جامعة ألزهر – غزة

Al-Azhar University – Gaza

FACULTY OF ENGINEERING AND INFORMATION TECHNOLOGY

DEPARTMENT OF SW ENG.

FINAL REPORT [FYP II]
ENGINEERING STUDENTS' ADVISOR APP

Bilal Othman Hamed Al Najjar	20171810
Saif eddeen Sami Deeb Obaid	20173065
Zaki Khaled Zaki Abo Qaoud	20172106

SUPERVISED BY
Dr Mosab M. Ayiad

September 2022
Semester I 2022/2023

FINAL YEAR PROJECT REPORT
ITSE 5330

ENGINEERING STUDENTS' ADVISOR APP

by

Bilal Othman Hamed Al Najjar	20171810
Saif eddeen Sami Deeb Obaid	20173065
Zaki Khaled Zaki Abo Qaoud	20172106

SUPERVISED BY
DDrMosab M. Ayiad

In partial fulfilments the requirement for the
Bachelor of

Department of SW Eng.
Faculty of Engineering and Information Technology

September 2022
Semester I 2022/2023

CERTIFICATE OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein has not been taken or done by unspecified sources or persons.

_____

Bilal Othman Hamed Al Najjar	20171810
Saif_Eddeen Sami Deeb Obaid	20173065
Zaki Khaled Zaki Abo Qaoud	20172106

ACKNOWLEDGEMENT

Praise and thanks to Allah first and foremost whose blessing enabled us to accomplish this project.

We wish to express our deepest appreciation to our supervisor DDrMosab M. Ayiad and our ex-supervisor Dr Maher Shamia for relentless guidance, helpful suggestion, close supervision and moral encouragement to complete this task.

A special thanks to our parents and to all otheteachers we have had. Thank you to DDrMosab M. Ayiad and Dr Maher Shamia.

My ssincerethanks to all those whowhorectly or indirectly help us to complete this project.

ABSTRACT

This project involves the development of aanapplication for supporting students of SW Eng. and MecTr Eng. majors in local universities in the Gaza Strip. The work aims to provide engineering students with general resources that contain descriptions, explanations, and various supporting materials for their studies. Also, the application allows students to keep track of their progress in their study plans and enables the creation of short-term schedules for their daily, weekly or even monthly sstudies

TABLE OF CONTENTS

Contents

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

Chapter One

Introduction

Chapter One: Introduction

In this chapter, the introduction of the document is written, also general information about project specification area and project team members are documented.

1. Preamble

Nowadays, different websites have been used in many different fields of life just like commercial, educational, social, entertainment and other different fields. And that because of the good view of information in the websites, also because the ease of using of, dealing with, and access to different applications.

In this document a development process of a new educational application software will be discussed and viewed in detailed. The next sections of this document will show the idea of this new application and how it will be analyzed.

2. Project Team

1.1. Team Members

The team that works on this project is a software engineering team from AL AZHAR University – Gaza. The team consists of three Software Engineers that are studying at AL AZHAR University – Gaza, and they are in their fifth level of the study. The team's name is ***Simple Soft***.

The following table shows the main skills of each team member whose work on this project and contact information:

Table 1: Team Members

Member Name	*Email*	*Mobile NO.*	
Saif eddeen Sami Deeb Obaid	sdsdseao99@hotmail.com	0592761267	***Leader***
Zaki Khaled Zaki Abo Qaoud	b.zaki181999@gmail.com	0595078069	
Bilal Othman Hamed Al Najjar	blalthmen75@gmail.com	0598242120	

1.2. Skills Baseline

Each member of the project team has some skills that enable him to work on this project. The following table shows the basic skills of each member of the project team.

Table 2: Team Members' Skills

Name	*Skills*
Saif eddeen Sami Deeb Obaid	• Technical Writing. • System analysis and design. • Web frontend developer using: (Html, CSS, JS). • Web backend developer using php Laravel framework, and MySQL. • Web backend programming using php Laravel framework. • Managerial skills in software management field.
Zaki Khaled Zaki Abo Qaoud	• Web frontend developer using: (Html, CSS, JS, jQuery, React JS). • Web backend developer using php Laravel framework, and MySQL. • Technical Writing. • System analysis and design. • Managerial skills in software management field.
Bilal Othman Hamed Al Najjar	• Web frontend developer using: (Html, CSS, JS, jQuery, Sass, Bootstrap, Vue.js, Livewire, React JS). • Web backend developer using php Laravel framework, and MySQL. • Technical Writing. • System analysis and design. • Managerial skills in software management field.

3. Project Scope

3.1. Area of Specification

The software is a guidance application that guide students of engineering in their study in the university. As a start of the application, applications' contents are divided into two departments, one for Software Eng., and the other for Mechatronics Eng. Each department in the website will contain many sections that guides students and support them in their study such like: major overview, major scientific plan, studying stages of the major, and a proposed path contains what, how, and when to study a specific external or internal resource for each subject in the major to guide students.

3.2. Targeted Users

The application discussed in this document will have three user classes: browsers, students, and admins. The characteristics of each user class are as follows:
First, the students: choose their engineering department, follow the study plan path, and benefit from the subject's content.

Second, the browsers: browse the site and its main contents without the need to be logging in to the site.

