كلية الحاسبات والذكاء الإصطناعي
Faculty of Computers & Artificial Intelligence

جامعة حلوان
HELWAN UNIVERSITY

Law

GET AND DISCOVER

[Encyclopedia Of Law]

Graduation Project 2022

Faculty of Computers and Artificial Intelligence
Information Systems Department

Supervised by:

Dr. Mohamed Marie

Team Members

1-Aya Hamed Zaki Haredi
2-Doaa Adel Mohamed Hassan
3-Tabark Mohamed ElSayed Ali
4-Salsapeel Ezzeldin Abdel Kader Sayed
5-Alaa Adel Mohamed Bakry
6-Hager Esam Khedr Mohamed

ACKNOWLEDGEMENT

First and foremost, we would like to express our great gratitude and immense joy at having the opportunity to work with such a dedicated person as **Dr. Mohamed Marie** for her support, outstanding guidance, and encouragement throughout our graduation project.

We would also like to thank our families, especially our parents, for their encouragement, patience, and assistance over the years. We are forever indebted to our parents, who have always kept us in their prayers and pushed us to Success.

Finally, for our faculty for providing the suitable environment that leaded us to represent the best image that computer science graduates of Helwan University are supposed to represent.

THANK YOU ALL

Table of Contents

ACKNOWLEDGEMENT 3
1. Introduction 6
1.1 Overview 7
1.2 Objectives 7
1.3 Project Purpose 8
1.4 Project Scope 8
1.4.1 Planning 8
1.4.2 Designing 9
1.4.3 Coding 9
1.4.4 Testing 10
1.4.5 Documentation 10
1.5 General Constraints 11
2. Project planning and analysis 12
2.1 Project Planning 13
2.1.1 Feasibility study 13
2.1.3 Gannt chart 15
2.2 Analysis and Limitation of existing system 19
2.3 Need for the new system 19
2.4 Analysis of a new systems 20
2.4.1 User Requirements 20
2.4.2 System Requirements 20
2.4.3 Domain Requirements 21
2.4.4 Functional Requirements 21
6. Result and Discussion 84
6.1 Results 85
6.1.1 Expected Result 85
6.1.2 Actual Result 85
6.2 Discussion 86
Bibliography 90
“If you do not like to read you have not found the right book.” 91

Abstract

Because of the spread of the Internet, people are searching for information easily, but it is difficult to find more accurate information especially if this is related about the law. Therefore, our project revolves around users who are primarily interested in the law, so this site tries to communicate with users to view the subjects he wants and it is displayed to him and he can download the low and also another feature has been added that he can search for lawyers to consult them legally and communicate with them, he also rated the lawyer, and this evaluation is very useful to users in order to help them choose a lawyer.

This project also targets students who have graduated from law and wanted to prove themselves and train with famous lawyers.

Thus, different types of users may benefit from the site: users, graduate students, and advertising for lawyers.

Chapter 1: Introduction

1. Introduction

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.

1.1 Overview

On this site, I am happy to say that we have contained all the laws and provided it to users. It is a site where the user can search through the laws and find the subjects, he is interested in. The site will help him find the best lawyers for different specialties, legal advice, book appointments and communicate with them, and helps lawyers in Advertisement about their specialty

1.2 Objectives

Our main objectives are:

- Our team aims to develop a high quality and fast performance website that will serve as an encyclopedia of laws which will provide the text of the original authoritative law In order to serve many public users and lawyers and to get their satisfaction by the end of July 2022.

- Provide the system users with an easy and speed search way to find the specific law subjects that he/she is looking for by providing our system with a real law database, the search results must display in a few seconds and must be accurate and correct in 100% for credibility.

- Our staff aims to adding all new upgrades and improvements that have been added to current law's subject's version during 2021, Which will be identical and modified according to the latest legal amendments in the country to provide a simplified and easy service for system users

- We aim to provide our users with professional lawyers to help them in their personal or public cases and give them response for there requests within 24 hours to have 100% user satisfaction

- The user can evaluate or rate and express his opinion or make review about the lawyer who resolved his case, and we can measure the extent of customer satisfaction with the lawyers through this evaluation and the comments given, also using this service, clients who are looking for a lawyer for a case can choose him through this evaluation or rate and read the comments.

- The admin can delete the user who abuses the system or its users, and we can identify this user through the violations he does, if the user receives three warning messages, he/she will be erased/deleted from the system in order to create a system free of problems, bad speech and destructive opinions.

1.3 Project Purpose

- Our project aims to facilitate communication between civilians and lawyers.
- it's an encyclopedia of law, which makes it easier for the user to view the contents of the law easily and download a book of law.
- The user can also consult or appoint a lawyer by sending a request to the lawyer, And the lawyer may respond with an appointment time or decline the request.
- Users can also rate the lawyer and make a review to help other users choose an experienced lawyer.

1.4 Project Scope

The project collects all the content in books about the law which is consuming to find a specific point in it we make it easier, the user can his work for advertising and connect to the customers, that will help find a job, we offer a user to select the right lawyer for his case from multiple choice on the website.

The approximate work involved to finish the project is divided into these five phases: -

1.4.1 Planning

- Collecting datasets about the project and the lack that made us in a need to the App.
- Searching for easy way to collect data in organized way that can be updated easily.
- Asking lawyers about law and understanding the structure of law.
- Determining the functional and non-functional requirements.
- Setting a Gantt chart for the project.

- Determining the resources of the team.
- Prepare a database to move data into it .
- Dividing the tasks on team using Trello.

1.4.2 Designing

- Designing recommendations algorithm:
  1. Reading data from pdf files.
  2. Understand the structure of law.
  3. Filtering data.
  4. Designing database collected from filtering data
  5. Normalize the database
- Determining the diagrams to be carried out within the project: -
  - ✓ Use-case Scenario
  - ✓ Activity Diagram
  - ✓ Class Diagram
  - ✓ Use-case Diagram
  - ✓ Sequence Diagram

1.4.3 Coding

The main supposed functions to be coded in this App are: -

- Sign up.
- Login.
- Logout.
- Search For Subjects
- Display
- Download
- View Subject
- Rate Lawyer
- View Profile
- Edit Profile
- Add Subject

- Edit Subject
- Send Request
- Receive Notifications
- View Lawyer profile
- View Schedule
- Edit Date
- Search For Lawyer
- Help

1.4.4 Testing

- Functional Testing: -
  - Unit testing
  - Regression testing
  - Integration testing
- Non-functional Testing: -
  - Performance testing
  - Street testing
  - Security testing

1.4.5 Documentation

Should mainly include these main chapters: -

- Introduction: includes an overview of the project and limitations.
- Project Planning: includes the tools and technologies as well as tasks and timeline plan.
- Project Requirements: includes the functional and nonfunctional requirements.
- Project Design: includes the diagrams.
- Project Implementation: includes user application and administrator system.
- Project Testing: includes testing types.

1.5 General Constraints

1.5.1-Time Constraints

- Collect data about laws.
- Understand the laws.
- Determine the relationships between them.

- The structure of laws varies from one law to another, which leads us to build many classes and the relationships between them are very complex.
- Indiscipline "Human factor" like being late in delivering tasks or attending meetings.
- Time management.

1.5.2 Software Constraints:

- make rules and conventions commonly agreed to in each programming environment explicit and automatically checkable.
- In frontend, we use ASP.NETCOR.MVC, Java Script, Razor, CSS, Bootstrap, C# .
- In Backend, we use Microsoft Sql Server.

1.5.3-Hardware constraints

- they enable run our application on web Application.

Chapter 2: Project Planning and Analysis

2. Project planning and analysis

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 application

2.1 Project Planning

2.1.1 Feasibility study

A Feasibility study is used to determine if a business or a specific project is achievable, so for determining the achievability of our project we will go deeper in the following points: -

1. Market Analysis

- We can make money from our website using ADS.
- Encyclopedia Of Law helps users to discover the law.
- Encyclopedia helps users search for the exact subjects we want and display their details.
- Encyclopedia helps users in download law book.
- Encyclopedia helps users search for lawyer in different specialization.
- Encyclopedia helps users evaluate the lawyers who contacted them in order to express their opinions.
- Encyclopedia helps users to connect with lawyers and take appointment.
- The user can also be a graduate student looking for lawyers to give her the opportunity to train for her.

2. Technical Feasibility

- The hardware and software for developing, maintaining, and launching the website will be reliable and available for us. The website will not be the need for training users also. The system will be maintainable in the future and easy to connect with external systems and work on many different platform.

3. Operational Feasibility

- In In the old systems, it was difficult for the user to interact with the system, and it was difficult in the search process to access the information he wanted
- Also, the old systems did not include searching for lawyers
- This system was created based on the user's need for this legal information in his life
- A solution has been proposed for users who are looking for lawyers in different specialties and communicate with them, and the user also evaluates the lawyer to help others
- The system will be easy to use for all users and does not need to train the user to use it, but through a small video he will discover the program, and there is also a help center for the user if he encounters any problem.
- The system will be isolated, meaning that no user can see any information that was important to another user.
- The software can provide the information that the user needs to complete his tasks.
- Over time, it is possible, based on user requests, to add some other features.

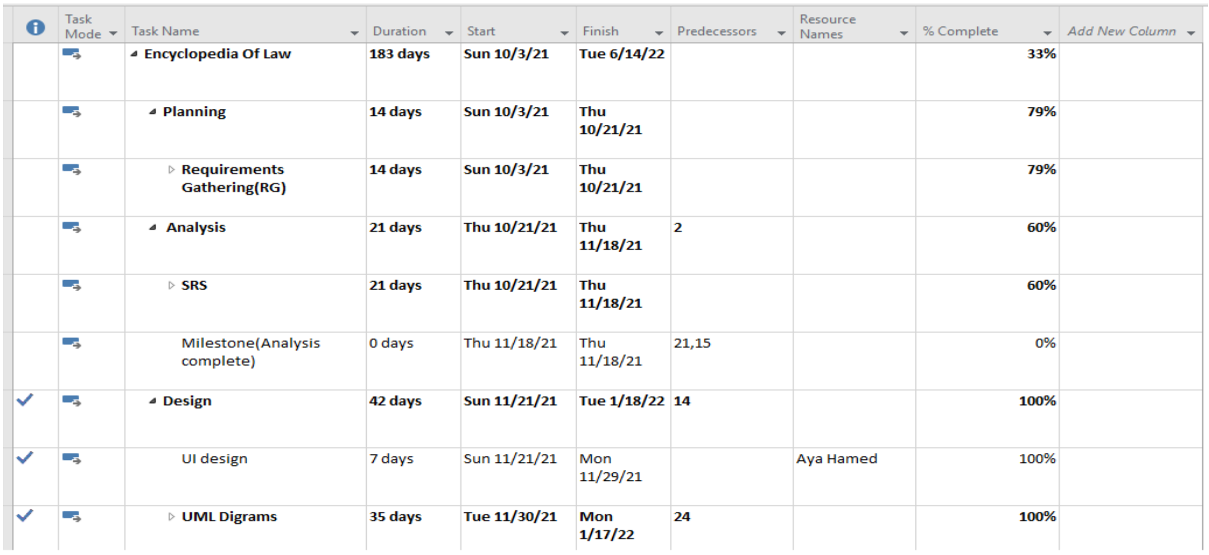
4.Schedule Feasibility

-We divided website into stages into smaller tasks with specific working duration with specific deadline

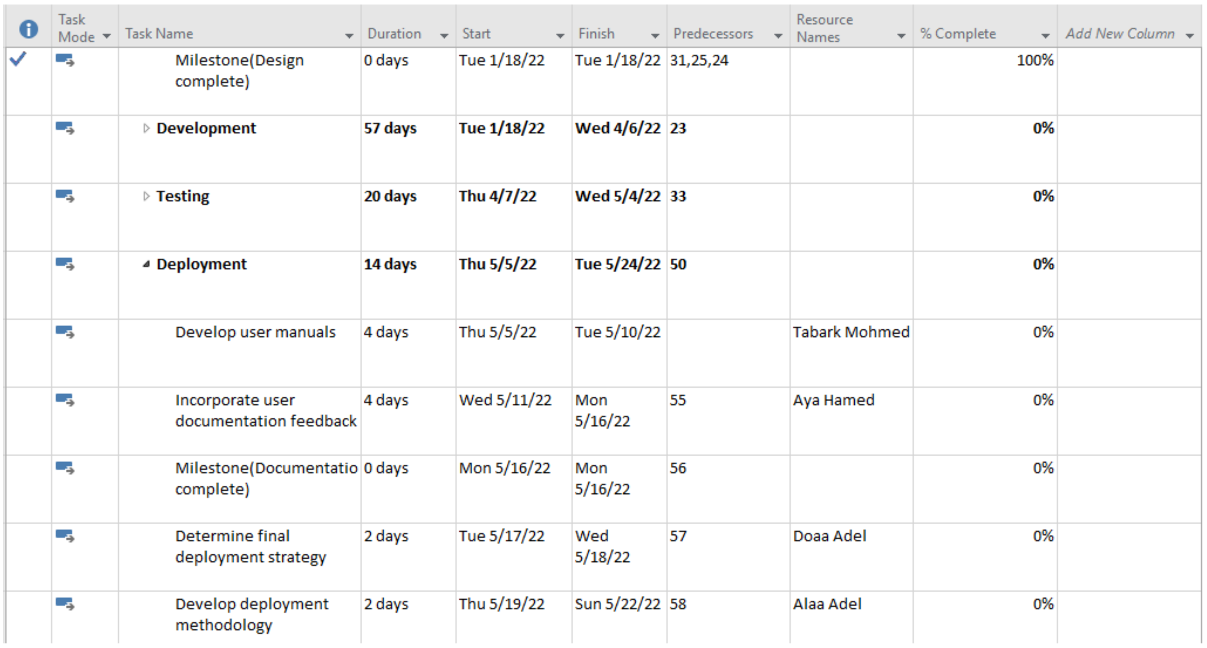
- ## 2.1.2 Estimated Cost

  -We may need some cost to buy software and dataset to help our project expand its scop. The development of any project must be organized and have a clear time plan for the entire work of project

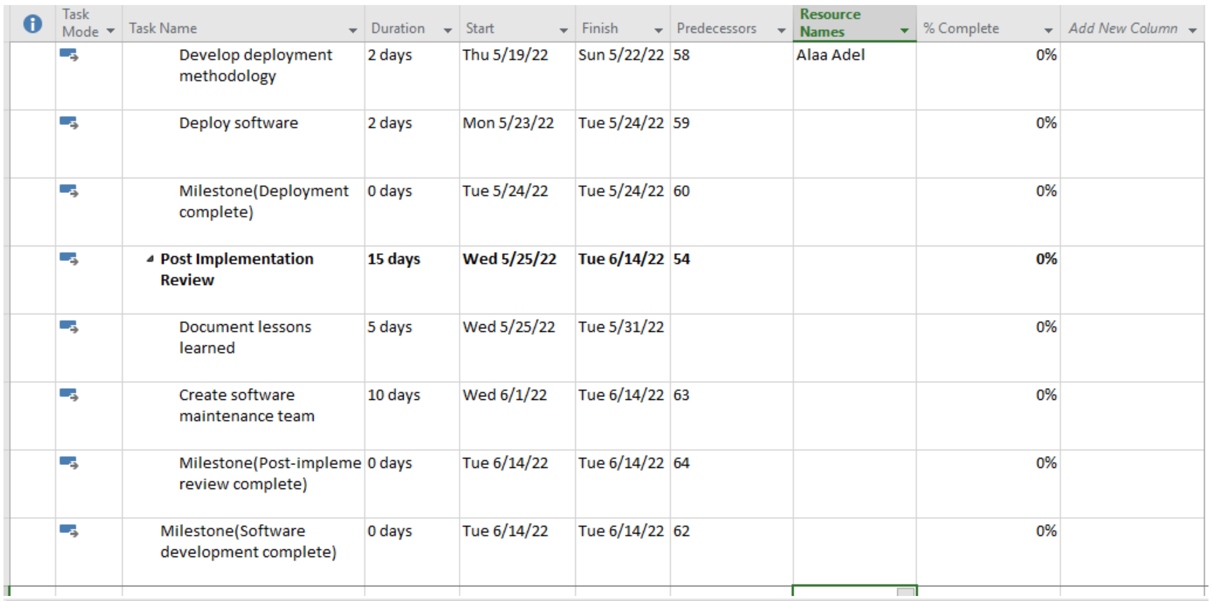
2.1.3 Gannt chart



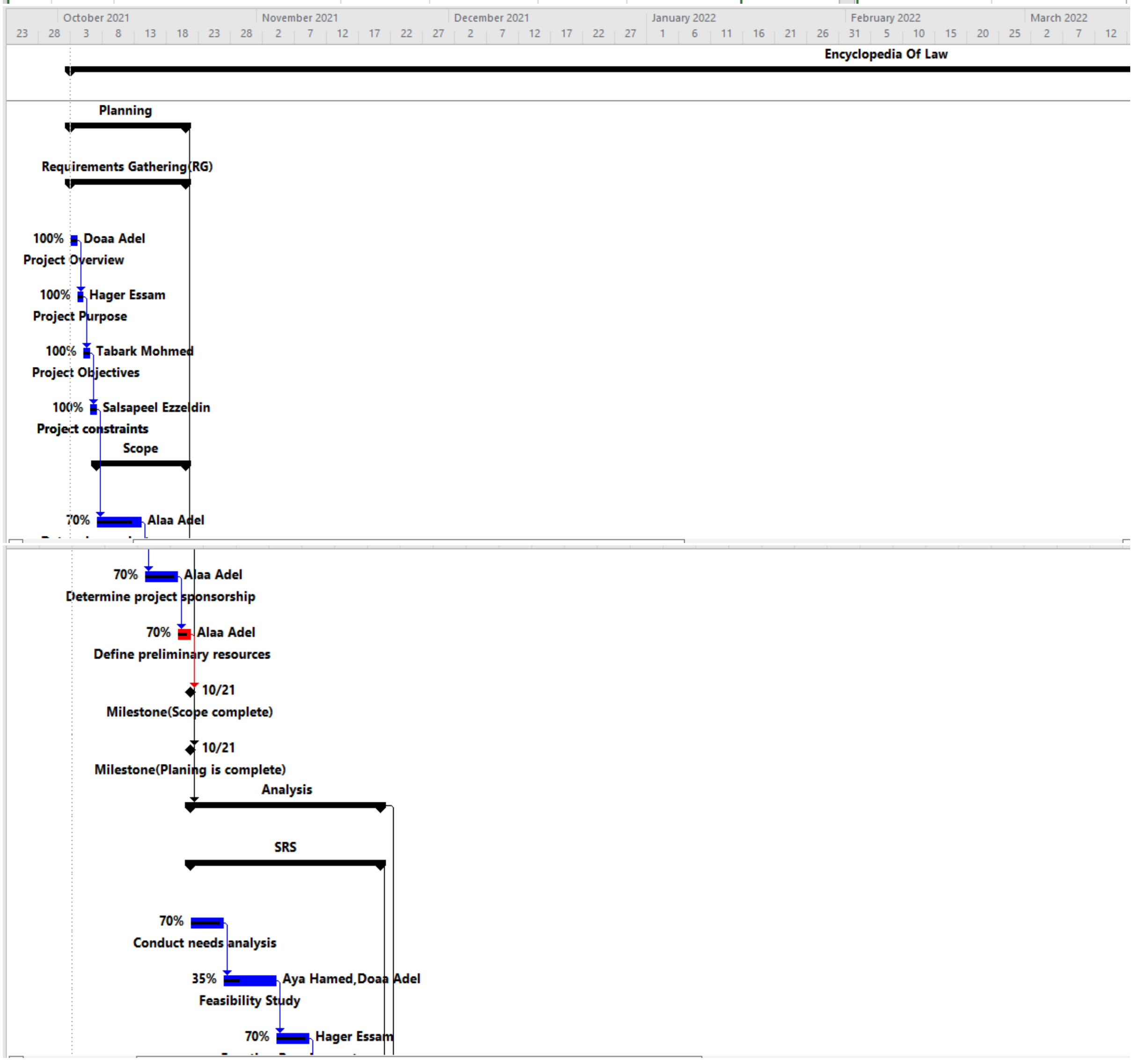
	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names	% Complete	Add New Column
		Encyclopedia Of Law	**183 days**	**Sun 10/3/21**	**Tue 6/14/22**			**33%**	
		Planning	**14 days**	**Sun 10/3/21**	**Thu 10/21/21**			**79%**	
		Requirements Gathering(RG)	**14 days**	**Sun 10/3/21**	**Thu 10/21/21**			**79%**	
		Analysis	**21 days**	**Thu 10/21/21**	**Thu 11/18/21**	**2**		**60%**	
		SRS	**21 days**	**Thu 10/21/21**	**Thu 11/18/21**			**60%**	
		Milestone(Analysis complete)	0 days	Thu 11/18/21	Thu 11/18/21	21,15		0%	
✓		**Design**	**42 days**	**Sun 11/21/21**	**Tue 1/18/22**	**14**		**100%**	
✓		UI design	7 days	Sun 11/21/21	Mon 11/29/21		Aya Hamed	100%	
✓		**UML Digrams**	**35 days**	**Tue 11/30/21**	**Mon 1/17/22**	**24**		**100%**	

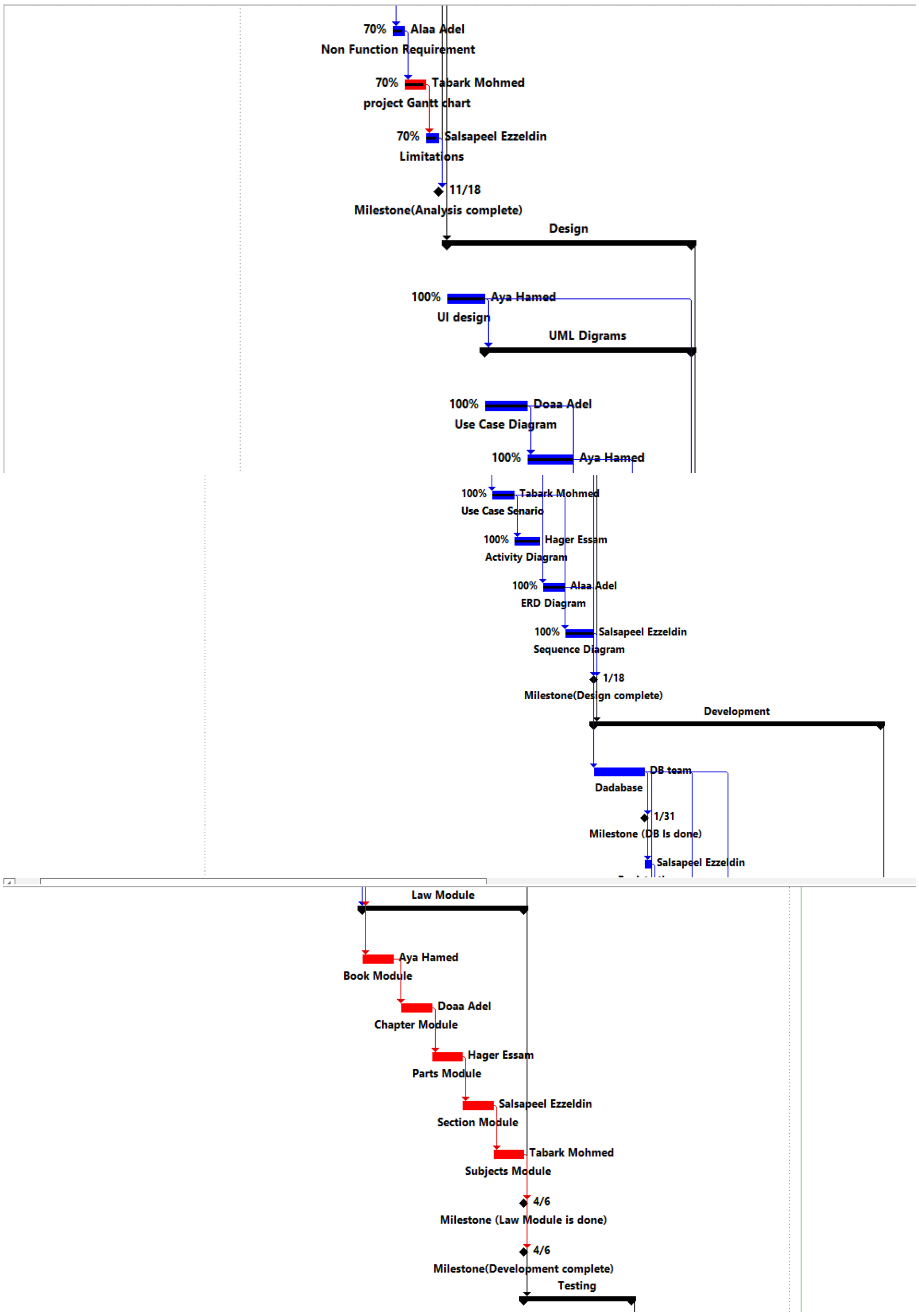


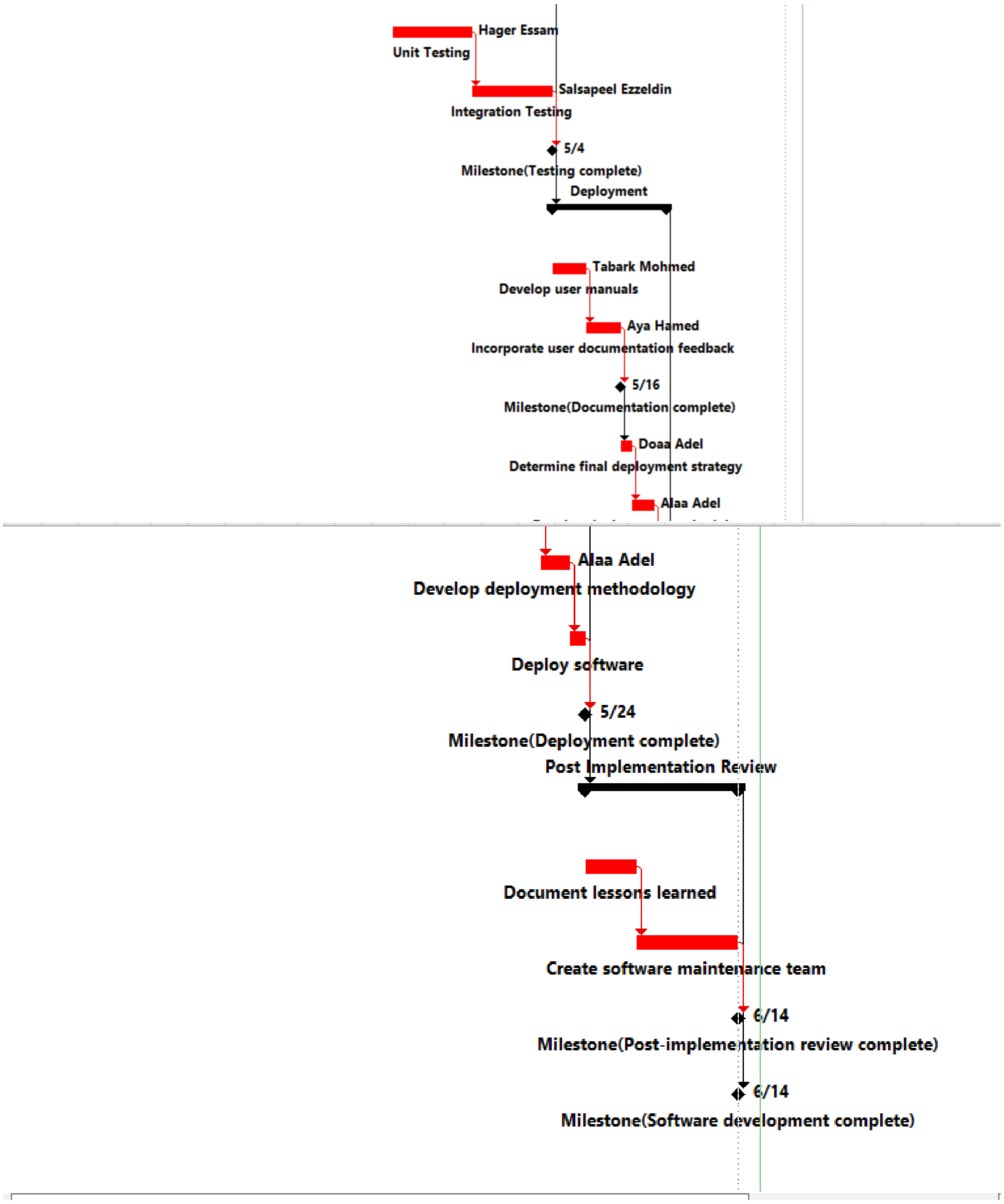
	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names	% Complete	Add New Column
✓		Milestone(Design complete)	0 days	Tue 1/18/22	Tue 1/18/22	31,25,24		100%	
		Development	**57 days**	**Tue 1/18/22**	**Wed 4/6/22**	**23**		**0%**	
		Testing	**20 days**	**Thu 4/7/22**	**Wed 5/4/22**	**33**		**0%**	
		Deployment	**14 days**	**Thu 5/5/22**	**Tue 5/24/22**	**50**		**0%**	
		Develop user manuals	4 days	Thu 5/5/22	Tue 5/10/22		Tabark Mohmed	0%	
		Incorporate user documentation feedback	4 days	Wed 5/11/22	Mon 5/16/22	55	Aya Hamed	0%	
		Milestone(Documentatio complete)	0 days	Mon 5/16/22	Mon 5/16/22	56		0%	
		Determine final deployment strategy	2 days	Tue 5/17/22	Wed 5/18/22	57	Doaa Adel	0%	
		Develop deployment methodology	2 days	Thu 5/19/22	Sun 5/22/22	58	Alaa Adel	0%	



	Task Mode	Task Name	Duration	Start	Finish	Predecessors	Resource Names	% Complete	Add New Column
		Develop deployment methodology	2 days	Thu 5/19/22	Sun 5/22/22	58	Alaa Adel	0%	
		Deploy software	2 days	Mon 5/23/22	Tue 5/24/22	59		0%	
		Milestone(Deployment complete)	0 days	Tue 5/24/22	Tue 5/24/22	60		0%	
		Post Implementation Review	**15 days**	**Wed 5/25/22**	**Tue 6/14/22**	**54**		**0%**	
		Document lessons learned	5 days	Wed 5/25/22	Tue 5/31/22			0%	
		Create software maintenance team	10 days	Wed 6/1/22	Tue 6/14/22	63		0%	
		Milestone(Post-impleme review complete)	0 days	Tue 6/14/22	Tue 6/14/22	64		0%	
		Milestone(Software development complete)	0 days	Tue 6/14/22	Tue 6/14/22	62		0%	




70% Alaa Adel
Non Function Requirement
70% Tabark Mohmed
project Gantt chart
70% Salsapeel Ezzeldin
Limitations
11/18
Milestone(Analysis complete)
Design
100% Aya Hamed
UI design
UML Digrams
100% Doaa Adel
Use Case Diagram
100% Aya Hamed
100% Tabark Mohmed
Use Case Senario
100% Hager Essam
Activity Diagram
100% Alaa Adel
ERD Diagram
100% Salsapeel Ezzeldin
Sequence Diagram
1/18
Milestone(Design complete)
Development
DB team
Dadabase
1/31
Milestone (DB Is done)
Salsapeel Ezzeldin
Law Module
Aya Hamed
Book Module
Doaa Adel
Chapter Module
Hager Essam
Parts Module
Salsapeel Ezzeldin
Section Module
Tabark Mohmed
Subjects Module
4/6
Milestone (Law Module is done)
4/6
Milestone(Development complete)
Testing


Hager Essam
Unit Testing
Salsapeel Ezzeldin
Integration Testing
5/4
Milestone(Testing complete)
Deployment
Tabark Mohmed
Develop user manuals
Aya Hamed
Incorporate user documentation feedback
5/16
Milestone(Documentation complete)
Doaa Adel
Determine final deployment strategy
Alaa Adel
Alaa Adel
Develop deployment methodology
Deploy software
5/24
Milestone(Deployment complete)
Post Implementation Review
Document lessons learned
Create software maintenance team
6/14
Milestone(Post-implementation review complete)
6/14
Milestone(Software development complete)

2.2 Analysis and Limitation of existing system

Non-Usability

There are also several other web portals created for the purpose of search in law

But it is difficult to use for a citizen who wants to access the text of a particular law subject as it expands For legislation, prosecutors, rulings, constitutions, etc

leads to error prone results:
-the systems are in-efficient to handle multiple requests at the same time and leads to error results.

not update of database:

-There are also several other web work as Law Encyclopedia But it works to display the content of one law It is not possible to add a new law.

lack of data security:

-There is no security for any of the information stored in the database

no interact

Most systems work as a book or document only to read the content of the laws and there is no interaction between the user and the system, The user cannot ask or appoint a lawyer

2.3 Need for the new system

The system was built with easy user interface so that the user and the law researcher can access the content with ease to save time and effort on the user.

