if instructor want to know more information.
Goal	To let student know that is rejected.
Pre-condition	Instructor is logged in, the team is available at the system and the student enrolled in level 4.
Post condition	Go to the page which would be able to communicate with doctors on it.

30) Enroll students (Bulk enroll)

Use case name	Enroll students
Actors	Admin/Super Admin
Description	This process is responsible to enroll a group of students inside a course.
Goal	To facilitate students enrollment to course whatever their number.
Pre-condition	Admin is logged in, there's a courses available.
Post condition	Students will be registered to that course.

31) Add Announcement

Use case name	Add announcement
Actors	Admin/Super Admin/Instructor
Description	This process is responsible to send the important announcements to students and all system users.
Goal	To facilitate sending the important announcements to system users.
Pre-condition	Admin/super admin/instructor is logged in. (and if instructor must be assigned to courses)
Post condition	Announcements sent to the chosen user easily.

32) View Announcement

Use case name	View announcements
Actors	All system users
Description	This process is responsible to allow all users see the important announcements sent to their related topics.
Goal	Facilitate telling a large group of people an important announcement
Pre-condition	User is logged in.
Post condition	List of all announcements sent to that user.

33) System Settings

Use case name	System settings
Actors	Admin/Super admin
Description	This process is responsible to allow admins manage what should happen and when.
Goal	To restrict user activities if it's not the time for doing that.
Pre-condition	Admin is logged in, and open settings.
Post condition	List of all settings to be managed by admin.

34) E-mail students

Use case name	E-mail Students
Actors	Instructor
Description	This process is responsible to allow instructors send emails to students.
Goal	To easily communicate with any student in course.
Pre-condition	Instructor is logged in, and registered in a course and there's a students in that course.
Post condition	An email sent to the chosen student.

35) Upload tutorial

Use case name	Upload tutorial
Actors	Instructor
Description	This process is responsible to allow instructors upload assignments or quizzes.
Goal	To upload for the students the assignment or quiz in an easy way.
Pre-condition	Instructor is logged in, and registered in a course.
Post condition	Upload assignment or quiz for the students.

36) View and Grade

Use case name	View and grade
Actors	Instructor
Description	This process is responsible to allow instructors view assignments or quizzes students answers.
Goal	To view the students answer to the assignment or quiz in an easy way.
Pre-condition	Instructor is logged in, and registered in a course, and add a tutorial on it.
Post condition	Assignment/quiz answer is viewed and add the student grade

37) Upgrade students level

Use case name	Upgrade students level
Actors	Admin/super Admin
Description	This process is responsible to allow admins to upgrade students levels.
Goal	To upgrade students level on system easily.
Pre-condition	Admins are logged in, and there's a students inside the system.
Post condition	Students are upgraded to the new level.

2.4.5 Non-Functional Requirements

- **Performance:**
  Include information such as cycle time, speed per transaction, test requirements, minimum bug counts, speed, reliability, utilization… etc.

- **Security:**
  This is security for such as security audits, cryptography, user data, system identification, system authentication, resources utilization… etc.

- **Availability**:
  Such as hours of operation, level of availability required, down-time impact, support availability… etc.

- **Usability:**
  Easy to use with good and easy interface.

- **Reliability:**
  The probability that a product will operate without failure for a specified number of uses (transactions) or for a specified period of time.

- **Scalability:**
  The ability to enhance the system by adding new functionality at minimal effort.

- **Capability:**
  Measure of the ability of an entity system to achieve its objectives, especially in relation to its overall mission.

2.5 Advantages of the new system

- It is an integration of more than one system.
- Facilitate communication between students and their instructors and cooperation among their peers.
- Doesn't need to be used in a specific place.
- Help users to get information about registered course.
- The system is easy to use by anyone.
- The resources are available from anywhere and at any time. It is a very efficient way of delivering courses content.
- Online quizzes and assignment with deadline.

2.6 Risk and Risk management

Risk	Definition	Management
Authorization	unauthorized party gaining access of the assets present in E-Learning system	Provide verification and authentication to all users
Integrity Violation	unauthorized party accessing and tempering with an asset used in E-Learning system	Provide verification and authentication to all users
Denial of Service	Prevention of legitimate access rights by disrupting traffic during the transaction among the users of E-Learning system	Make the system updated and kept under supervision for maintenance
Repudiation	Persons denial of participation in any transaction of documents	Keep previous sessions in case of failure
Illegitimate use	Exploitation of privileges by legitimate users	Only display the legitimate views for each user
Traffic analysis	Leakage of information by abusing communication channel.	Secure the communication channels

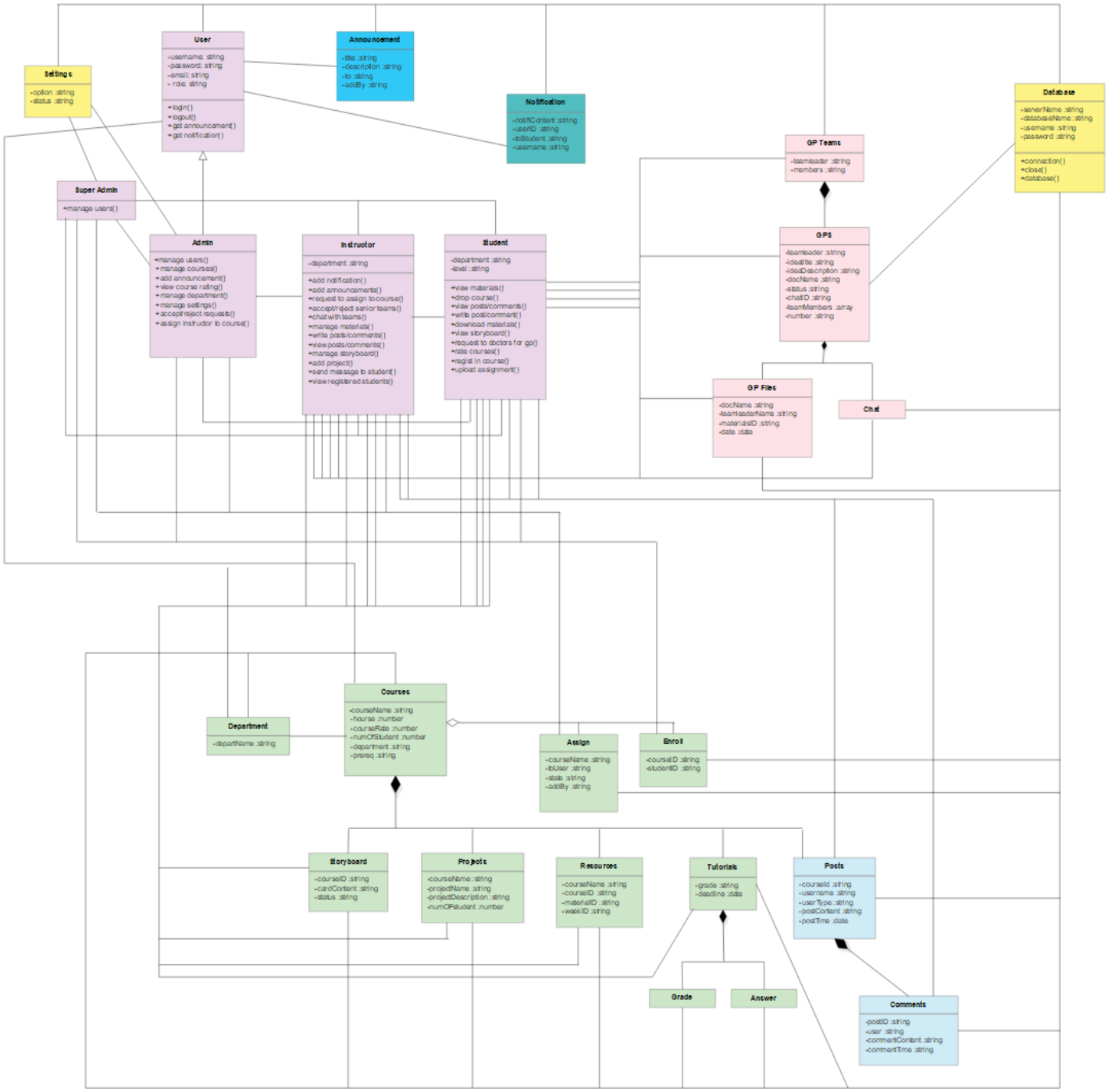
Chapter Three

In this chapter, we are going to discuss and go deeper in our website's design, and present its diagrams and database.

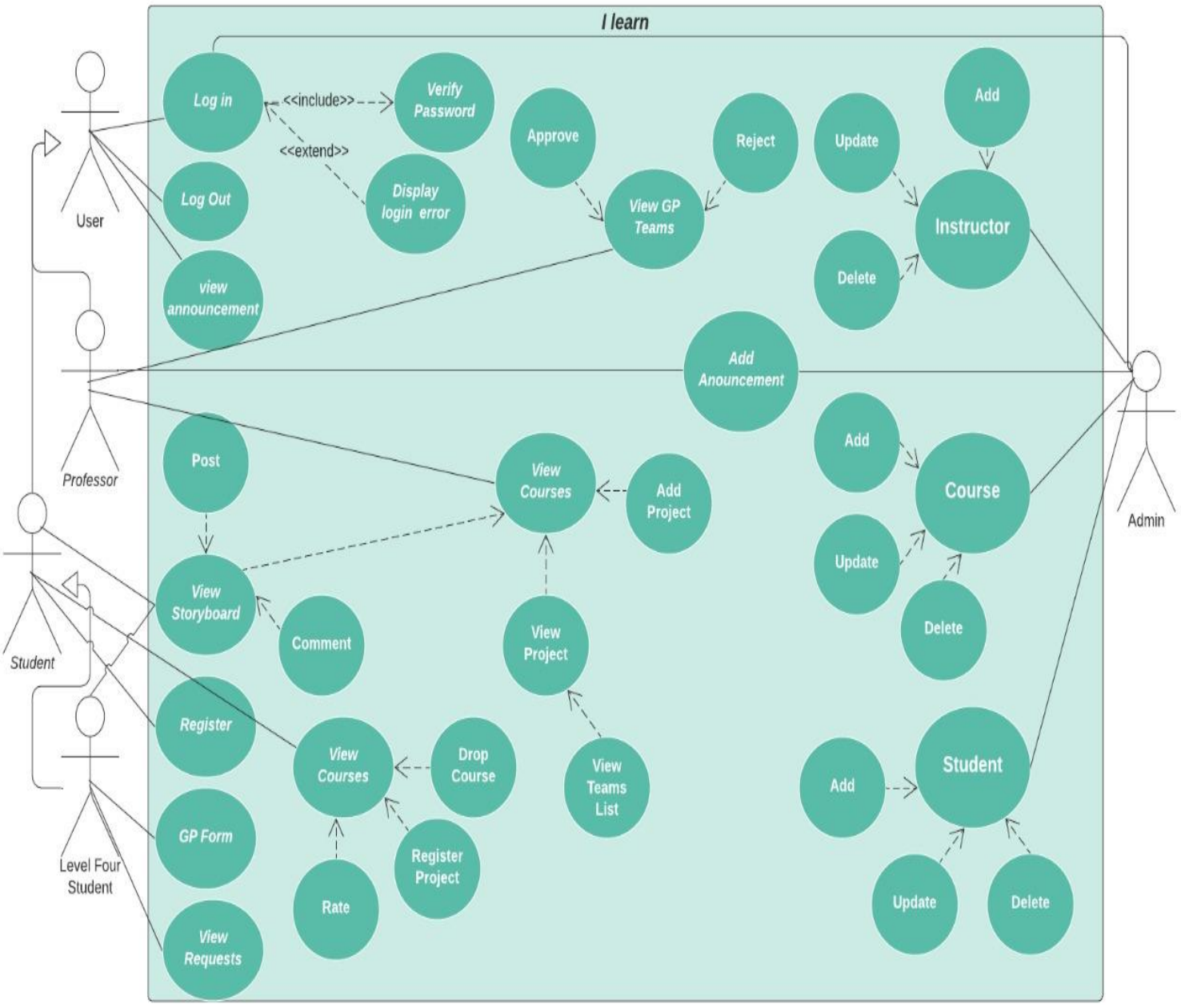
Chapter Headlines:

1. Class Diagram
2. Use Case Diagram
3. Activity Diagram
4. Sequence Diagram

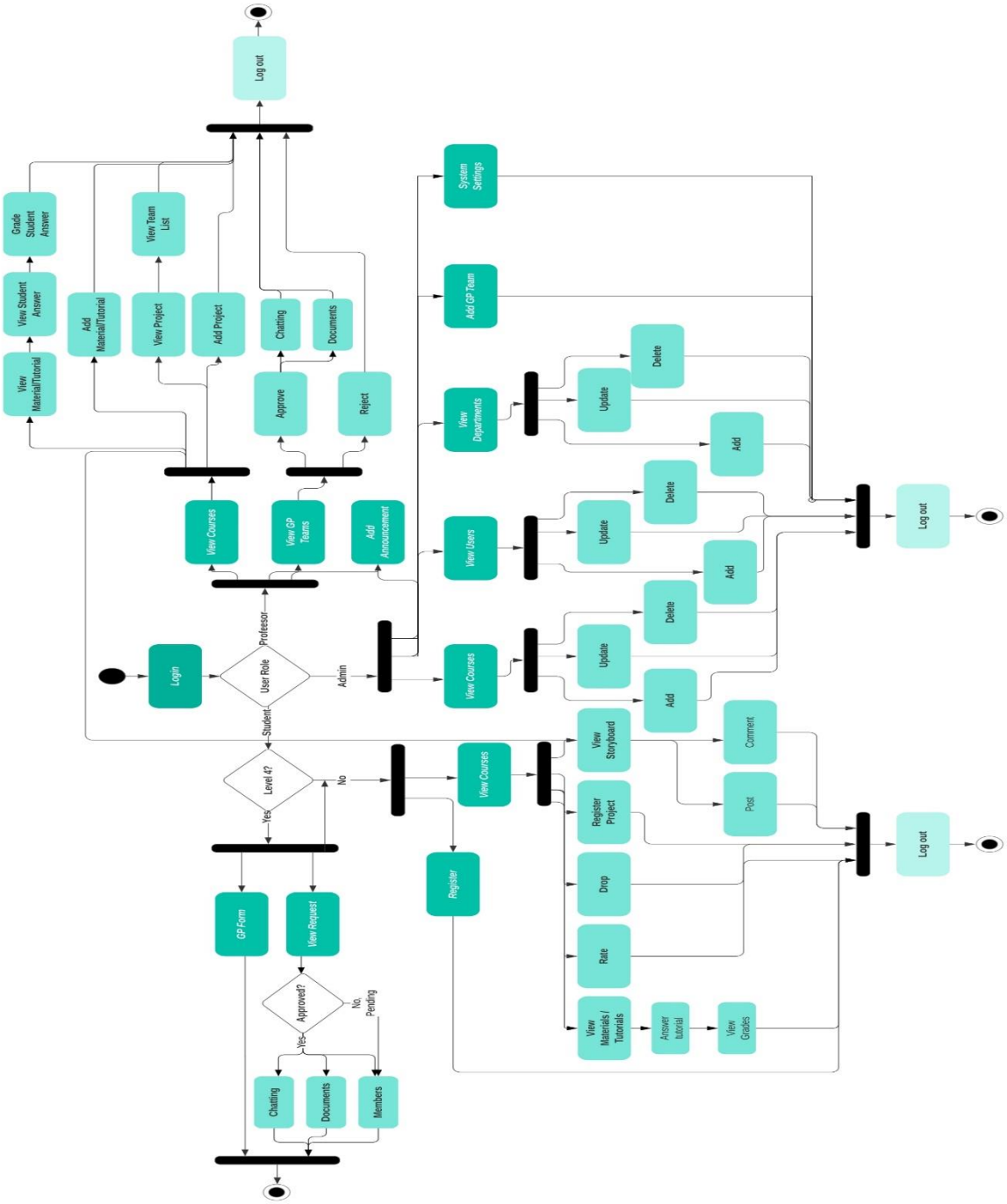
3.1 Class Diagram



3.2 Use Case Diagram

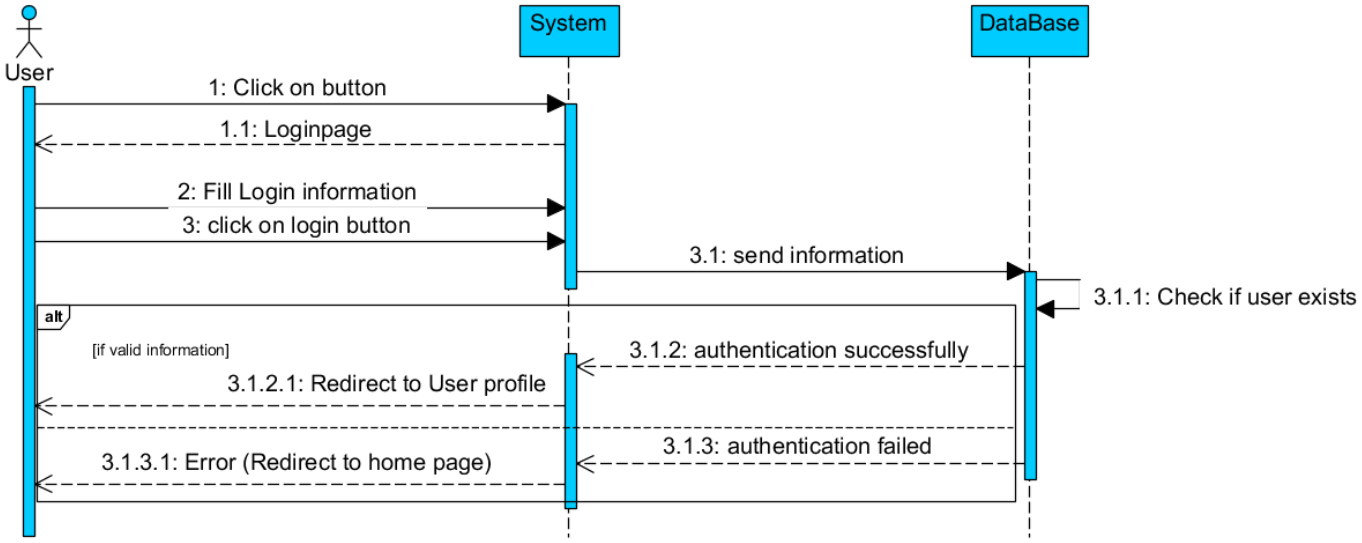


3.3 Activity Diagram

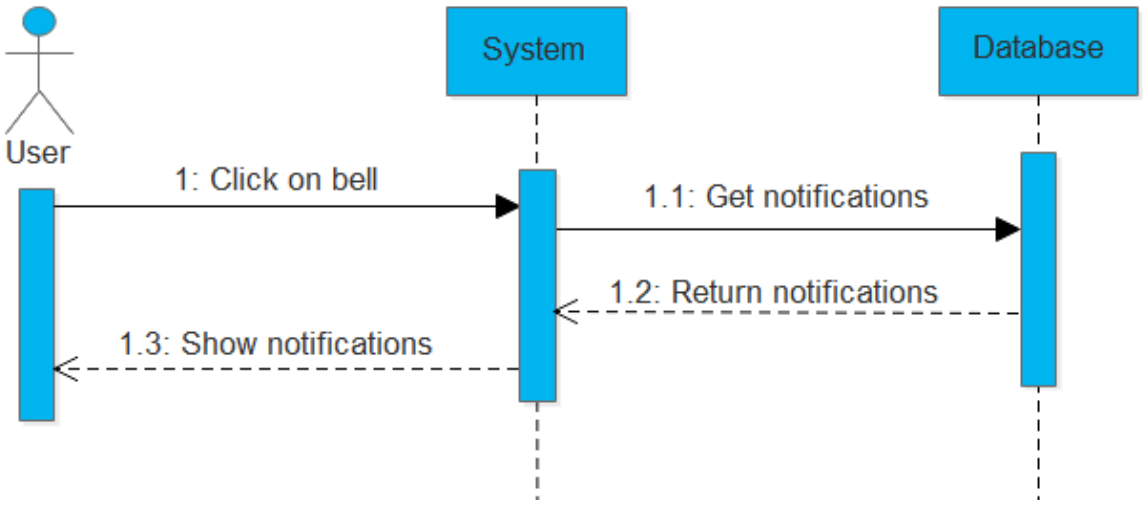


3.4 Sequence Diagram

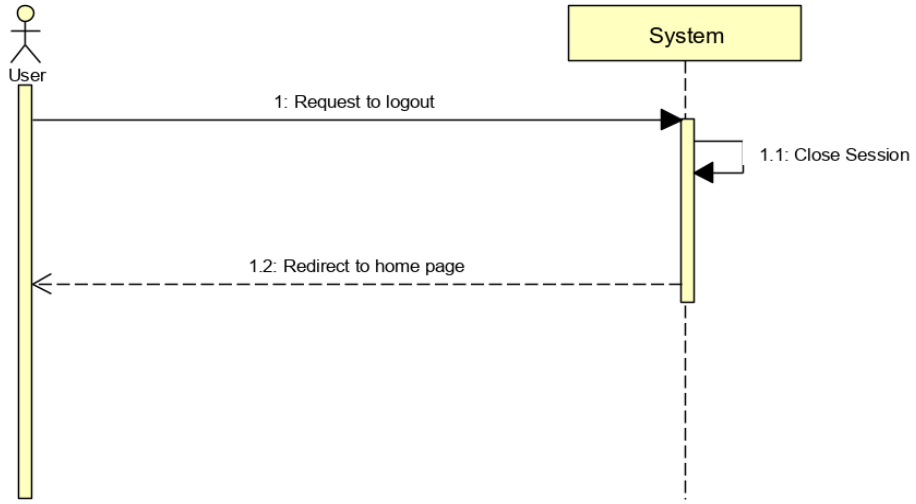
Login



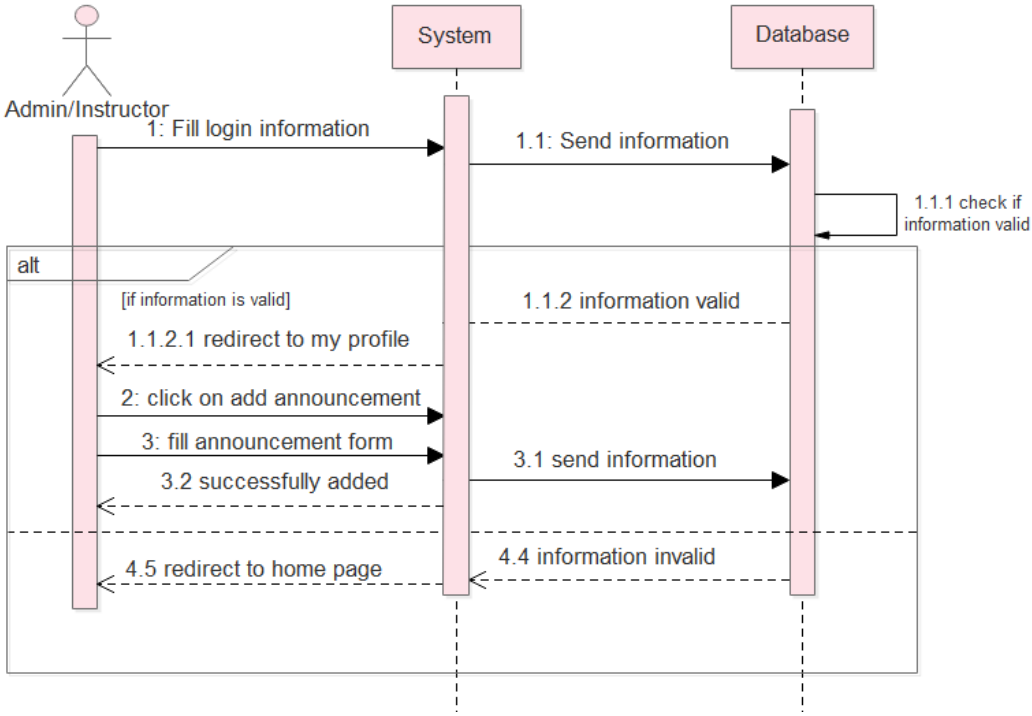
Show Notification



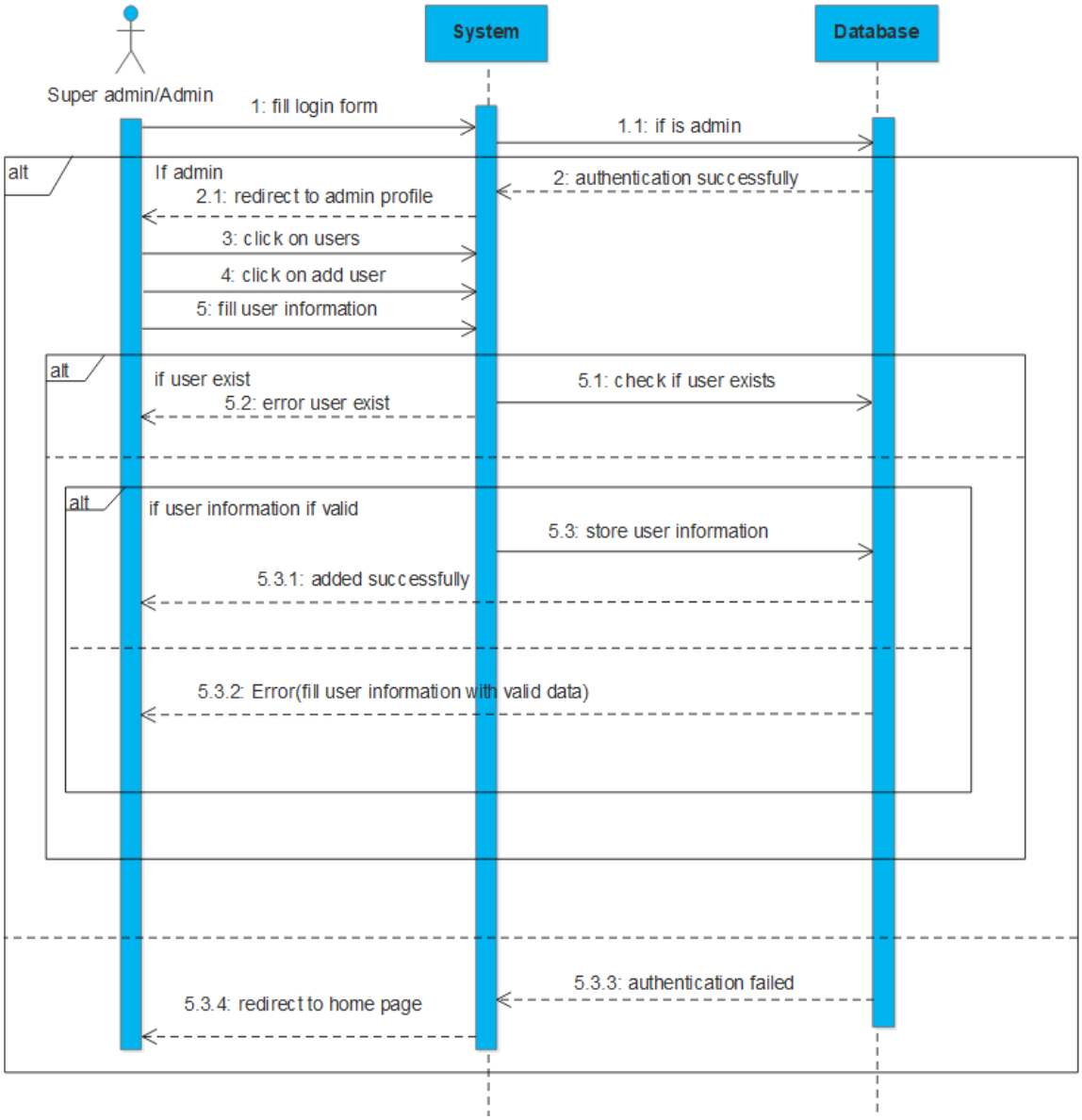
Logout



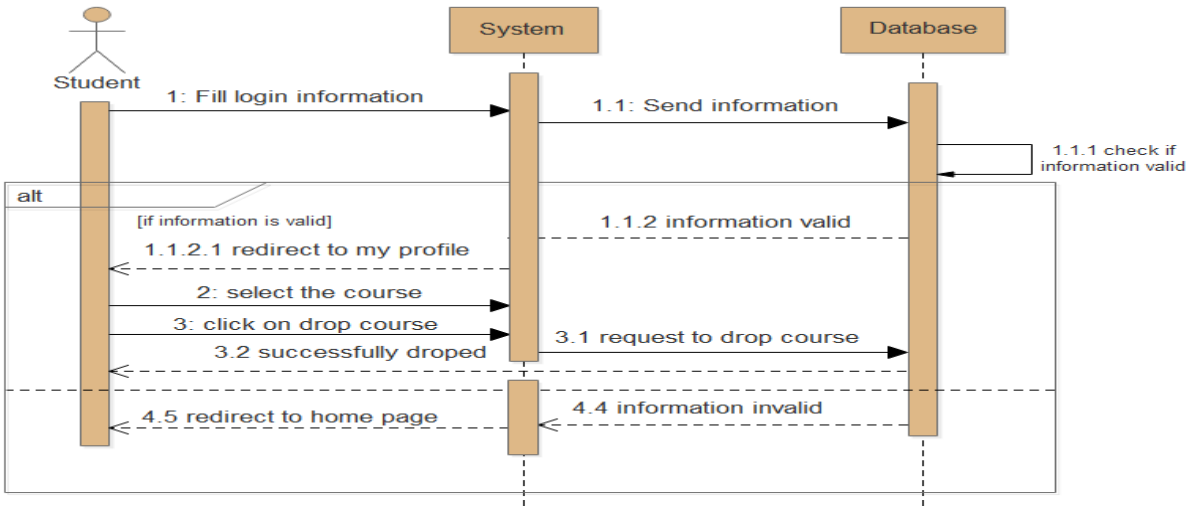
Add Announcement



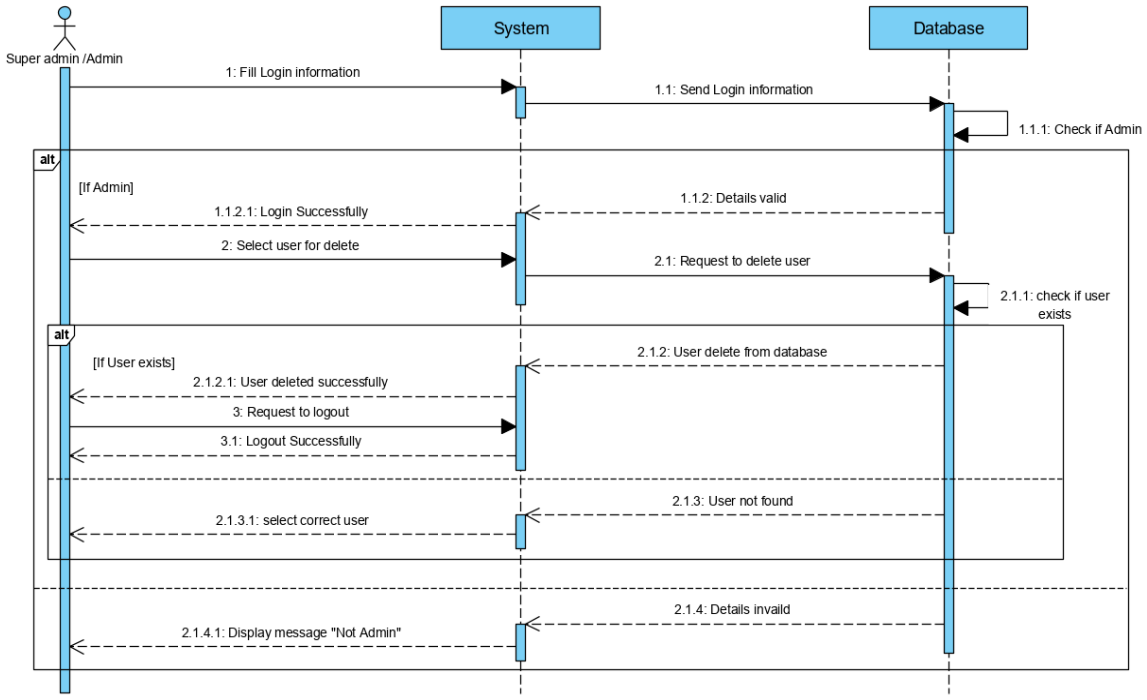
Add User


System
Database
Super admin/Admin
1: fill login form
1.1: if is admin
alt
If admin
2: authentication successfully
2.1: redirect to admin profile
3: click on users
4: click on add user
5: fill user information
alt
if user exist
5.1: check if user exists
5.2: error user exist
alt
if user information if valid
5.3: store user information
5.3.1: added successfully
5.3.2: Error(fill user information with valid data)
5.3.3: authentication failed
5.3.4: redirect to home page