Third, the admins: they are the only personals who can add to and manipulate the contents of the application.

3.3. Platform Requirements

There is no special platform needed for this application to work, just like any other web application it needs a domain server to store the contents of the site on. Also, in the client and user side, the application needs a good browser that support modern JavaScript to make the application works in the best possible way.

4. Motivation

While the idea of this project has been produced by the team works on this project, the team members believe that finishing this project will help engineering students, especially students of software engineering and mechatronics engineering at the beginning, in their study.

There were two main reasons that motivate the project team members to work on this project, the first is that they need an useful idea for their final year project in their university study, and they believe that this project will benefit the users of its production.

The other reason is that they want to solve the problem that the most of engineering students suffer from which is the problem of not finding a one source, especially local source, that have a clear and wide explanation for engineering majors and their subjects with various supporting sources and materials that guides students of those departments to finish their study with the largest possible benefits from their study.

5. Report Structure

This document constructed from six main chapters, each chapter talks about standard documentation phase, software development phase, or a customed phase needed for the project work.

In the document chapter one, the introduction of the document is written, also general information about project specification area and project team members are documented.

Chapter two introduces the project through the project abstract and the project background, with a detailed introduction of the problem that intended to be solved through this work.

Chapter three demonstrates the methodologies that used in analyzing the application, and the specifying the main characteristics of the application as well as the content of the application.

In chapter four, the requirements analysis for the application system is introduced in detailed through specifying functional and non-functional requirements, and doing an UX quick analysis for the system.

After that, chapter five came to illustrates the specification of the system functional requirements by specifying stakeholders, system actors, and system use cases. Also, it specifies the system use cases by describing it and then drawing the use cases diagram.

Chapter six came then to take the project to the design phase, it describes the system overall structure, demonstrates the class diagram and sequence diagrams, and then introduces the database design for the application.

Next, chapter seven is written to demonstrate the implementation process of the application. It illustrates the implementation phases which are: the backend development, the frontend development, and the application execution.

6. Glossary

The following table will contain the special terms that used in this project and its intended meaning to provide the user a clear understand of this project.

Table 3: Project Terms and Definitions

Term	***Definition***
The web application	The website that will be developed.
The application	The web application.
SW	Software.
MecTr	Mechatronics.
Eng.	Engineering.
Particular plan	Plan that a student makes by himself.
Schedule	Particular plan.
Special plan	Particular plan.
Scientific plan	A major plan
General plan	Scientific plan
Major	Engineering major, ordinary SW Eng. or MecTr Eng..
Department	Major.
Subject	Major subjects or materials.
Sub subject	A main branch of the subject.
Application pages	Application web pages.

Chapter Two

Background

Chapter Two: Background

Nowadays, the use of learning websites in education becomes essential, especially the supporting educational applications.

The project assumed to be worked on in this documentation is suggested to be a supporting application to help students of Mechatronics and SW Eng. and guide them in their study. This application will contain many plans to guide students in their study, and also will contain many supporting procedures to help students finish their study with the largest possible benefits.

1. Project Background

The software is an application that guides students of engineering majors in their study. In the first version of the application, contents will be divided into two departments, one for SW Eng., and the other for MecTr Eng.

In general, the contents of each department in the application are major overview, major scientific plan, studying stages of the major, sub majors of the major, and a proposed path containing what, how, and when to study a specific internal or external resource for each subject in the major.

This application will also enable students to make their schedule and will enable them to keep tracking what they finish and where they reach in their major depending on the scientific plan of the major.

2. Problem Statement

2.1. Problem in General

In the local community, lot of the engineering students suffer from the problem that there isn't a clear and specific resource that guides them in their study and the problem of not knowing what is the best practices and the benefits of a specific subject unless they finish studying it.

Another problem is the difficulty of finding a lot number of different resources and supporting materials for a specific subject in the major in a specific place.

Also, the randomness of studying because of the lack of student planning is one of the most problems that any engineering student could face.

2.2. Related Work

Many applications include the learning and guiding goals in their content, and in the way that they introduce their content. The next lines will show an application that contains some points that are similar to points in this application and will describe these similarities and the advantages of this application.

EDX application.

1) Introduction:
   EDX application offers the highest quality online subjects from institutions that share our commitment to excellence in teaching and learning.

   EDX website link: https://www.edx.org/new/search.

   Here is the logo of the website:

   Figure 1: EDX Website Logo

2) Background:
   Harvard University and the Massachusetts Institute of Technology (MIT) established the EDX application in 2012 to

make the education system available to all people and solve both the cost and distance problems.
EDX application provides very strong subjects and accredited certificates to students from different universities and institutions all over the world such as Chicago University, Oxford University, IBM Company, and Microsoft Company. It gives the possibility of translating lectures into the student's language and downloading them to the student's computer. It also provides programs for bachelor's, master's, and doctoral degrees.

3) Objectives:
   1. Promoting student self-education.
   2. Saving the student's time and effort.
   3. Enhancing the students' knowledge and expanding their awareness.
   4. Producing a more efficient student.

4) Similarities:
   Both applications offer a set of subjects, each containing an overview, books, videos, and notes.

   Also, both of those support the style of containing many supporting ways of teaching their subjects.