-In this system, the database was created so that the staff can add any other law.

-In addition to viewing the contents of the law, the user can interact with the system, The user can also consult or appoint a lawyer by sending a request to the lawyer, And the lawyer may respond with an appointment time or decline the request.

-Users can also rate the lawyer and make a review to help other users choose an experienced lawyer.

2.4 Analysis of a new systems

2.4.1 User Requirements

1. Mandatory Requirements

- the user can connect with lawyer throw the app and rate his lawyer
- only admins can update the subjects
- Lawyer can post his work to show up in users page

2. Desirable Requirements

- The user needs to use Search Bar to make the process easier so we will provide this feature in our app.
- The user can browse the app any time so our app should be available in any place and any time with any browser.

3. Optional Requirements

- The users prefer registration with special accounts like Facebook or Gmail so we will provide this feature in our app.

2.4.2 System Requirements

1. Web Browser

You can browse the application in any web browser

2. Internet Connections

This app use Internet to browse the web and restore the data and message other users

3. Storage space

To keep the user's data in addition to the application itself, we need a fine storage.

2.4.3 Domain Requirements

1. Multiple users must be able to use the application simultaneously without corrupting the database (whatever form it may be).
2. The application must verify all values before making the change in the database.
3. The database should be backed up occasionally in case the original does become corrupt.
4. application runs in any browser
5. The application must have update capabilities for future models and accessories.

2.4.4 Functional Requirements

1. Signup

Name	Signup
Actor	Client ,lawyer
input	Fill registration form (Email, username, password, roll name ,Address, phone…etc.)
output	User will go directly to this profile page
description	user can create an account to be a part of the application, and login to the application for the second time he opens the application without the need of sign up again.

precondition	open the application, client should enter a valid data to create an account such username, email, password, and phone.
Post condition	the user should have an account and enter to home page and his profile that has its data which saved when the account created.

2. Login

name	Login
Actor	Client, admin, lawyer, staff
input	e-mail & password
output	Send the user into the home page
description	user should enter his email and his password for authorizing that he has an account
preconditions	open the application, enter login Button, and enter a valid e-mail and password.
Post conditions	Logged into the system then enter to home page then he can go to his profile that has its data which saved in the database.

3.Update profile

name	Update profile
Actor	Lawyer, Client
input	The updated information
output	Update the new data in DB
description	If user wants to change any data in his profile the app will update Them and save in the database
preconditions	There is actual account sign up or login, open his profile
Post conditions	the updated information will be displayed in the profile and saved in the firebase.

4.Logout

name	Logout
Actor	Client , lawyer ,admin, staff
input	User token in data base
output	End user session
description	After the user login to the application, they can logout easily from account.
preconditions	open the application, login, enter log out Button you have been logged in first.
Post conditions	user will log out successfully and could not use the application without doing login again

5.Download

Name	Download You must have an account and logged in successfully then go to home page and press on your books to see details
Actor	Client ,lawyer, staff
Input	Press download button on subject details page
Output	Subject is downloaded
description	The user view the subject details depend on his need and download the subject
preconditions	You must have an account and logged in successfully then go to subject details page and press on download button to download
Post conditions	The subject pdf download successfully

6.View subject

Name	View subject
Actor	Client ,Admin , lawyer, staff
Input	Press on any subject
Output	Send the user into the subject details
description	The user choose subject depend on his interests and want to see subject details to discover information about this book.
preconditions	You must have an account and logged in successfully then go to home page and press on subjects icon to see details
Post conditions	The subjects' details will display.

Search

Name	Search
Actor	Admin ,Client ,lawyer, staff
Input	Write subject's name, subject's part of the content , lawyer name
Output	List of all results related to the search
description	Help the user to search about subjects that have the same subject name, and search for the lawyer in the department that they need
preconditions	Sign up or login and go to search icon then write name in text field.
Post conditions	All the subjects or lawyer that have same name will display

7.Rete lawyer

Name	Rate lawyer
Actor	Client
Input	Choose from 1 to 5 stars
Output	Send the rate to lawyer

description	Share opinions and suggestions with each other
preconditions	Open application, you must have an account and logged in successfully then go to lawyer profile page and rate
Post conditions	Your rate sent successfully

8.Add subject

name	Add subject
Actor	Staff
input	Fill Add subject Form (Title, description, law name ,part name , etc..).
output	The data saved in DB
description	The staff wants to display subjects in home page to discover it the users.
preconditions	Open the application, login as a staff with valid email and password and enter to Add subject page.
Post conditions	The subject saved in DB and sent to view subjects page

9.Edit subject

name	Edit subject
Actor	staff
input	Fill edit subject form(Title, description, content)
output	Old subject deleted and new one saved into data base
description	staff go to subject press edit button fill the form then the new one added to database
preconditions	staff logged in and the subject already recorded in the database
Post conditions	The subject is successfully updated in a specific law

10.Delete subject

name	Delete subject
Actor	staff
input	staff press delete button in the subject details page
output	Subject removed from database
description	staff must be logged in then go to subject which will be removed the delete it
preconditions	staff logged in and the subject already recorded in the database
Post conditions	The subject is successfully removed

11.Send request

name	Send request
Actor	Client
input	Client view the lawyer profile page and press the request button
output	Request appears in lawyer page
description	Client send a request to have an appointment with a lawyer
preconditions	Client must be logged in the system
Post conditions	Request have been sent successfully

12.Cancel request

name	Cancel request
Actor	Client
input	Client go to his page and press cancel button in requests
output	Request removed from database

description	Client want to cancel a request
preconditions	Client must be logged in the system and already sent a request
Post conditions	The request cancelled successfully

name	Display profile
Actor	Admin ,lawyer ,client, staff
input	User go to his profile page and view his information
output	Profile page is displayed
description	User want to view his information
preconditions	User is logged in the system
Post conditions	Profile is displayed successfully

13.display profile

14.Edit profile

name	Edit profile
Actor	Client ,lawyer ,admin, staff
input	Edit user information
output	Profile information updated in database
description	User want to change data in his profile
preconditions	User must be logged in successfully
Post conditions	Profile updated with the new data

15.Set a date

name	Set date
Actor	Lawyer
input	Lawyer go to his page and view his clients request then press set appointment button and set a data

output	Appointment saved in database
description	Lawyer want to assign a date for a user request
preconditions	Lawyer logged in successfully and has request
Post conditions	Appointment saved successfully

16. cancel date

name	Cancel date
Actor	Lawyer
input	Lawyer go to his schedules and press in specific appointment then press cancel button
output	Date deleted from database
description	Lawyer want to cancel a date with the client
preconditions	Lawyer must be logged in and data already set
Post conditions	Date removed successfully

Edit date

name	Edit date
Actor	Lawyer
input	Lawyer go to his schedules and press in specific appointment then press edit button and set the new date
output	Date updated and send the new appointment to the client
description	Lawyer want to update his appointment with the client
preconditions	Lawyer must be logged in and data already set
Post conditions	Date updated successfully

Ask for support

name	Ask for support
Actor	Client ,lawyer
input	User go to support page and write his email and message to support stuff in the system
output	Message saved in database
description	User want to ask for a help in the system
preconditions	None
Post conditions	Message sent successfully

View user requests

name	View requests
Actor	Lawyer
input	Lawyer go to his page and view requests
output	Requests displayed
description	Lawyer want to view the requests that sent by clients
preconditions	Lawyer must be logged in
Post conditions	Requests displayed successfully

2.4.5 Nonfunctional Requirement

1. Availability:

the system should be available all the time in case of a hardware failure or database or delay a replacement page will be shown, and in case of hardware or database delay backups will be retrieved from application data folder -it means 24/7 available.

2. Flexibility:

if the team intends to increase the application functionality and extends it, that would be easy as the application is prepared from the beginning to be expanded for future.

3. Robustness:

Our application can handle error conditions gracefully, without failure. This includes a tolerance of invalid data, software defects, and unexpected operating conditions.

4. Scalability:

Our application is scalable can handle a wide variety of system configuration sizes. By adding new features and new machines.

Usability:

Our application is Ease to use requirements address the factors that constitute the capacity of the software to be understood, learned, and used by its intended users.

5. Reliability:

the system provides the right tools for problem solving it is made in such as the way that the system is reliable in its operations and for securing the sensitive details.

2.5 Advantages of the new system

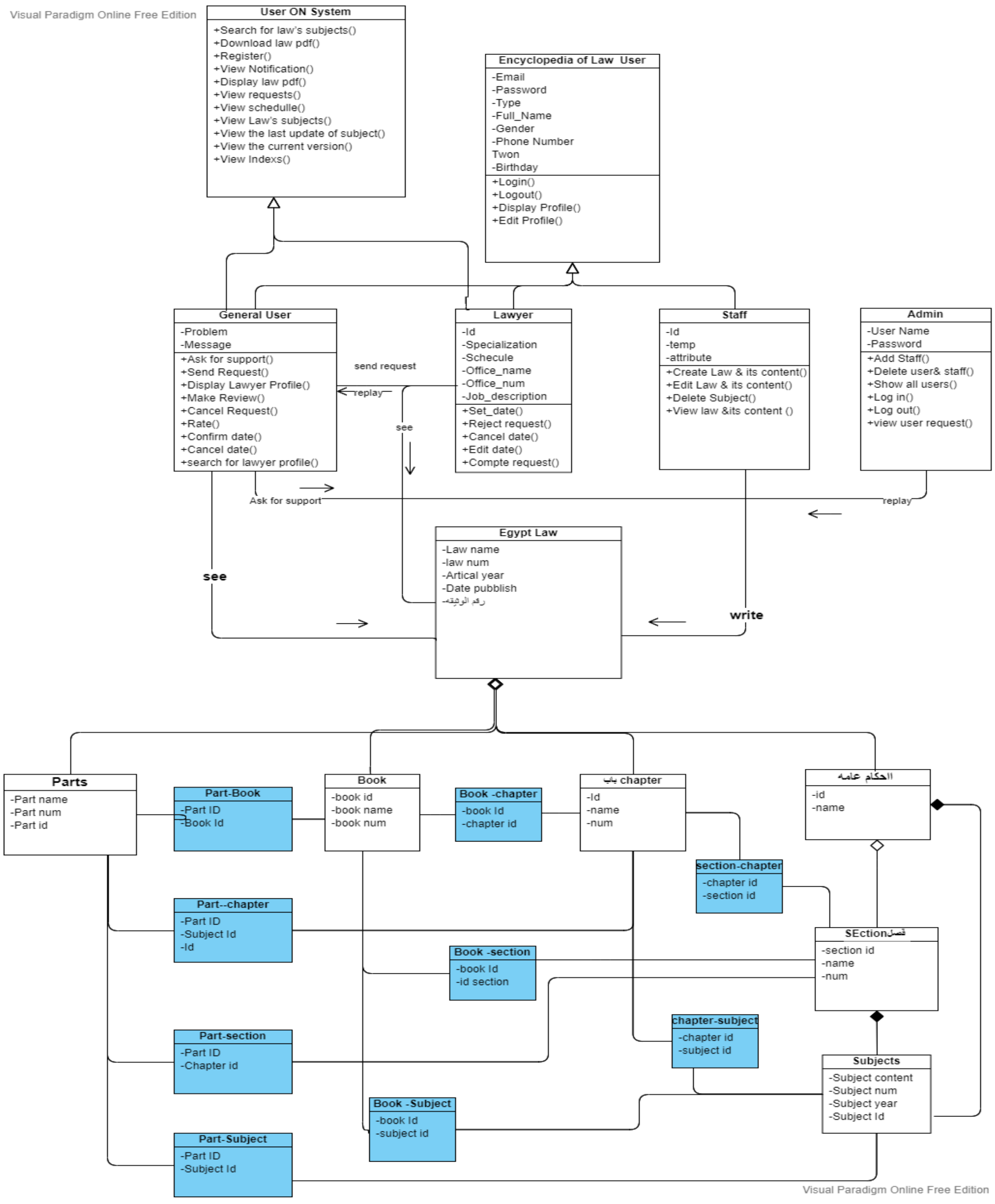
- We can make money from our website using ADS.
- Encyclopedia Of Law helps users to discover the law.
- Encyclopedia helps users search for the exact subjects we want and display their details .
- Encyclopedia helps users in download law book.

- Encyclopedia helps users search for lawyer in different specialization.
- Encyclopedia helps users evaluate the lawyers who contacted them to express their opinions.
- Encyclopedia helps users to connect with lawyers and take appointment.
- The user can also be a graduate student looking for lawyers to give her the opportunity to train for her.

Chapter 3: Design

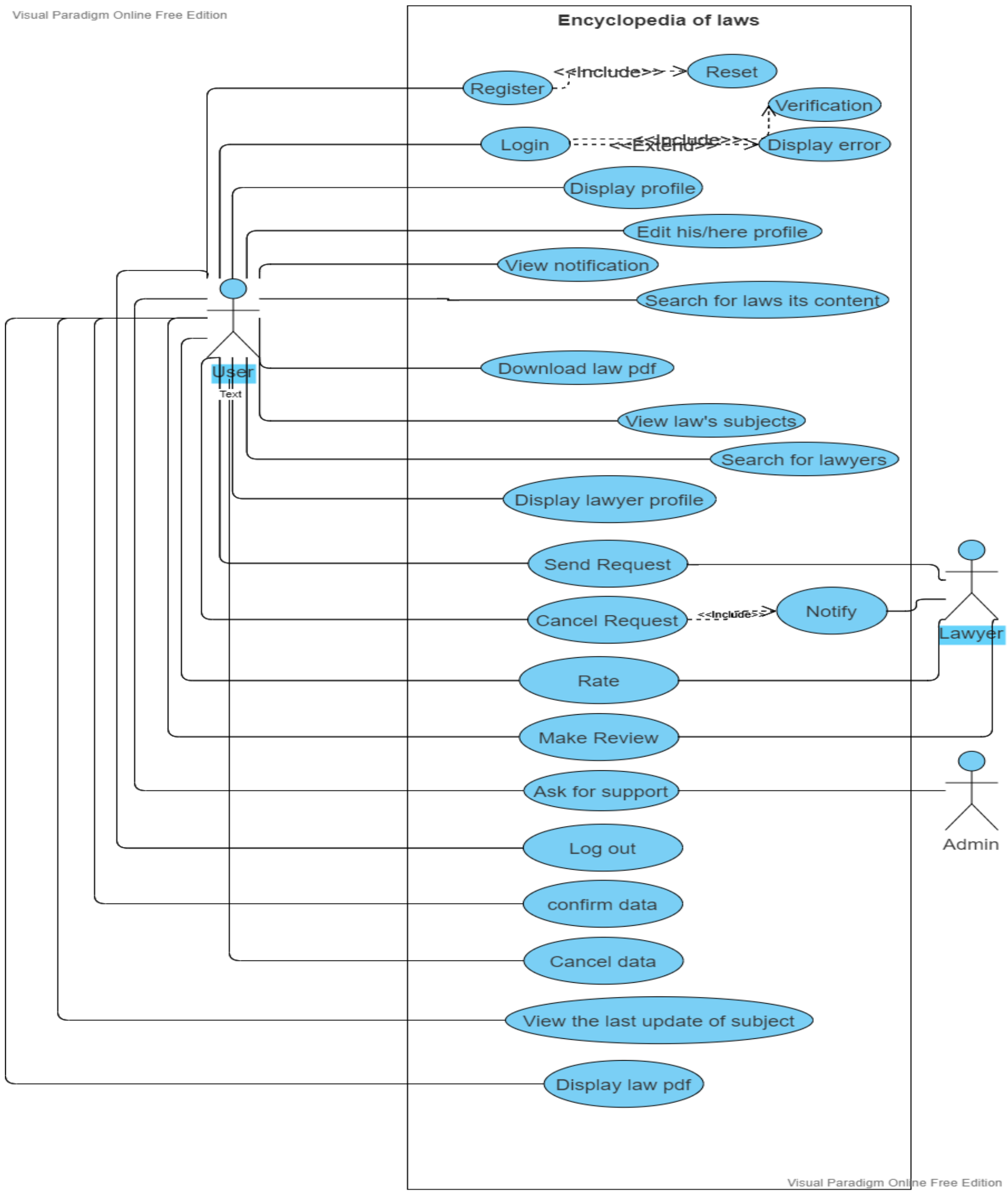
In this chapter we are going to discuss and go deeper in Settle mobile application's design and present its diagrams and database.

3.1 Class Diagram

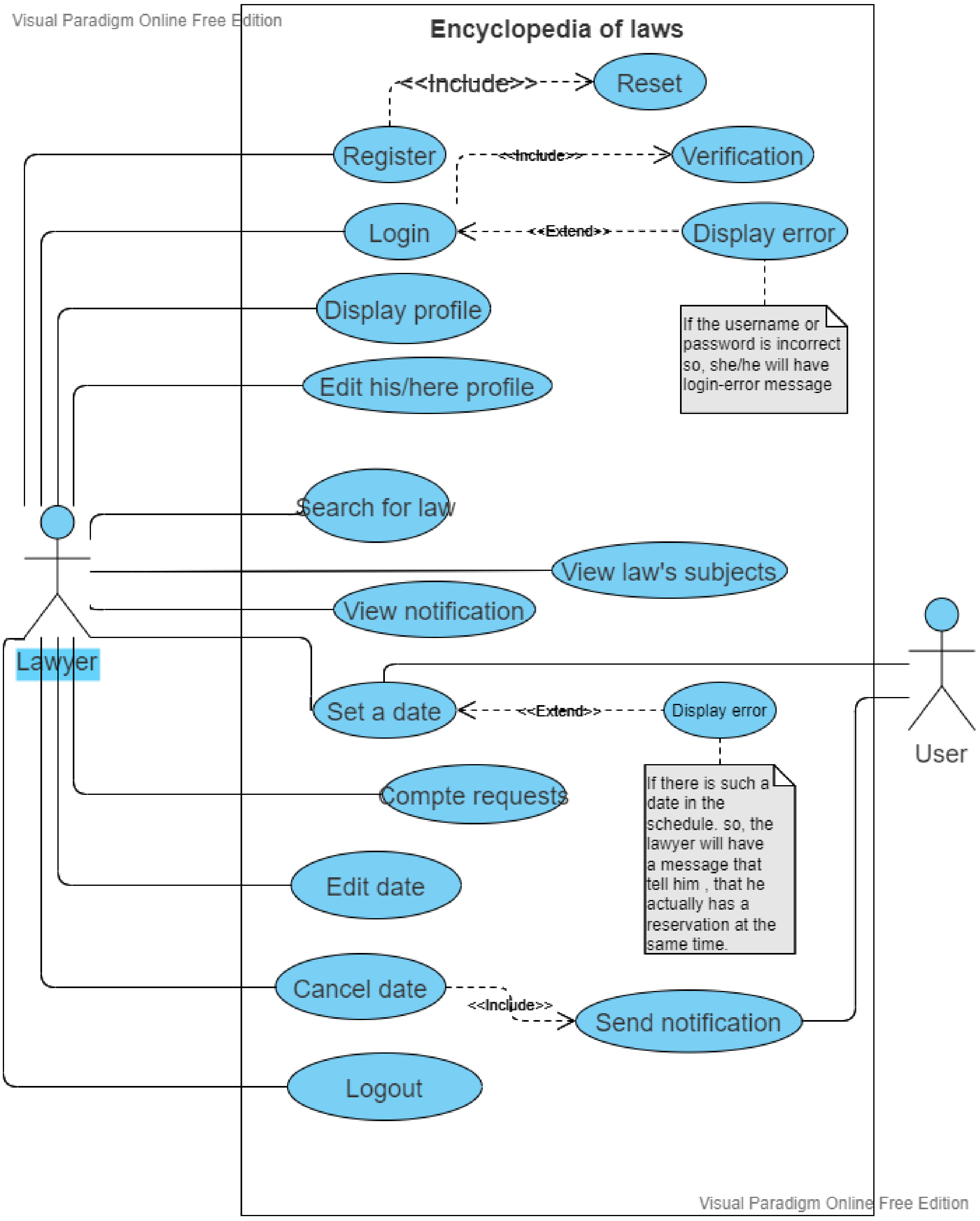


3.2 Use case Diagram

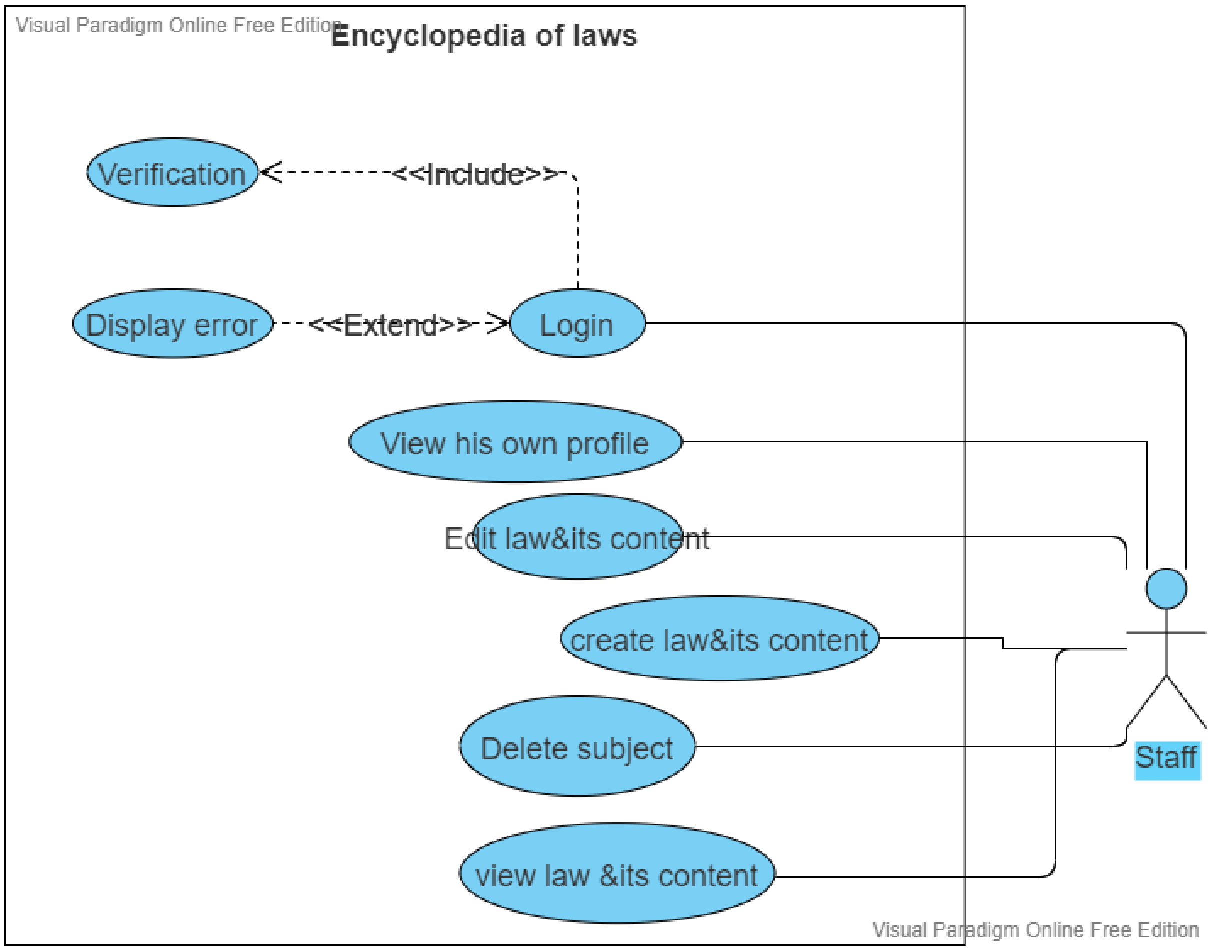
3.2.1 User Use case


Visual Paradigm Online Free Edition
Encyclopedia of laws
Register
<<Include>>
Reset
Verification
Login
<<Include>>
<<Extend>>
Display error
Display profile
Edit his/here profile
View notification
Search for laws its content
Download law pdf
View law's subjects
Search for lawyers
Display lawyer profile
Send Request
Cancel Request
<<Include>>
Notify
Rate
Make Review
Ask for support
Log out
confirm data
Cancel data
View the last update of subject
Display law pdf
User
Text
Lawyer
Admin
Visual Paradigm Online Free Edition

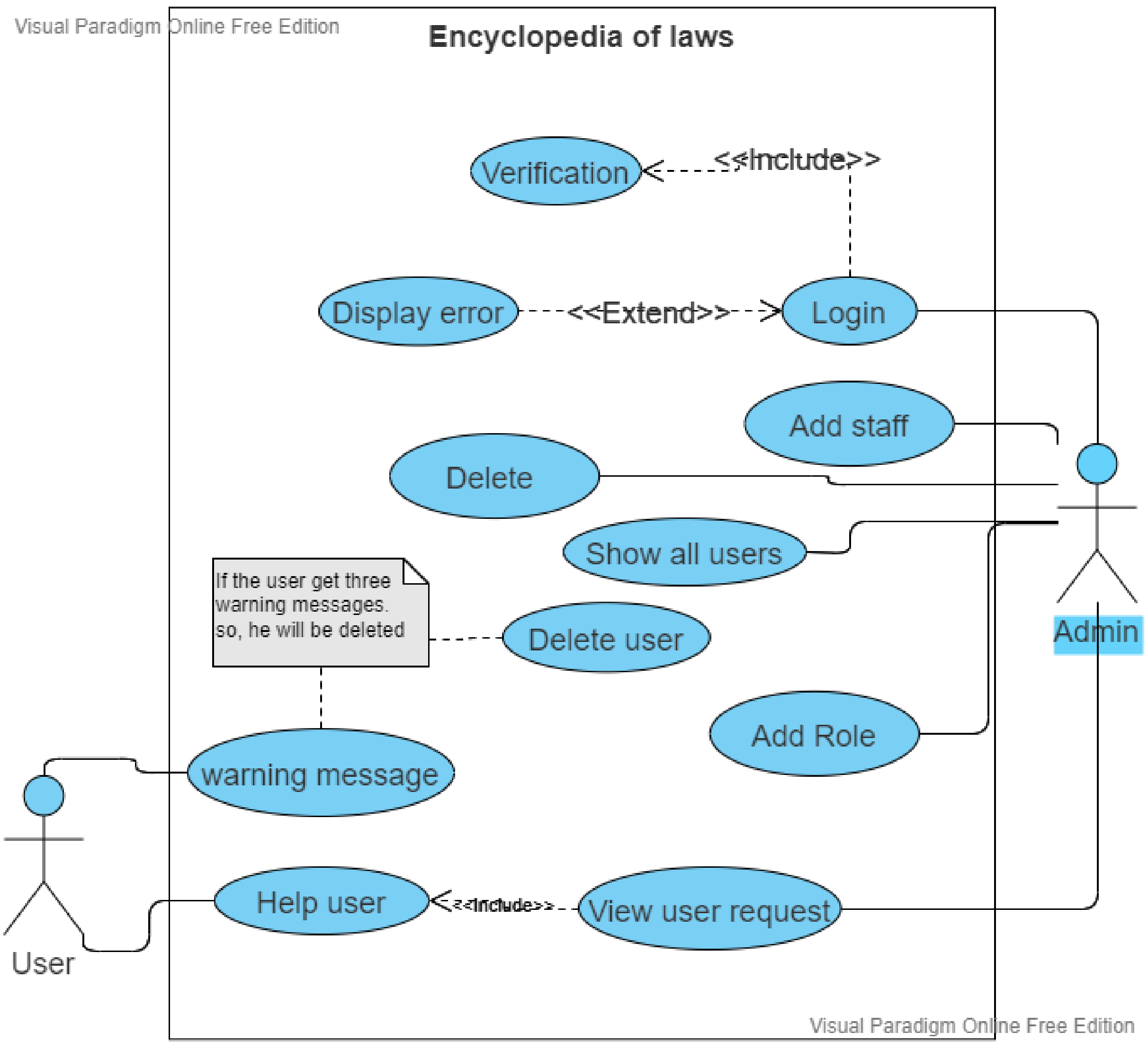
3.2.2 Lawyer use case


Visual Paradigm Online Free Edition
Encyclopedia of laws
<<Include>>
Reset
Register
<<Include>>
Verification
Login
<<Extend>>
Display error
If the username or password is incorrect so, she/he will have login-error message
Display profile
Edit his/here profile
Search for law
View law's subjects
View notification
Lawyer
Set a date
<<Extend>>
Display error
If there is such a date in the schedule. so, the lawyer will have a message that tell him , that he actually has a reservation at the same time.
User
Compte requests
Edit date
Cancel date
<<Include>>
Send notification
Logout
Visual Paradigm Online Free Edition

3.2.3 Staff use case



3.2.4 Admin use case



3.3 Use Case Scenario

ID &Use Case Name	1 Register
Initiator/Actors	The primary actor: (user) The secondary actor: (staff). The third actor :lawyer
Goal	To enter the application and use its services.
Precondition	None
Postcondition	The registration is saved in the database then he/she could login in the application and use its services.
Description of main sequence scenario	1. Any of the three actors (user, staff, lawyer) fill the registration from. 2. The registration is saved in the database record. 3. Then they can login to the system and use its services.
Description of alternative sequence	-

Table 2: for "Login" use case

ID &Use Case Name	2 Login
Initiator/Actors	The primary actor: (user) The secondary actor: (staff). The third actor :lawyer the forth actor : admin.
Goal	To enter the application and use its services.
Precondition	Any of the actors must be registered before.
Postcondition	They have an access to the system &they

	could use the application.
Description of main sequence scenario	1. Enter username and password. 2.user is verify to use the system
Description of alternative sequence	2.a the user is not allowed to use the system. 2. a .1 Message error is appear.