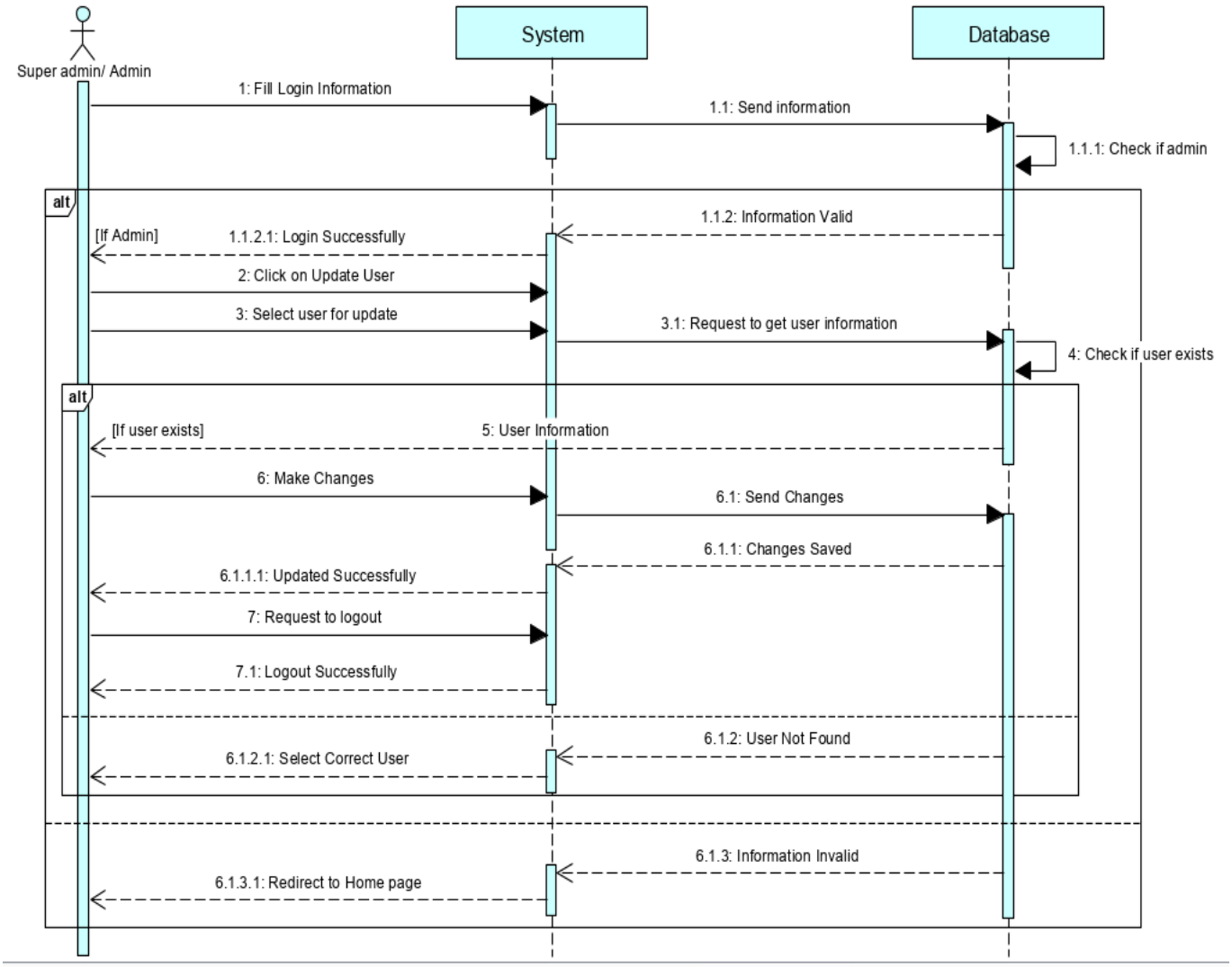
Drop Course



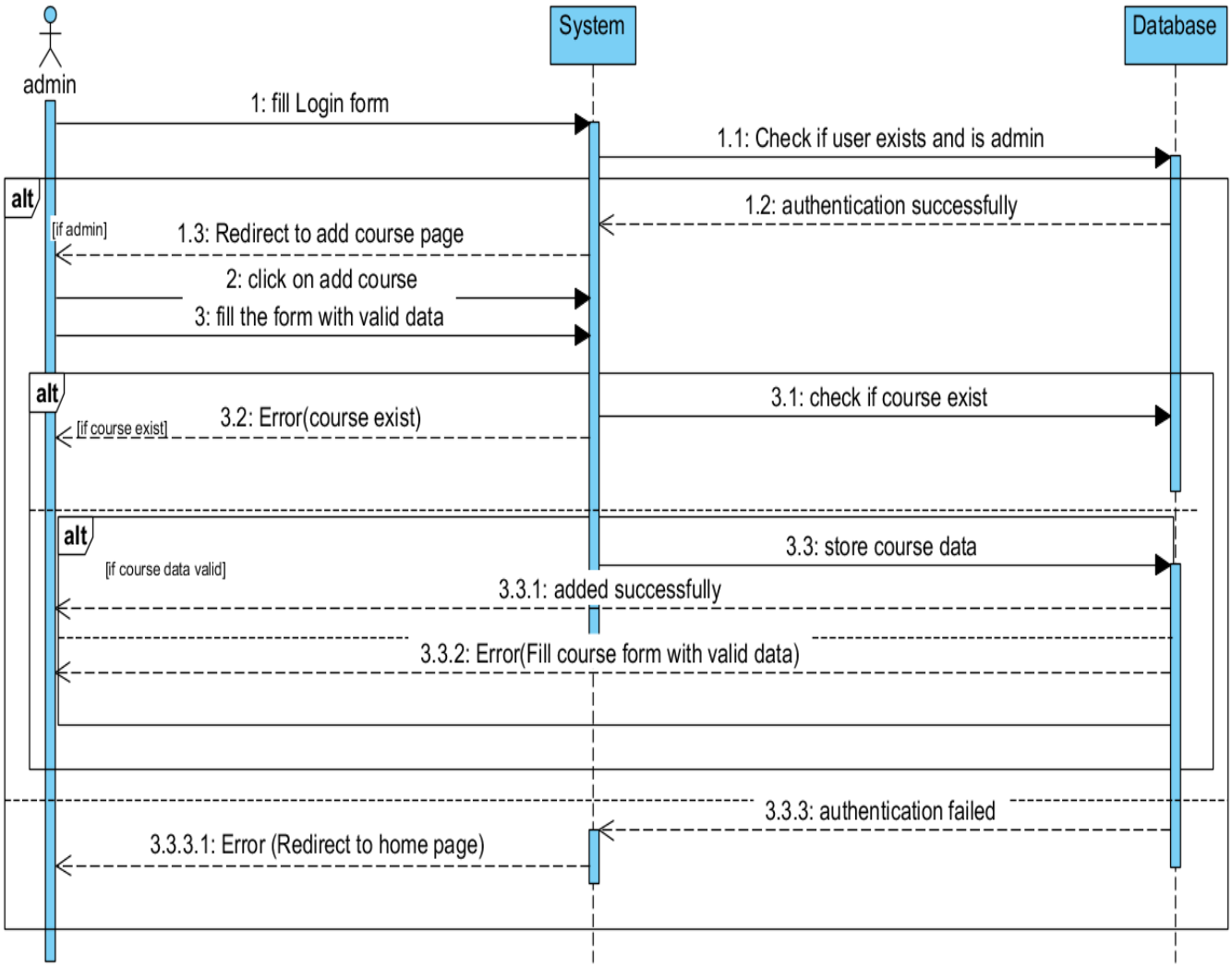
Delete User



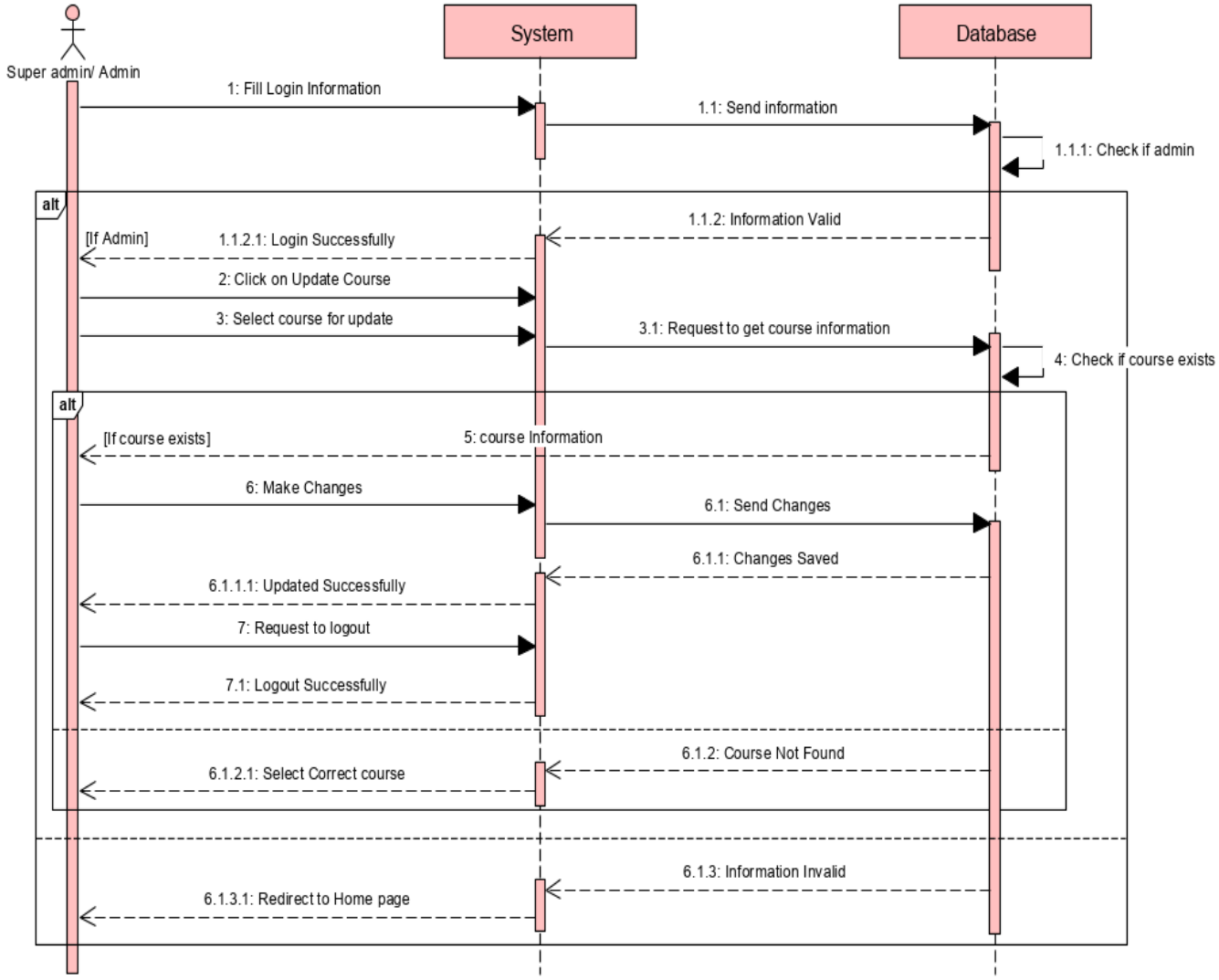
Update User


Super admin/ Admin
System
Database
1: Fill Login Information
1.1: Send information
1.1.1: Check if admin
alt
[If Admin]
1.1.2: Information Valid
1.1.2.1: Login Successfully
2: Click on Update User
3: Select user for update
3.1: Request to get user information
4: Check if user exists
alt
[If user exists]
5: User Information
6: Make Changes
6.1: Send Changes
6.1.1: Changes Saved
6.1.1.1: Updated Successfully
7: Request to logout
7.1: Logout Successfully
6.1.2: User Not Found
6.1.2.1: Select Correct User
6.1.3: Information Invalid
6.1.3.1: Redirect to Home page

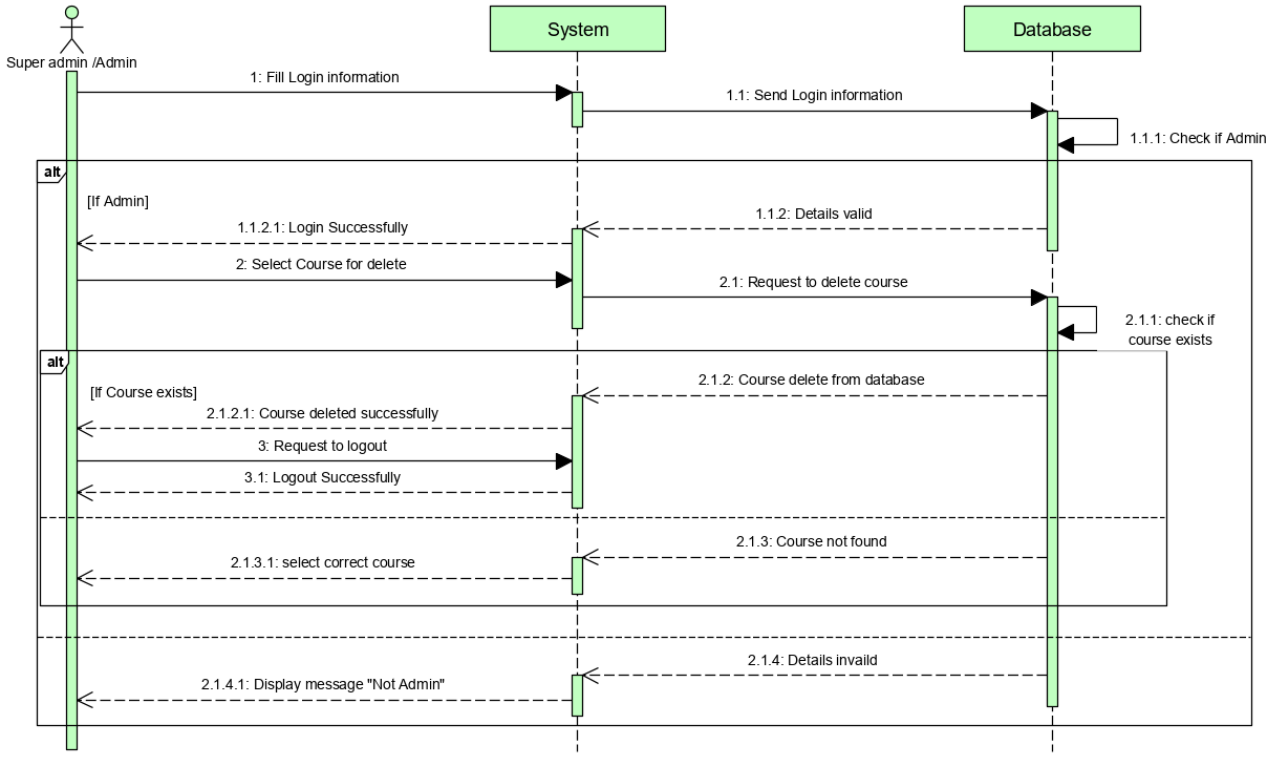
Add Course


admin
System
Database
1: fill Login form
1.1: Check if user exists and is admin
alt
[if admin]
1.2: authentication successfully
1.3: Redirect to add course page
2: click on add course
3: fill the form with valid data
alt
[if course exist]
3.1: check if course exist
3.2: Error(course exist)
alt
[if course data valid]
3.3: store course data
3.3.1: added successfully
3.3.2: Error(Fill course form with valid data)
3.3.3: authentication failed
3.3.3.1: Error (Redirect to home page)