   Also, both of those supporting the style of containing many of supporting ways of teaching their subjects.

5) Advantages of Engineer Guidance Application:
   - Engineer Guidance Application is instructive and educational, unlike the EDX Application, which is limited to the educational aspect only.

   - Engineer Guidance Application provides a general plan for each major divided into levels that are divided into subjects with the ability to track them.

   - Engineer Guidance Application enables the student to create his special plan and track his achievement path.

2.3. The Proposed Solution

After understanding the problem, it appears that engineering students need a guide in their study trip, and the perfect guide should contain a clear and concise path for studying, a lot of different supporting resources, and a feature that enables the student to schedule his studying plans, and the proposed application will be the best guider.

3. Project Objectives

After finishing this project, many objectives and goals should be accomplished, these objectives are:

1) ***Providing quick access to educational resources***: such as books and videos: by uploading links to other helping websites, videos and books, also by uploading any available documents.

2) ***Saving students time and effort***: because of the different supporting resources the student found in the application.

3) ***Enhancing the student's ability to search for different resources***: since that the application contains links to different types of resources.

4) ***Promoting learning by modern means through applications***: by using the application and the learning techniques that will develop.

5) ***Enhancing students' knowledge and expanding their awareness*** by expanding the explanations of a special object.

6) ***Producing a more efficient and more experienced student***: by explaining the path of the student major and providing him with needed resources throughout his study.

Chapter Three

Application Analysis

Chapter Three: Application Analysis

1. Analysis Methodology

In the analysis phase, sophisticated brainstorming and analysis are made before reaching the complete specification of the application. This section presents two methods that have been used to collect some functional requirements for the application. The full list of functional requirements will be in *chapter four*.

1.1. Elementary Analysis of The Applications' Pages

The first step for starting collecting functional requirements and putting a general concept for this new application was this general and quick analysis. *See Appendix A*.

This quick analysis starts with putting the main web pages in the application and the main purpose for each page. Also specifying the content of the essential pages in this analysis has been done.

After that in this quick analysis, a simple sitemap for the application pages that have been imagined before in this quick analysis has been drawn. This sitemap describes how the transmitting can be done between the application pages, from any page to other possible pages.

At the end of this quick analysis, an approximate overall structure of the application has been imagined, as well as many functional requirements have been collected.

Here are the functional requirements that appear from this quick analysis:

1. The Application must describe a general plan for each engineering major (SW Eng. and MecTr Eng. initially).

2. The Application must allow the user to contact application owners from any application page.

3. Each engineering department must contain an overview of its science.

4. The application must enable users to be students in the application (registration).

5. The application must have at least one admin to manage its contents.

6. Students have a profile page in the application.

7. Only when a student signs in can go to his profile page.

8. The application should allow the student to create a study plan.

9. The application should allow the student to update his study plan.

This quick analysis with full details is in *Appendix A*.

1.2. Surveying Engineering Students

As this new application will be made to help students of engineering in their study, it was essential to know their imaginations and needs that should be included in this new application. See *Appendix B*.

These questions were put in a way that enables the project team to collect some functional requirements in specific fields of the application.

The students that the questions made for are students of SW Eng. and MecTr Eng. from AL AZHAR University – Gaza, because this university is the most famous and the oldest university that studying those majors.

❖ **How these questions were put:**

The questions were put in a way that got answers for specific fields in this project.

The questions were created in two groups as follows; first group contains five general questions created:

1. Knowing the importance of such an application to those students.
2. Collecting some suggestions for features that could benefit the users.
3. Collecting information about how students plan could be created in the application.

The second group is yes or no questions that are created to specify the number of supporting methods to be in the application that can support students in their study.

- **Results of this step:**

The analysis of the answers to this question was resulting in two things, the first is the importance of such an application for the students while all the asked students encourage to create such an application.
The other thing that results in some functional requirements, these functional requirements are:

1. The application should enable students to create a short-term schedule.

2. The application must provide a comprehensive, general plan for the overall department.

3. The application must enable students to mark their progress in the general plan.

4. Students should be able to update their short-term schedules.

5. The application should provide users with videos links for a specific subject.

6. The application should enable users to ask questions about specific subject throw the application.

7. The application should enable students to answer others' questions about a specific subject.

8. The application should provide links to other websites that describe a specific subject.

9. The application should provide supporting documents for a specific subject.

The full survey in details found in *Appendix B*.

2. The Application's Characteristics

By analyzing the application, and after thinking with its main characteristics and properties, here is the main characteristics of it:

1) The Name: ***EnGuider***.

   This name interprets two important questions about the application, the first is For Whom This Application, and the other question is The Purpose of This Application.
   So, '***En***' stands for engineer, which is the person that proposed to benefit from the application. And '***Guider***' explain the purpose of the application, which is a guiding process for engineering students.

2) Main Content:

   The main content of any application must accomplish the main purpose that made up for.

   The first version of the application aims to help guiding the students of Software Eng. and MecTr Eng. in Gaza in their study, the main content of it will be distributed as follows:

   First, for any user of the application:
   a) Two main departments one for SW Eng., and the other for MecTr Eng.

   b) For each department, these main parts must be contained:

      1. General well-structured plan for the department.
      2. Levels of the department and levels' materials.
      3. Materials definitions, importance, benefits and any possible good supporting docs, videos and websites.