Table 3: for (Add new subject) use case

ID &Use Case Name	3 Add new subject
Initiator/Actors	The primary actor: staff
Goal	New subject will be added.
Precondition	There is a law recorded in the database.
Postcondition	The subject is successfully added to a specific law.
Description of main sequence scenario	1. Staff must login in the system. 2. Then he/she could select the law. 3. Staff chose a part, book , chapter or section in law if exists. 4. Add the subject.
Description of alternative sequence	-none

Table 4: for (edit subject) use case

ID &Use Case Name	4 edit subject
Initiator/Actors	The primary actor: staff
Goal	Staff need to edit an existing subject.
Precondition	The subject is already recorded in the database.
Postconditions	The subject is successfully updated in

	a specific law.
Description of main sequence scenario	1. Staff must login in the system. 2. Then he/she could select the subject. 3. Enter the subject’s new text.
Description of alternative sequence	-none

Table 5: for (display subject) use case

ID &Use Case Name	5 display subjects.
Initiator/Actors	The primary actor: user /lawyer
Goal	User can show the subjects.
Precondition	The subject is already recorded in the database.
Postconditions	The subject is successfully displayed.
Description of main sequence scenario	1. user enter the system. 2. User can select the law. 3. Then select the subject Id. 4. Then chose the display subject option.
Description of alternative sequence	-none

Table 6: for (delete subject) use case

ID &Use Case Name	6delete subjects.
Initiator/Actors	The primary actor: staff
Goal	Staff need to remove an existing subject
Precondition	The subject is already recorded in the database.

Postconditions	The subject is successfully removed.
Description of main sequence scenario	1. Staff must login in the system. 2. Then he/she could select the subject. 3. Then delete the subject.
Description of alternative sequence	-none

Table 7: for (Request Appointment) use case

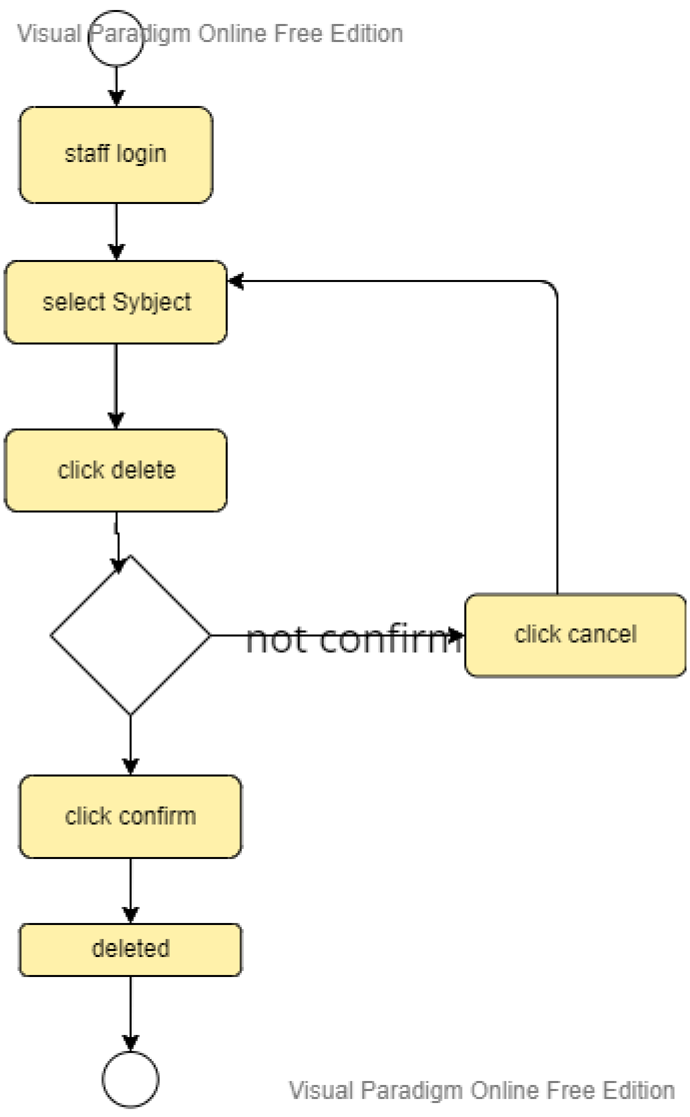
ID &Use Case Name	7 Request Appointment.
Initiator/Actors	The primary actor: user
Goal	User send request to have an appointment with a lawyer.
Precondition	User must be login in the system.
Postconditions	The request have been sent successfully.
Description of main sequence scenario	1. User must login in the system. 2. Then he/she could select any lawyer. 3. Then sent the request.
Description of alternative sequence	-none

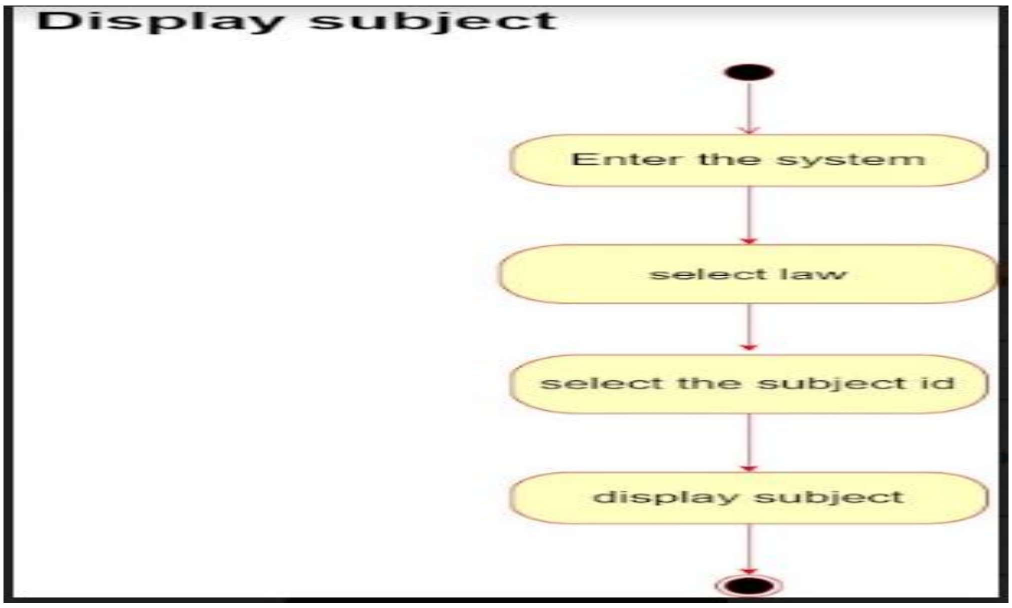
Table 8: for (assign a date) use case

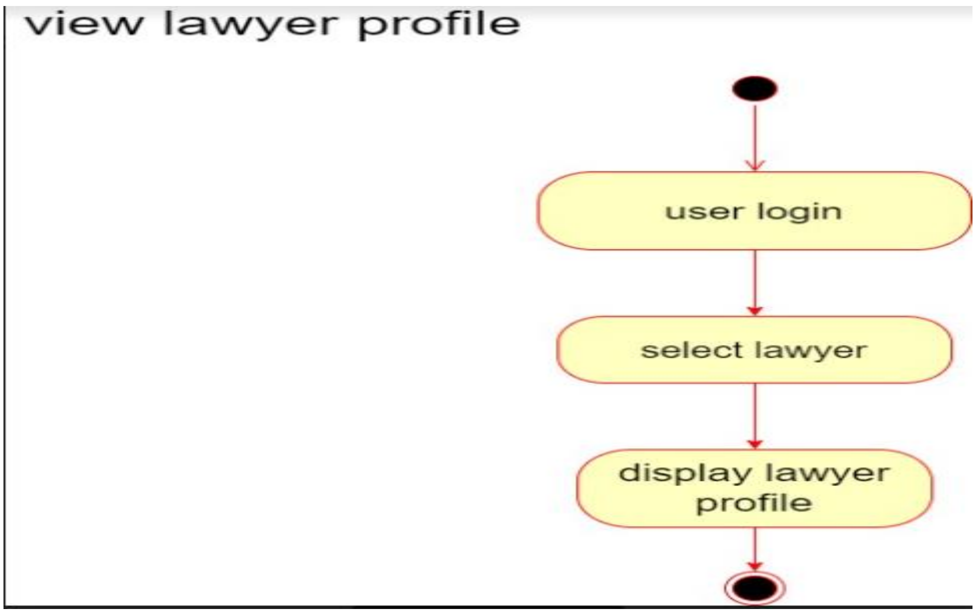
ID &Use Case Name	8 assign a date.
Initiator/Actors	The primary actor: lawyer
Goal	Lawyer assign a date for the user request.
Precondition	Lawyer Is already on the system & user

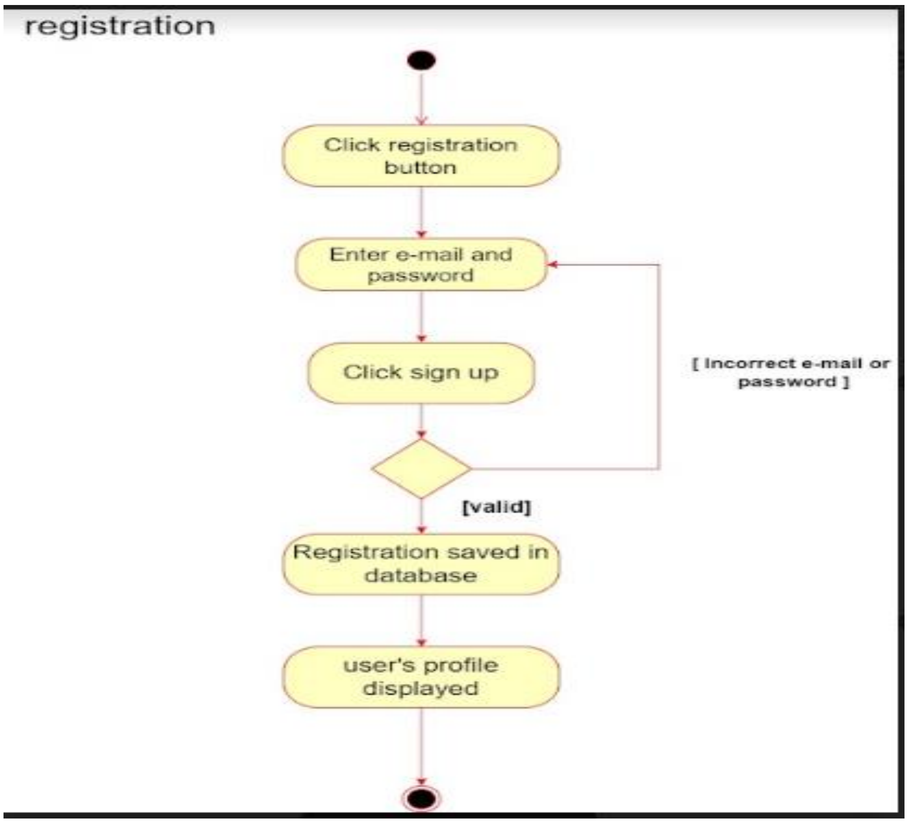
	have sent his/her request.
Postconditions	The date have been assigned successfully.
Description of main sequence scenario	1. lawyer must login in the system. 2. Then lawyer display his/her schedule (show user requests). 3. Then assign an appointment or date for user.
Description of alternative sequence	-none

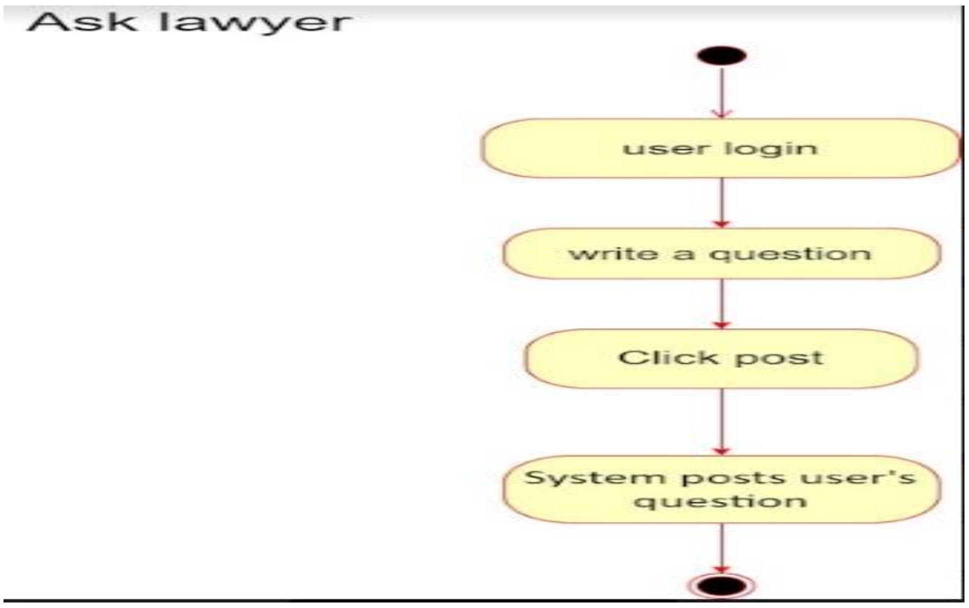
3.4 Activity Diagram

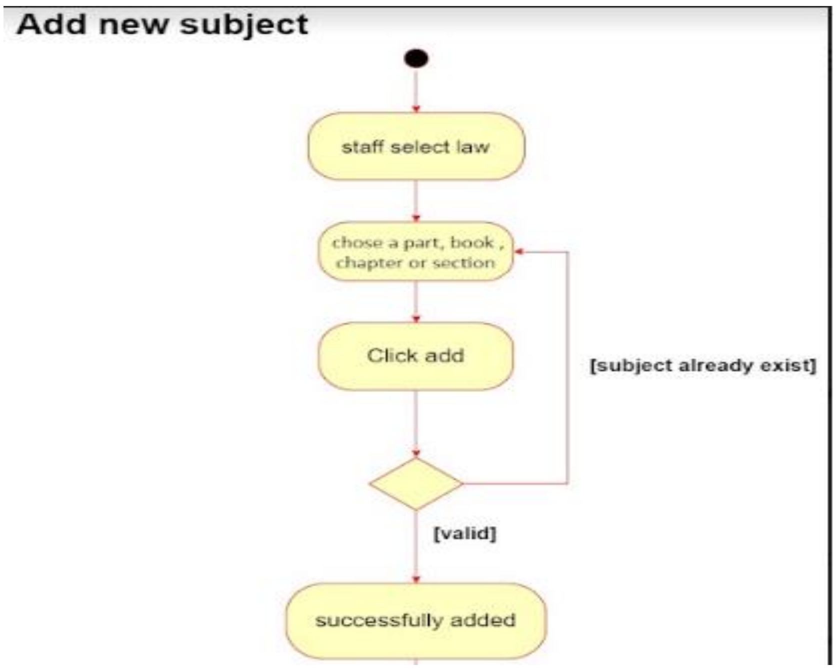


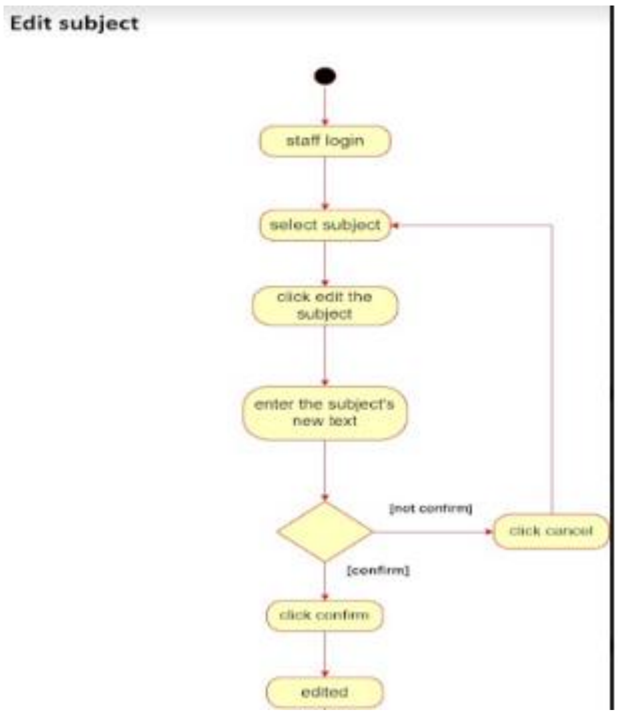

Display subject
Enter the system
select law
select the subject id
display subject

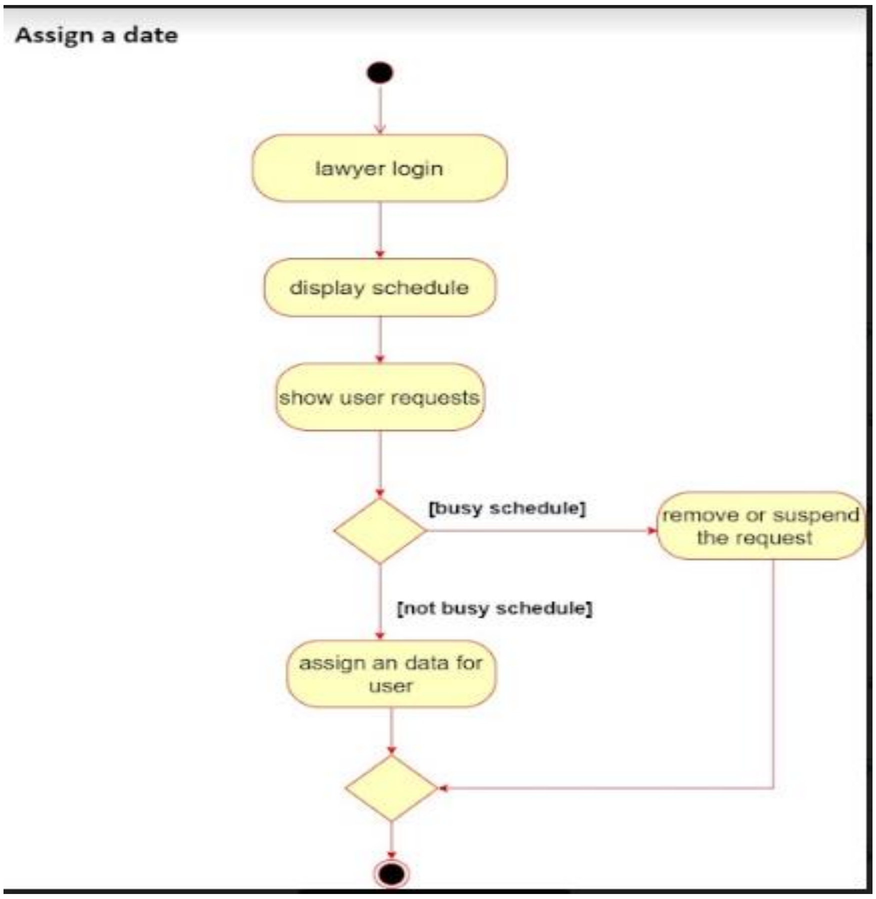

view lawyer profile
user login
select lawyer
display lawyer profile

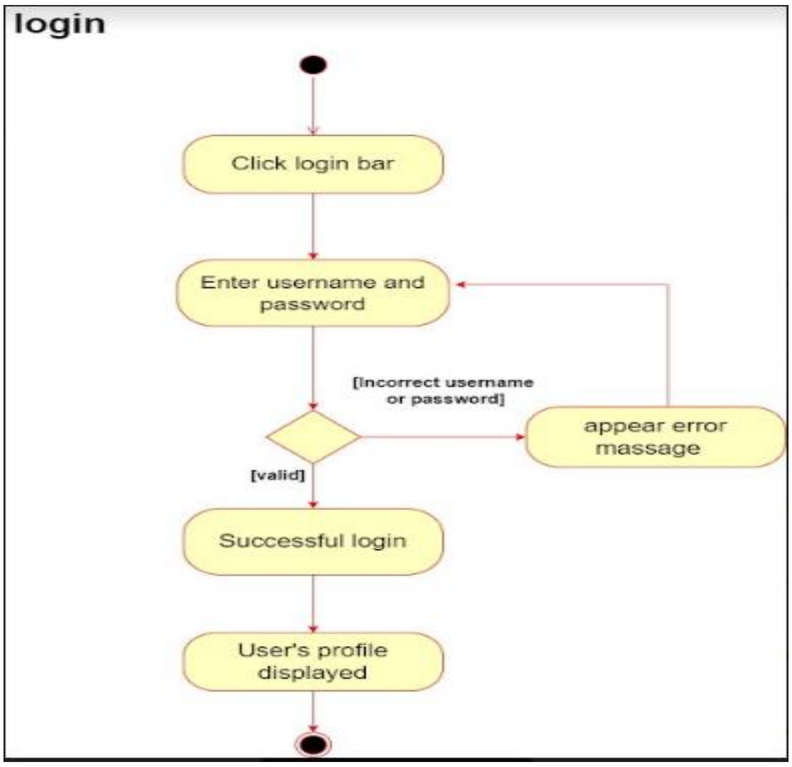

registration
Click registration button
Enter e-mail and password
Click sign up
[Incorrect e-mail or password]
[valid]
Registration saved in database
user's profile displayed

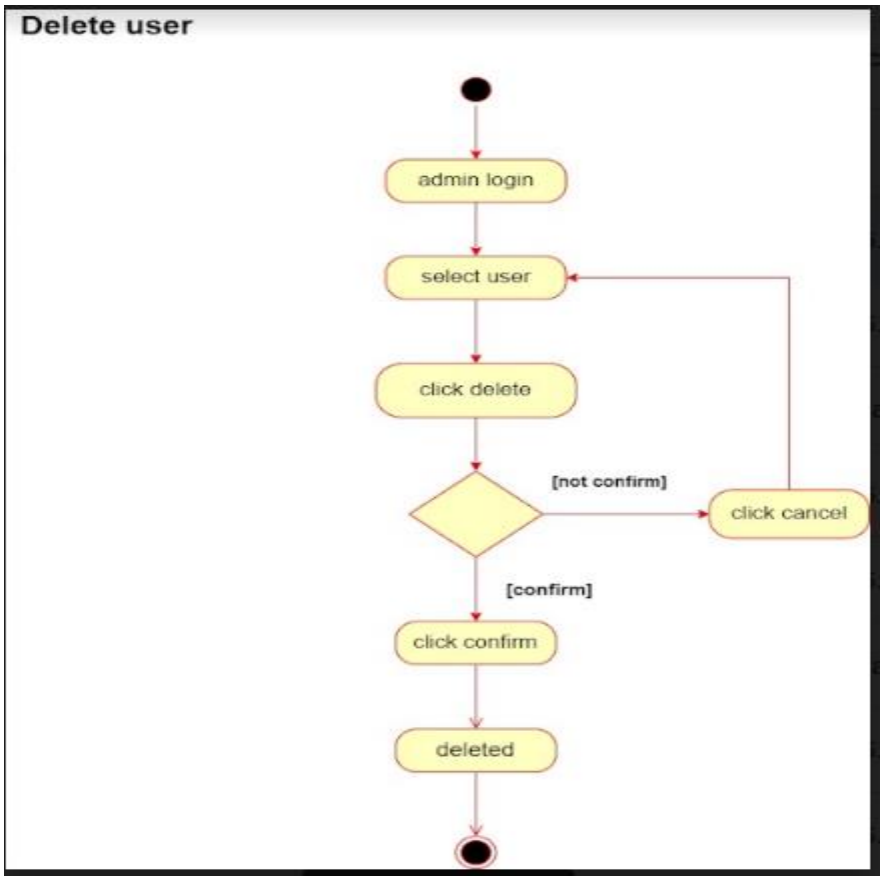

Ask lawyer
user login
write a question
Click post
System posts user's question

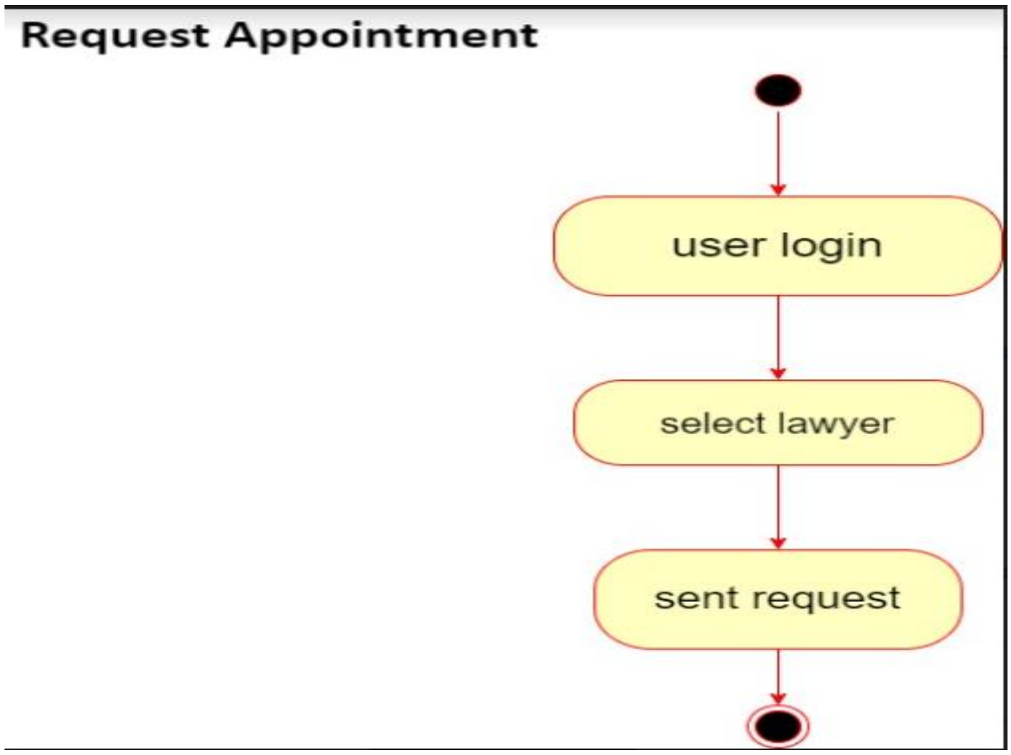

Add new subject
staff select law
chose a part, book , chapter or section
Click add
[subject already exist]
[valid]
successfully added