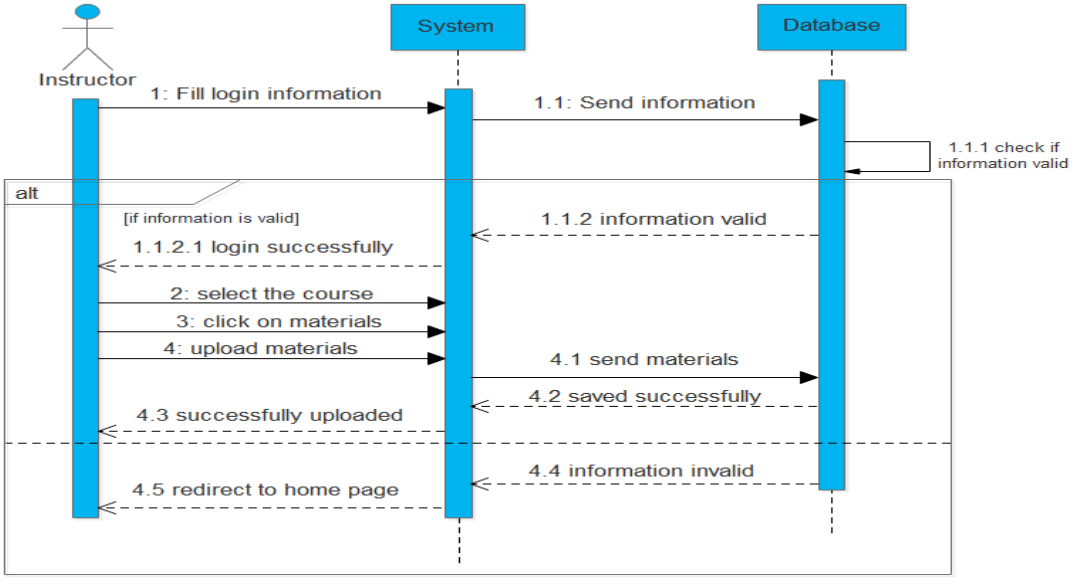
Update Course



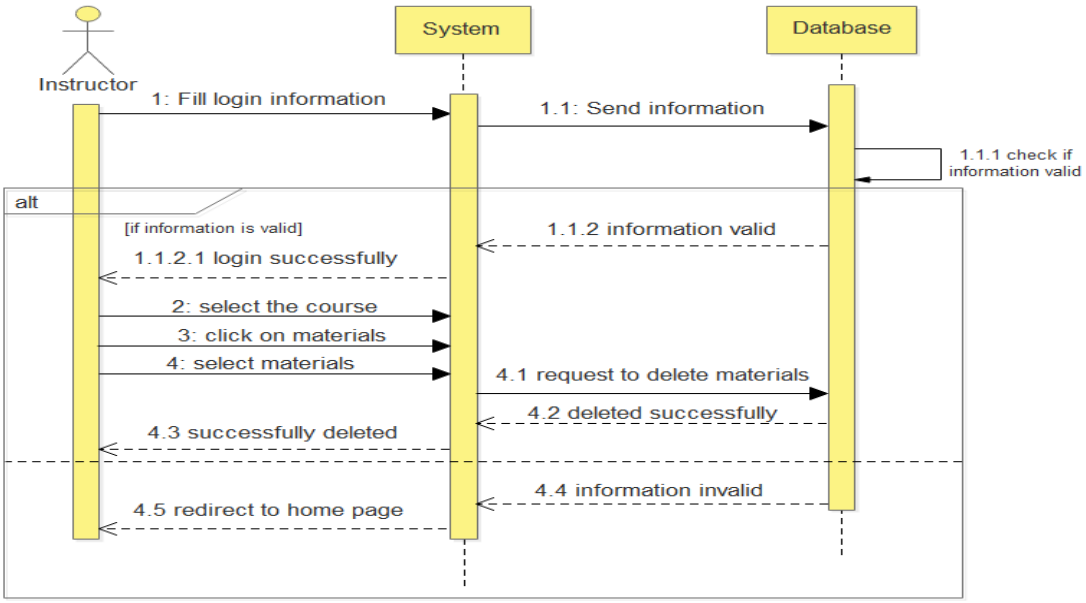
Delete Course



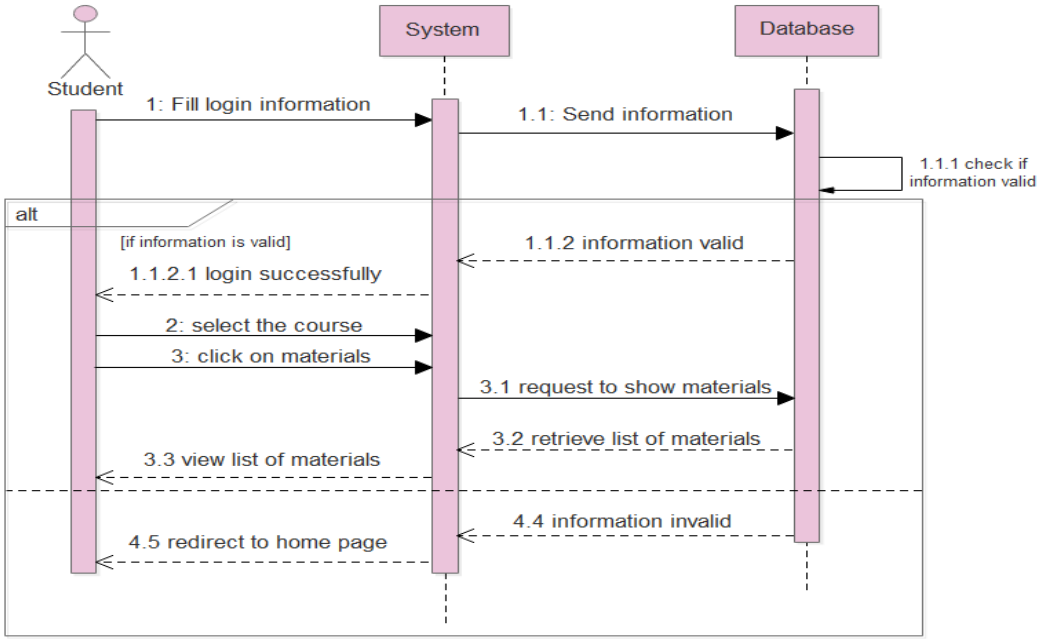
Add Materials



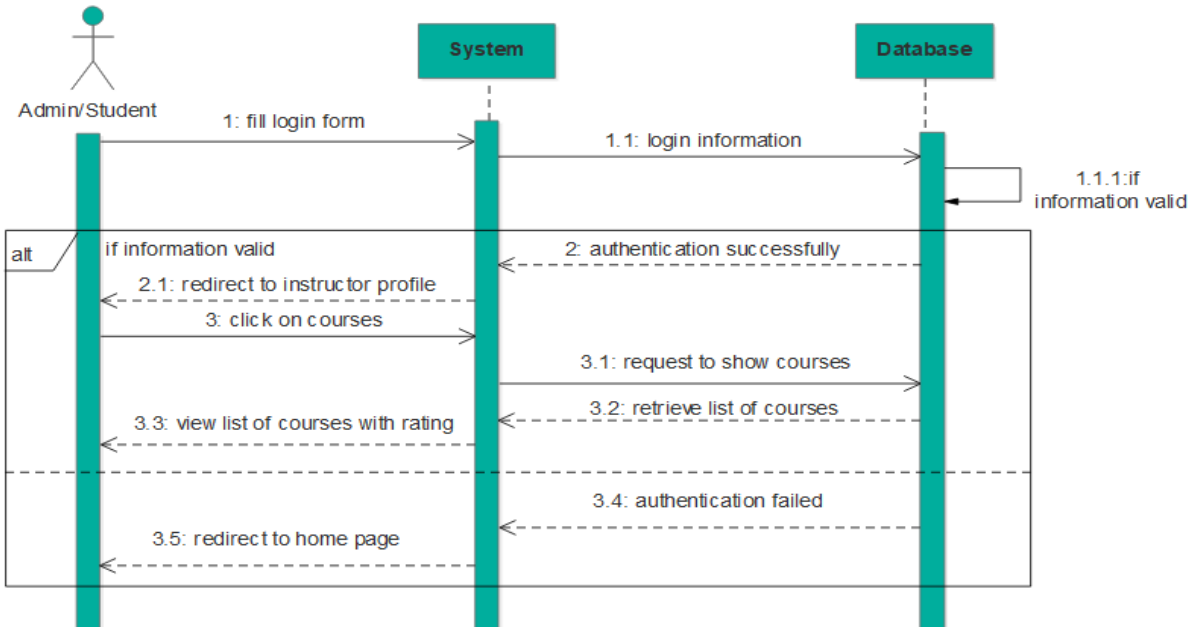
Delete Materials



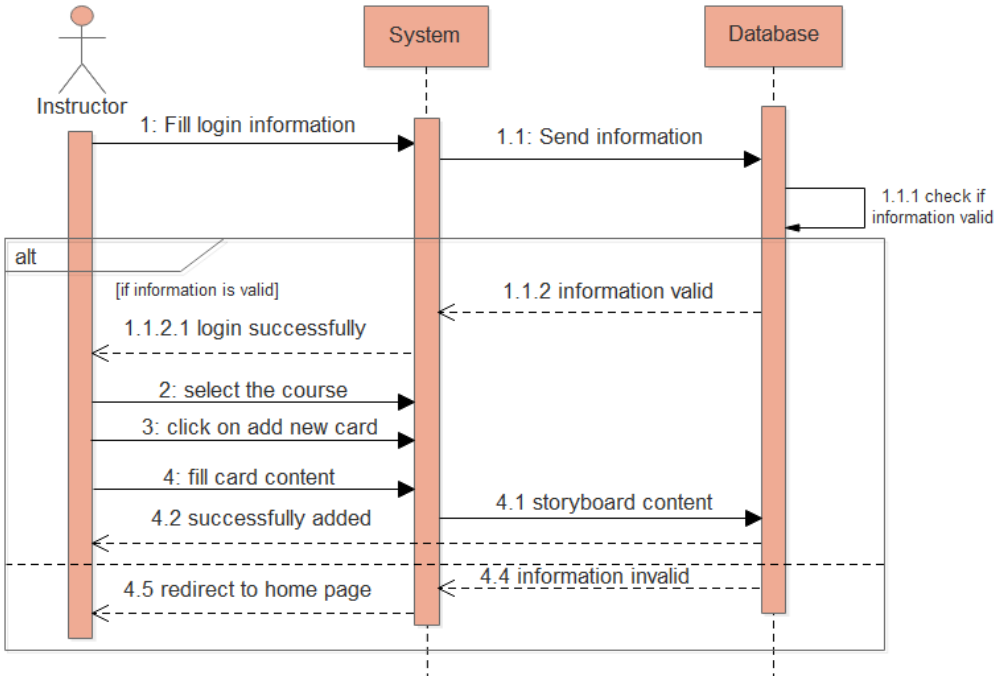
View Materials



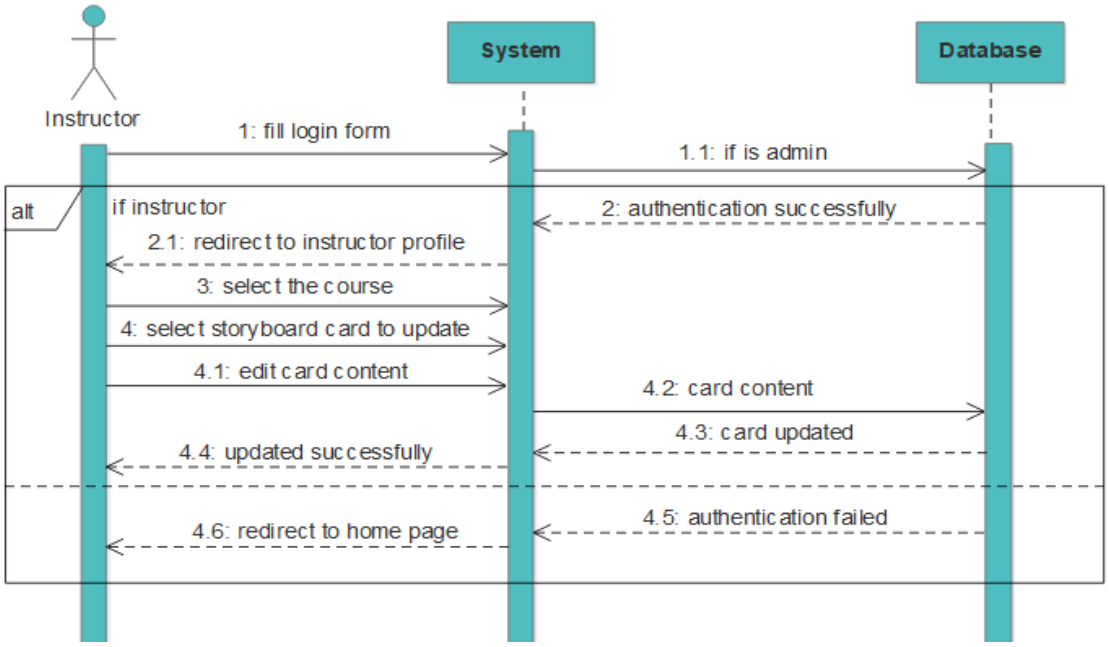
View Course Rating



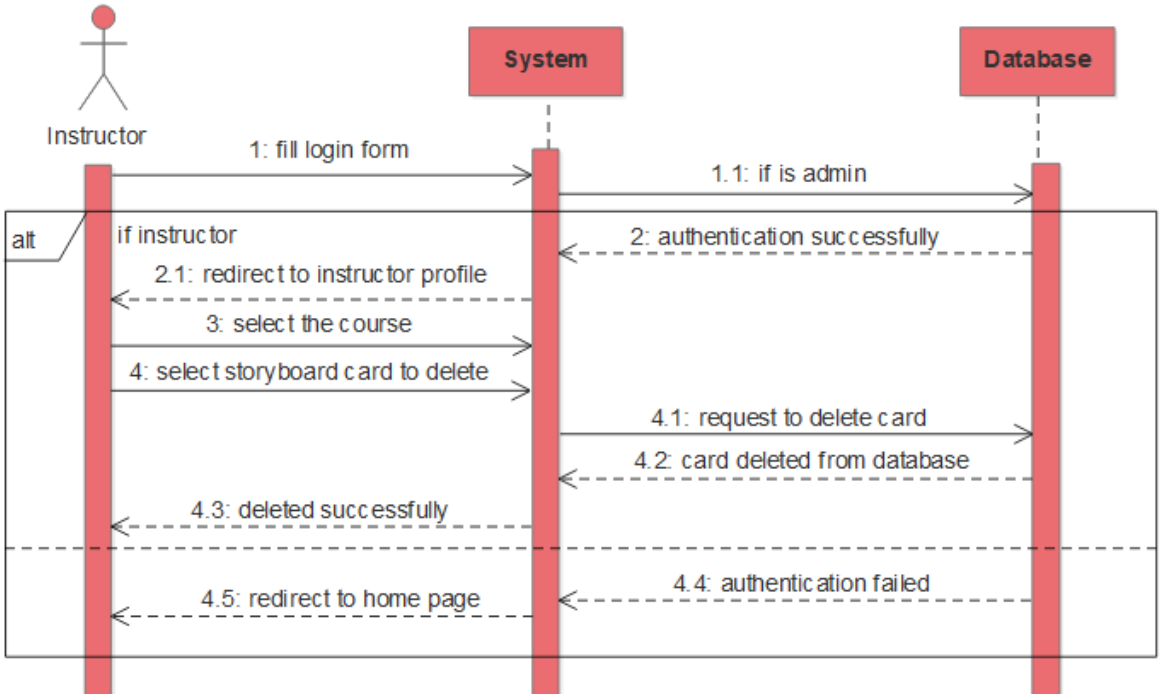
Add Storyboard Card



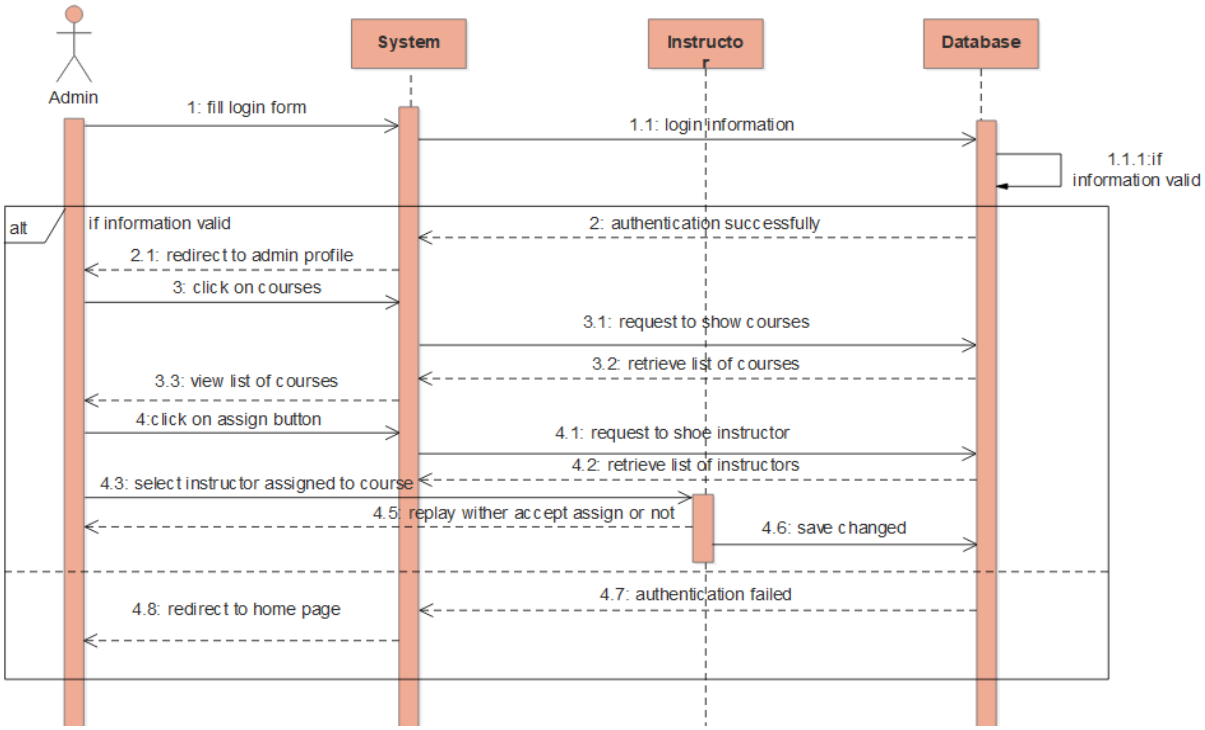
Update Storyboard Card Status



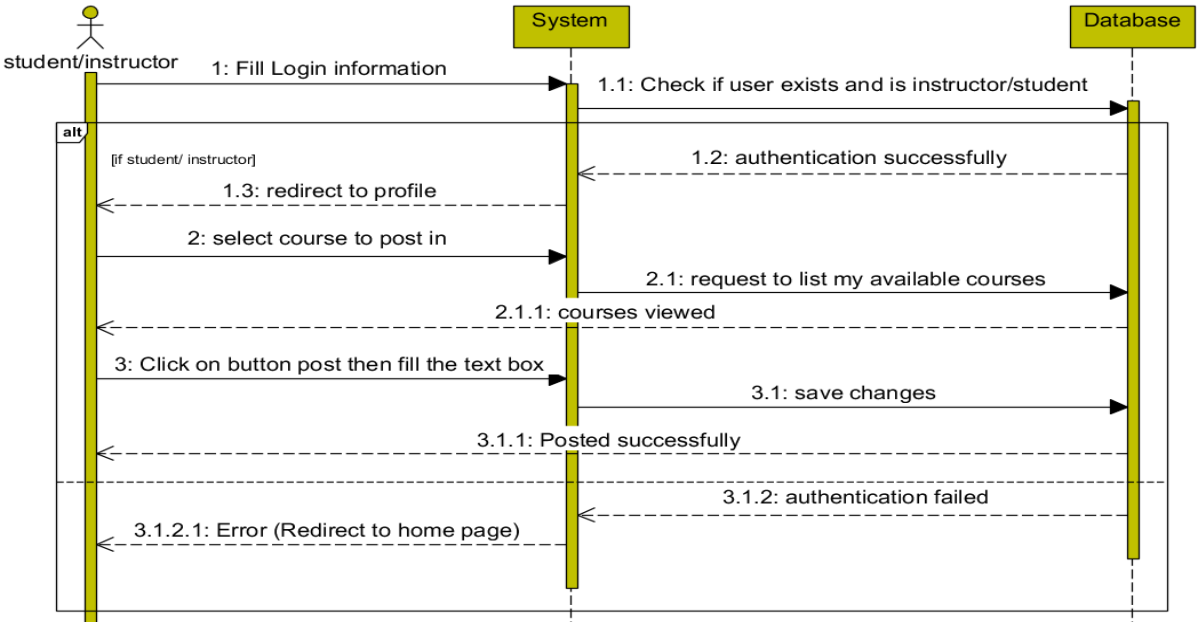
Delete Storyboard Card



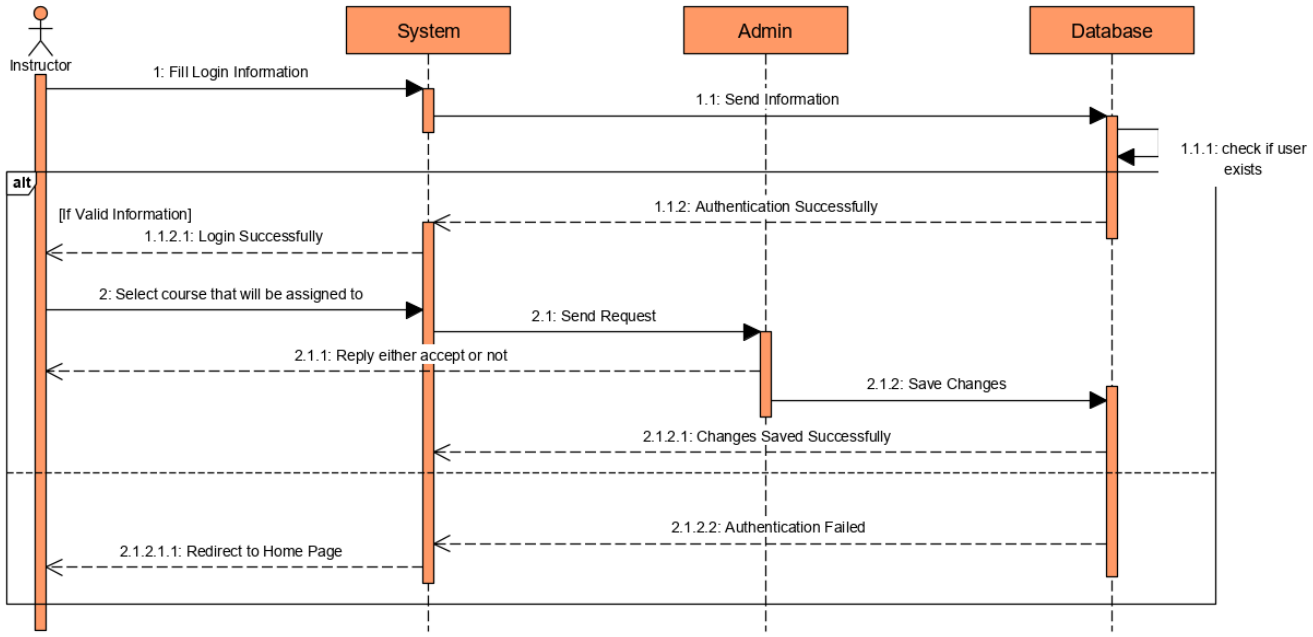