   Second, for users as students:
   Students are users that already have been registered in the application, and after that, they signed in with their accounts information that is stored in the registration. This type of user has special content in addition to the previous content. This content is:

   a) Profile page: contains its main info.

   b) Presentation of his department major plan: to keep track of his progress in its major.

c) Particular plan (schedule) page: contains his studying schedules, and the main functions of controlling them.

d) Current materials: materials that exist in its plans currently and not finishing it yet.

3) Others:

Other characteristics such as domain name and application URI will be contained in the *implementation section* of this project.

3. Application's Content Analysis

For any application, the main block of constructing it is web pages, these pages contain the content of the application and are stored in the application's server. Anyone using the application deals with its web pages and interacts with the system through it. This section will contain the final analysis of the application's pages and its content.

3.1. List of Application Pages

For this application, and after analyzing it by the previous methods, it appears that the pages of this application are categorized into three main categories depending on the users of the application, pages for any user, pages for students, and pages for admins.

The first collection of application pages are pages that can be reached and used by any user of the application. The following table describes these pages:

Table 4: Application Pages - Any User Pages

Web Page	Description
Home Page	The main page in the application appears when opening it.
SW Eng. Page	The page contains the department of SW Eng. in the application.
MecTr Eng. Page	The page contains the department of MecTr Eng. in the application.
Subjects Pages	Contains all materials in the applications but categorized.
About Page	The page contains information about the application and its owners.
Sign up Page	Through this page a user can register to the application as a student.
Sign in Page	The student can enter his session through this page.

The second collection of the application's pages is for students, the user can access these pages after signing in to the system as a student. These pages are described in the following table:

Table 5: Application Pages - Student Pages

Web Page	Description
Student Profile Page	The page contains student profile information and its profile actions.
Plans Page	The page contains the student department's plan and other student schedules.
***Subjects* Page**	The page contains the student's current materials.

The last group of pages is the admins' web pages, which enable the application admins to manage the content and the registered students in the application. Those pages are described in the following table:

Table 6: Application Pages - Admin Pages

Web Page	Description
Admin Login Page	Through this page, the admin can enter the system as an admin.
Admin Profile Page	The page contains admin profile information and its profile actions.
Departments Content Page	The page that enables the admin to manage SW Eng. and MecTr Eng. general content.
***Subjects* Management Page**	The page that enables the admin to manage subjects, materials and their contents.
Students Page	The page enables the admin to manage registered student accounts in the application.
Admins Page	Only SUPER admin can enter this page, and that is because of the purpose of managing admins' accounts, adding new, and deleting existing admins.

3.2. Content of Each Page

In this section, the structure of the content of the application's pages will be defined and specified to help the development team in the rest of the development stages.

In this application, each page has its own structure, which means that some pages contain the scientific content directly, and others have been divided into sections and the sections contain the actual content. Also, some pages contain sections and subsections. In the next lines, a detailed structure for each page of this application will be specified as well as the purpose of each page.

Home Page	
Content	Presentation of the application content.
Section:	**Section Content:**

Page Architecture 1: Home Page

SW Eng. Page	
Content	SW Eng. major description, plan, levels, subjects and sub majors.
Section:	**Section Content:**
Overview	Major definition, short description
Major Plan	Short plan description, general major plan
Major Levels	Levels of the major, materials description in each level, materials links
	Subsection: specific level with its materials links
Sub majors	Sub major overview, sub major plan, sub major materials links
	Subsection: specific sub major

Page Architecture 2: Software Eng. Page

MecTr Eng. Page	
Content	MecTr Eng. major description, plan, levels, subjects and sub majors.
Section:	**Section Content:**
Overview	Major definition, short description
Major Plan	Short plan description, general major plan
Major Levels	Levels of the major, materials description in each level, materials links
	Subsection: specific level with its materials links
Sub majors	Sub major overview, sub major plan, sub major materials links
	Subsection: specific sub major

Page Architecture 3:: Mechatronics Eng. Page

A Subject Page	
Content	Material definition, explanations, supporting things.
Section:	**Section Content:**
Overview	Material definition, short description, main topics, main documents or websites
Supporting videos	Videos links and description divided on the main material's topics
Supporting docs	Documents, supporting websites links
Questions and Answers	Questions and answers from students on this material

Page Architecture 4:: A Material Page

About Page	
Content	Information about the application and its owners.
Section:	**Section Content:**

Page Architecture 5:About Page

Sign up Page	
Content	Fields of information to register new student to the system.
Section:	**Section Content:**

Page Architecture 6: Sign up Page

Sign in Page	
Content	Fields for student sign in information.
Section:	**Section Content:**

Page Architecture 7: Sign in Page

Student Profile Page	
Content	Student information and profile actions.
Section:	**Section Content:**

Page Architecture 8: Student Profile Page

Student Plans Page	
Content	Student general plan progress, student study schedules.
Section:	**Section Content:**
General plan	Student major overall plan, student progress
Particular schedules	Student short-term schedules, progress in schedules, schedules actions

Page Architecture 9: Student Plans Page

Subjects Page	
Content	Student major materials and current materials.
Section:	**Section Content:**
Current materials	Materials that are currently in student plans with specifying material plan
All materials	All student major materials, materials pages links, short description for the material in its major plan