Edit subject
staff login
select subject
click edit the subject
enter the subject's new text
[not confirm]
click cancel
[confirm]
click confirm
edited


Assign a date
lawyer login
display schedule
show user requests
[busy schedule]
remove or suspend the request
[not busy schedule]
assign an data for user

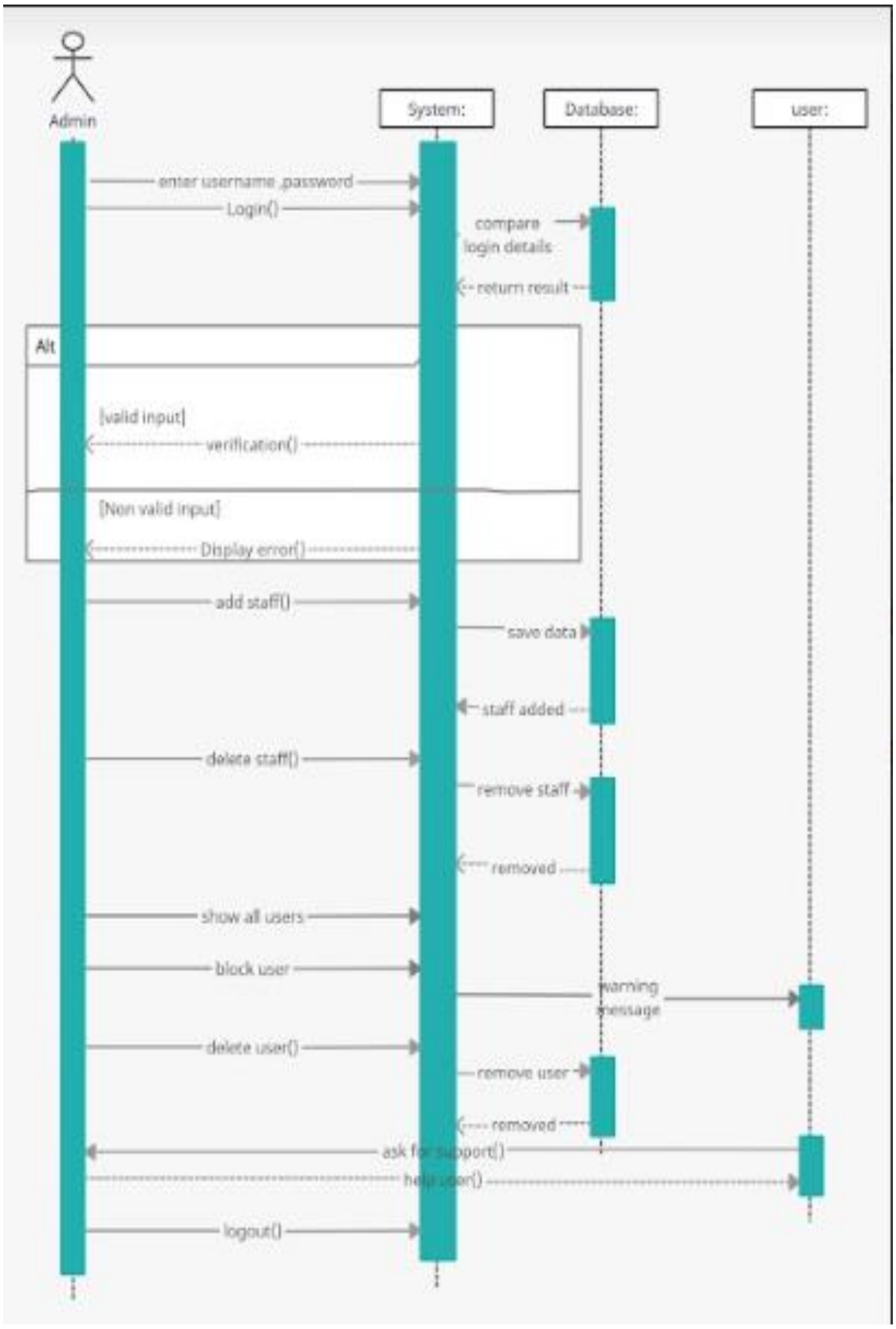

login
Click login bar
Enter username and password
[Incorrect username or password]
appear error massage
[valid]
Successful login
User's profile displayed


Delete user
admin login
select user
click delete
[not confirm]
click cancel
[confirm]
click confirm
deleted

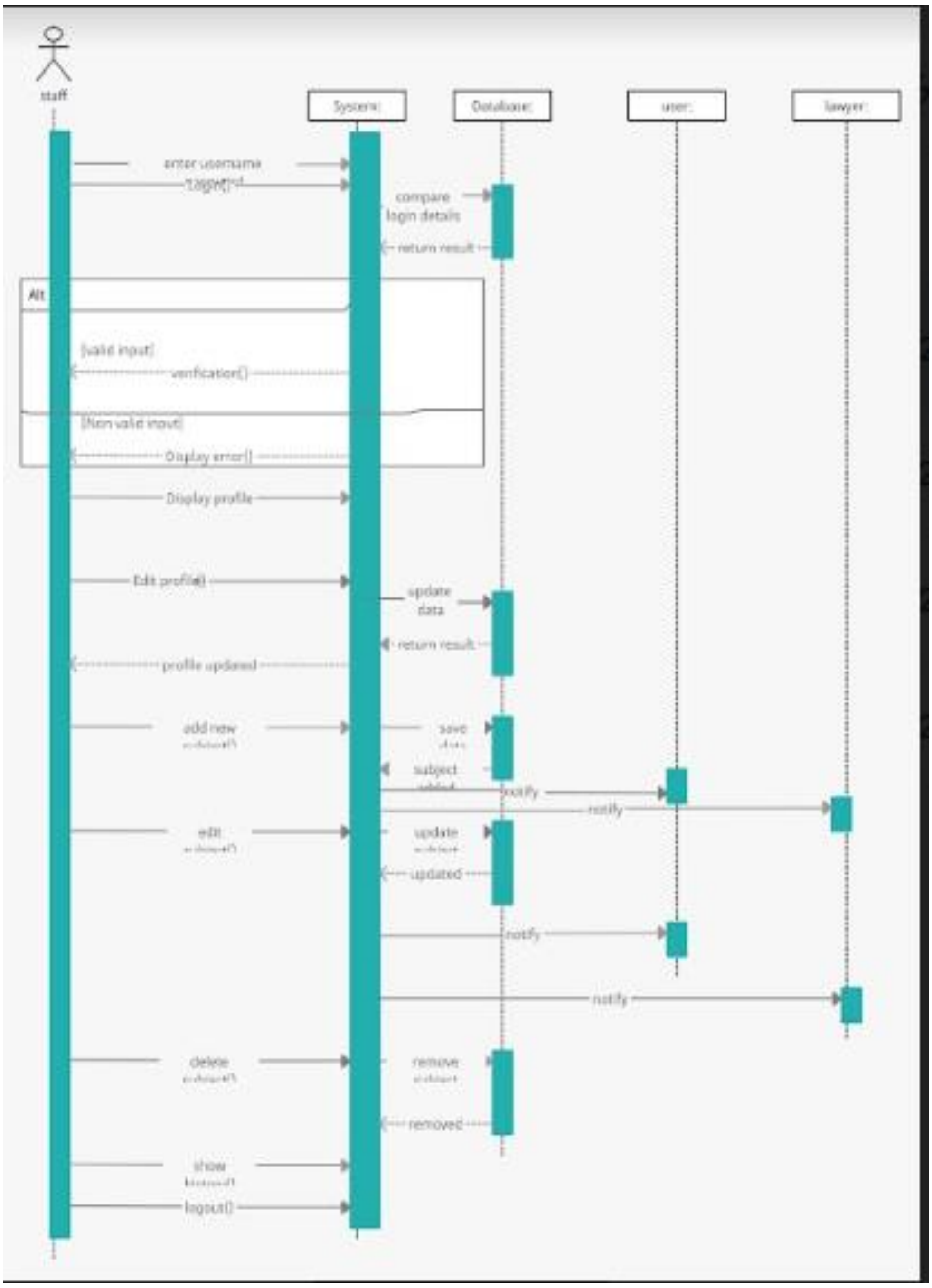

Request Appointment
user login
select lawyer
sent request

3.5 Sequence Diagram

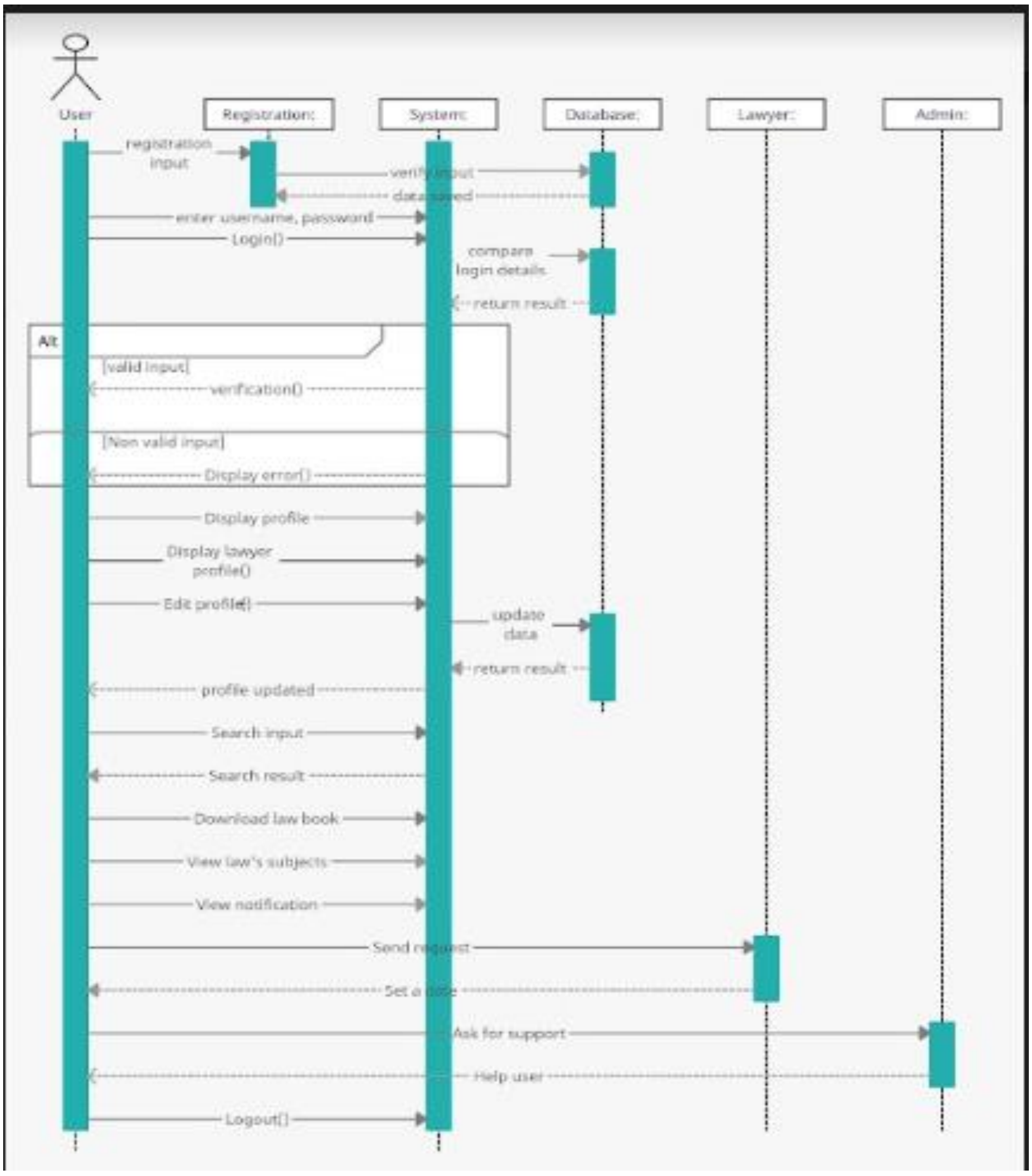
-3.5.1 Admin



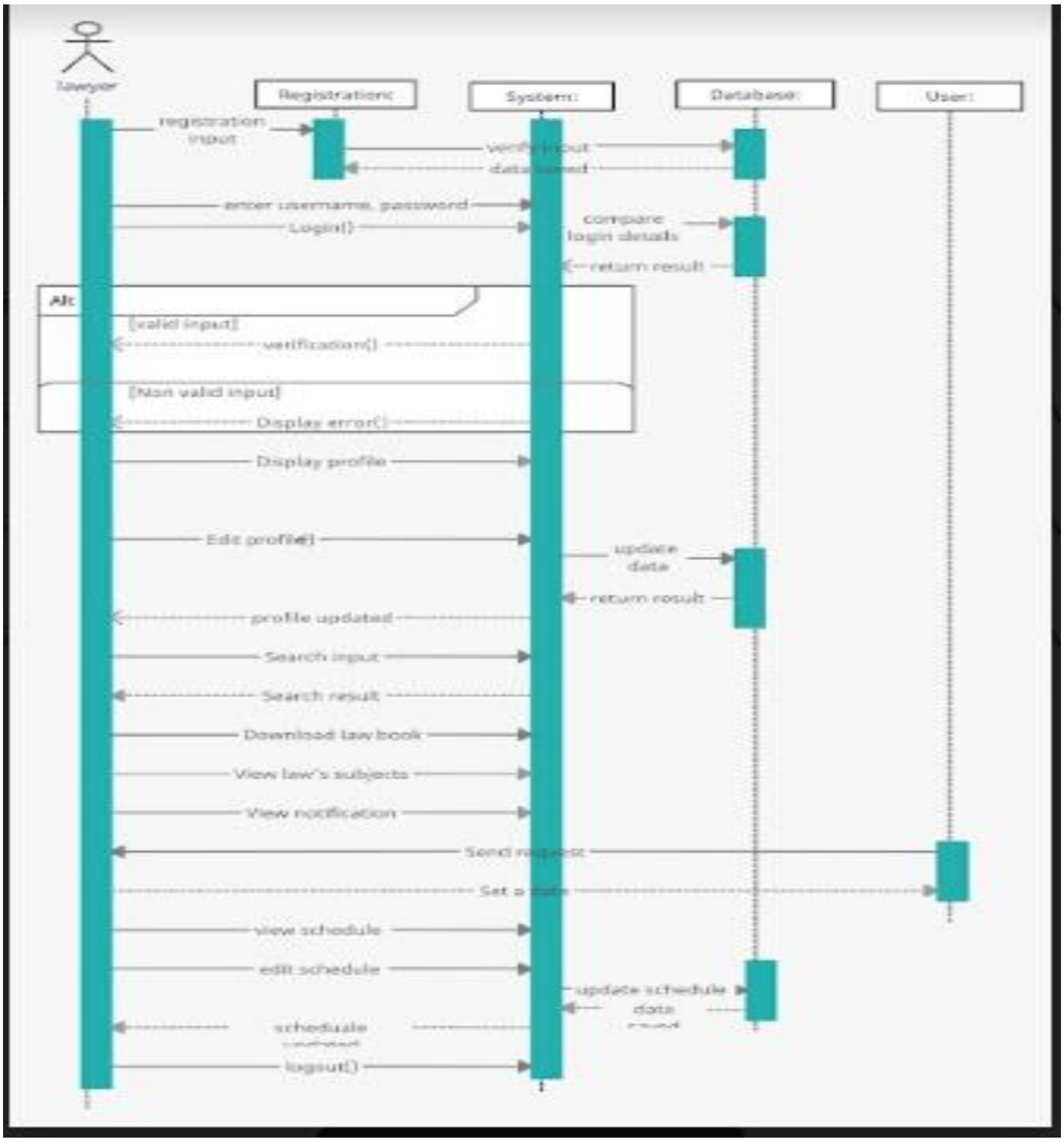
-3.5.2 Staff


staff
System:
Database:
user:
lawyer:
enter username
Login()
compare login details
return result
Alt
[valid input]
verification()
[Non valid input]
Display error()
Display profile
Edit profile()
update data
return result
profile updated
add new
save
subject
notify
notify
edit
update
updated
notify
notify
delete
remove
removed
show
logout()

3.5.3 User


User
Registration:
System:
Database:
Lawyer:
Admin:
registration input
verify input
data saved
enter username, password
Login()
compare login details
return result
Alt
[valid input]
verification()
[Non valid input]
Display error()
Display profile
Display lawyer profile()
Edit profile()
update data
return result
profile updated
Search input
Search result
Download law book
View law's subjects
View notification
Send request
Set a date
Ask for support
Help user
Logout()

-3.5.4 Lawyer

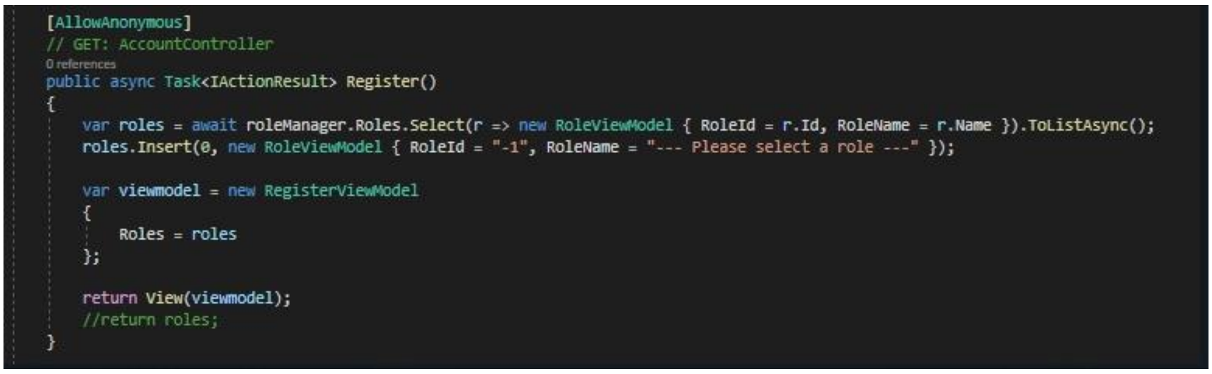

lawyer
Registration:
System:
Database:
User:
registration input
verify input
data saved
enter username, password
Login()
compare login details
return result
Alt
[valid input]
verification()
[Non valid input]
Display error()
Display profile
Edit profile()
update data
return result
profile updated
Search input
Search result
Download law book
View law's subjects
View notification
Send request
Set a date
view schedule
edit schedule
update schedule data
data saved
schedule updated
logout()

Chapter 4: Implementation

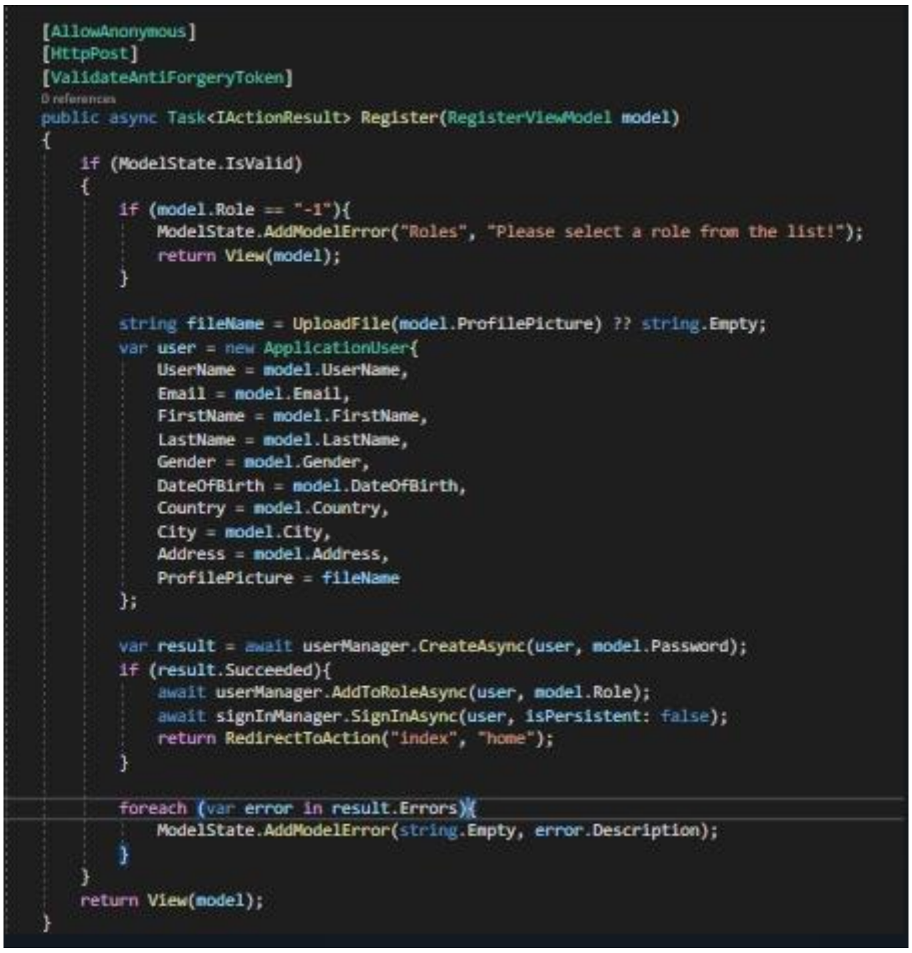
In this chapter we are going to discuss and go deeper in Encyclopedia Of Law implementation, and present its code and the algorithms used to build it.

4.1 Software Architecture

4.1.1 Register

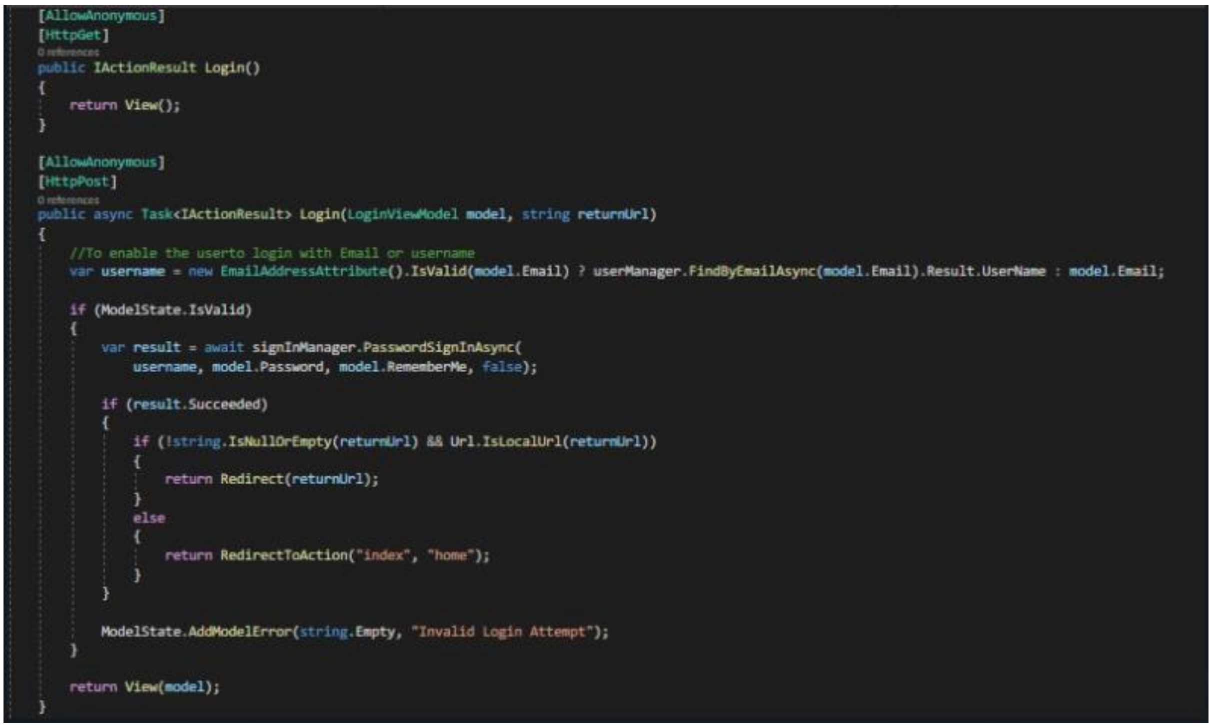


```
[AllowAnonymous]
// GET: AccountController
0 references
public async Task<IActionResult> Register()
{
    var roles = await roleManager.Roles.Select(r => new RoleViewModel { RoleId = r.Id, RoleName = r.Name }).ToListAsync();
    roles.Insert(0, new RoleViewModel { RoleId = "-1", RoleName = "--- Please select a role ---" });

    var viewmodel = new RegisterViewModel
    {
        Roles = roles
    };

    return View(viewmodel);
    //return roles;
}
```



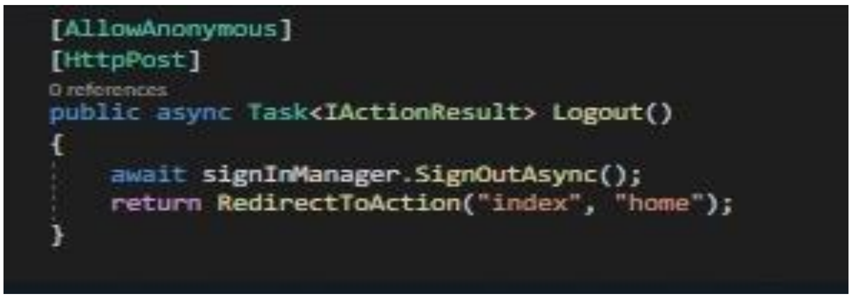
```
[AllowAnonymous]
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        if (model.Role == "-1"){
            ModelState.AddModelError("Roles", "Please select a role from the list!");
            return View(model);
        }

        string fileName = UploadFile(model.ProfilePicture) ?? string.Empty;
        var user = new ApplicationUser{
            UserName = model.UserName,
            Email = model.Email,
            FirstName = model.FirstName,
            LastName = model.LastName,
            Gender = model.Gender,
            DateOfBirth = model.DateOfBirth,
            Country = model.Country,
            City = model.City,
            Address = model.Address,
            ProfilePicture = fileName
        };

        var result = await userManager.CreateAsync(user, model.Password);
        if (result.Succeeded){
            await userManager.AddToRoleAsync(user, model.Role);
            await signInManager.SignInAsync(user, isPersistent: false);
            return RedirectToAction("index", "home");
        }

        foreach (var error in result.Errors){
            ModelState.AddModelError(string.Empty, error.Description);
        }
    }
    return View(model);
}
```

4.1.2 Login



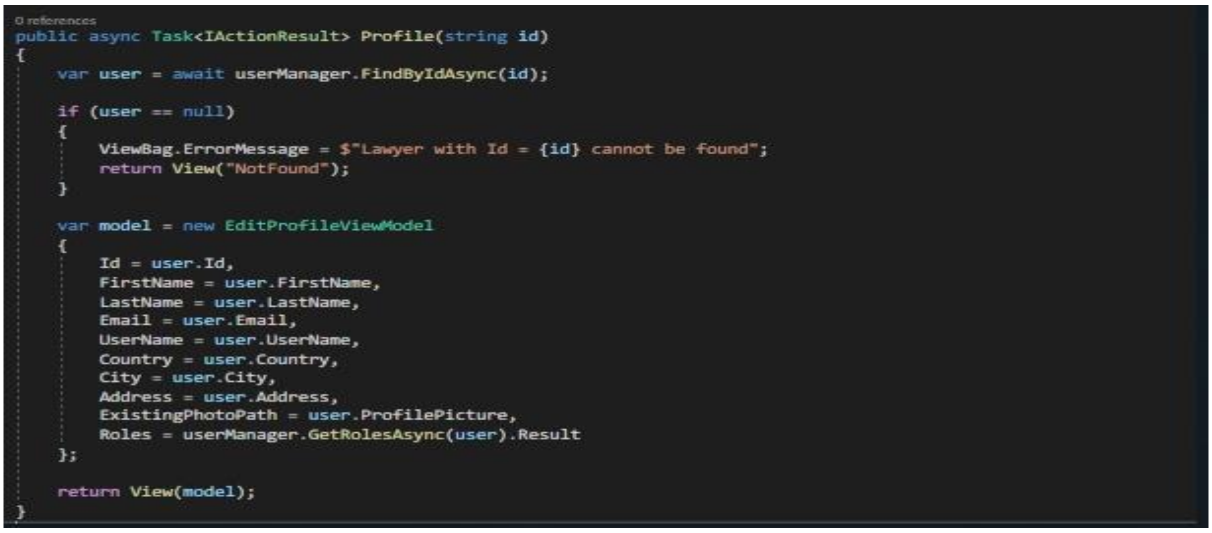
```
[AllowAnonymous]
[HttpGet]
0 references
public IActionResult Login()
{
    return View();
}

[AllowAnonymous]
[HttpPost]
0 references
public async Task<IActionResult> Login(LoginViewModel model, string returnUrl)
{
    //To enable the userto login with Email or username
    var username = new EmailAddressAttribute().IsValid(model.Email) ? userManager.FindByEmailAsync(model.Email).Result.UserName : model.Email;

    if (ModelState.IsValid)
    {
        var result = await signInManager.PasswordSignInAsync(
            username, model.Password, model.RememberMe, false);

        if (result.Succeeded)
        {
            if (!string.IsNullOrEmpty(returnUrl) && Url.IsLocalUrl(returnUrl))
            {
                return Redirect(returnUrl);
            }
            else
            {
                return RedirectToAction("index", "home");
            }
        }

        ModelState.AddModelError(string.Empty, "Invalid Login Attempt");
    }

    return View(model);
}
```