Assign Course to Instructor



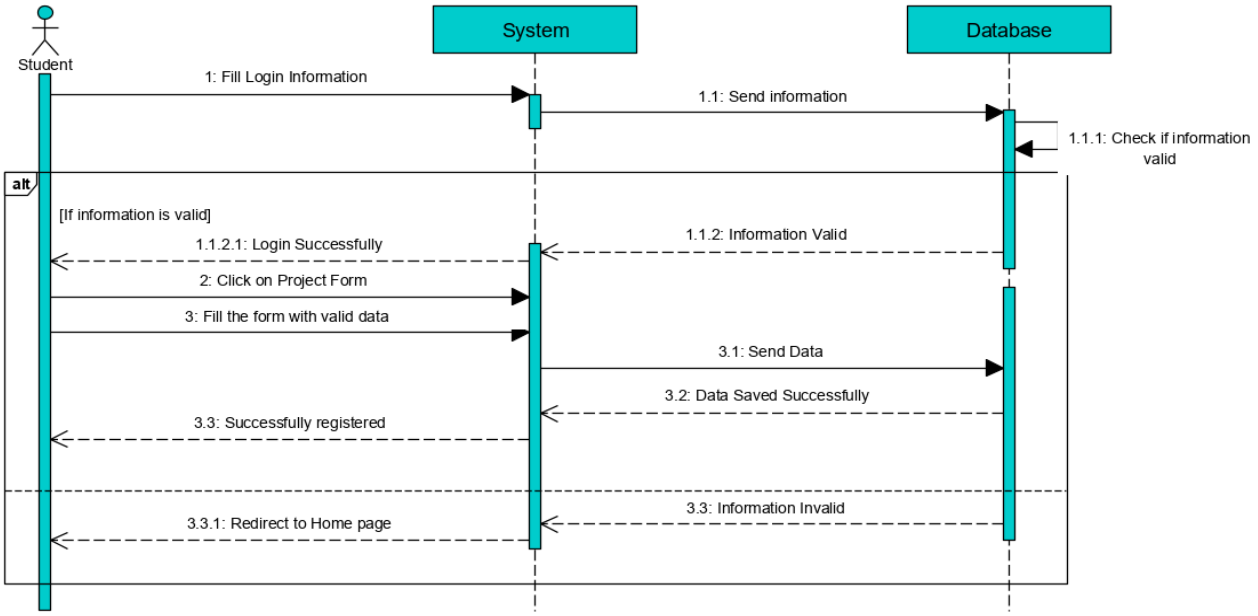
Post on Course



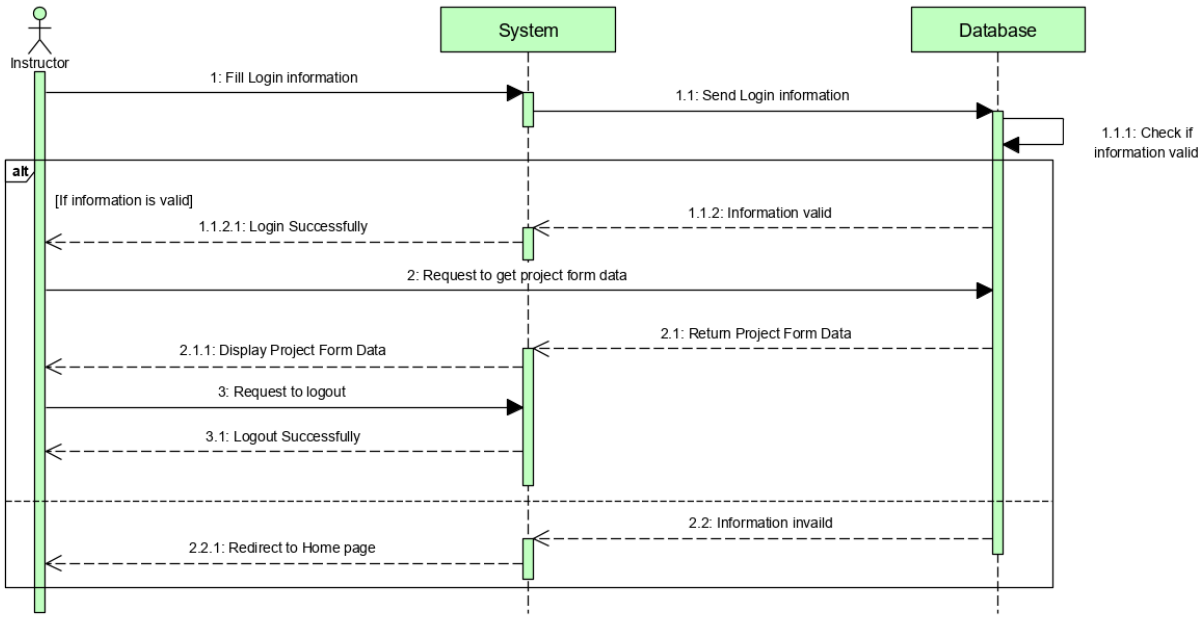
Request to be assigned to a course



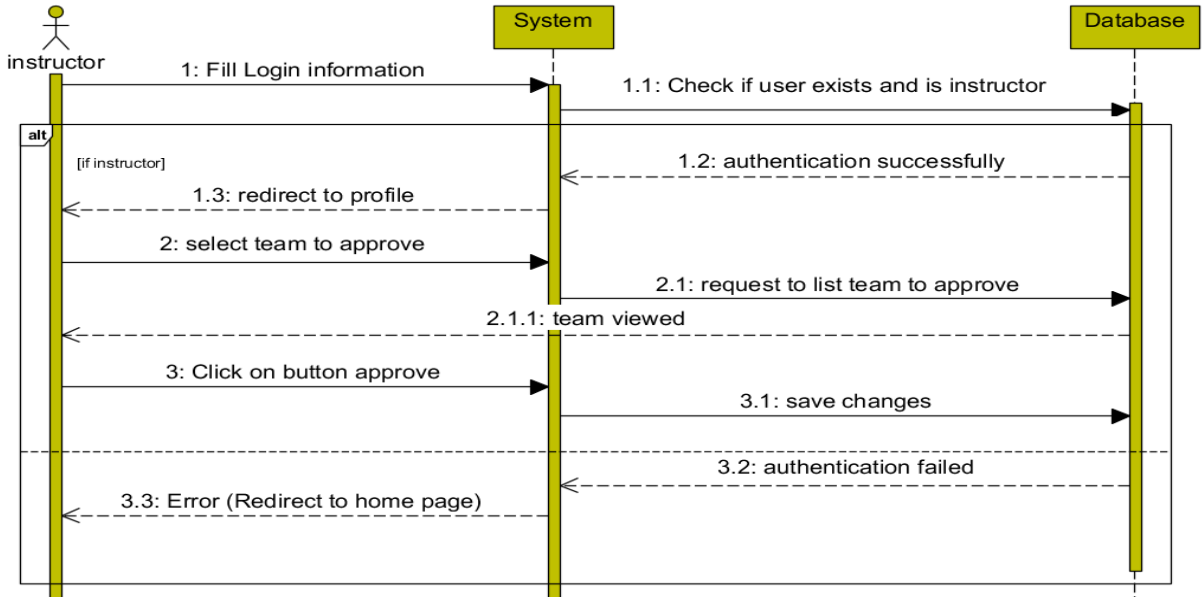
Registration in Project Form



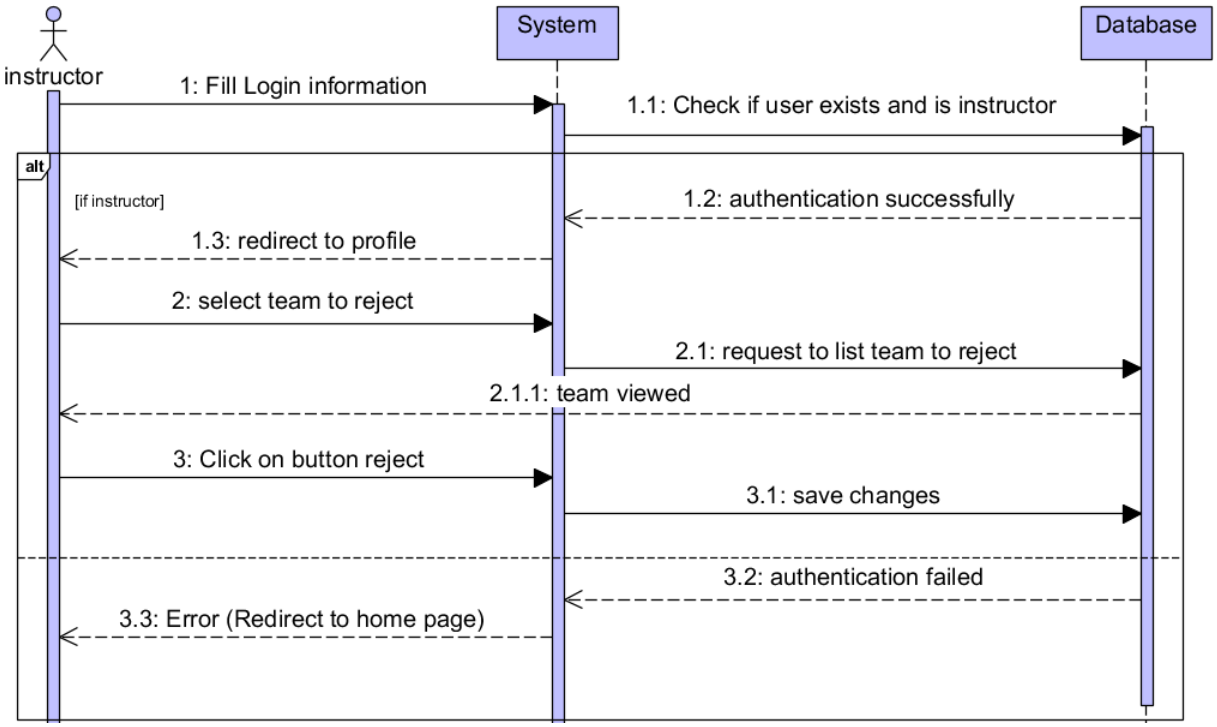
View Project Form Data



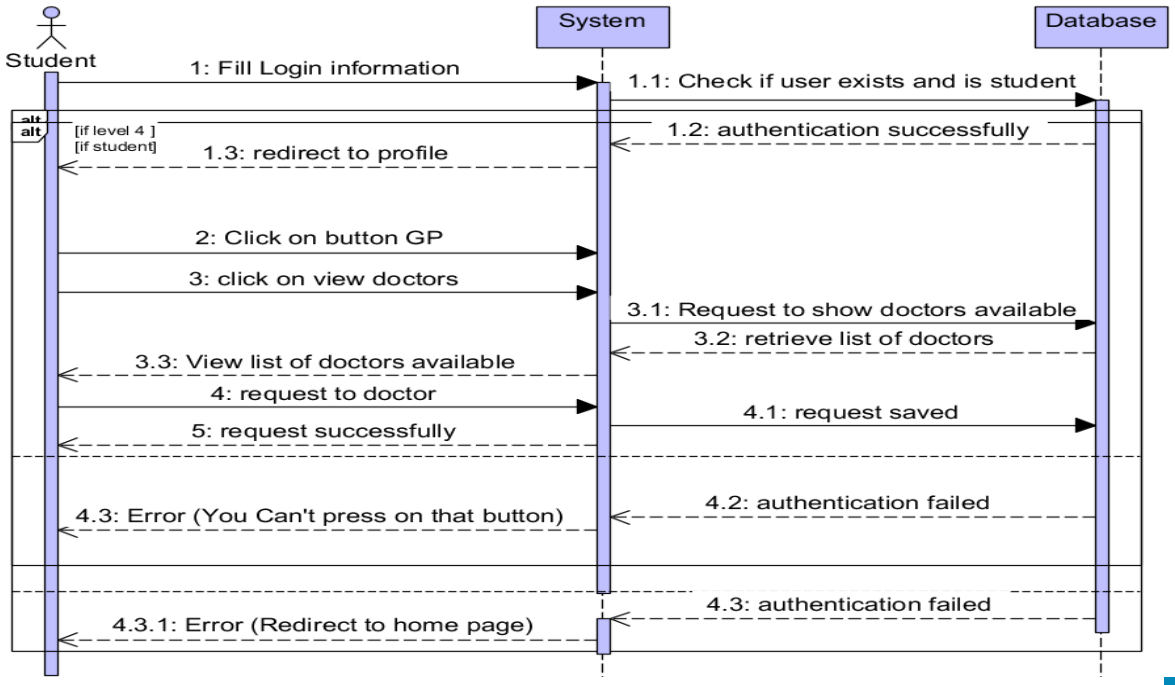
Approve Team GP



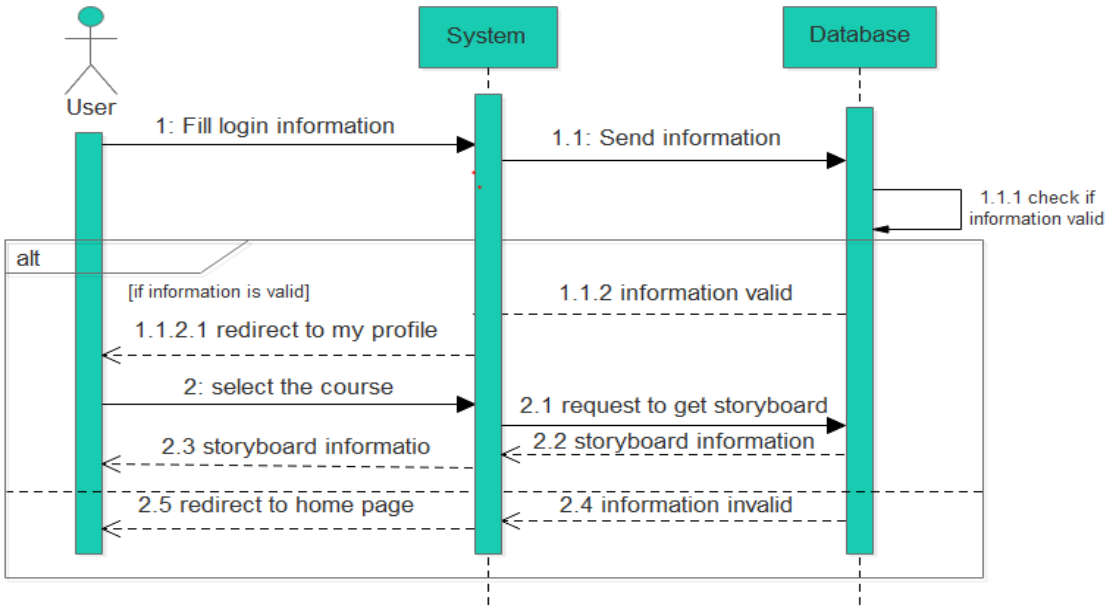
Reject Team GP



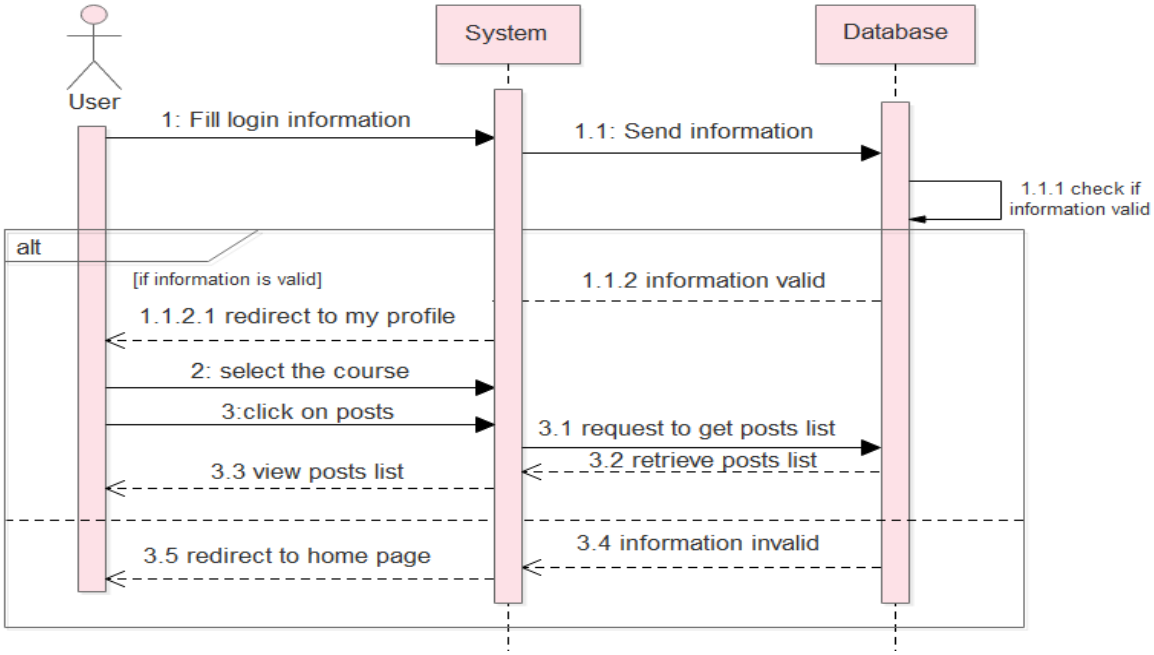
Send request to doctor to be supervisor



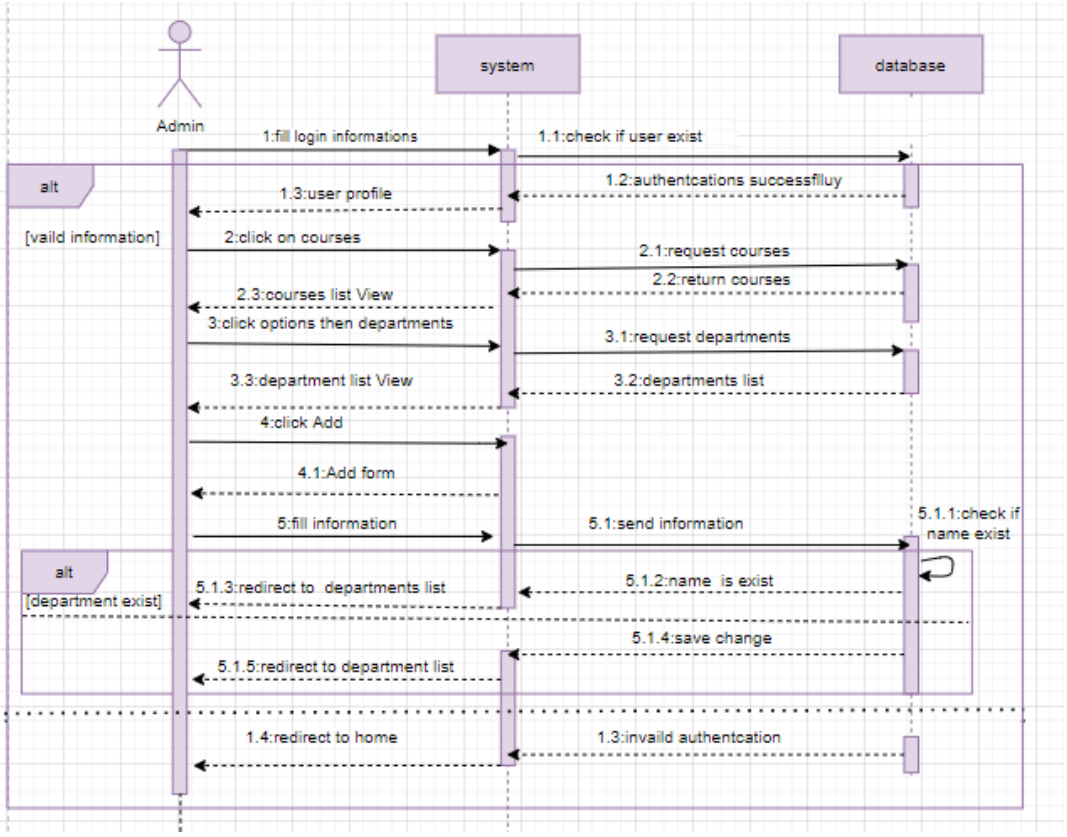
View Storyboard



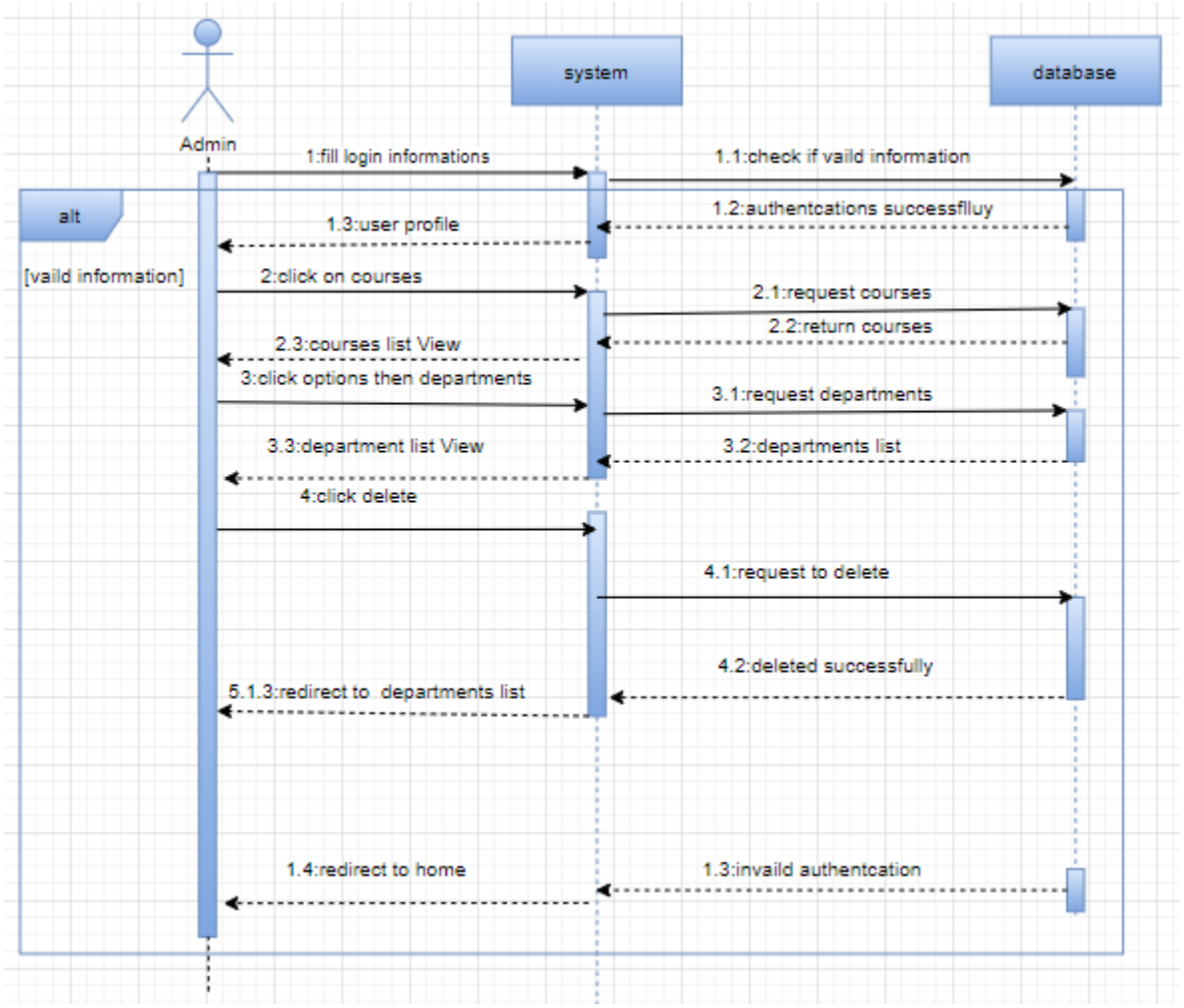