Page Architecture 10: Materials Page

Admin Login Page	
Content	Fields for admin login information.
Section:	**Section Content:**

Page Architecture 11: Admin Login Page

Admin Profile Page	
Content	Admin information and profile actions.
Section:	**Section Content:**

Page Architecture 12: Admin Profile Page

Departments Content Page	
Content	Choice to specify a department, department description, levels and overall plan.
Section:	**Section Content:**
Department description	Name, definition, description
General plan	Contents of general plan and actions of manipulating it
Department levels	Specific level, its materials and description

Page Architecture 13: Departments Content Page

Materials Management Page	
Content	Choice for materials and its information and manipulating actions.
Section:	**Section Content:**
Material overview	Name, definition, description, level
Supporting videos	Videos links and description
Supporting docs	Sites links and docs
Questions and answers	Questions and answers with the author of each one

Page Architecture 14: Materials Management Page

Students Management Page	
Content	Registered students accounts and its actions.
Section:	**Section Content:**

Page Architecture 15: Students Management Page

Admins Management Page	
Content	Admins accounts and its actions.
Section:	**Section Content:**

Page Architecture 16: Admins Management Page

3.3. Application Sitemap

The application sitemap describes the application pages, and how to transmit between these pages. The following shape describes the sitemap of the application that was analyzed before in this chapter:

Sitemap keys:

1. Oval Shape: For pages that are accessed directly when opening the application.
2. Rounded Rectangle: main pages that are accessed by any user.
3. Regular Rectangle: admins pages after entering the application as an admin.
4. Circle: students' pages after entering the application as a student.
5. Filled Shapes: just connectors between arrows and other arrows.
6. Arrows: connectors between pages showing how can transfer between pages.

Notes:

- All black bordered shapes are for admin pages.
- Any user can transmit between the home page and any users' pages, also between any users' pages.

The final sitemap for this application:

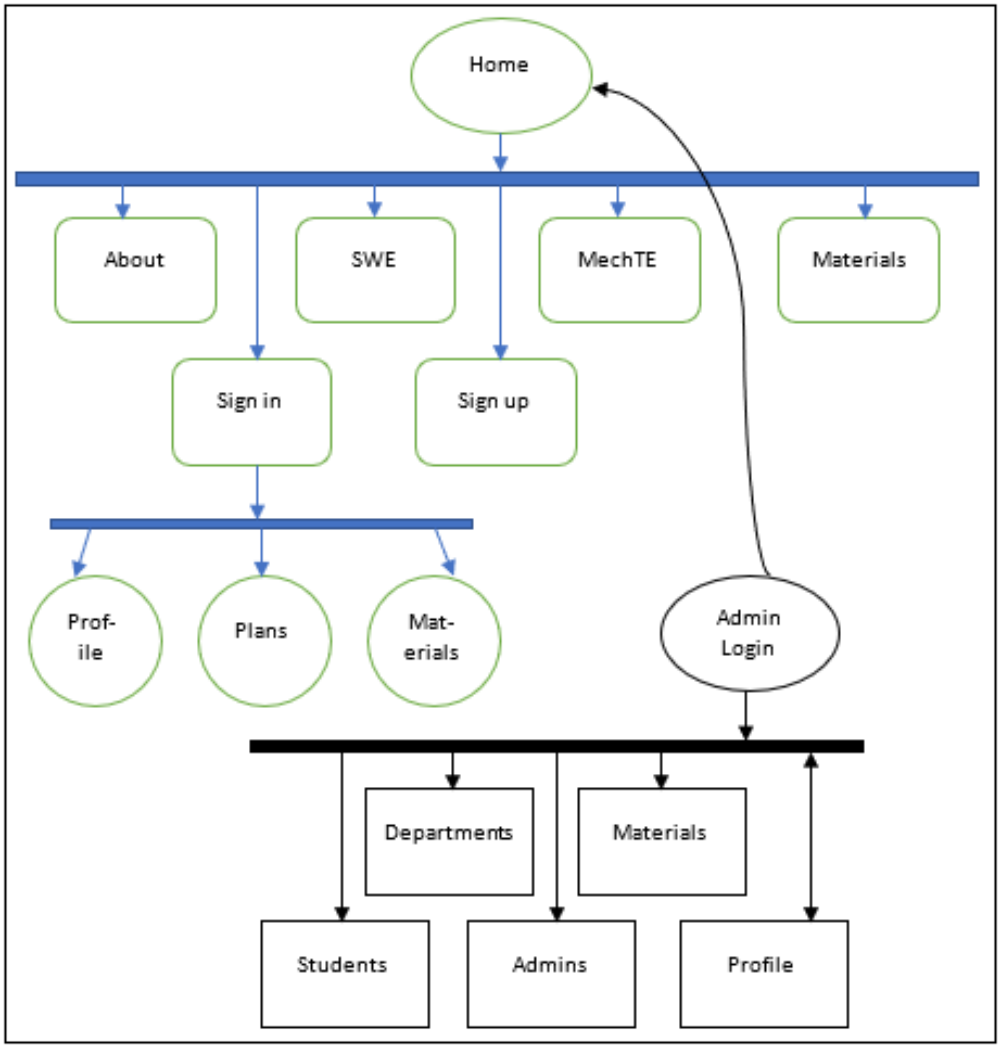


Figure 2: Application's Final Sitemap

Chapter Four

System Requirements

Chapter Four: System Requirements

This chapter introduces the functional and non-functional requirements as they elicited during the mainstreaming and analysis phase which is accomplished previously. The requirements are identified and briefly specified in this chapter. The complete derived requirement specification is presented in next chapter.