4.1.3 Logout



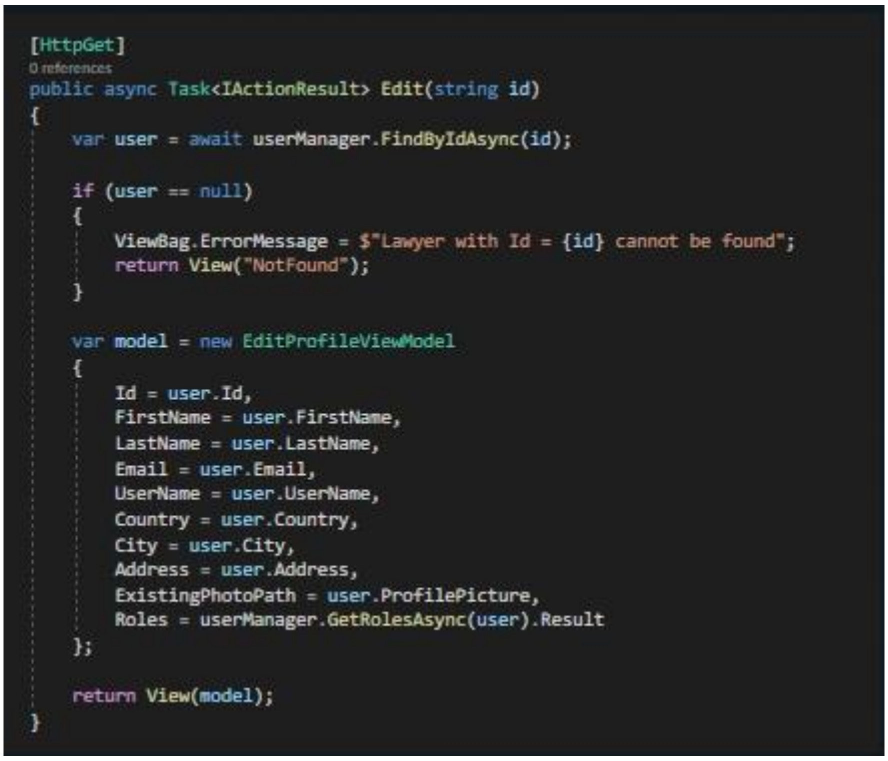
```
[AllowAnonymous]
[HttpPost]
0 references
public async Task<IActionResult> Logout()
{
    await signInManager.SignOutAsync();
    return RedirectToAction("index", "home");
}
```

4.1.4 Profile

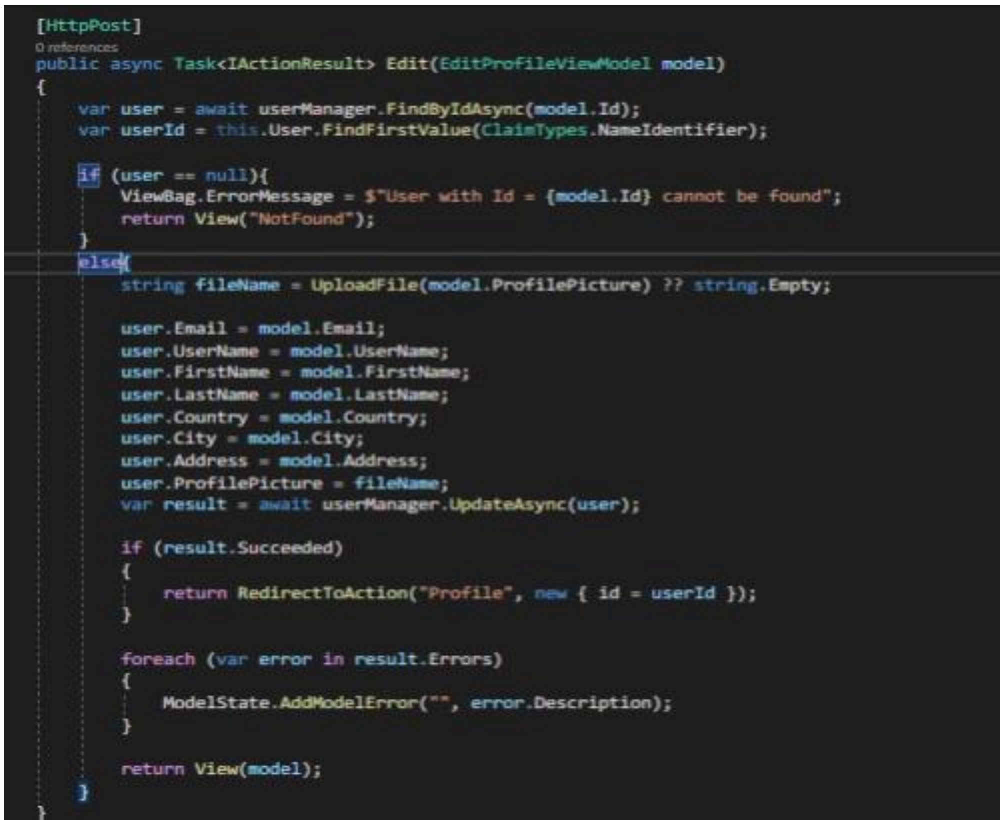


```
0 references
public async Task<IActionResult> Profile(string id)
{
    var user = await userManager.FindByIdAsync(id);

    if (user == null)
    {
        ViewBag.ErrorMessage = $"Lawyer with Id = {id} cannot be found";
        return View("NotFound");
    }

    var model = new EditProfileViewModel
    {
        Id = user.Id,
        FirstName = user.FirstName,
        LastName = user.LastName,
        Email = user.Email,
        UserName = user.UserName,
        Country = user.Country,
        City = user.City,
        Address = user.Address,
        ExistingPhotoPath = user.ProfilePicture,
        Roles = userManager.GetRolesAsync(user).Result
    };

    return View(model);
}
```

4.1.5 Edit Profile

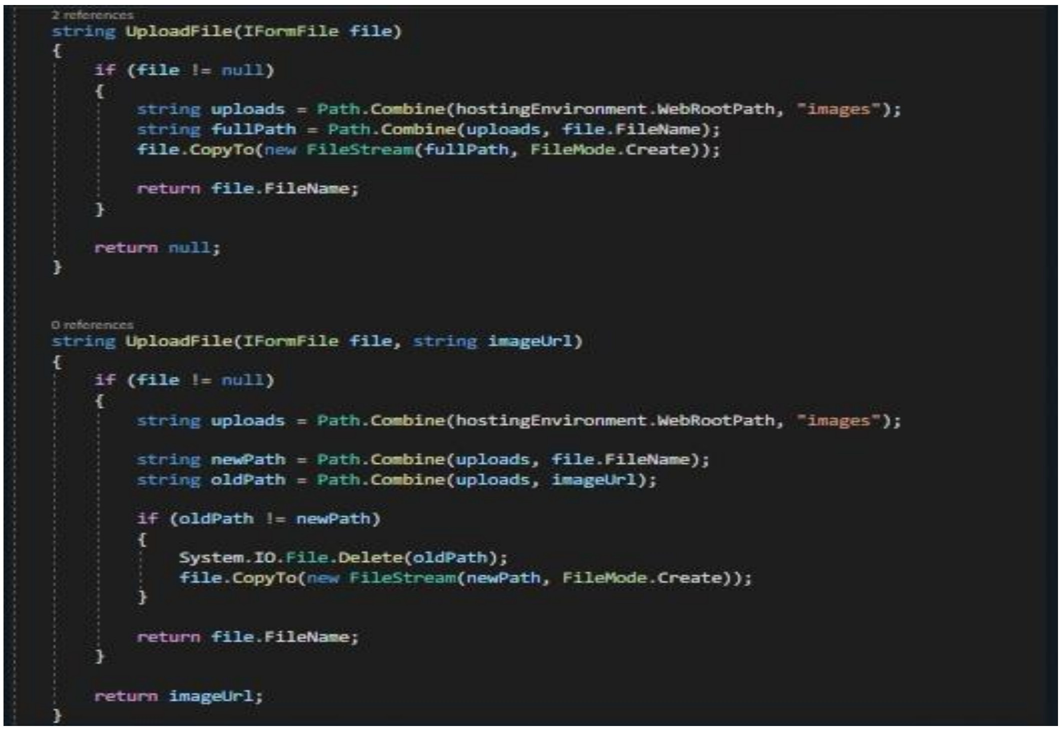


```
[HttpGet]
0 references
public async Task<IActionResult> Edit(string id)
{
    var user = await userManager.FindByIdAsync(id);

    if (user == null)
    {
        ViewBag.ErrorMessage = $"Lawyer with Id = {id} cannot be found";
        return View("NotFound");
    }

    var model = new EditProfileViewModel
    {
        Id = user.Id,
        FirstName = user.FirstName,
        LastName = user.LastName,
        Email = user.Email,
        UserName = user.UserName,
        Country = user.Country,
        City = user.City,
        Address = user.Address,
        ExistingPhotoPath = user.ProfilePicture,
        Roles = userManager.GetRolesAsync(user).Result
    };

    return View(model);
}
```



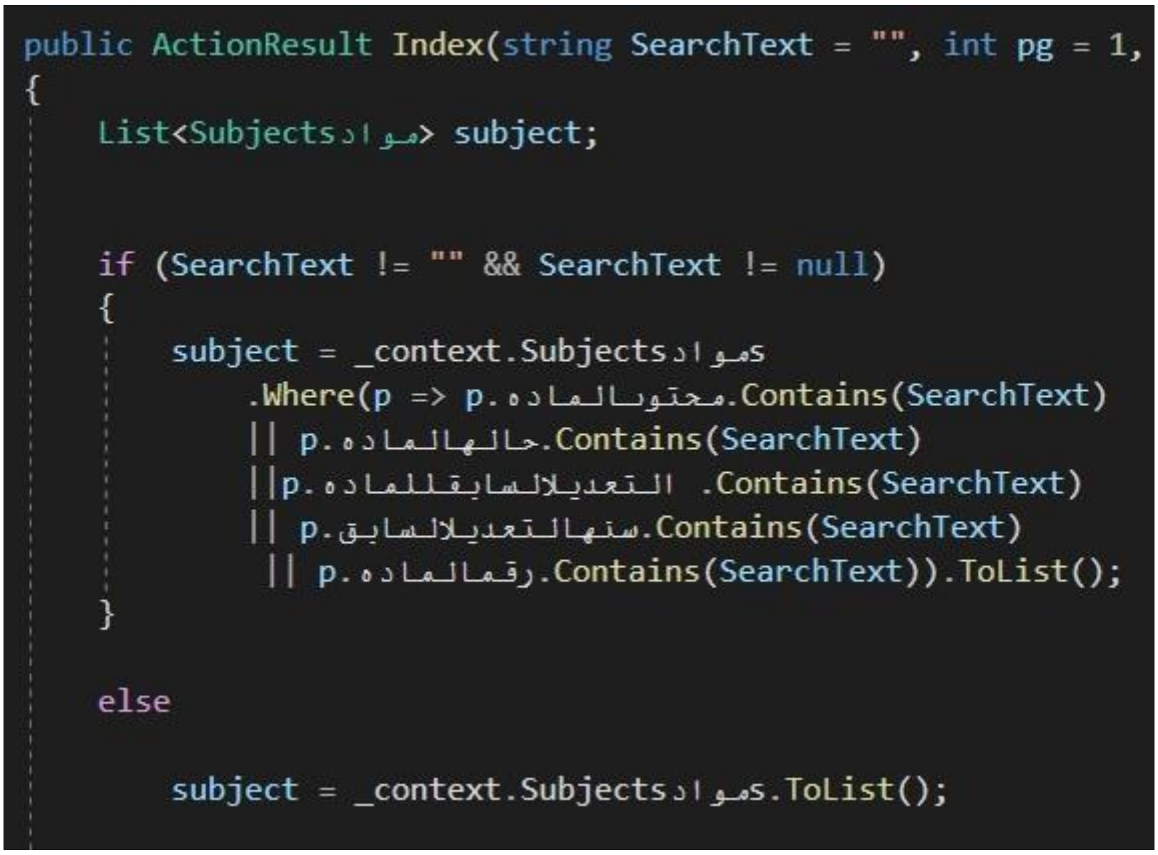
```
[HttpPost]
0 references
public async Task<IActionResult> Edit(EditProfileViewModel model)
{
    var user = await userManager.FindByIdAsync(model.Id);
    var userId = this.User.FindFirstValue(ClaimTypes.NameIdentifier);

    if (user == null){
        ViewBag.ErrorMessage = $"User with Id = {model.Id} cannot be found";
        return View("NotFound");
    }
    else{
        string fileName = UploadFile(model.ProfilePicture) ?? string.Empty;

        user.Email = model.Email;
        user.UserName = model.UserName;
        user.FirstName = model.FirstName;
        user.LastName = model.LastName;
        user.Country = model.Country;
        user.City = model.City;
        user.Address = model.Address;
        user.ProfilePicture = fileName;
        var result = await userManager.UpdateAsync(user);

        if (result.Succeeded)
        {
            return RedirectToAction("Profile", new { id = userId });
        }

        foreach (var error in result.Errors)
        {
            ModelState.AddModelError("", error.Description);
        }

        return View(model);
    }
}
```

4.1.6 Upload file



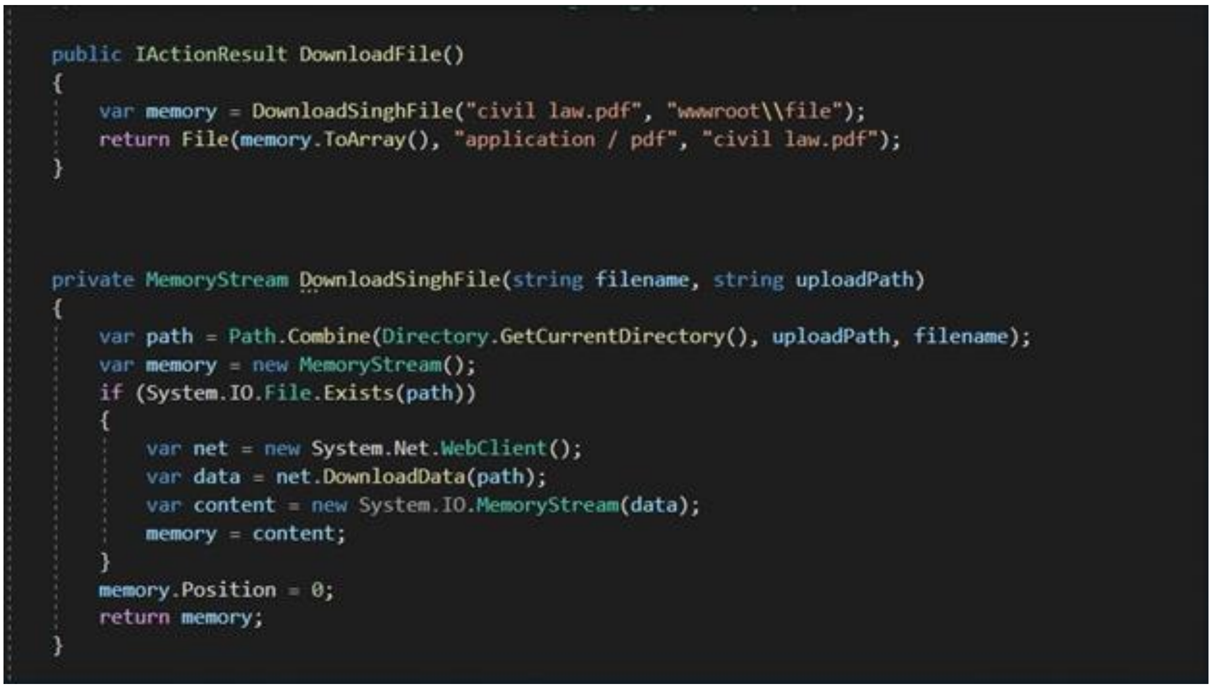
```
2 references
string UploadFile(IFormFile file)
{
    if (file != null)
    {
        string uploads = Path.Combine(hostingEnvironment.WebRootPath, "images");
        string fullPath = Path.Combine(uploads, file.FileName);
        file.CopyTo(new FileStream(fullPath, FileMode.Create));

        return file.FileName;
    }

    return null;
}

0 references
string UploadFile(IFormFile file, string imageUrl)
{
    if (file != null)
    {
        string uploads = Path.Combine(hostingEnvironment.WebRootPath, "images");

        string newPath = Path.Combine(uploads, file.FileName);
        string oldPath = Path.Combine(uploads, imageUrl);

        if (oldPath != newPath)
        {
            System.IO.File.Delete(oldPath);
            file.CopyTo(new FileStream(newPath, FileMode.Create));
        }

        return file.FileName;
    }

    return imageUrl;
}
```

4.1.7 Index



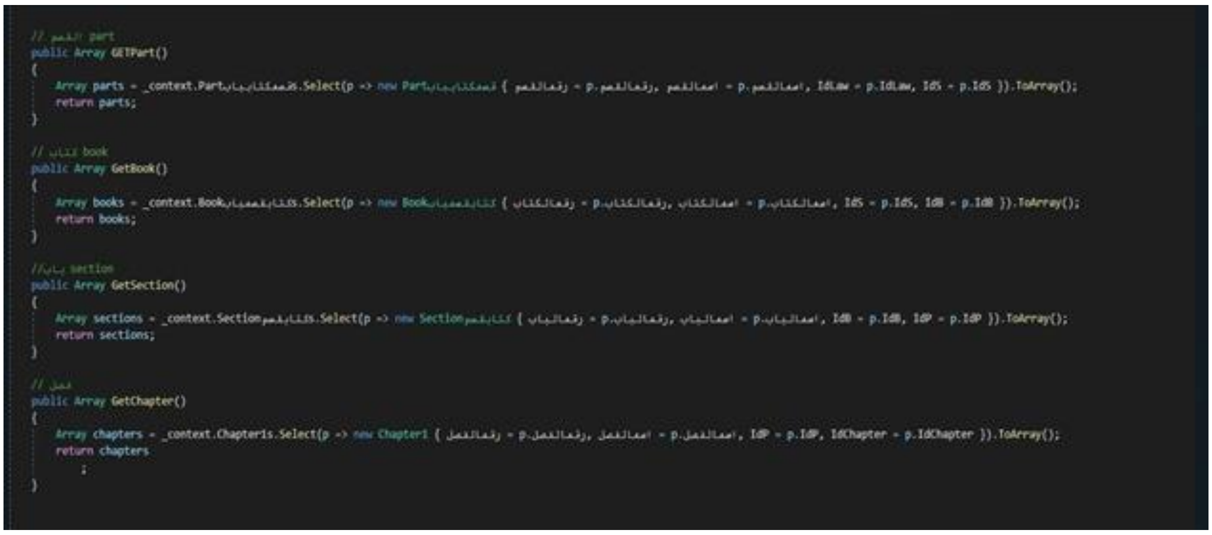
```
public ActionResult Index(string SearchText = "", int pg = 1,
{
    List<Subjectsمواد> subject;


    if (SearchText != "" && SearchText != null)
    {
        subject = _context.Subjectsمواد
            .Where(p => p.محتوى_المادة.Contains(SearchText)
            || p.حاله_المادة.Contains(SearchText)
            ||p.التعديلات_السابقة_للمادة .Contains(SearchText)
            || p.سنه_التعديل_السابق.Contains(SearchText)
             || p.رقم_المادة.Contains(SearchText)).ToList();
    }

    else

        subject = _context.Subjectsمواد.ToList();
```

4.1.8 Download File



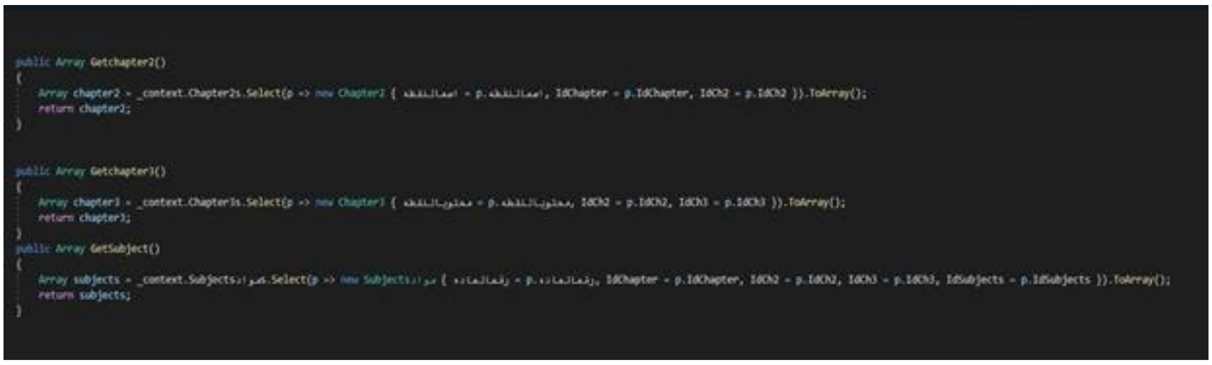
```
public IActionResult DownloadFile()
{
    var memory = DownloadSinghFile("civil law.pdf", "wwwroot\\file");
    return File(memory.ToArray(), "application / pdf", "civil law.pdf");
}



private MemoryStream DownloadSinghFile(string filename, string uploadPath)
{
    var path = Path.Combine(Directory.GetCurrentDirectory(), uploadPath, filename);
    var memory = new MemoryStream();
    if (System.IO.File.Exists(path))
    {
        var net = new System.Net.WebClient();
        var data = net.DownloadData(path);
        var content = new System.IO.MemoryStream(data);
        memory = content;
    }
    memory.Position = 0;
    return memory;
}
```

4.1.9 GetPart



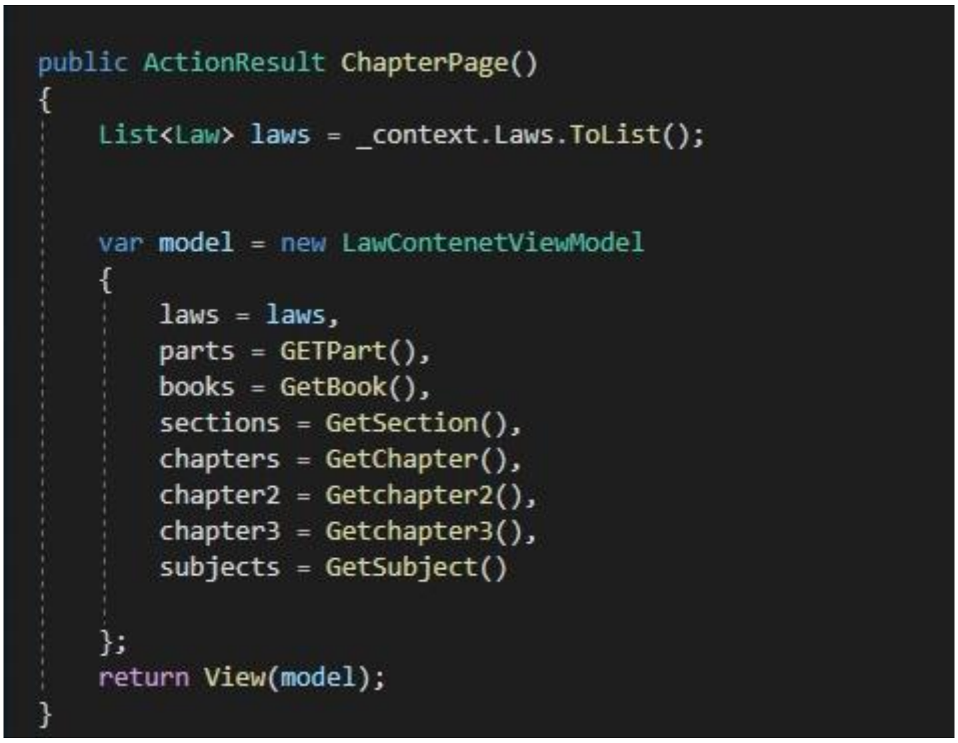
```
// القسم part
public Array GETPart()
{
    Array parts = _context.Partبابكتابقسمs.Select(p => new Partبابكتابقسم { رقمالقسم = p.رقمالقسم, اسمالقسم = p.اسمالقسم, IdLaw = p.IdLaw, IdS = p.IdS }).ToArray();
    return parts;
}

// كتاب book
public Array GetBook()
{
    Array books = _context.Bookبابقسمكتابs.Select(p => new Bookبابقسمكتاب { رقمالكتاب = p.رقمالكتاب, اسمالكتاب = p.اسمالكتاب, IdS = p.IdS, IdB = p.IdB }).ToArray();
    return books;
}

//باب section
public Array GetSection()
{
    Array sections = _context.Sectionكتابقسمs.Select(p => new Sectionكتابقسم { رقمالباب = p.رقمالباب, اسمالباب = p.اسمالباب, IdB = p.IdB, IdP = p.IdP }).ToArray();
    return sections;
}

// فصل
public Array GetChapter()
{
    Array chapters = _context.Chapter1s.Select(p => new Chapter1 { رقمالفصل = p.رقمالفصل, اسمالفصل = p.اسمالفصل, IdP = p.IdP, IdChapter = p.IdChapter }).ToArray();
    return chapters
        ;
}
```

4.1.10 GetChapter



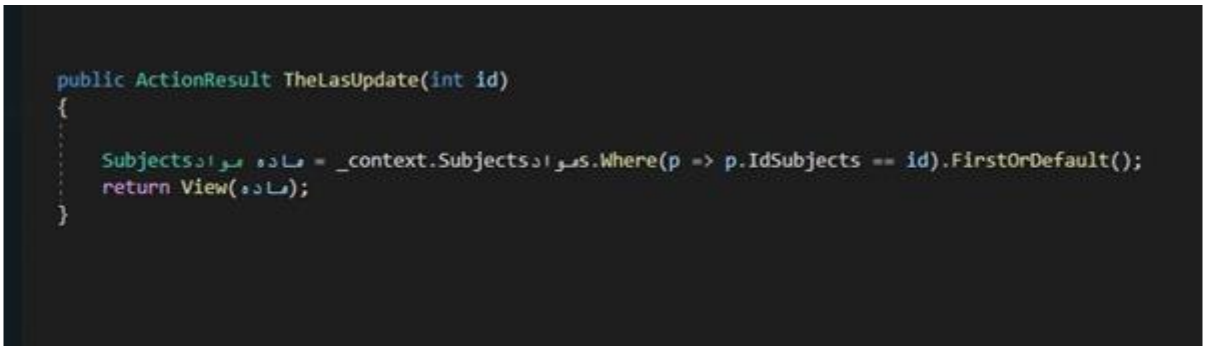
```
public Array Getchapter2()
{
    Array chapter2 = _context.Chapter2s.Select(p => new Chapter2 { اسمالنقطة = p.اسمالنقطة, IdChapter = p.IdChapter, IdCh2 = p.IdCh2 }).ToArray();
    return chapter2;
}


public Array Getchapter3()
{
    Array chapter3 = _context.Chapter3s.Select(p => new Chapter3 { محتوىالنقطة = p.محتوىالنقطة, IdCh2 = p.IdCh2, IdCh3 = p.IdCh3 }).ToArray();
    return chapter3;
}
public Array GetSubject()
{
    Array subjects = _context.Subjectsموادs.Select(p => new Subjectsمواد { رقمالمادة = p.رقمالمادة, IdChapter = p.IdChapter, IdCh2 = p.IdCh2, IdCh3 = p.IdCh3, IdSubjects = p.IdSubjects }).ToArray();
    return subjects;
}
```