View Posts



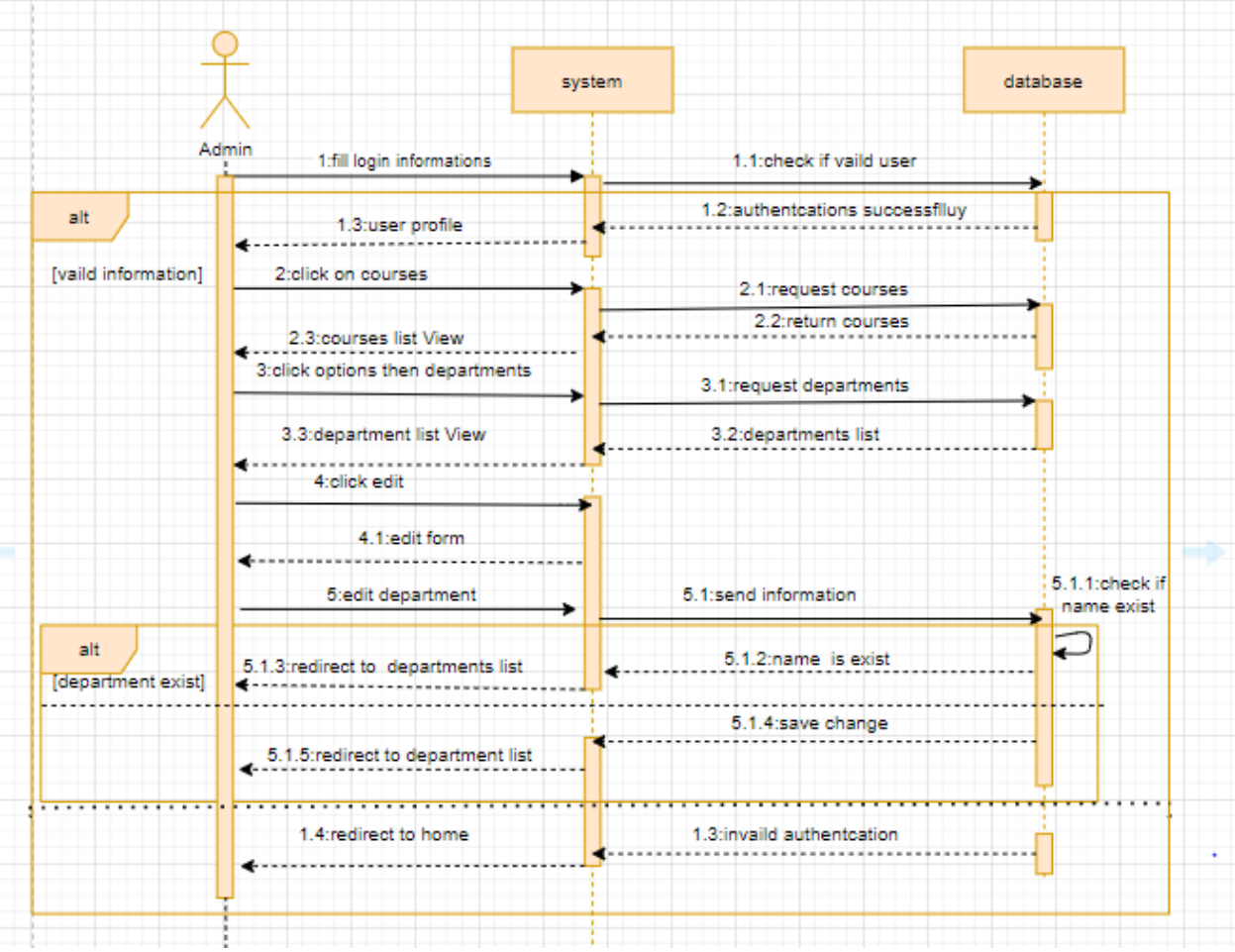
Add Department



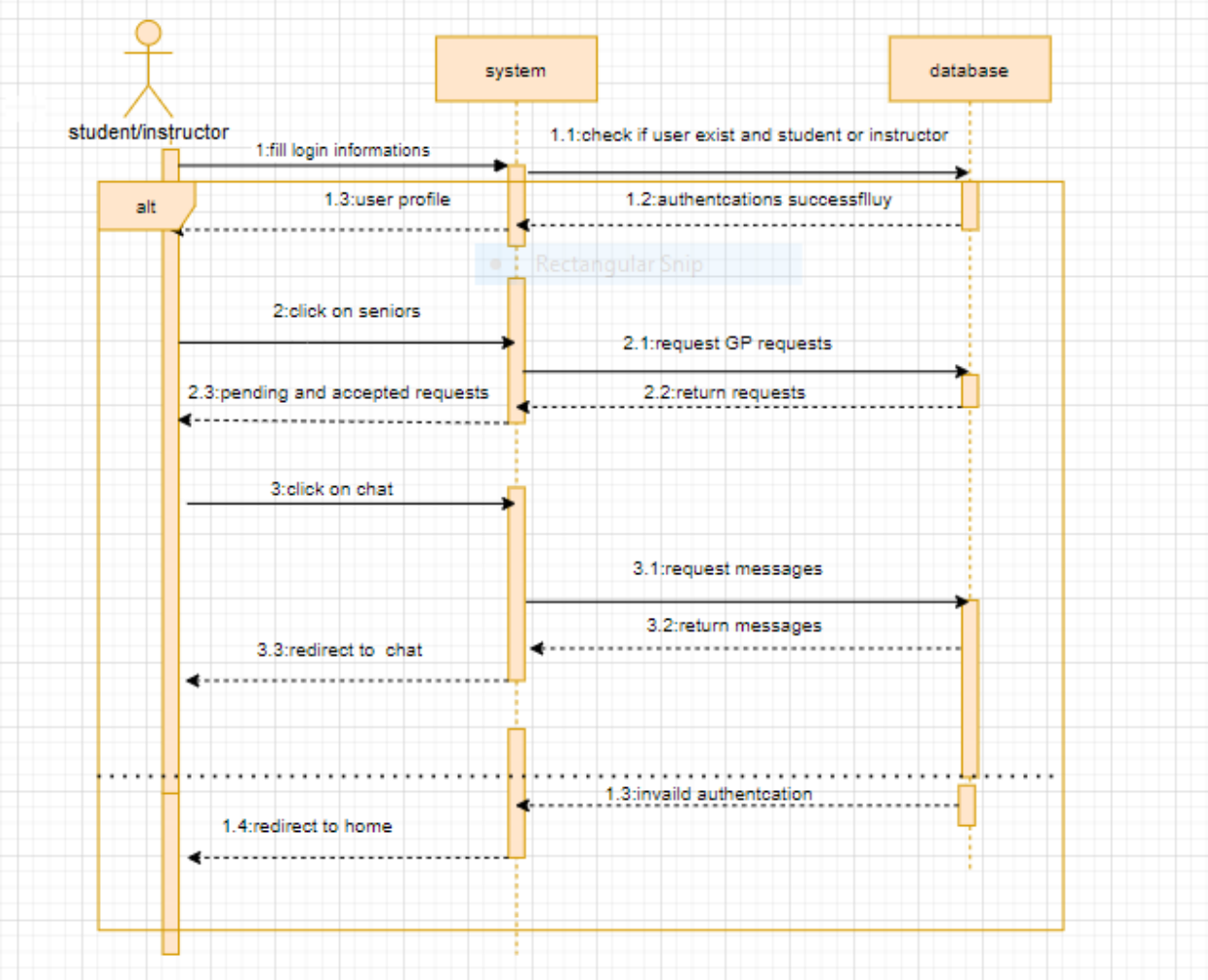
Delete Department



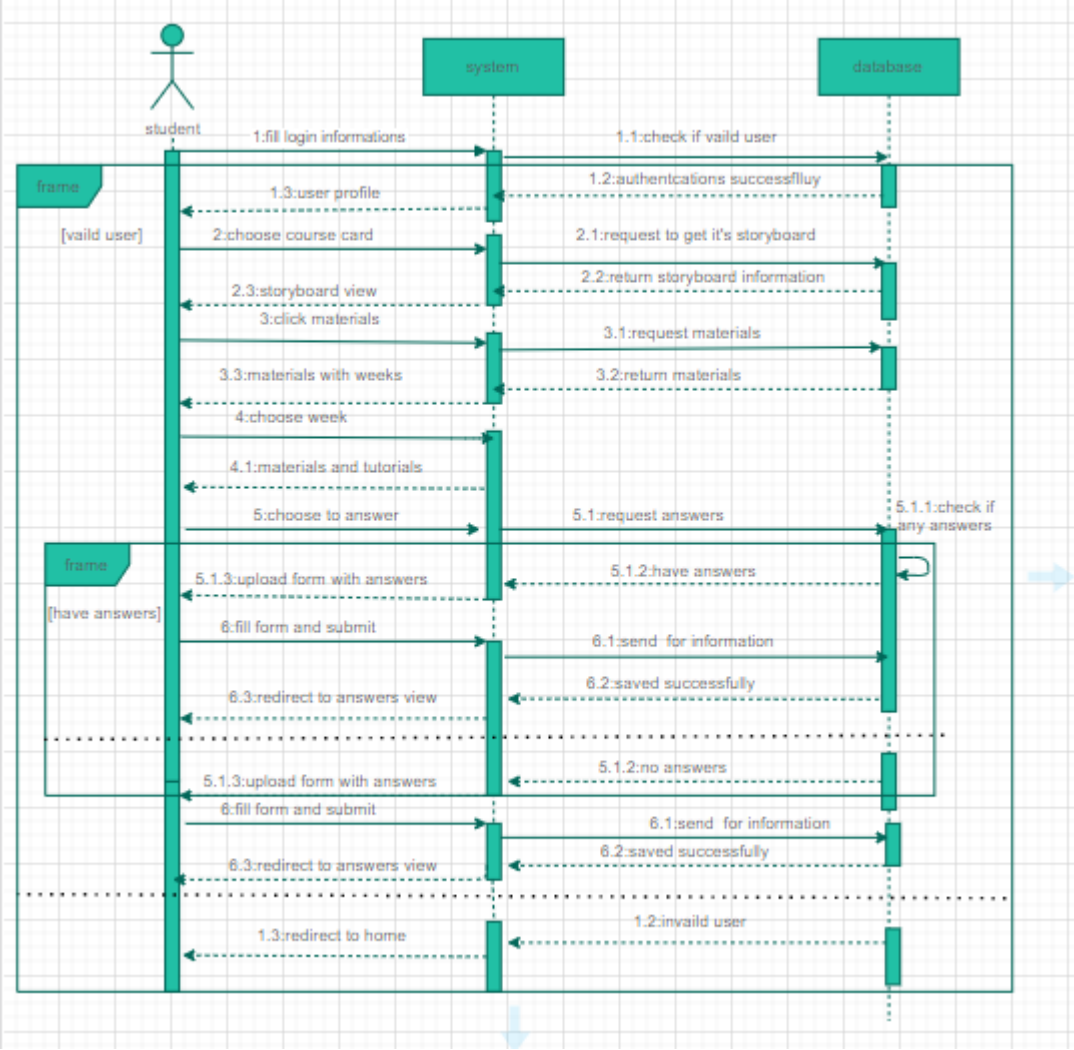
Edit Department



Start Chat


student/instructor
system
database
1:fill login informations
1.1:check if user exist and student or instructor
alt
1.3:user profile
1.2:authentcations successfilluy
2:click on seniors
2.1:request GP requests
2.3:pending and accepted requests
2.2:return requests
3:click on chat
3.1:request messages
3.2:return messages
3.3:redirect to chat
1.3:invaild authentcation
1.4:redirect to home