1. Functional Requirements

From the previous methods, a number of functional requirements of this web application have been collected.

The following tables show the system functional requirements divided into five main categories, which are **general**, **user-department**, **user-subject**, **student**, and **admin**.

The first group of the system functional requirements mentions general system requirements that are required in any web application. The following table shows the first group.

Table 7: System Functional Requirements – First Group

FRQ no.	***Definition***
FRQ1	The user should be acknowledged of how to use the web application.
FRQ2	The web application must define engineering majors on the home page.
FRQ3	The user should be able to contact the web application admins.
FRQ4	The content of the web application should be in Arabic in addition to English.
FRQ5	The user must be able to access the SW Eng. department.
FRQ6	The user must be able to access the MecTr Eng. department.
FRQ7	The web application should allow users to know about its owner.
FRQ8	The user should be able to contact the web application owner.

The second group of the system functional requirements mentions the requirements from the SW Eng., and MecTr Eng. departments in the application. These requirements are listed in the following table.

Table 8: System Functional Requirements - Second Group

FRQ no.	***Definition***
FRQ9	The Software Engineering section must describe Software Engineering in detail.
FRQ10	The Software Engineering section must define the general plan for the major.
FRQ11	The Software Engineering section must divide the plans into a number of levels.
FRQ12	The web application must enable users to access Software Engineering subjects.
FRQ13	Software Engineering subjects must be defined in the Software Engineering section.
FRQ14	The Software engineering section should contain sub majors of it (for example: Systems Analyst major).
FRQ15	The Mechatronics Engineering section must describe Software Engineering in detail.
FRQ16	The Mechatronics Engineering section must define the general plan for the major.
FRQ17	The Mechatronics Engineering section must divide the plans into a number of levels.
FRQ18	The web application must enable users to access Mechatronics Engineering subjects.
FRQ19	Mechatronics Engineering subjects must be defined in the Software Engineering section.
FRQ20	The Mechatronics engineering section should contain sub majors of it.

The third group of the system functional requirements contains requirements related to subjects in the application. The next table shows the third group of the system functional requirements.

Table 9: System Functional Requirements - Third Group

FRQ no.	***Definition***
FRQ21	Each subject in the web application should be divided into sections.
FRQ22	The Main sections of each subject should be explained.
FRQ23	In any subject of any major, main books and docs should be included if can.
FRQ24	In any subject of any major, supported videos for that subject should be included.
FRQ25	Any questions or answers on any subject should be displayed for any user.
FRQ26	The web application should provide dependencies between subjects.

The fourth group of the system functional requirements contains the requirements related to users as students who register in the application. This group is shown in the following table.

Table 10: System Functional Requirements - Fourth Group

FRQ no.	***Definition***
FRQ27	The web application must allow users to register as students using email.
FRQ28	The web application must allow users to register as students using a Facebook account.
FRQ29	The web application must allow users to register as students using a Google Account.
FRQ30	The web application must allow users as students to log in using a email.
FRQ31	The web application must allow users as students to log in using a Facebook account.
FRQ32	The web application must allow users as students to log in using a Google account.
FRQ33	The Signed-in student should be able to update progress in his major general plan.
FRQ34	The Signed-in student must be able to create a particular plan for his subjects.
FRQ35	The Signed-in student should be able to update his particular plan.
FRQ36	The Signed-in student must be able to delete his particular plan.
FRQ37	The web application must not allow conflict between user particular plan.
FRQ38	The Signed-in student should be able to add parts of a subject to his particular plan.
FRQ39	The Signed-in student should be able to add any subject tohis particular plan.
FRQ40	The Signed-in student must be able to update his profile information (profile image, name, account password).
FRQ41	The Signed-in student should be able to ask questions in a specific subject.
FRQ42	The Signed-in student should be able to replay questions in a specific subject.
FRQ43	The web application should provide suggestions to the student when he creates his particular plan about his major.
FRQ44	The web application should provide a status of a created particular plan.
FRQ45	The web application should provide a status of a created particular plan.
FRQ46	The web application must allow the student to reset his password when he forgets it.
FRQ47	The student must be able to view his special plan.

The fifth group of the system functional requirements is the last group which contains all the requirements for the application admins. The following table contains the fifth group of functional requirements.