4.1.11 Chapter Page



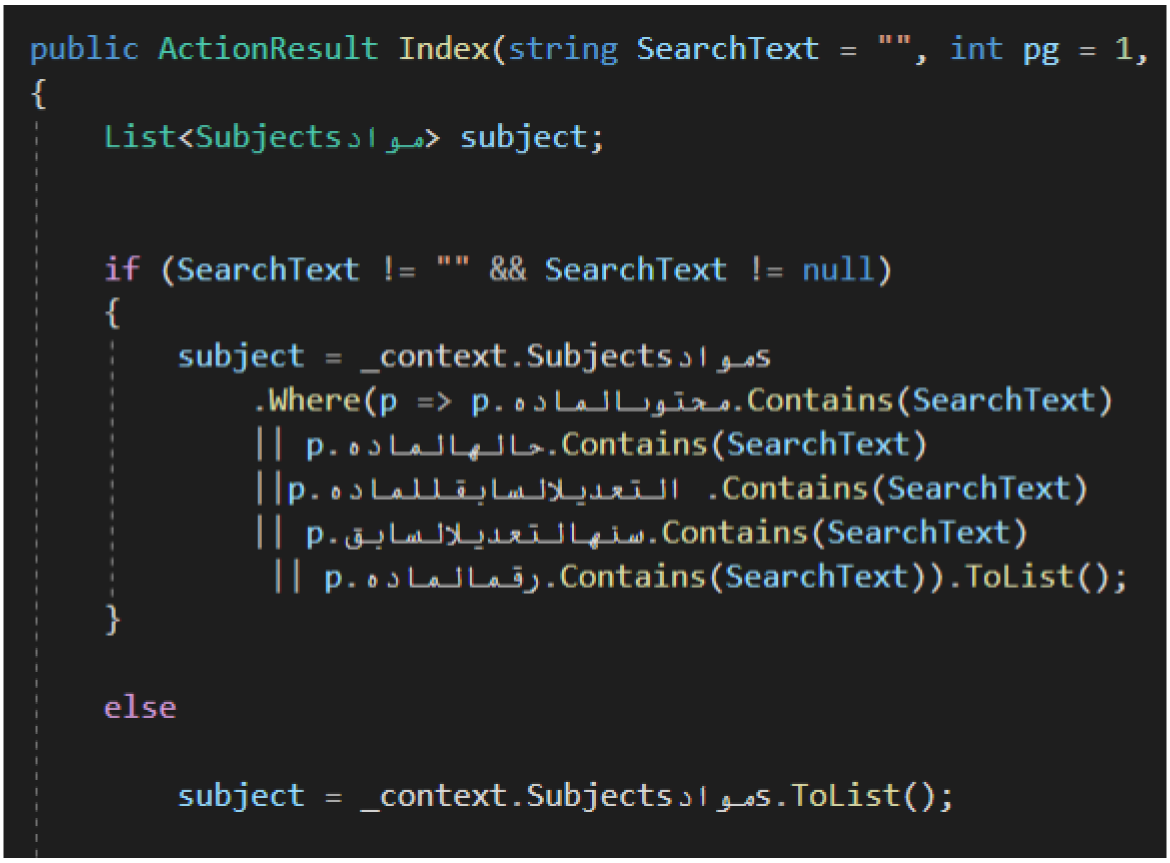
```
public ActionResult ChapterPage()
{
    List<Law> laws = _context.Laws.ToList();


    var model = new LawContenetViewModel
    {
        laws = laws,
        parts = GETPart(),
        books = GetBook(),
        sections = GetSection(),
        chapters = GetChapter(),
        chapter2 = Getchapter2(),
        chapter3 = Getchapter3(),
        subjects = GetSubject()

    };
    return View(model);
}
```

4.1.12 The last update



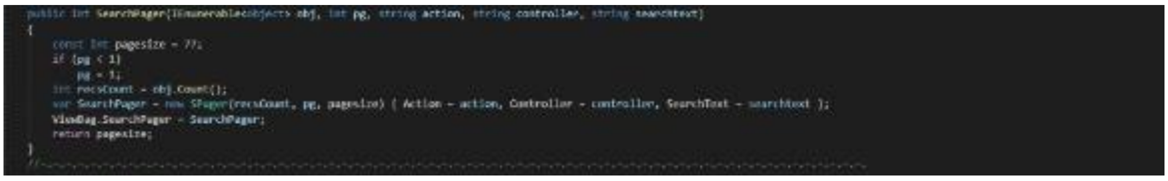
```
public ActionResult TheLasUpdate(int id)
{

    Subjectsمواد مادة = _context.Subjectsموادs.Where(p => p.IdSubjects == id).FirstOrDefault();
    return View(مادة);
}
```

4.1.13 Search For law subject



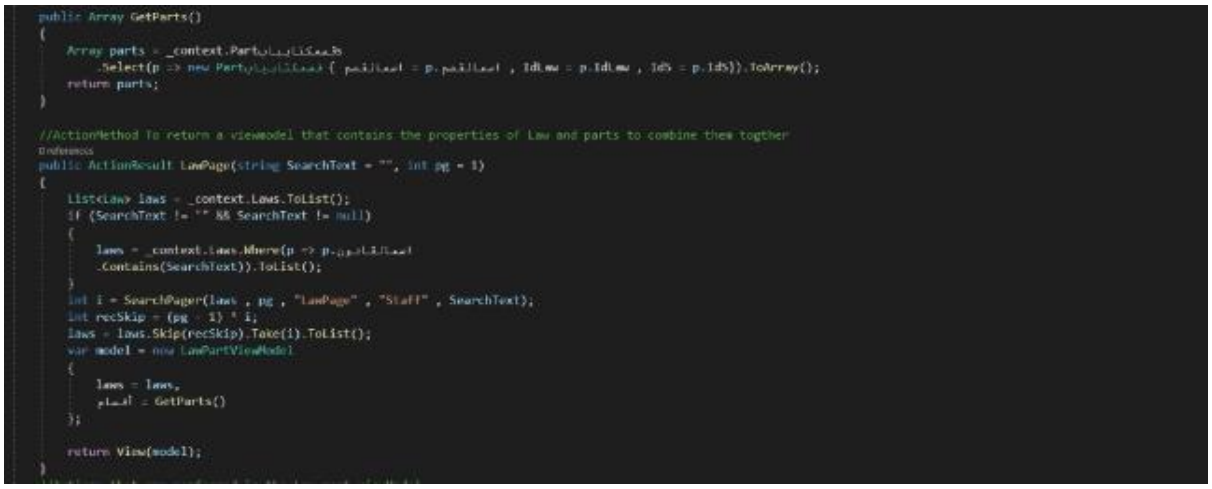
```
public ActionResult Index(string SearchText = "", int pg = 1,
{
    List<Subjectsمواد> subject;

    if (SearchText != "" && SearchText != null)
    {
        subject = _context.Subjectsموادs
            .Where(p => p.محتوىالمادة.Contains(SearchText)
            || p.حالهالمادة.Contains(SearchText)
            ||p.التعديلاتالسابقةللمادة .Contains(SearchText)
            || p.سنهالتعديلالسابق.Contains(SearchText)
             || p.رقمالمادة.Contains(SearchText)).ToList();
    }

    else

        subject = _context.Subjectsموادs.ToList();
```

4.1.14 search Pager



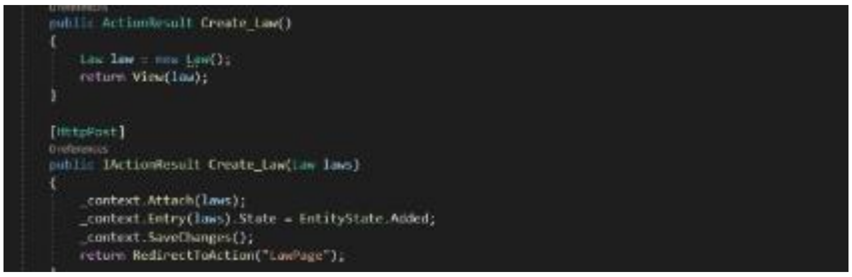
```
public int SearchPager(IEnumerable<objects> obj, int pg, string action, string controller, string searchtext)
{
    const int pagesize = 7;
    if (pg < 1)
        pg = 1;
    int recsCount = obj.Count();
    var SearchPager = new SPager(recsCount, pg, pagesize) { Action = action, Controller = controller, SearchText = searchtext };
    ViewBag.SearchPager = SearchPager;
    return pagesize;
}
//------------------------------------------------------------------------------------------------
```

4.1.15 Get Part & Law Page



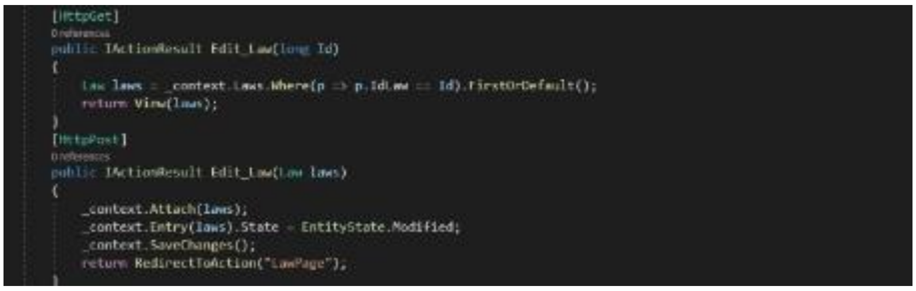
```
public Array GetParts()
{
    Array parts = _context.Part[illegible]s
        .Select(p => new Part[illegible] { [illegible] = p.[illegible] , IdLaw = p.IdLaw , IdS = p.IdS}).ToArray();
    return parts;
}

//ActionMethod To return a viewmodel that contains the properties of Law and parts to combine them togther
0 references
public ActionResult LawPage(string SearchText = "", int pg = 1)
{
    List<Law> laws = _context.Laws.ToList();
    if (SearchText != "" && SearchText != null)
    {
        laws = _context.Laws.Where(p => p.[illegible]
        .Contains(SearchText)).ToList();
    }
    int i = SearchPager(laws , pg , "LawPage" , "Staff" , SearchText);
    int recSkip = (pg - 1) * i;
    laws = laws.Skip(recSkip).Take(i).ToList();
    var model = new LawPartViewModel
    {
        laws = laws,
        أقسام = GetParts()
    };

    return View(model);
}
```

4.1.16 Create-Law



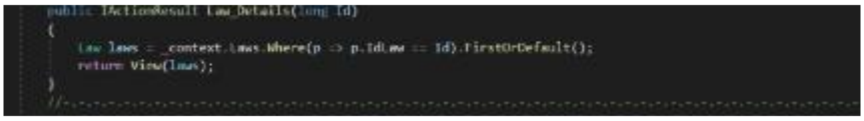
```
0 references
public ActionResult Create_Law()
{
    Law law = new Law();
    return View(law);
}

[HttpPost]
0 references
public IActionResult Create_Law(Law laws)
{
    _context.Attach(laws);
    _context.Entry(laws).State = EntityState.Added;
    _context.SaveChanges();
    return RedirectToAction("LawPage");
```

4.1.17 Edit-Law



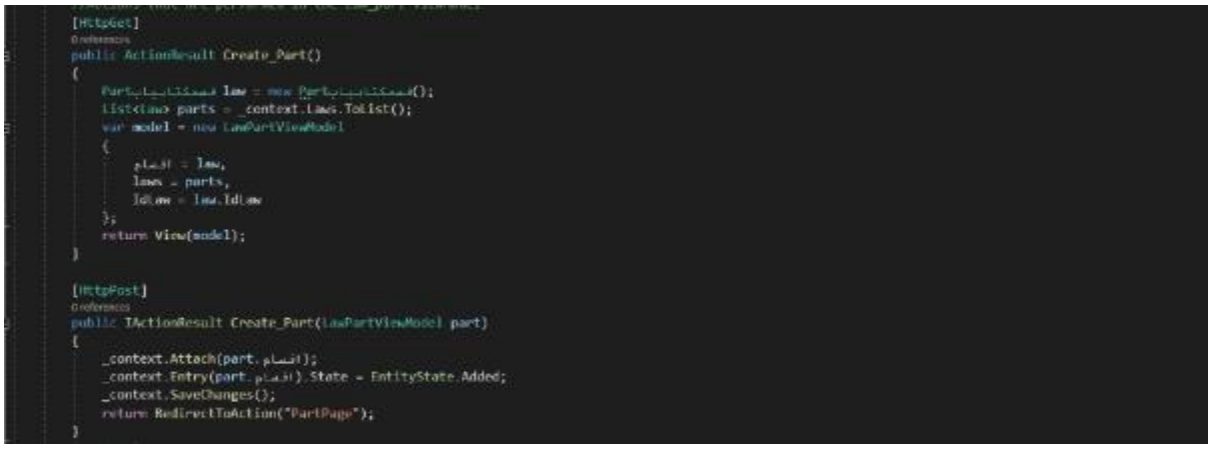
```
[HttpGet]
0 references
public IActionResult Edit_Law(long Id)
{
    Law laws = _context.Laws.Where(p => p.IdLaw == Id).FirstOrDefault();
    return View(laws);
}
[HttpPost]
0 references
public IActionResult Edit_Law(Law laws)
{
    _context.Attach(laws);
    _context.Entry(laws).State = EntityState.Modified;
    _context.SaveChanges();
    return RedirectToAction("LawPage");
}
```

4.1.18 Law-Details



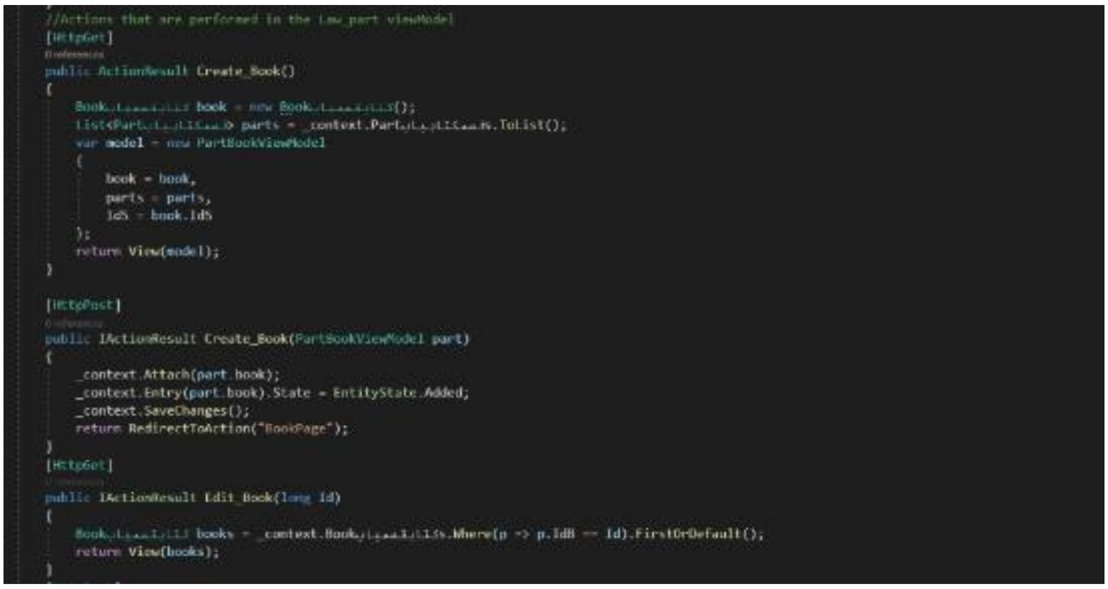
```
public IActionResult Law_Details(long Id)
{
    Law laws = _context.Laws.Where(p => p.IdLaw == Id).FirstOrDefault();
    return View(laws);
}
//-----------------------------------------------------------------------
```

4.1.19 Create-Part



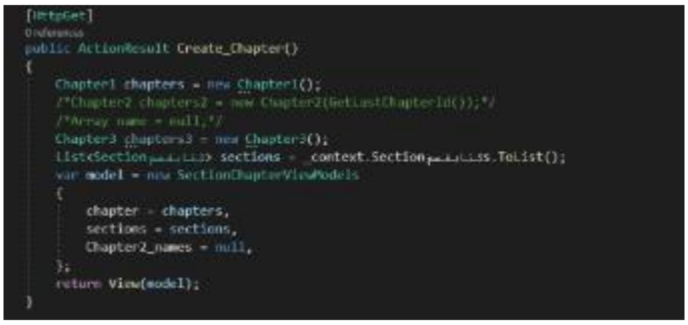
```
[HttpGet]
0 references
public ActionResult Create_Part()
{
    Part[illegible] law = new Part[illegible]();
    List<Law> parts = _context.Laws.ToList();
    var model = new LawPartViewModel
    {
        القسم = law,
        laws = parts,
        IdLaw = law.IdLaw
    };
    return View(model);
}

[HttpPost]
0 references
public IActionResult Create_Part(LawPartViewModel part)
{
    _context.Attach(part.القسم);
    _context.Entry(part.القسم).State = EntityState.Added;
    _context.SaveChanges();
    return RedirectToAction("PartPage");
}
```

4.1.20 Create-Book

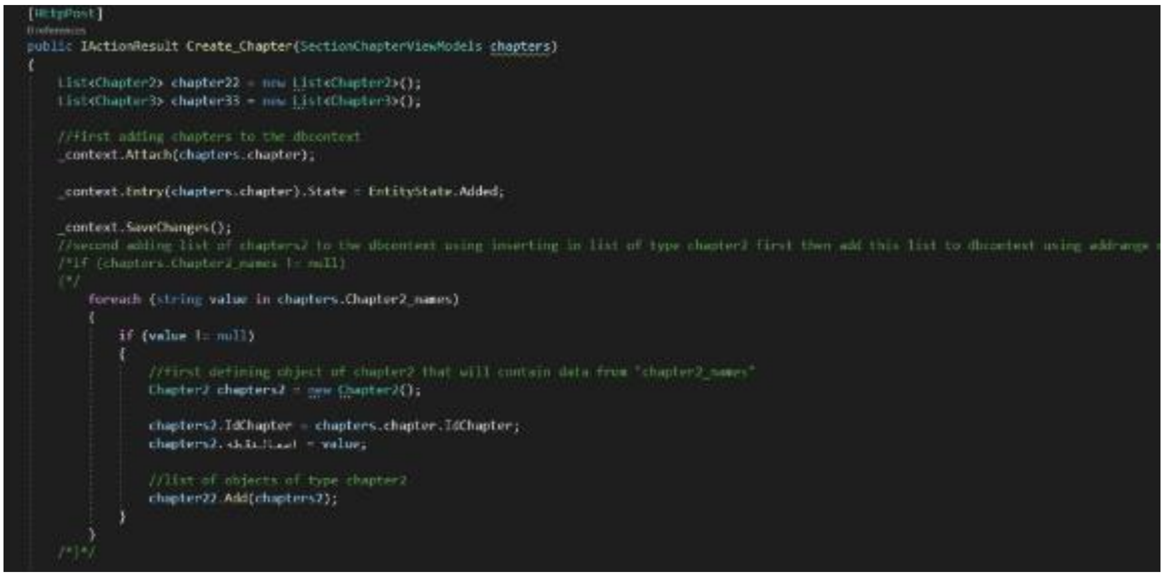


```
//Actions that are performed in the Law_part viewModel
[HttpGet]
0 references
public ActionResult Create_Book()
{
    Book_[illegible] book = new Book_[illegible]();
    List<Part_[illegible]> parts = _context.Part_[illegible].ToList();
    var model = new PartBookViewModel
    {
        book = book,
        parts = parts,
        IdS = book.IdS
    };
    return View(model);
}

[HttpPost]
0 references
public IActionResult Create_Book(PartBookViewModel part)
{
    _context.Attach(part.book);
    _context.Entry(part.book).State = EntityState.Added;
    _context.SaveChanges();
    return RedirectToAction("BookPage");
}
[HttpGet]
0 references
public IActionResult Edit_Book(long Id)
{
    Book_[illegible] books = _context.Book_[illegible].Where(p => p.IdB == Id).FirstOrDefault();
    return View(books);
}
```

4.1.21 Create-Chapter

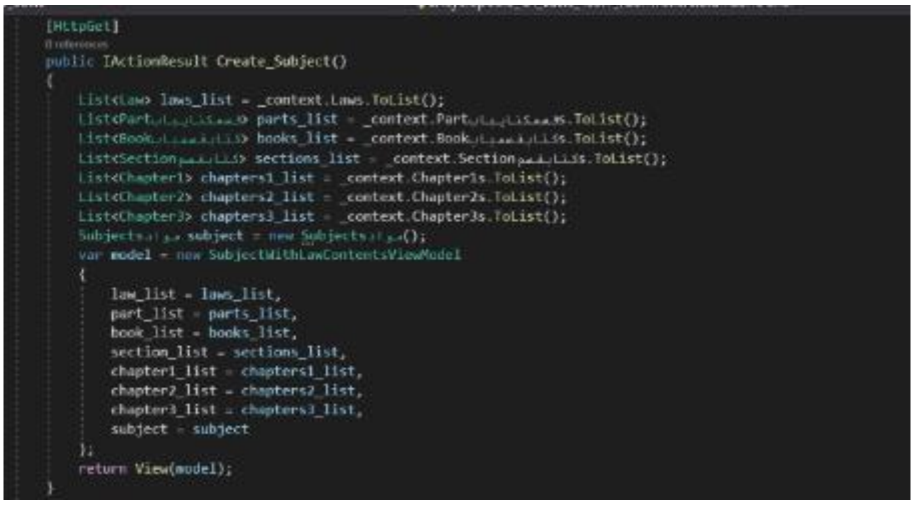


```
[HttpGet]
0 references
public ActionResult Create_Chapter()
{
    Chapter1 chapters = new Chapter1();
    /*Chapter2 chapters2 = new Chapter2(GetLastChapterId());*/
    /*Array name = null;*/
    Chapter3 chapters3 = new Chapter3();
    List<Section_[illegible]> sections = _context.Section_[illegible].ToList();
    var model = new SectionChapterViewModels
    {
        chapter = chapters,
        sections = sections,
        Chapter2_names = null,
    };
    return View(model);
}
```

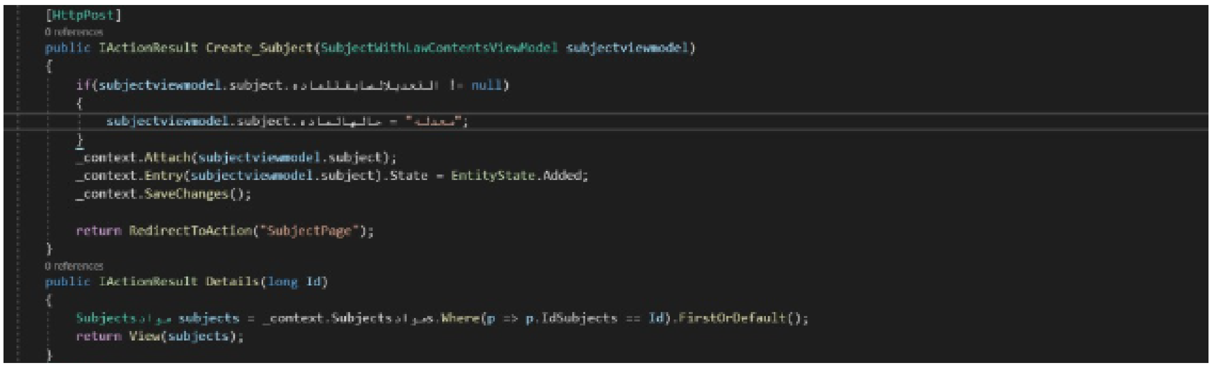


```
[HttpPost]
0 references
public IActionResult Create_Chapter(SectionChapterViewModels chapters)
{
    List<Chapter2> chapter22 = new List<Chapter2>();
    List<Chapter3> chapter33 = new List<Chapter3>();

    //first adding chapters to the dbcontext
    _context.Attach(chapters.chapter);

    _context.Entry(chapters.chapter).State = EntityState.Added;

    _context.SaveChanges();
    //second adding list of chapters2 to the dbcontext using inserting in list of type chapter2 first then add this list to dbcontext using addrange
    /*if (chapters.Chapter2_names != null)
    {*/
        foreach (string value in chapters.Chapter2_names)
        {
            if (value != null)
            {
                //first defining object of chapter2 that will contain data from "chapter2_names"
                Chapter2 chapters2 = new Chapter2();

                chapters2.IdChapter = chapters.chapter.IdChapter;
                chapters2.[illegible] = value;

                //list of objects of type chapter2
                chapter22.Add(chapters2);
            }
        }
    /*}*/
```

4.1.22 Create-Subject

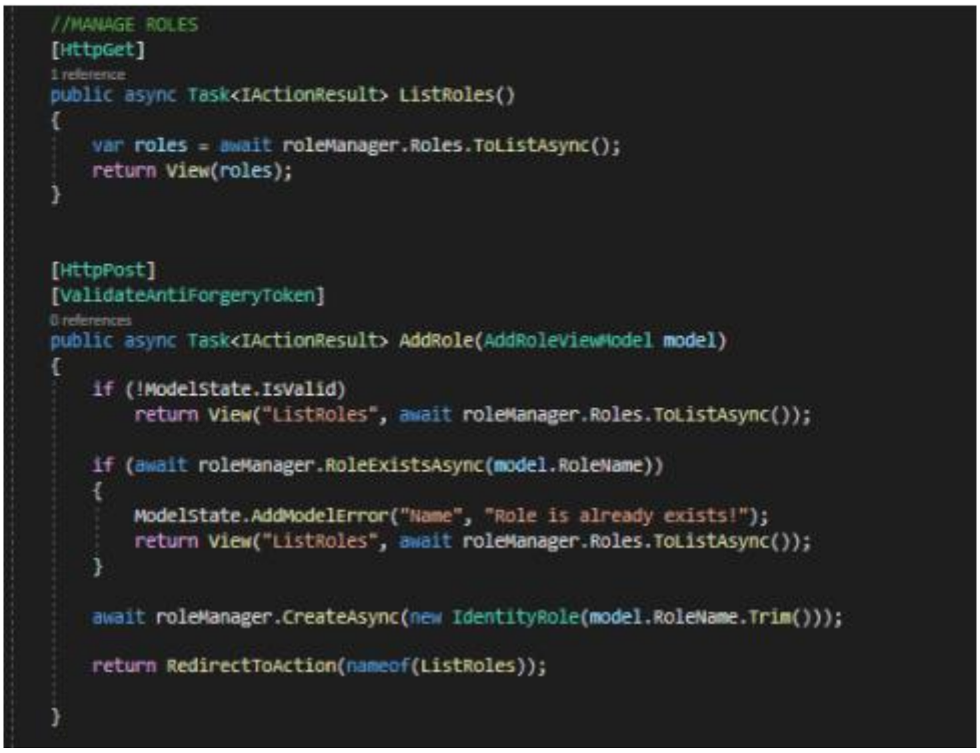


```
[HttpGet]
0 references
public IActionResult Create_Subject()
{
    List<Law> laws_list = _context.Laws.ToList();
    List<Part[illegible]> parts_list = _context.Part[illegible]s.ToList();
    List<Book[illegible]> books_list = _context.Book[illegible]s.ToList();
    List<Section[illegible]> sections_list = _context.Section[illegible]s.ToList();
    List<Chapter1> chapters1_list = _context.Chapter1s.ToList();
    List<Chapter2> chapters2_list = _context.Chapter2s.ToList();
    List<Chapter3> chapters3_list = _context.Chapter3s.ToList();
    Subjectsمواد subject = new Subjectsمواد();
    var model = new SubjectWithLawContentsViewModel
    {
        law_list = laws_list,
        part_list = parts_list,
        book_list = books_list,
        section_list = sections_list,
        chapter1_list = chapters1_list,
        chapter2_list = chapters2_list,
        chapter3_list = chapters3_list,
        subject = subject
    };
    return View(model);
}
```



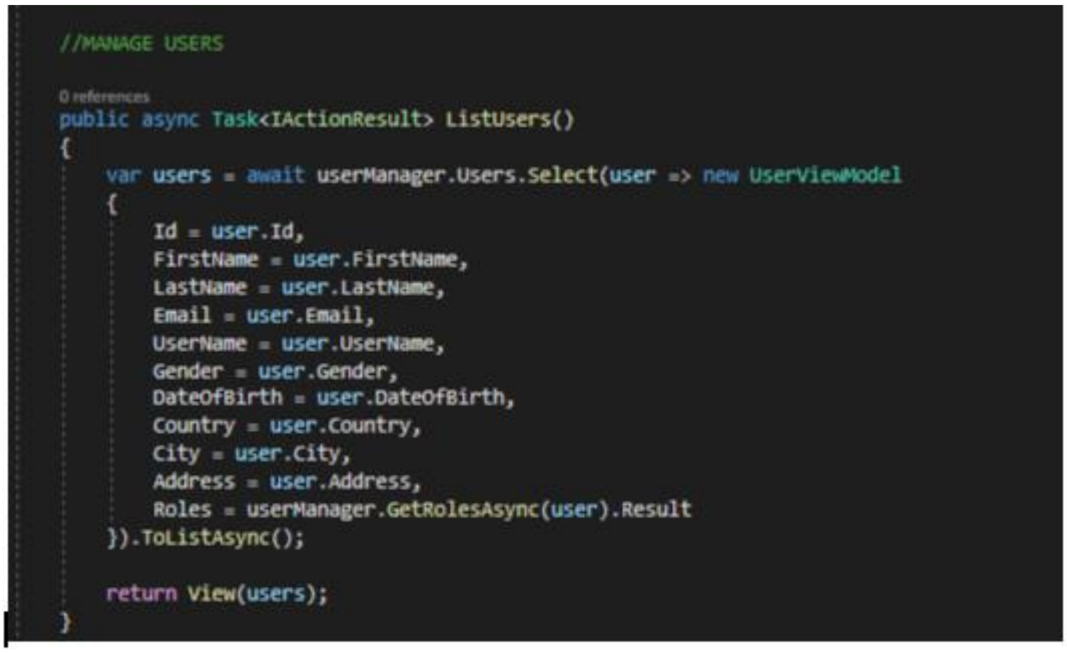
```
[HttpPost]
0 references
public IActionResult Create_Subject(SubjectWithLawContentsViewModel subjectviewmodel)
{
    if(subjectviewmodel.subject.[illegible] != null)
    {
        subjectviewmodel.subject.[illegible] = "[illegible]";
    }
    _context.Attach(subjectviewmodel.subject);
    _context.Entry(subjectviewmodel.subject).State = EntityState.Added;
    _context.SaveChanges();

    return RedirectToAction("SubjectPage");
}
0 references
public IActionResult Details(long Id)
{
    Subjectsمواد subjects = _context.Subjectsمواد.Where(p => p.IdSubjects == Id).FirstOrDefault();
    return View(subjects);
}
```