Answer Tutorial



Chapter Four

In this chapter we're going to discuss and go deeper in I-Learn system's implementation, and present its code and the algorithms used to build it.

Chapter Headlines:

1. Software Architecture
2. Flowchart

4.1 Software Architecture

Login

```
exports.postLogin = async (req, res, next) => {
    var studnotfi=[];
    const password =req.body.Password;
    const userName=req.body.Username;

    var arrayy=[];
    var myCourses=[];
    var userrrrrrr;
    exports.getAllinst(req, res, next);
    await assaign.find({toUser : req.body.Username,state:'accepted'})
    .then(mycourse => {
        req.session.mycourse=mycourse;
        res.locals.assaign=req.session.mycourse;
        for (var x of mycourse){
            arrayy.push(x.courseName)
        }
        Course.find({
            'courseName': { $in: arrayy}
        },function(err, docs){
            myCourses=docs;
            req.session.c=docs;
            res.locals.cour=req.session.c; });
        res.locals.cour=req.session.c; })
    await notfi.find().then(notificationn=>{

    res.locals.notfi=req.session.notificationn;
    for(i=0;i<notificationn.length;i++){
    studnotfi.push(notificationn[i]);}
    }).catch(err => console.log(err));
    await User.findOne({ userName: userName })
        .then(user => {
            if (!user) {
                req.flash('error', 'Invalid email or password.');
                return res.redirect('/login');
            }
```

Login Cont…

```
        return  bcrypt
            .compare(password, user.password)
            .then(doMatch => {
                if (doMatch) {
                req.session.isLoggedIn = true;
                req.session.user = user;
                req.session.save();
                res.locals.user = req.session.user;
                userrrrrrr= req.session.user.userName;
                res.locals.myannounce=[];
                res.locals.mynotfi=[];
                res.locals.mycoursess=[];
                exports.dashboard(req, res, next);
                    switch (user.userRole) {

                        case 'admin':
                            admin();
                            res.locals.myannounce=adminann;
                            res.redirect('admin');
                            break
                        case'student':
                            student(req.session.user._id);
                            res.locals.myannounce=stuann;
                            res.locals.mynotfi=studnotfi;
                            res.redirect('student');
                            break
                        case 'instructor':
                            instructor();
                            res.locals.myannounce=instann;      You, 4 months ago • annour
                            res.locals.mycoursess=arrayy;
                            res.redirect('instructor');
                            break
                        case 'super admin':
                            admin();
                            res.locals.myannounce=adminann;
                            res.redirect('admin');
                            break
```

Register Course

```
exports.register=function(req,res,next){
    res.locals.user=req.session.user.userName;
    const courseid =req.params.id;
    const coursename =req.params.name;
    const studentid= req.session.user._id;
    const studentname = req.session.user.userName;

    const enrollreq = new EnrollReq({
        courseid: courseid,
        coursename:coursename,
        studentid: studentid,
        studentname:studentname
    })
    enrollreq
        .save()
        .then(result => {
            console.log('Request Sent!');
            res.redirect(req.get('referer'));
        })
        .catch(
            err => {
                console.log(err);
            }
        )
};
```

Get my GP teams

```
exports.getmyteams= async function(req, res, next) {
    res.locals.user=req.session.user;
    var option=Array;
    await Settings.find({option : 'Delete My GP Team/Doctor'})
    .then(setting => {
        option=setting;
    })
    .catch(err => console.log(err));
    await Gp.find({doctorname:req.session.user.userName})
        .then(requestgp => {
                res.render('gp/myteams', {
                    list: requestgp,
                    role:req.session.user.userRole,
                    status:option[0].status
                });
            })
            .catch(err =>{ console.log(err);
        });
};
```

Send GP requests

```
exports.addrequest=function(req, res,next) {
    const teamleader =req.session.user.userName;
    const ideatitle= req.body.ideatitle;
    const ideadesc=req.body.ideadesc;
    const doctorname=req.body.doctorname;
    const status = "Pending";
    const chatID='pending';
    const teammembers = req.body.teammembers;
    const number = req.body.number;

    const gpreq= new Gp({
        teamleader: teamleader,
        ideatitle: ideatitle,
        ideadesc: ideadesc,
        doctorname:doctorname,
        status:status,
        chatID:chatID,
        teammembers:teammembers,
        number:number,
    })
    stdnt=req.session.user._id;
    User.findOne({userName:doctorname})
        .then(instrobj =>{
            if (instrobj){
               return  dctr=instrobj._id;
            }});
    gpreq
        .save()
        .then(result => {
            res.redirect('/gp/list');
        })
        .catch(
            err => {
                console.log(err);}
        )
```

Accept/Reject GP teams

```
exports.rejectreq=function(req,res,next){
    Gp.findById(req.params.id, (err, doc) => {
        if (!err) {
            Gp.updateOne(
                {
                    _id: req.params.id,
                },
                {
                    $set: { 'status': 'Rejected'} },
                function(err, count) {
                    if (err) return next(err);      aliaayasser, 15 days ago • admin View
            });
            res.redirect('/gp/list');
        }
        else { console.log('Error in Rejecting the request :' + err); }
    });
}
```

Clear all Registered Courses

```
exports.clearenroll= async function (req,res,next){
    var backup=Array;
    await Enroll.find()
    .then(enrolled => {
        backup=enrolled;
    })
    .catch(err => console.log(err));

    await Enrollbackup.insertMany(backup)
    .then(enrolled => {
    })
    .catch(err => console.log(err));

    await Enroll.remove()
        .then(enrolled => {
            res.redirect('../settings/settings');
        })
        .catch(err => console.log(err));
};
```

Get my Courses

```
    //courses
    var mycourses=Array;
    var ids=[];
    await Enroll.find({studentid : req.session.user._id})
        .then(mycourse => {
            mycourses=mycourse;
        })
        .catch(err => console.log(err));

        for(i=0;i<mycourses.length;i++)
        {
            ids.push(mycourses[i].courseid);
        }

    await Course.find({
            '_id': { $in: ids}
        }, function(err, docs){
            res.locals.courses = docs;
            console.log(res.locals.courses);
        });
```

Add Storyboard

```
exports.getAddstoryboard = (req, res, next) => {
    res.locals.user=req.session.user;
    const idd = req.params.id;
    res.locals.user=req.session.user;
    Storyboard.find().then(storyboard=>{
        storyboards=storyboard;
    })

    Course.findById(idd).then(course => {
        if (!course) {
            return res.render('course/storyboard');
        } res.render('course/storyboard', {
            courses: course
            , pageTitle: 'Storyboard',
            path: 'course/storyboard'
        });
    }).catch(
        err => console.log(err)
    );
};


exports.postAddstoryboard = (req, res, next) => {

    const Courseid = req.body.Courseid;
    const CardContent = req.body.CardContent;
    const status = "To Do";

    const storyboard = new Storyboard({
        Courseid: Courseid,
        CardContent: CardContent,
        status: status
    });
    storyboard.save().then(result => {
        console.log('storyboard added');
        res.redirect(req.get('referer'));
    }).catch(err => {
        console.log(err);
    });
};
```

Storyboard cards status

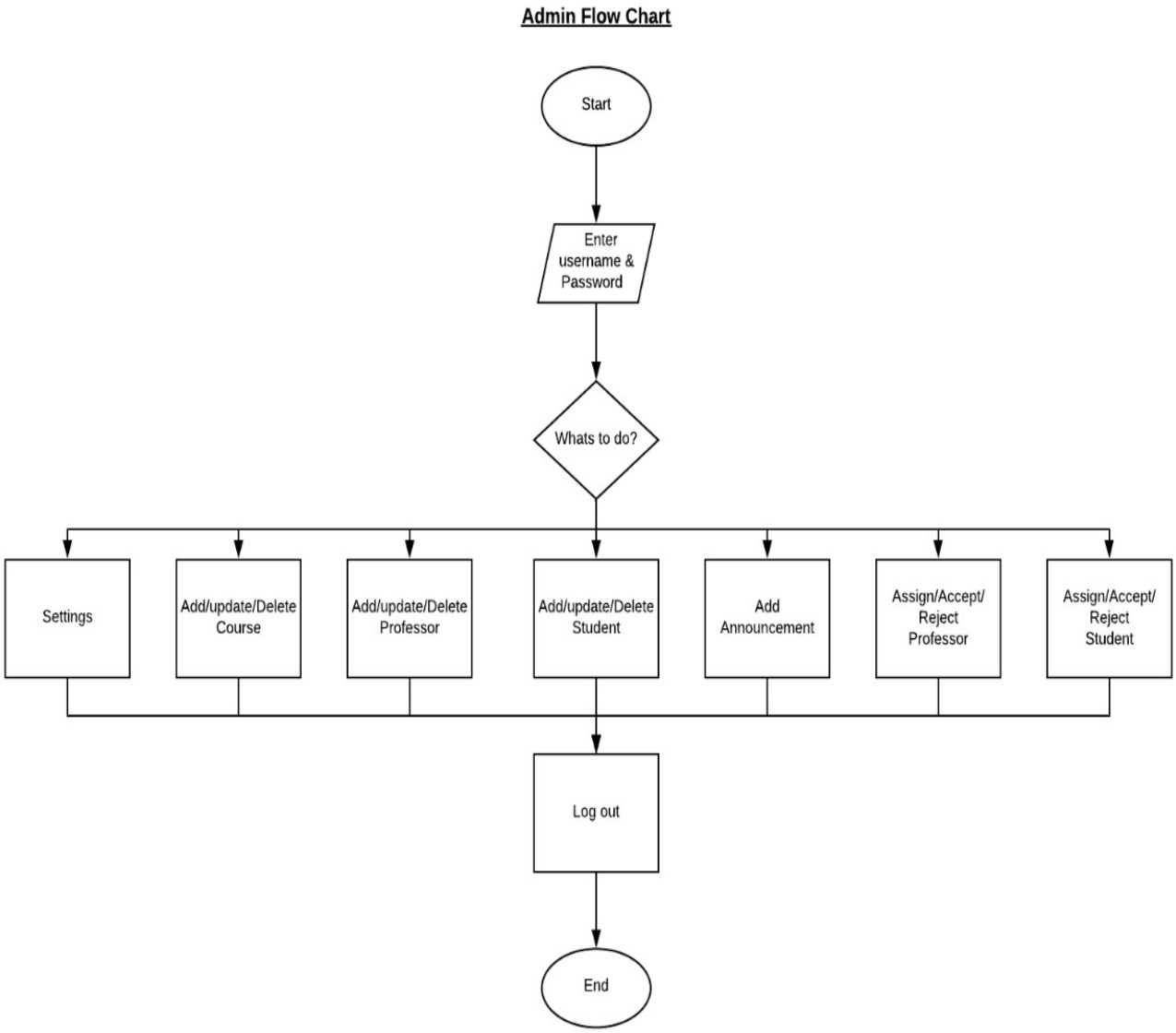
```
exports.donereq=function(req,res,next){
    Storyboard.findById(req.params.id, (err, doc) => {
        if (!err) {
            Storyboard.updateOne(
                {
                    _id: req.params.id,
                },
                {
                    $set: { 'status': 'Done'} },
                function(err, count) {
                    if (err) return next(err);
            });
            res.redirect(req.get('referer'));
        }
        else { console.log('Error in Done the request :' + err); }
    });
};

exports.onprogressreq=function(req,res,next){
    Storyboard.findById(req.params.id, (err, doc) => {
        if (!err) {
            Storyboard.updateOne(
                {
                    _id: req.params.id,
                },
                {
                    $set: { 'status': 'On Progress'} },
                function(err, count) {
                    if (err) return next(err);
            });
            res.redirect(req.get('referer'));
        }
        else { console.log('Error in On Progress the request :' + err); }
    });
};

exports.todoreq=function(req,res,next){
    Storyboard.findById(req.params.id, (err, doc) => {
        if (!err) {
            Storyboard.updateOne(
                {
                    _id: req.params.id,
                },
                {
                    $set: { 'status': 'To Do'} },
                function(err, count) {
                    if (err) return next(err);
            });
            res.redirect(req.get('referer'));
```

Add project

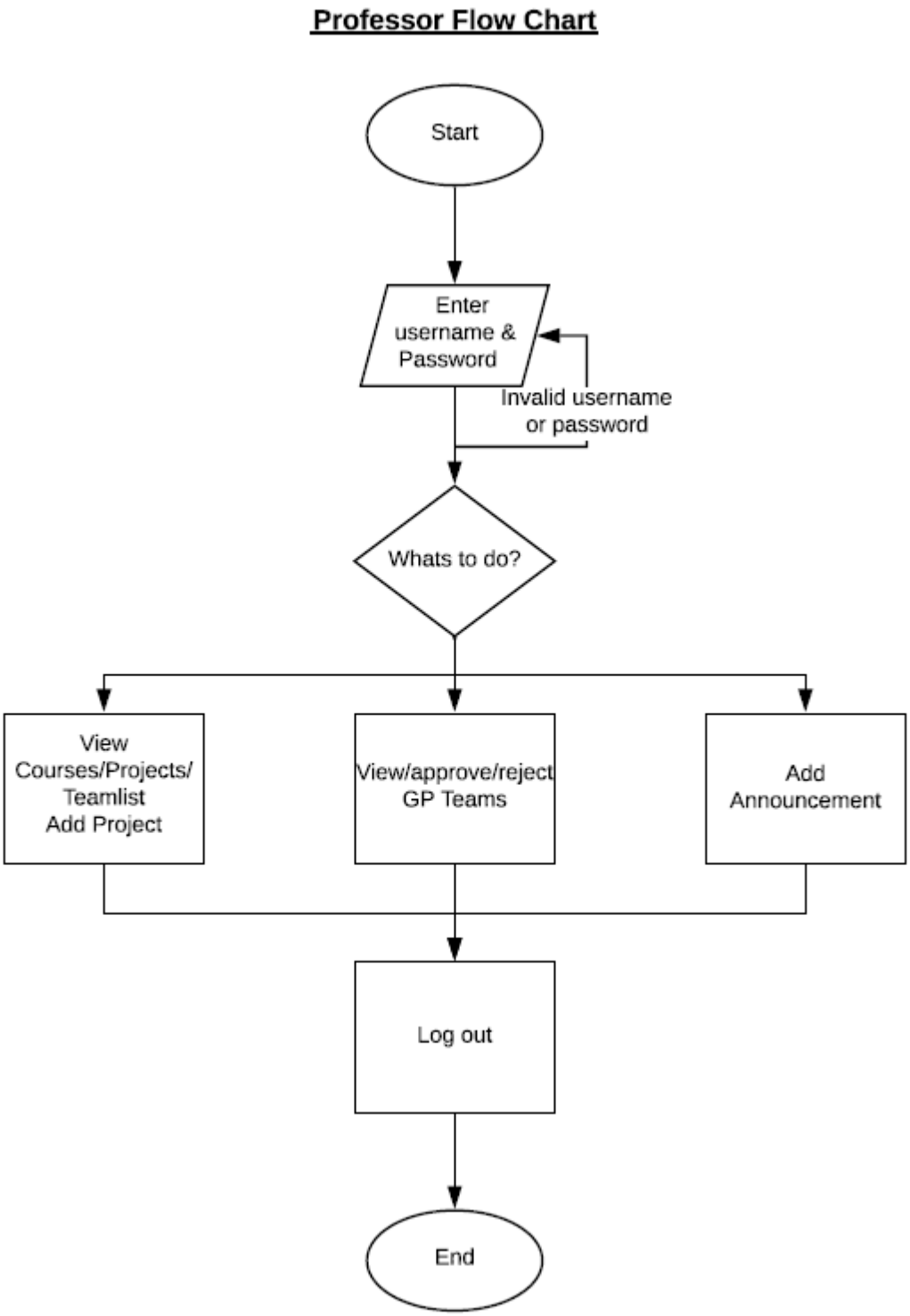
```
exports.postAddproject=(req,res,next)=>{
const CourseName=req.body.CourseName;
const projectName=req.body.projectName;      aliaayasser, 5 months ago • assign
const projectDescription=req.body.projectDescription;
const NumberOfStudent= req.body.NumberOfStudent;

const project=new Project(
    {
        CourseName:CourseName,
        projectName:projectName,
        projectDescription:projectDescription,
        NumberOfStudent:NumberOfStudent
    }
);
project.save().then(
    result=>{
        console.log('project added');
        res.redirect(req.get('referer'));
    }).catch(err=>{
    console.log(err);
});
};
```