Table 11: System Functional Requirements - Fifth Group

FRQ no.	***Definition***
FRQ48	The web application must have a super admin.
FRQ49	A super admin must be able to add admins to the web application.
FRQ50	A super admin must be able to change any content on the website.
FRQ51	The web application must allow the admins to sign in using email and password.
FRQ52	The web application must allow the admin to update his profile information.
FRQ53	An admin must be able to add full subjects to Engineering departments.
FRQ54	An admin must be able to remove full subjects from Engineering departments.
FRQ55	An admin must be able to add documents for any subject.
FRQ56	An admin must be able to add videos for any subject.
FRQ57	An admin must be able to remove documents from any subject.
FRQ58	An admin must be able to remove videos from any subject.
FRQ59	An admin must be able to remove questions from any subject.
FRQ60	An admin must be able to remove students from the web application.
FRQ61	A super admin must be able to change student password.
FRQ62	An admin must be able to see user's messages.
FRQ63	An admin must be able to message the students on their emails from the web application.

2. Non-functional Requirements

It is a number of general requirements that are needed in the application to ensure the quality of the application. There are many aspects that can write under it the non-functional requirements.

For this application, the next lines mentioned some non-functional requirements that are needed for this application, and how could achieve these requirements.

1) **Efficiency:**
   One of the most important features for a website user is the page load speed, for this application, the load speed for a single page must not exceed 2 seconds.

   This non-functional requirement will be provided by using the **React JS** library in the frontend development of the application, which provides a concept of Routing.

2) **Performance:**
   For this application, high performance is needed because there are many download processes by users and many interactions between users through asking questions and answering them.

   Also, the use of **React JS** library will ensure high performance through the concept of Single Page Application provided by it.

3) **Security:**
   Since authentication operations are needed for students and admins in this application, security becomes an important requirement for the application.

   The use of **PHP Laravel Framework** for application implementation will provide a high degree of security, and that is one of the best features of the Laravel framework.

3. UX Analysis and Prototyping

A method of analyzing any new system and collecting its functional requirements is designing the main user interfaces in the system. This method will help developers to specify functional requirements in the system, as well as know the system use cases.

For this web application, the project team designed the main web pages in the web application to help them imagine how would the user interact with the web application, and what functional requirements exactly should be in this web application.

The following pages will show the UI design for the main web pages in this web application. Should know that this UI design is not the final user interfaces but it's approximately as much like as possible to the final UI design for this web application.

Before showing the UI design, here is a list of the web application's pages that are designed for this section:

Table 12: Designed UI - Designed Application's Pages

Page Name	***Purpose***
Home Page	Define the web application and enable accessing other pages
Materials Page	Containing all materials and subjects for each engineering department on the web application.
A Material Page	Describing specific material and its contents
Student General Plan Page	Enable the student to access his general plan.
Student Special Plans Page	Enable the student to access his particular schedules.
Student Subjects Page	Enable the student to access his current subjects.
Sign in Page	Enabling the student to sign in.
Register Page	Enabling users to create students accounts.

The following figures show the design of the mentioned web pages in this web application with a short description for each web page:

1. Home Page:

   Is an introductory page for the web application contents and how to use it, and also contains an overview of the site's engineering departments (Software Engineering and Mechatronics Engineering).

Figure 3: UI Design - Home Page

2. Materials Page:

   Example: Software Engineering Materials:
   Is a page that contains all the subjects on the web application for the Software Engineering department.