4.1.23 List-Roles



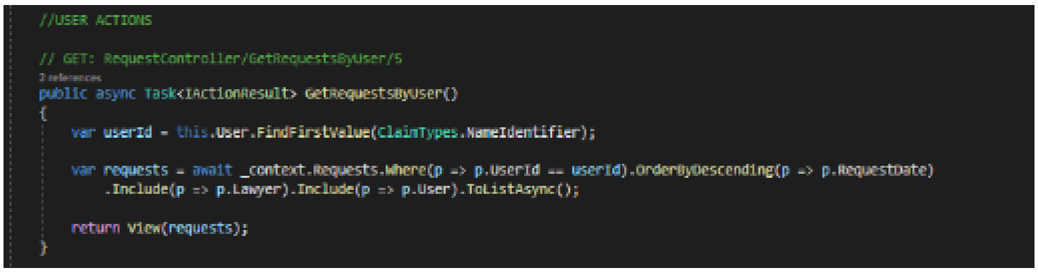
```
//MANAGE ROLES
[HttpGet]
1 reference
public async Task<IActionResult> ListRoles()
{
    var roles = await roleManager.Roles.ToListAsync();
    return View(roles);
}


[HttpPost]
[ValidateAntiForgeryToken]
0 references
public async Task<IActionResult> AddRole(AddRoleViewModel model)
{
    if (!ModelState.IsValid)
        return View("ListRoles", await roleManager.Roles.ToListAsync());

    if (await roleManager.RoleExistsAsync(model.RoleName))
    {
        ModelState.AddModelError("Name", "Role is already exists!");
        return View("ListRoles", await roleManager.Roles.ToListAsync());
    }

    await roleManager.CreateAsync(new IdentityRole(model.RoleName.Trim()));

    return RedirectToAction(nameof(ListRoles));

}
```

4.1.24 List-User



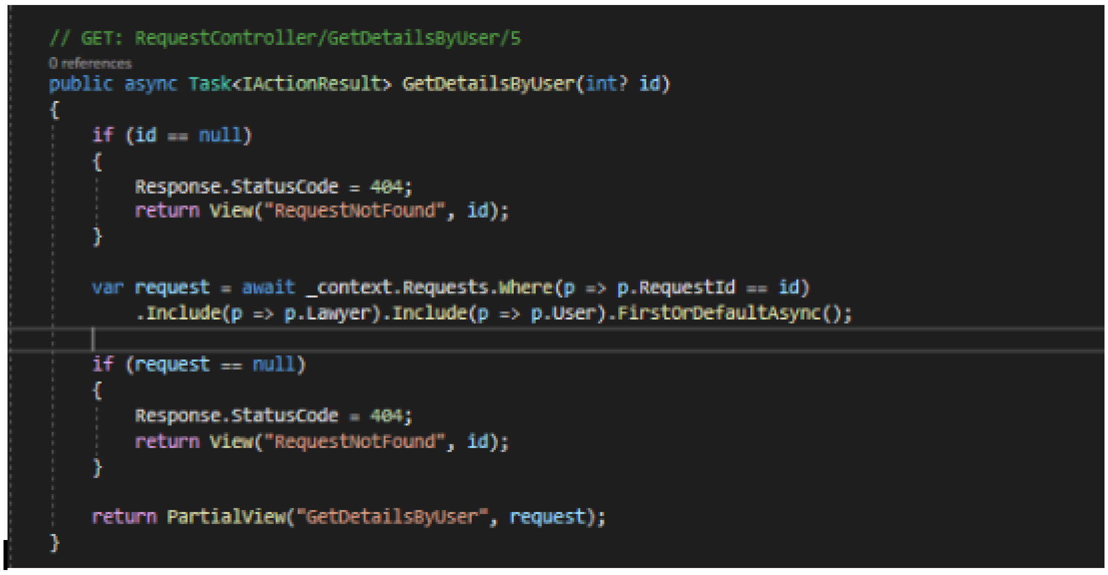
```
//MANAGE USERS

0 references
public async Task<IActionResult> ListUsers()
{
    var users = await userManager.Users.Select(user => new UserViewModel
    {
        Id = user.Id,
        FirstName = user.FirstName,
        LastName = user.LastName,
        Email = user.Email,
        UserName = user.UserName,
        Gender = user.Gender,
        DateOfBirth = user.DateOfBirth,
        Country = user.Country,
        City = user.City,
        Address = user.Address,
        Roles = userManager.GetRolesAsync(user).Result
    }).ToListAsync();

    return View(users);
}
```

4.1.25 Get Request By user



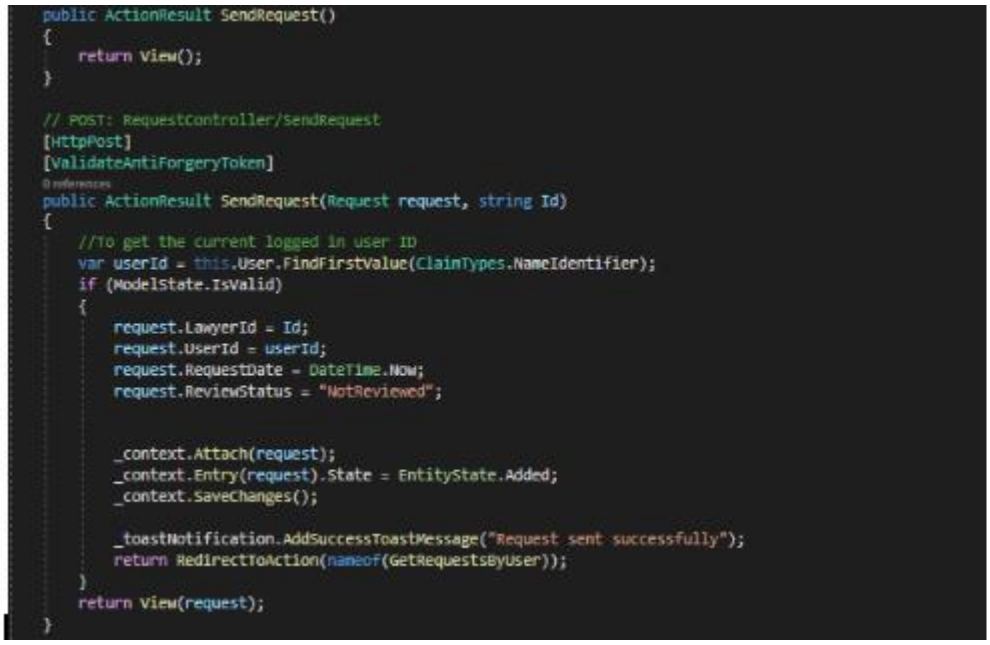
```
//USER ACTIONS

// GET: RequestController/GetRequestsByUser/5
2 references
public async Task<IActionResult> GetRequestsByUser()
{
    var userId = this.User.FindFirstValue(ClaimTypes.NameIdentifier);

    var requests = await _context.Requests.Where(p => p.UserId == userId).OrderByDescending(p => p.RequestDate)
        .Include(p => p.Lawyer).Include(p => p.User).ToListAsync();

    return View(requests);
}
```

4.1.26 Get Details by user



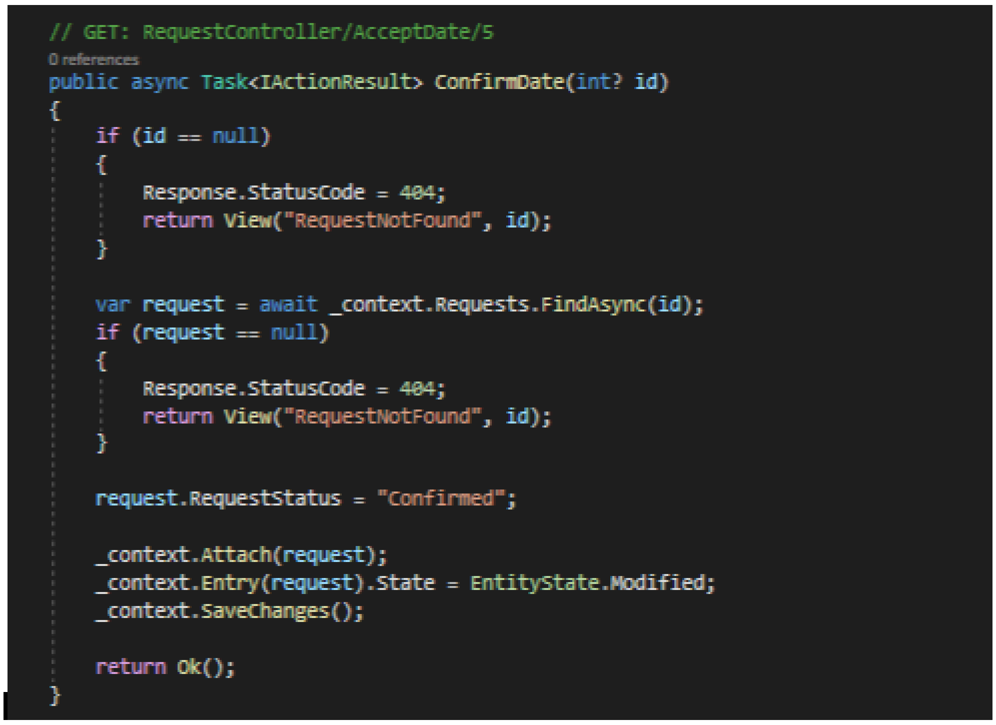
```
// GET: RequestController/GetDetailsByUser/5
0 references
public async Task<IActionResult> GetDetailsByUser(int? id)
{
    if (id == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    var request = await _context.Requests.Where(p => p.RequestId == id)
        .Include(p => p.Lawyer).Include(p => p.User).FirstOrDefaultAsync();

    if (request == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    return PartialView("GetDetailsByUser", request);
}
```

4.1.27 send Request



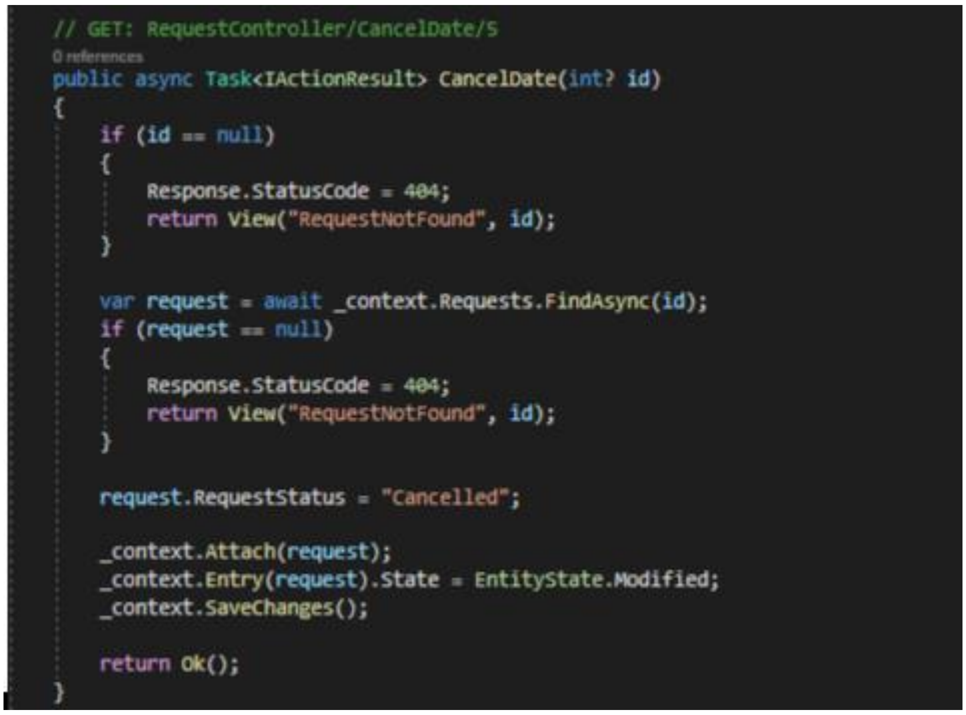
```
public ActionResult SendRequest()
{
    return View();
}

// POST: RequestController/SendRequest
[HttpPost]
[ValidateAntiForgeryToken]
0 references
public ActionResult SendRequest(Request request, string Id)
{
    //To get the current logged in user ID
    var userId = this.User.FindFirstValue(ClaimTypes.NameIdentifier);
    if (ModelState.IsValid)
    {
        request.LawyerId = Id;
        request.UserId = userId;
        request.RequestDate = DateTime.Now;
        request.ReviewStatus = "NotReviewed";


        _context.Attach(request);
        _context.Entry(request).State = EntityState.Added;
        _context.SaveChanges();

        _toastNotification.AddSuccessToastMessage("Request sent successfully");
        return RedirectToAction(nameof(GetRequestsByUser));
    }
    return View(request);
}
```

4.1.28 confirm date



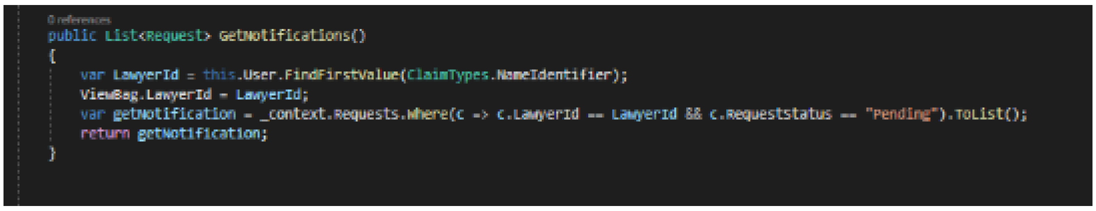
```
// GET: RequestController/AcceptDate/5
0 references
public async Task<IActionResult> ConfirmDate(int? id)
{
    if (id == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    var request = await _context.Requests.FindAsync(id);
    if (request == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    request.RequestStatus = "Confirmed";

    _context.Attach(request);
    _context.Entry(request).State = EntityState.Modified;
    _context.SaveChanges();

    return Ok();
}
```

4.1.29 cancel date



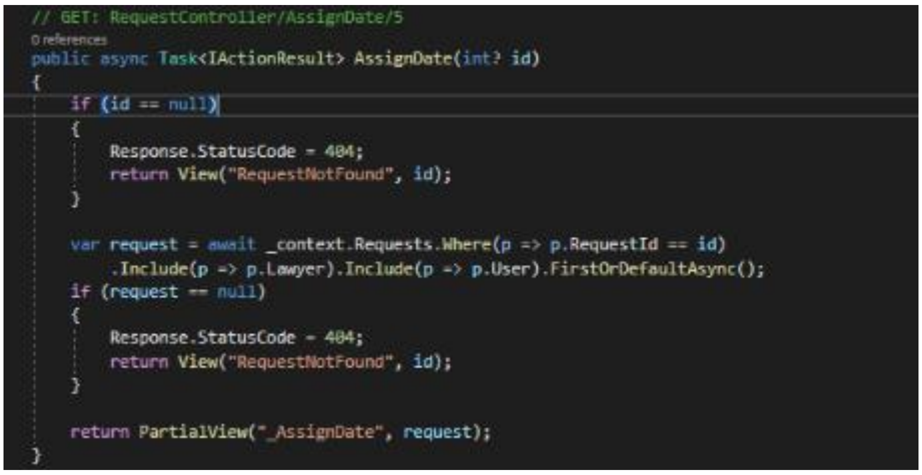
```
// GET: RequestController/CancelDate/5
0 references
public async Task<IActionResult> CancelDate(int? id)
{
    if (id == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    var request = await _context.Requests.FindAsync(id);
    if (request == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    request.RequestStatus = "Cancelled";

    _context.Attach(request);
    _context.Entry(request).State = EntityState.Modified;
    _context.SaveChanges();

    return Ok();
}
```

4.1.30 Notification



```
0 references
public List<Request> GetNotifications()
{
    var LawyerId = this.User.FindFirstValue(ClaimTypes.NameIdentifier);
    ViewBag.LawyerId = LawyerId;
    var getNotification = _context.Requests.Where(c => c.LawyerId == LawyerId && c.Requeststatus == "Pending").ToList();
    return getNotification;
}
```

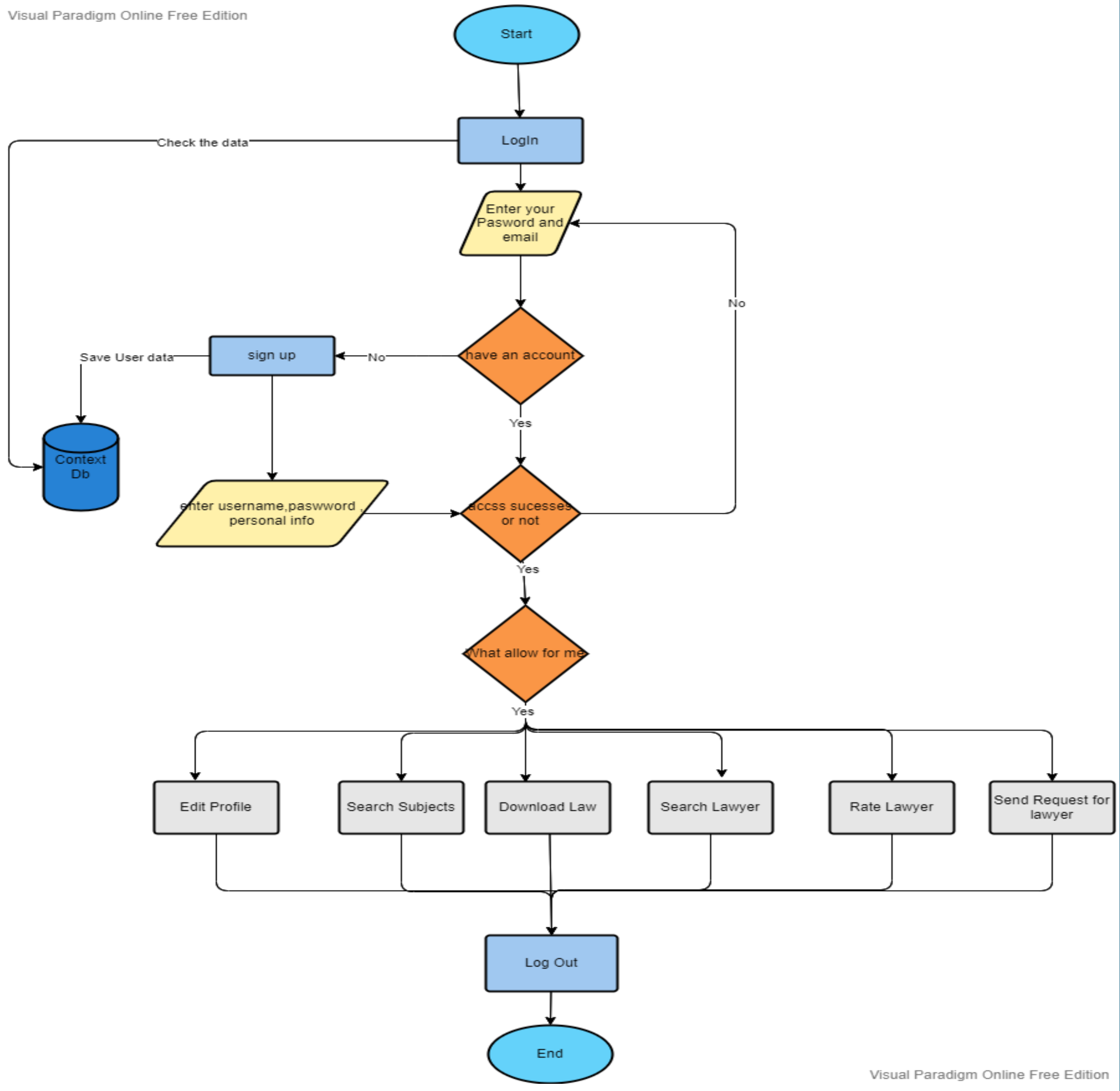
4.1.31 Assign Date



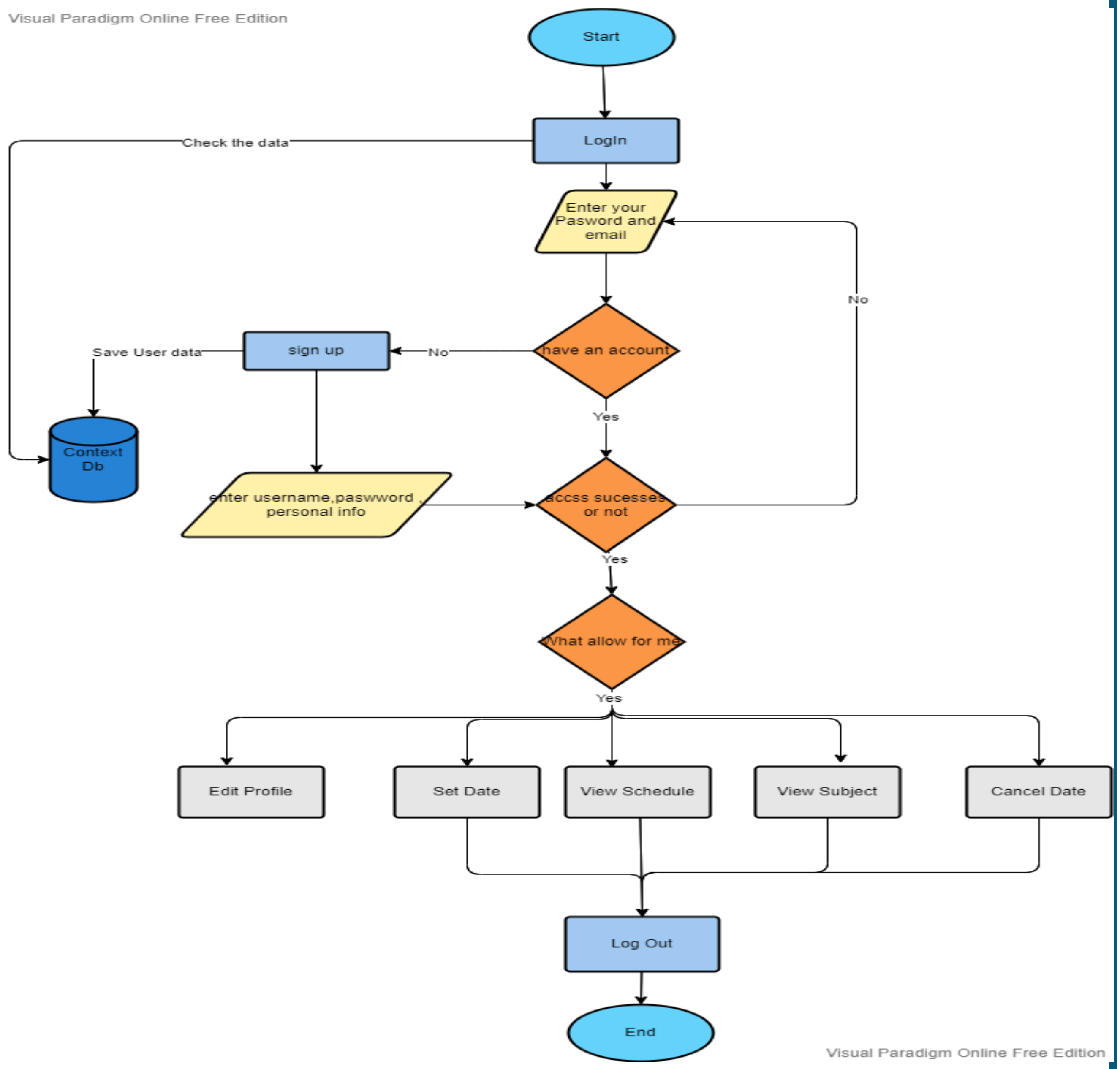
```
// GET: RequestController/AssignDate/5
0 references
public async Task<IActionResult> AssignDate(int? id)
{
    if (id == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    var request = await _context.Requests.Where(p => p.RequestId == id)
        .Include(p => p.Lawyer).Include(p => p.User).FirstOrDefaultAsync();
    if (request == null)
    {
        Response.StatusCode = 404;
        return View("RequestNotFound", id);
    }

    return PartialView("_AssignDate", request);
}
```

4.2 Flow Chart

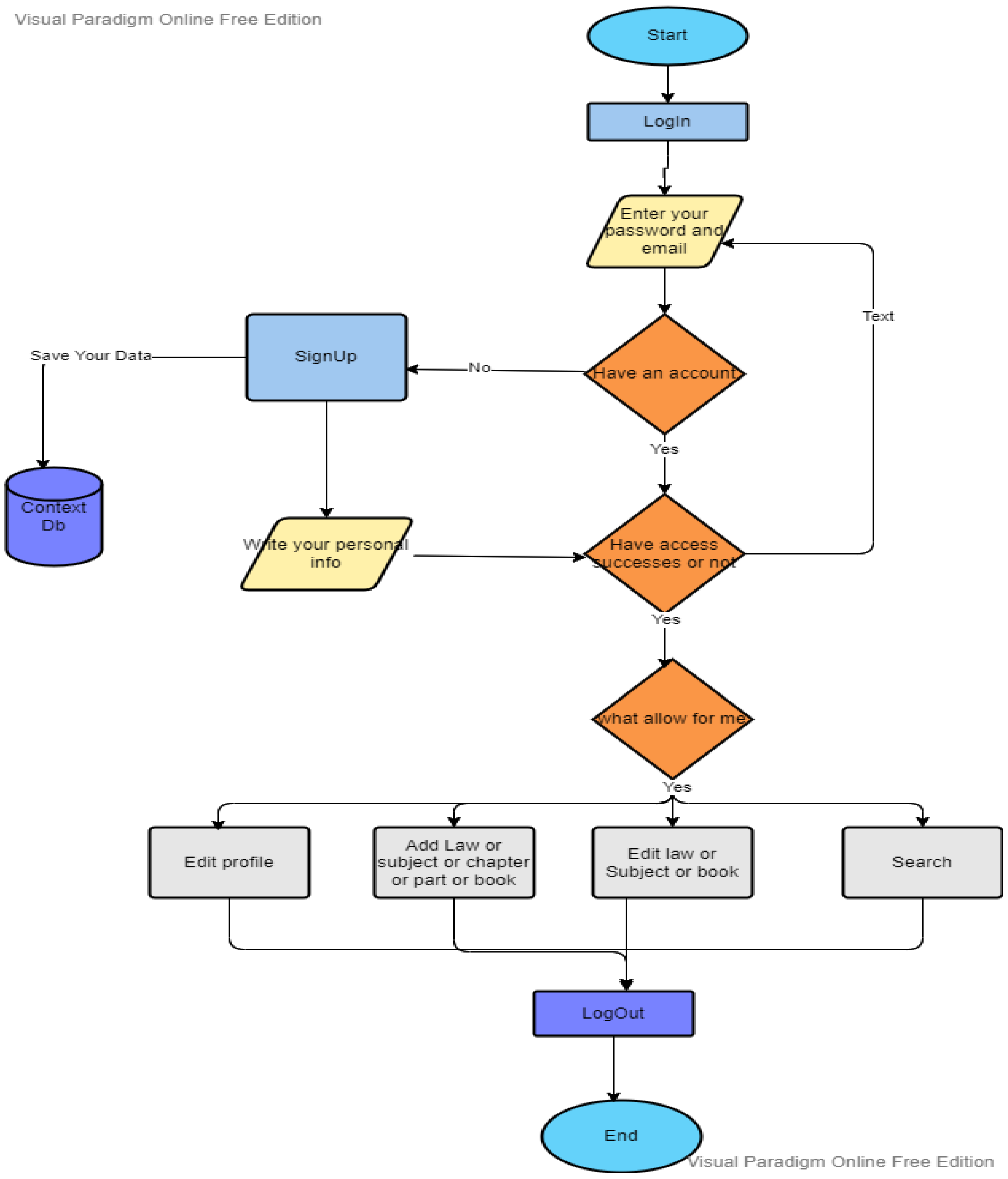
User flow chart



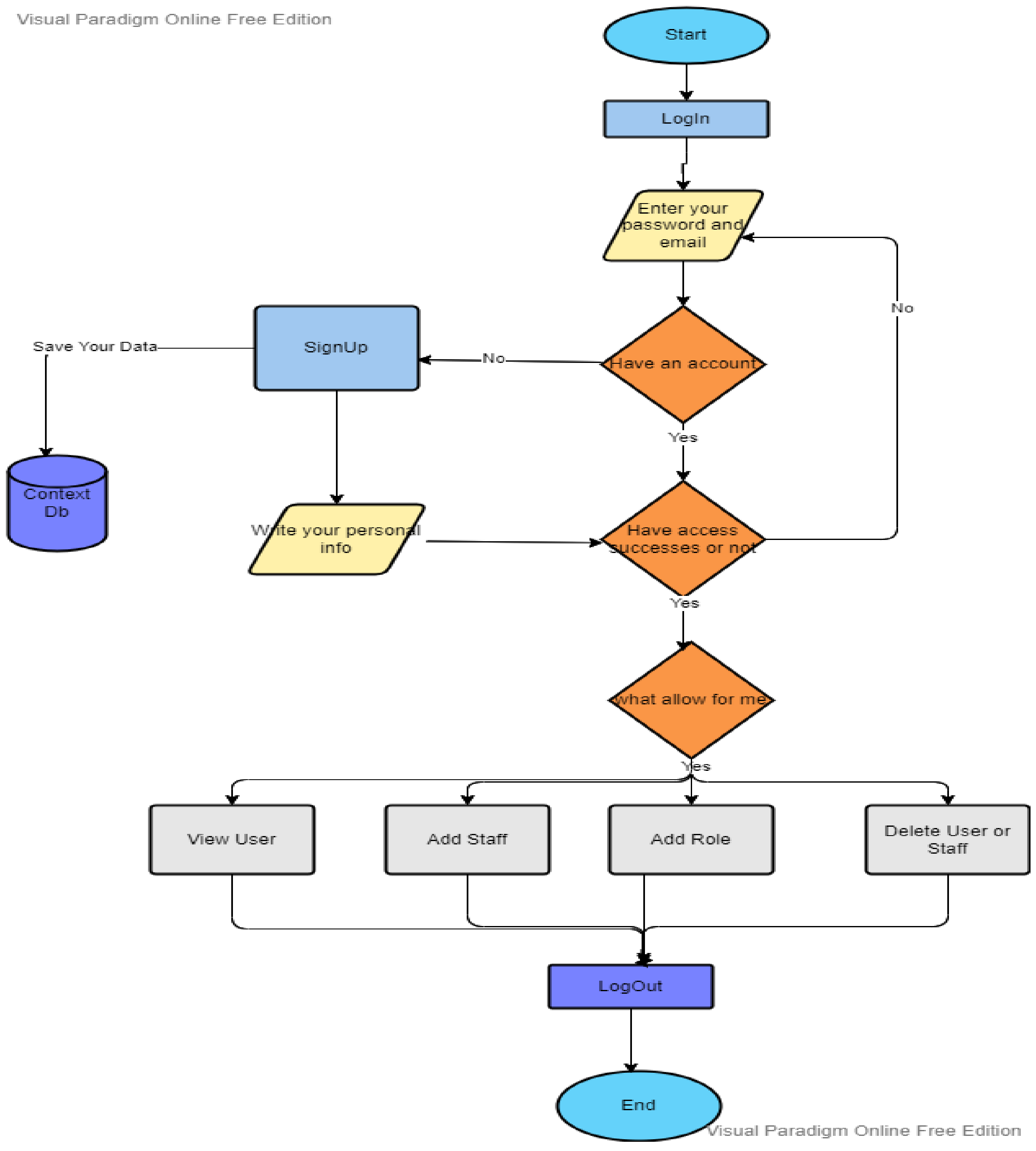
Lawyer flow chart



Staff Flow Chart


Visual Paradigm Online Free Edition
Start
LogIn
Enter your password and email
Text
Have an account
No
SignUp
Save Your Data
Context Db
Yes
Write your personal info
Have access successes or not
Yes
what allow for me
Yes
Edit profile
Add Law or subject or chapter or part or book
Edit law or Subject or book
Search
LogOut
End
Visual Paradigm Online Free Edition

Admin Flow Chart



Chapter 5:

Testing

5. Testing

In this chapter we are going to discuss and go deeper in Encyclopedia Of Law website testing, and present the types of testing to be used and test cases we examined our website through.