4.2 Flowchart

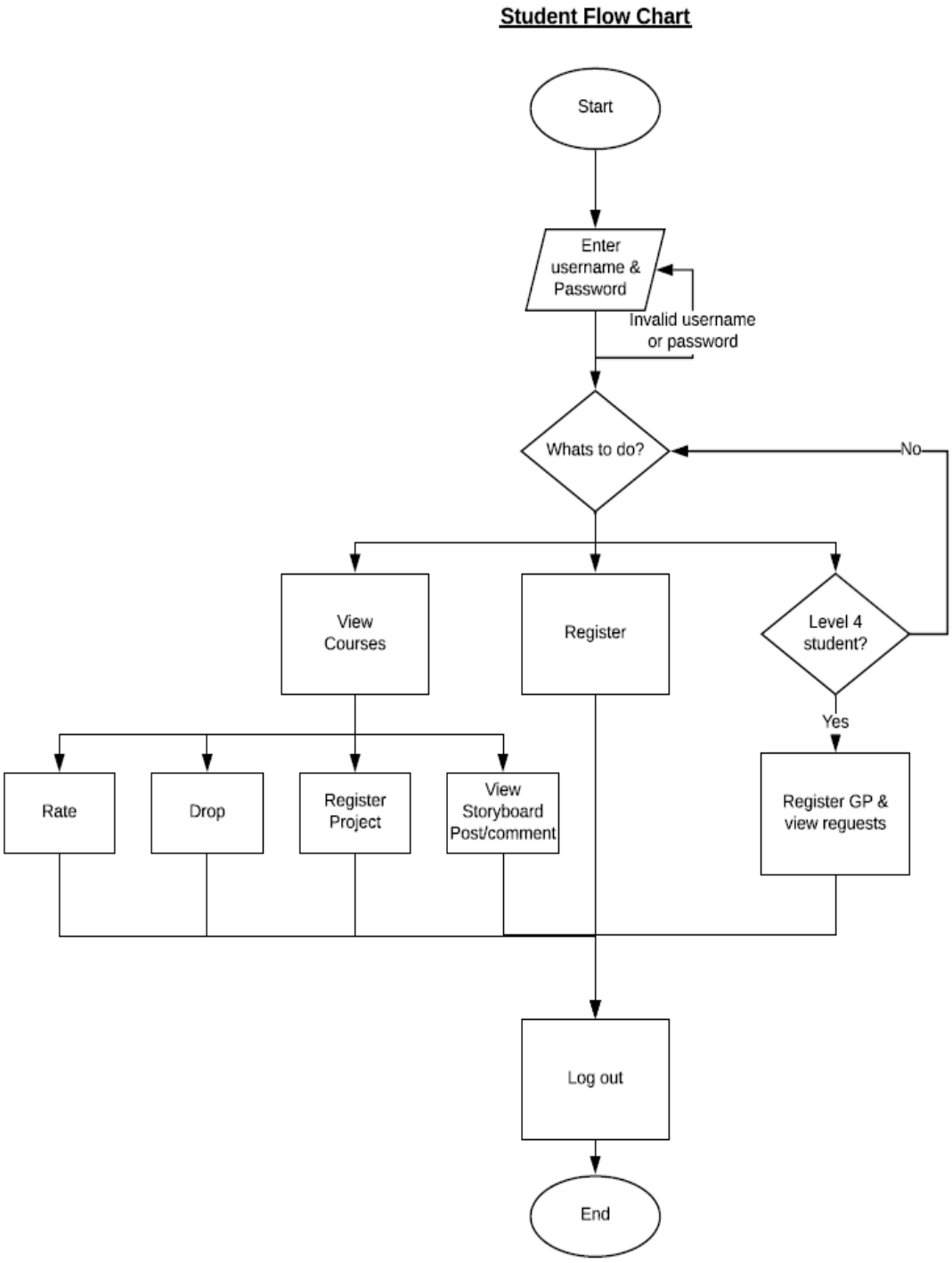
Admin



Instructor


Professor Flow Chart
Start
Enter username & Password
Invalid username or password
Whats to do?
View Courses/Projects/ Teamlist Add Project
View/approve/reject GP Teams
Add Announcement
Log out
End

Student


Student Flow Chart
Start
Enter username & Password
Invalid username or password
Whats to do?
No
View Courses
Register
Level 4 student?
Yes
Rate
Drop
Register Project
View Storyboard Post/comment
Register GP & view reguests
Log out
End

Chapter Five

In this chapter we're going to discuss and go deeper in I-Learn system testing, and present the types of testing to be used and test cases we examined our application through.

Chapter Headlines:

1. Unit Testing
2. Integrated Testing
3. Additional Testing

Here's a diagram that represent types of functional and non-functional testing and will discuss some of them in details.

5.1 Functional Testing

1. **Unit Testing**
   Testing of individual items (e.g. modules, programs, objects, classes, etc.)Usually as part of the coding phase, in isolation from other development item sand the system as a whole.

2. **Integration testing**
   Testing the interfaces between major (e.g. systems level application modules) and minor (e.g. individual programs or components) items within an application or system which must interact with each other.

3. **System testing**
   Testing a system behavior as a whole when development is finished and the system can be tested as complete entity.

4. **Regression Testing**
   To check older functionality after integrating new functionality.

5. **Acceptance testing**
   Testing to ensure that a development is ready to be deployed into the business, operational or production environment.

5.2 Non-Functional Testing

1. **Performance Testing**

   Accomplished a designated function regarding processing time and through put rate.

2. **Load Testing**

   Measuring the behavior of within creasing load which can be handled by the component or system.

3. **Stress Testing**

   Evaluate a system or component at or beyond the limits of its specified requirements.

4. **Security Testing**

   Testing how well the system protects against unauthorized internal or external access.

5.3 Test Cases

Test scenario objective

Verify **login** for a User with valid username and password

Assumptions:

Valid username: 2020001

Valid password: p@ssW0rd

Title: tc_ilearn_login001

Steps	Description	expected result	Actual result	Error type
1	Navigate to the URL (localhost:3000/login)	User navigates to login page		
2	Type valid username 2020001 in username field	User enters username correctly		
3	Type valid password p@ssW0rd	User enters password correctly		
4	Click submit	Login succeeds and user navigates to home page according to the user role.	Passed	

Test case scenario objective

Verify a student can submit a **Graduation Project form** containing idea, description and sending request to an instructor he chose

Assumptions

Student is logged in

Title: tc_ilearn_gpforrm_001

Steps	Description	Expected result	Actual result	Error Type
1	Student navigates to the Seniors button then form button	Student is navigated to the GP form		
2	Student chooses an instructor from the drop down list	List of instructors appears and he's able to choose one		
3	Student types his idea title	Student can type his idea title		
4	Student types the description	Student can add description		
5	Student enter his team members	Student can add team members names		
6	Student clicks on send request button	The request is sent successfully	passed	
7	Student clicks on Seniors button.	Student can view his sent requests	passed	

Test case scenario objective

Verify a student can cancel a sent request to an instructor

Assumptions

User is logged in and submitted a previous form

Title: tc_gpform_002

Steps	Description	Expected results	Actual results	Error type
1	User navigates to seniors' button	Student can see his sent requests		
2	Click on delete request button	The request is deleted	passed	

Test case objective scenario

Verify than an instructor actually receives the request and can accept a request

Assumptions/dependencies

Instructor is logged in

A student made a request for that instructor

Tile: tc_gpform_003

steps	Description	Expected results	Actual results	Error type
1	Instructor navigates to seniors button and check requests list	The instructor should find the list of requests sent to him		
2	Instructor clicks on accept button	Instructor can accept the request		
3	Instructor navigates to senior button and check my team list	The teams which are accepted are listed	passed	

Test case objective

Verify an instructor can decline (delete) a request

Assumptions/dependencies

Instructor is logged in

A student made a request for that instructor

Tile: tc_gpform_004

Steps	Description	Expected results	Actual results	Error type
1	Instructor navigates to seniors	Instructor views list of requests		
2	Instructor clicks on delete button	This request is deleted and disappears from the list	passed	

Test case objective

Verify appearing of the notification list when clicking on bill icon

Assumptions/dependencies

Student is logged in

An instructor send a notification for students

Tile: tc_ilearn_notificationlist_001

Steps	Description	Expected results	Actual results	Error type
1	Click on bill icon on top right	Notification list should open	passed	

Test case objective

Verify view all my announcements when open the dashboard.

Assumptions/dependencies

Student is logged in

An instructor send an announcement for students

Tile: tc_ilearn_announcementlist_001

Steps	Description	Expected results	Actual results	Error type
1	Click on home button on top left	Announcements should be listed in the home page.	passed	

Test case objective

Verify navigate to course registration page when clicking on register

Assumptions/dependencies

Student is logged in

Tile: tc_ilearn_register_001

Steps	Description	Expected results	Actual results	Error type
1	Click on my profile button on top right	Student can see his courses list		
2	Click on register button on right	Registration page should open	passed	

Test case objective

Verify appearing of “Registration is not available now!” when it’s not the time to register courses.

Assumptions/dependencies

Student is logged in

Tile: tc_register_002

Steps	Description	Expected results	Actual results	Error type
1	Click on my profile button on top right	Student can see his courses list		
2	Click on register button on right	Registration page should open with a msg that registration is closed now.	passed	

Test case objective

Verify on sending request to register courses available

Assumptions/dependencies

Student is logged in

It’s time of registration

Courses available is viewed

Tile: tc_registee_003

Steps	Description	Expected results	Actual results	Error type
1	Click on Send request of the course you need to register	Request should be sent and the course disappeared	passed	

Test case objective

Verify dropping the course selected when the student click “Drop course”.

Assumptions/dependencies

Student is logged in

Student has registered course

Tile: tc_ilearn_dropcourse_001

Steps	Description	Expected results	Actual results	Error type
1	Click on my profile button on top right	Student can see his courses list		
2	Click on course card that he want to drop	Storyboard page should open		
3	Click on drop button at top	The course selected should disappear	passed	

Test case objective

Verify navigate to rate courses page when clicking on “rate courses” and complete rating process.

Assumptions/dependencies

Student is logged in

Student has registered course

Courses page opens and views all registered courses

Tile: tc_ilearn_ratecourse_001

Steps	Description	Expected results	Actual results	Error type
1	Click on “Rate Courses”	The website should navigate to rating page		

2	Select rating from the drop box	The rate should be selected		
3	Click on "Submit"	The rate must be submitted	passed	

Test case objective

Verify navigate to GP registration page when clicking on "GP Form".

Assumptions/dependencies

Senior student is logged in

Tile: tc_ilearn_gpform_005

Steps	Description	Expected results	Actual results	Error type
1	Click on seniors on left then form	GP registration page should open	**Passed**	

Test scenario objective

Verify viewing courses when open home page and view courses when one of them is chosen

Assumptions:

Logged in an instructor/student account

Title: tc_ilearn_courseslist_001

Steps	Description	expected result	Actual result	Error type
1	Go to home page	Courses of instructor should appear in courses list		
2	Select and click on one of the courses	Navigate course storyboard	passed	

Test scenario objective

Verify navigate to students registered in the project.

Assumptions:

Logged in an instructor account

Open the file view of a specific course

Title: tc_ilearn_registerlist_001

Steps	Description	expected result	Actual result	Error type
1	Click on course needed, then click on list students at the end of project info list.	Project register list page should open and a table with registered teams' information should appear.	passed	

Test scenario objective

Verify navigating to storyboard when select any course

Assumptions:

Logged in an instructor account

Open the file view of a specific course

Title: tc_ilearn_storyboard_001

Steps	Description	expected result	Actual result	Error type
1	Click MIS course card.	Storyboard of the course should be opened		
2	Mark a task that it's "on progress"	The task must appear in "on progress" list.	Passed	

3	Mark a task that it's "Done"	The task must appear in "Completed" list.	Passed	
4	Mark a task to be on "Tasks to do"	The task must appear in "Tasks to do" list.	Passed	
5	Click on "add a task"	A pop-up window will appear		
6	Fill out the name of the new card then click "add a card"	The card must be added in "Tasks to do"	Passed	

Test scenario objective

Verify navigating to Add project form when select it from storyboard options.

Assumptions:

Logged in an instructor account

Open the view of a specific course

Title: tc_ilearn_projectform_001

Steps	Description	expected result	Actual result	Error type
1	Click Add project button.	Add a project form should be opened		
2	Fill out the name of the new project, number of students and description then click "Add"	The project must appear in Project list of this course inside the storyboard.	passed	

Test scenario objective

Verify viewing projects information when select any course.

Assumptions:

Logged in an instructor account

Open the file view of a specific course

Title: tc_ilearn_projectform_002

Steps	Description	expected result	Actual result	Error type
1	Click MIS course.	Storyboard will be opened with a project list listed all available projects.		

Test scenario objective

Verify navigating to Posts page when select it from storyboard options and add post/comment

Assumptions:

Logged in an instructor/student account

Open the file view of a specific course

Title: tc_ilearn_post_001

Steps	Description	expected result	Actual result	Error type
1	Click “Posts” button	Posts page should be opened		
2	Write a post then click on “Post”	The post must appear posted in the page.	passed	
3	Write a comment then click on “Submit”	The comment must appear down the post commented at	passed	

Test scenario objective

Verify adding a week when clicking on “Add a week” from Options

Assumptions:

Logged in an instructor account

Open the file view of a specific course

Title: tc_ilearn_addweeks_001

Steps	Description	expected result	Actual result	Error type
1	Click on “Add a week” from Option	Text label should be appeared		
2	Fill out the label with the week number, then click add week	Another week should appear.	passed	

Test scenario objective

Verify sending announcement when clicking on “send an announcement”

Assumptions:

Logged in an instructor/admin account

Open the file view of a specific course

Navigated to courses list

Title: tc_ilearn_sendingannouncement_001

Steps	Description	expected result	result	Error type
1	Click on “send an announcement”	Pop-up window should open		
2	Add title, description and the course.	Announcement must be posted	passed	

Chapter Six

In this chapter we're going to find out the results of the project whether they're achieved or not and also the differences between the desired results and the actual ones.

Chapter Headlines:

1. The Final Results
2. Discussion

6.1 Final Results

1. After Login as an admin, the system show all the options that admin could do (courses management, users management, settings, departments ...etc.).

2. After Login as an Instructor, system will show all the option that could do (view all his courses, add storyboards, projects, resources, get GP requests, Accept/reject GP).

3. After Login as student, will be ready to see all available courses to be registered, register needed courses, drop courses, view tutorials grades, view and download materials …etc.).

4. This system shows the suitable data for each logged user according to his identity and role.

5. Each important activity is managed by admin or super admin.

6.2 Discussion

We managed to get the same expected result except for making students register courses by codes as we thought that it would be more useful in the next improvements for the project.

Chapter Seven

In this chapter we're going to write our recommendations that will be added to the system in the future, also will write the project conclusion clearly.

Chapter Headlines:

1. Conclusion

7.1 Conclusion

Eventually, in this project we tried to make the best to avoid need for expensive LMS system and facilitate the management and communication with all people who seek to learn in an efficient way, it's just a way of expressing our duty towards our society as we felt responsible enough to participate in making this community a better place with one of the most powerful tools we can use and we chose as our studying field years ago which is TECHNOLOGY.

In summary, our society would be more convenient if we use what we have learned to help our society, brothers and sisters to have a good education with a modern tools. We should teach everyone to use all his knowledge to help build his community.

Finally, realizing the fact that "a few clicks" can do a lot to a society, will definitely make us more aware of the technology role and become better versions of ourselves.

Chapter Eight

In this chapter we're going to write our recommendations that will be added to our system in the future, also will write the project conclusion clearly.

Chapter Headlines:

1. Recommendation for the Future

8.1 Recommendation for Future

We are going to add more and more features like (Grading reports, Virtual Classes, GP catalogue, etc…) to our System and try to get a sponsor company for the financial and technical support for more enhancements.

We are seeking for joining competitions to spread our project idea that it support all the government schools or colleges with low budget.

So, we won't miss any chance to keep working as a team on that project and enhancing it after graduation as we're looking forward to turning this system into a start-up, where we start a company with this team members.

We have a dream, and we will reach it INSHALLAH with the power of team working and passion.

I-Learn

“Where You Discover Your Best”

Thank You!