5.1 Functional Testing:

5.1.1 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.

5.1.2 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 which must interact with each other.

5.1.3 System Testing

Testing a system behavior as a whole when development is finished, and the system can be tested as complete entity.

5.1.4 Acceptance Testing

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

5.2 Non-Functional

5.2.1 Performance Testing

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

5.2.2 Load Testing

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

5.2.3 Stress Testing

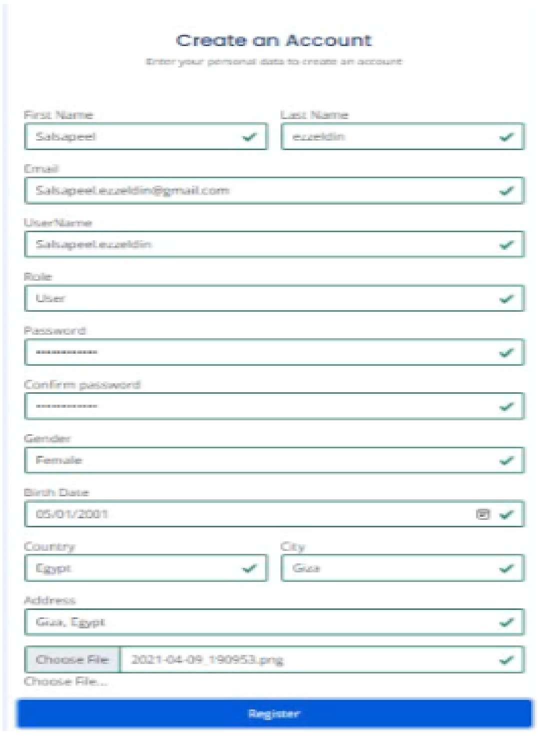
Evaluate a system or component at or beyond the limits of its specified requirements.

5.2.4 Security Testing

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

5.3Additional Testing

Project Name: Encyclopedia Of law



Create an Account

Enter your personal data to create an account

First Name: Salsapeel
Last Name: ezzeldin
Email: Salsapeel.ezzeldin@gmail.com
UserName: Salsapeel.ezzeldin
Role: User
Password: ••••••••••••
Confirm password: ••••••••••••
Gender: Female
Birth Date: 05/01/2001
Country: Egypt
City: Giza
Address: Giza, Egypt
Choose File: 2021-04-09_190953.png
Choose File...

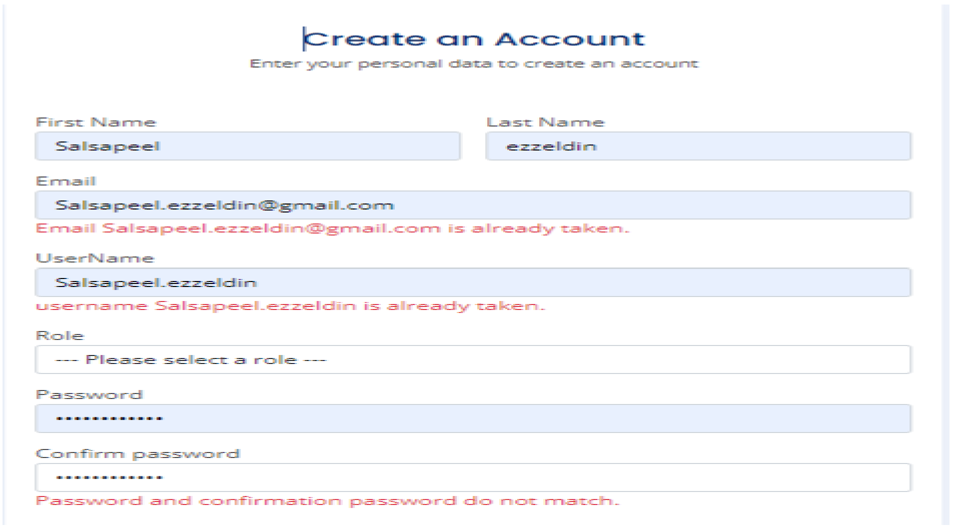
Register

Module Name: Register

Test case id: Register_1 Test case scenario: try to registration with valid and invalid data

Test Case	Test steps	Test data	Result
Register with complete and valid data	1-press sign up 2-enter data 3-press sign up	Personal data(name ,phone number, e-mail, password)	1-Account created successfully -logged in

Test Case	Test steps	Test data	Result
Register with complete and invalid data	1-press sign up 2-enter data 3-press sign up	Personal data(name ,phone number, email(already exist) ,password(invalid)), Username(already exist)	1-Account created successfully logged in



Create an Account

Enter your personal data to create an account

First Name: Salsapeel

Last Name: ezzeldin

Email: Salsapeel.ezzeldin@gmail.com

Email Salsapeel.ezzeldin@gmail.com is already taken.

UserName: Salsapeel.ezzeldin

username Salsapeel.ezzeldin is already taken.

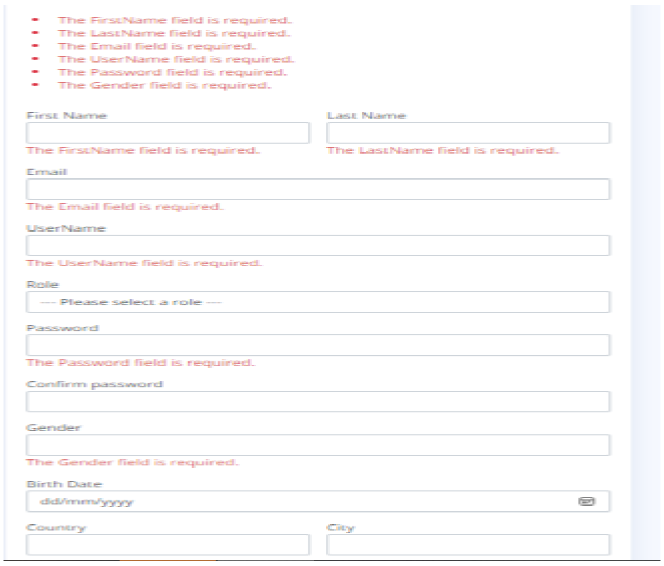
Role: --- Please select a role ---

Password: ••••••••••••

Confirm password: ••••••••••••

Password and confirmation password do not match.

Test Case	Test steps	Test data	Result
Register with incomplete and valid data	1-press sign up 2-enter data 3-press sign up	Personal data(,phone number, e-mail, password)	Display error message to co complete empty fields



- The FirstName field is required.
- The LastName field is required.
- The Email field is required.
- The UserName field is required.
- The Password field is required.
- The Gender field is required.

First Name
The FirstName field is required.

Last Name
The LastName field is required.

Email
The Email field is required.

UserName
The UserName field is required.

Role
--- Please select a role ---

Password
The Password field is required.

Confirm password

Gender
The Gender field is required.

Birth Date
dd/mm/yyyy

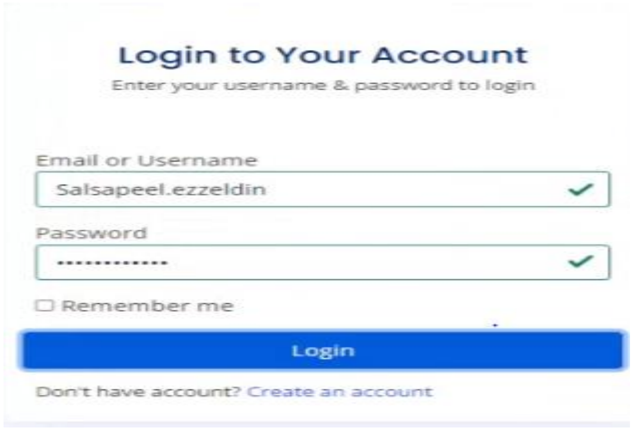
Country

City

Module Name: login

Test case id: login_2

Test case scenario: try to login with valid and valid email and password



Login to Your Account

Enter your username & password to login

Email or Username
Salsapeel.ezzeldin

Password
•••••••••••••

☐ Remember me

Test Case	Test steps	Test data	Result
login with valid email and valid password	1-enter email 2-enter password 3-press login	Valid Email Valid password	Logged successfully

Test Case	Test steps	Test data	Result
login with invalid email and invalid password	1-enter email 2-enter password 3-press login	invalid Email invalid password	Display invalid Login

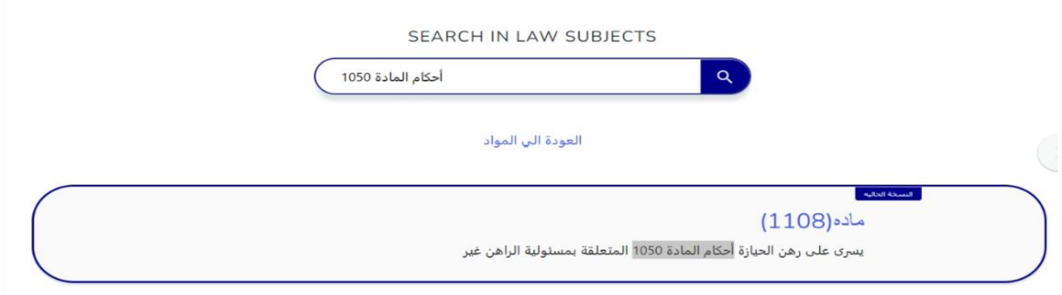
---

Module Name: search

Test case id: search

Test case scenario: try to search for law

Test Case	Test steps	Test data	Result
Search with part of subject	1-enter system 2-search	Part of subject	Display subject that contain this part


SEARCH IN LAW SUBJECTS

أحكام المادة 1050

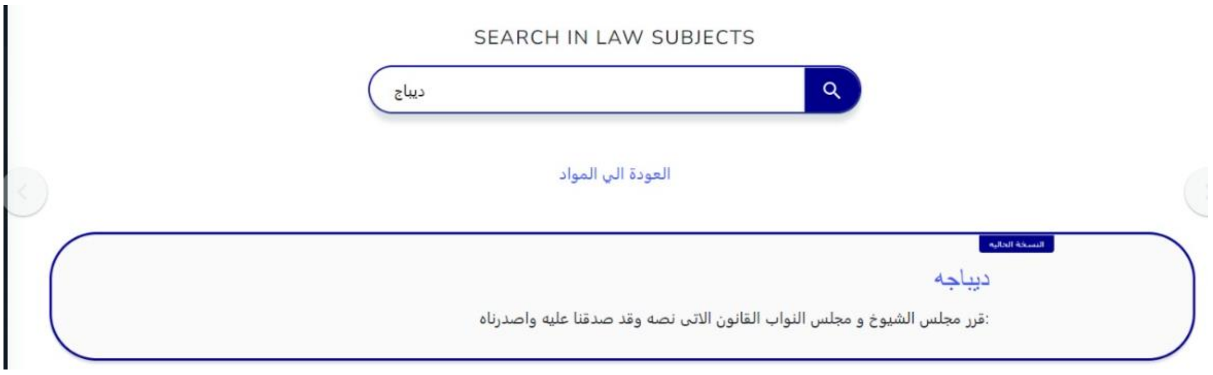
العودة الي المواد

النسخة الحاليه

ماده(1108)

يسرى على رهن الحيازة أحكام المادة 1050 المتعلقة بمسئولية الراهن غير

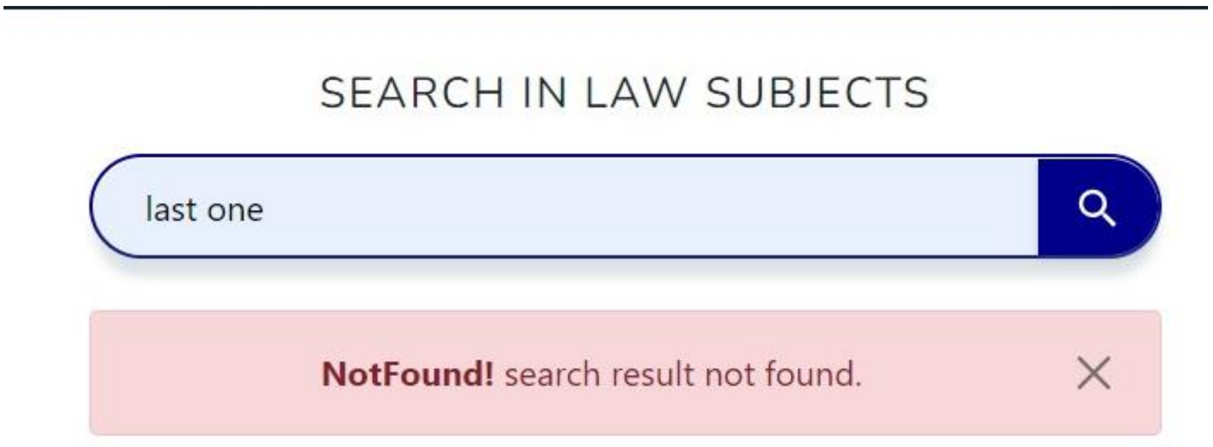
Test Case	Test steps	Test data	Result
Search with name of subject	1-enter system 2-search	valid name of subject	Display subject



ديباجه

قرر مجلس الشيوخ و مجلس النواب القانون الاتى نصه وقد صدقنا عليه واصدرناه:

Test Case	Test steps	Test data	Result
Search with invalid name	1-enter system 2-search	Invalid name of subject	Display message not found



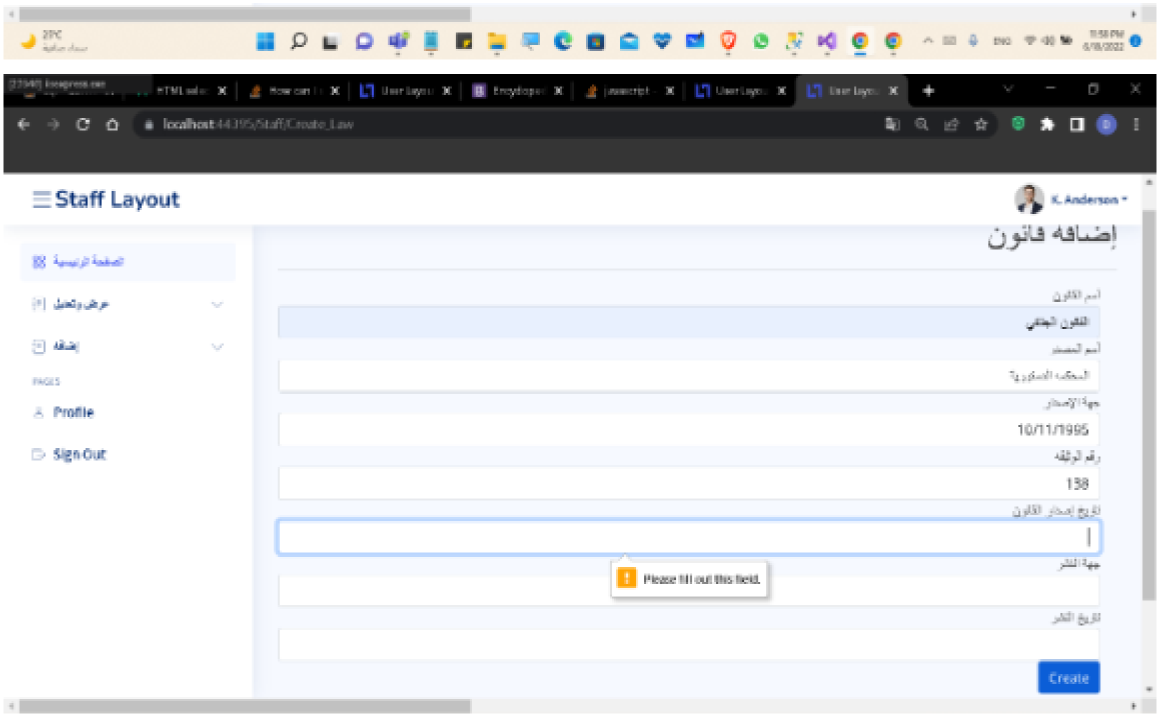
NotFound! search result not found.

Module Name: create-law

Test case id: create-law

Test case scenario: try to create

Test Case	Test steps	Test data	Result
Create law with incomplete data	1-log in as staff 2- create law	Incomplete data	Do not create


localhost:44395/Staff/Create_Law

Staff Layout

K. Anderson

إضافه قانون

اسم القانون
القانون الجنائي
اسم المصدر
المحكمة الدستورية
جهة الإصدار
10/11/1995
رقم الوثيقة
138
تاريخ إصدار القانون

Please fill out this field.

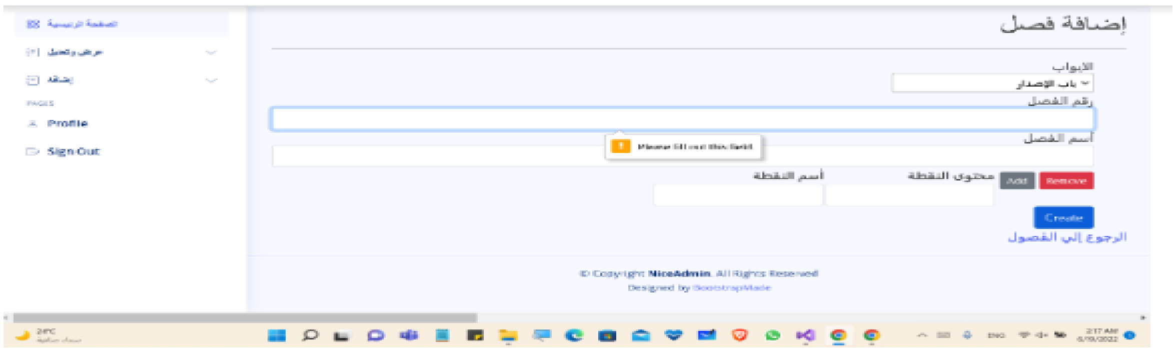
جهة النشر
تاريخ النشر

Module Name: create-chapter

Test case id: create

Test case scenario: try to create

Test Case	Test steps	Test data	Result
Create chapter with incomplete data	1-log in as staff 2- create chapter	Incomplete data	Do not create


إضافة فصل

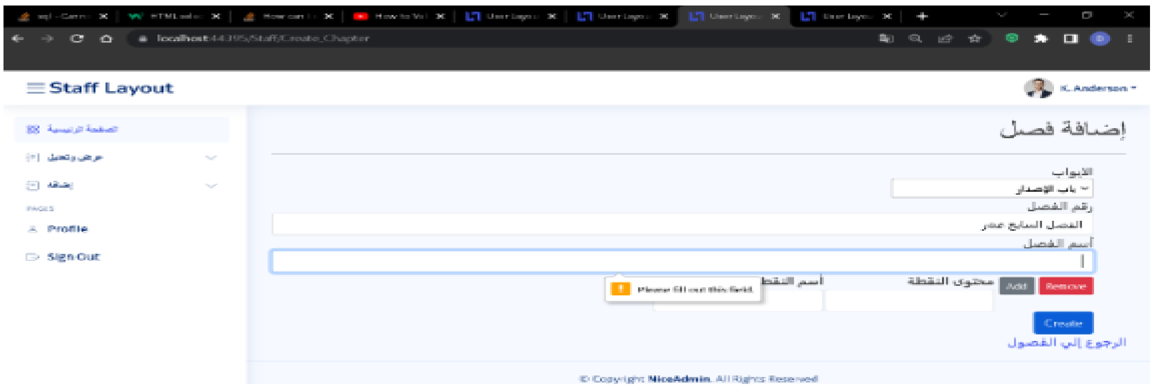
الابواب: باب الإصدار

رقم الفصل:

Please fill out this field.

أسم الفصل:

محتوى النقطة / أسم النقطة


localhost:44395/Staff/Create_Chapter

Staff Layout

إضافة فصل

الابواب: باب الإصدار

رقم الفصل: الفصل السابع عشر

أسم الفصل:

Please fill out this field.

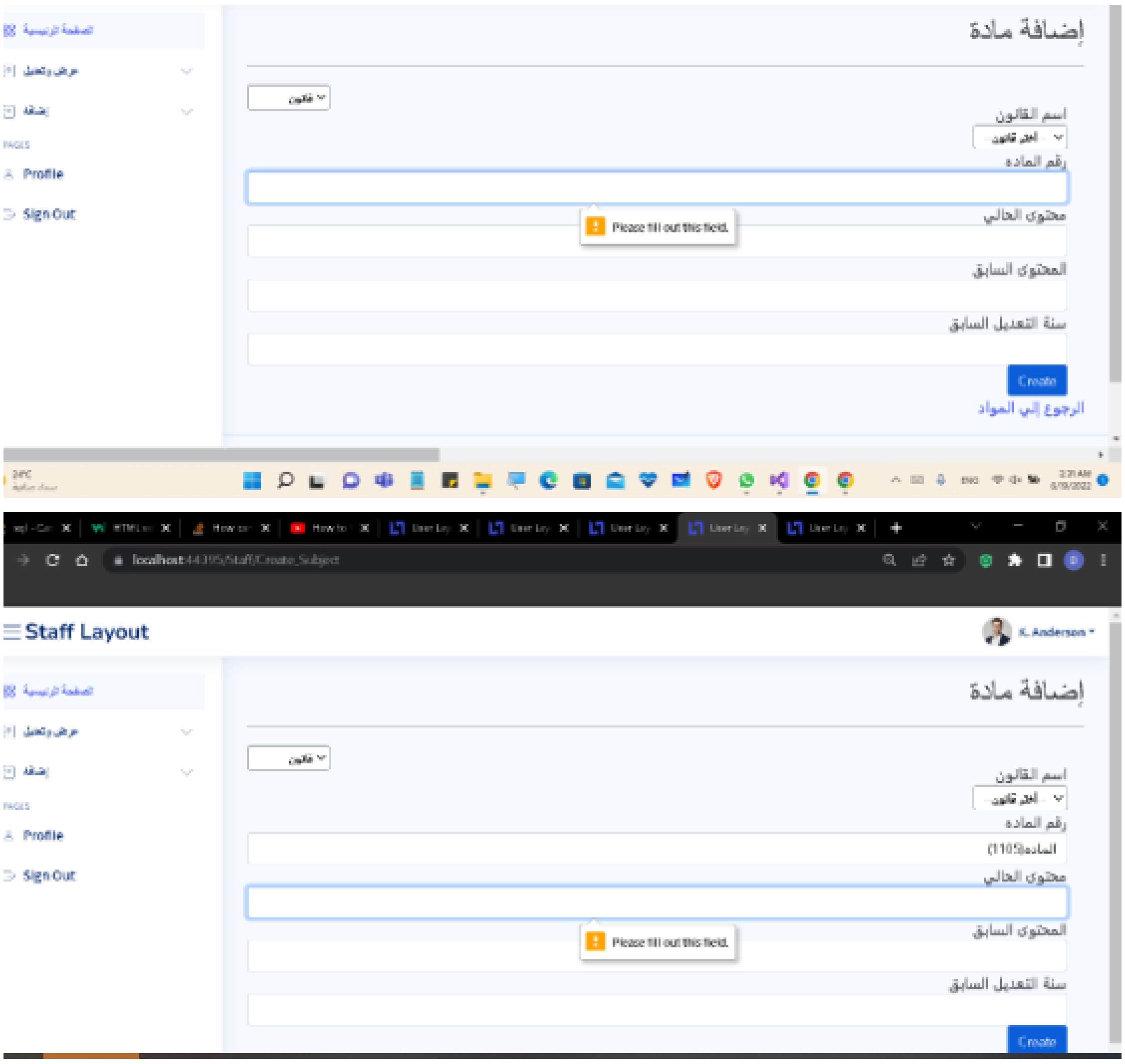
محتوى النقطة / أسم النقطة

Module Name: create-subject

Test case id: create

Test case scenario: try to create

Test Case	Test steps	Test data	Result
Create subject with incomplete data	1-log in as staff 2- create subject	Incomplete data	Do not create



Chapter 6: Result and Discussion

6. Result and Discussion

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

6.1 Results

6.1.1 Expected Result

- Different users can sign in ,log in and log out
- Users can edit their profile
- Users can search for a specific subject
- Users can search for a specific lawyer
- Users can send feedback
- Users can set new password (forget password)
- Clients can send request to lawyers to set an appointment
- Clients can cancel the request
- Lawyers can set or cancel an appointment with the clients
- Users can download the law book
- Only admins can edit ,add or update a subject
- Admins can add or delete users
- User can ask for support in the system to help them in their issues
- Clients can view lawyers profile and vise versa
- Clients can rate the lawyer on his work
- Lawyers can view their schedules with clients
- All users have notifications in their page
- Applicant have both dark and light mode

6.1.2 Actual Result

- Different users can sign in ,log in and log out
- Users can edit their profile
- Users can search for a specific subject
- Users can search for a specific lawyer
- Users can send feedback
- Users can set new password (forget password)
- Clients can send request to lawyers to set an appointment
- Clients can cancel the request
- Lawyers can set or cancel an appointment with the clients
- Users can download the law book
- Only admins can edit ,add or update a subject
- Admins can add or delete users users
- User can ask for support in the system to help them in their issues
- Clients can view lawyers profile and vise versa

- Clients can rate the lawyer on his work
- Lawyers can view their schedules with clients
- All users have notifications in their page
- Applicant have both dark and light mode

6.2 Discussion

There is a little difference between the expected and actual result except because of weakness of local server for allowing to reading books and see what your friends are interesting.

We have added new features into our application such as: Chatting and reporting.

Chapter 7: Conclusion

7. Conclusion

Eventually, during this project we tried to create the most effective we will avoid exploitation and it is simply the way of expressing our duty towards our society as we felt accountable enough to participate in creating this community a stronger place with one amongst the foremost powerful tools we
can use and that we selected as our finding out field years past that is **TECHNOLOGY**.

In summary, we tried to do our best in Encyclopedia of law web application and provide a lot of real and helpful resources and features we help the lawyers and also the Citizen and they first sign up with the name and age and email address and also a unique password

Then he can log in and search for the content of the law, or search for a lawyer or work through the application if the user is a lawyer.

Our project aims to connect the online world with the offline one to help lawyers and citizens.

Hope that is a satisfied application for you and hope we continue to develop it to help The Judicial and Legal Authority

Chapter 8: Future Work

8. Future Work

We are going to add more and more features to our application such as:

1.Adding other laws alongside the civil law

2.Using a new and updated datasets to add provisions, legislation and any document related to the law and the judiciary.

3. Chat Room To educate the user about their rights regarding an issue.

4. Increase information about recommendation lawyers

So, we will not miss any chance to keep working as a team on

that project and enhancing it after graduation as we are looking forward to turning this application into work environment that serves the citizen.

Our dream started with this idea months ago, and we will insist on not letting go of it and not letting it be just a dream

Bibliography

https://www.laweg.net//Default.aspx?action=HP

http://muqtafi.birzeit.edu/links.aspx

https://elpai.idsc.gov.eg/

https://manshurat.org/taxonomy/term/29

https://dostour.eg/

“If you do not like to read you have not found the right book.”

Thank You!