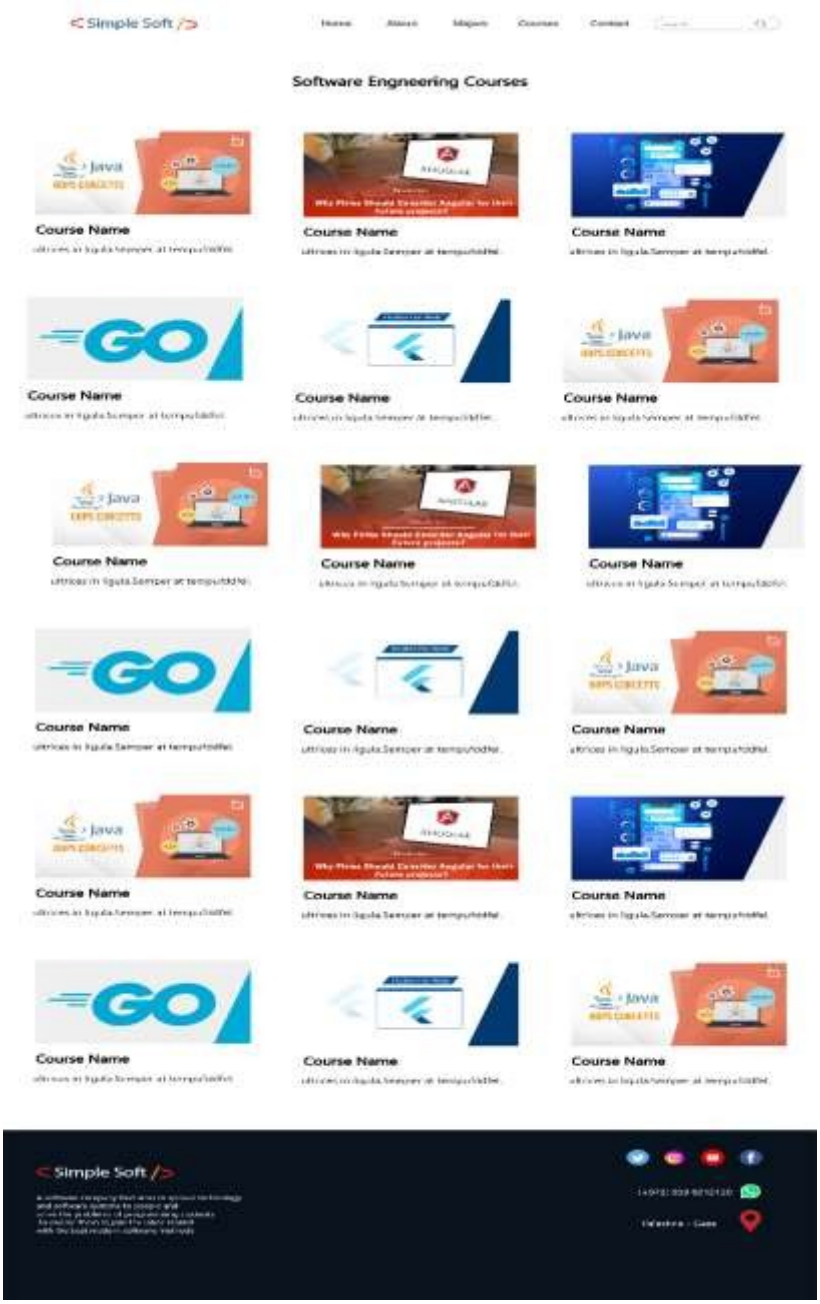


Figure 4: UI Design - SW Eng. Materials

3. A Material Page:

   It is a page that contains all the contents of a specific material, including the introduction of the material, books, documents, and videos for it. It also contains a special section for students' questions about this subject and their answers.

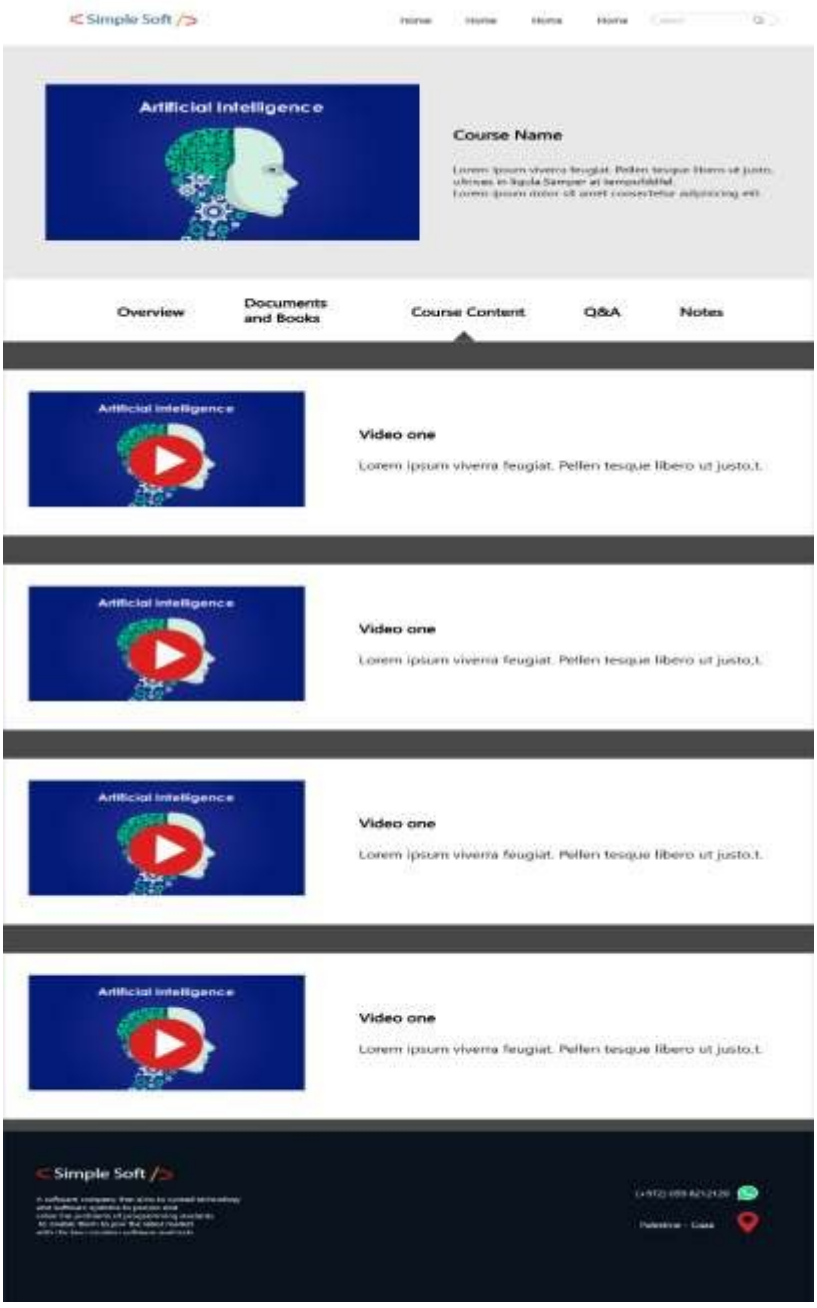


Figure 5: UI Design - a Specific Material Page

4. Student General Plan Page:

   It is the student's home page, through which the student can monitor his progress in the general plan, in addition to the presence of a special section for the materials in his plans.

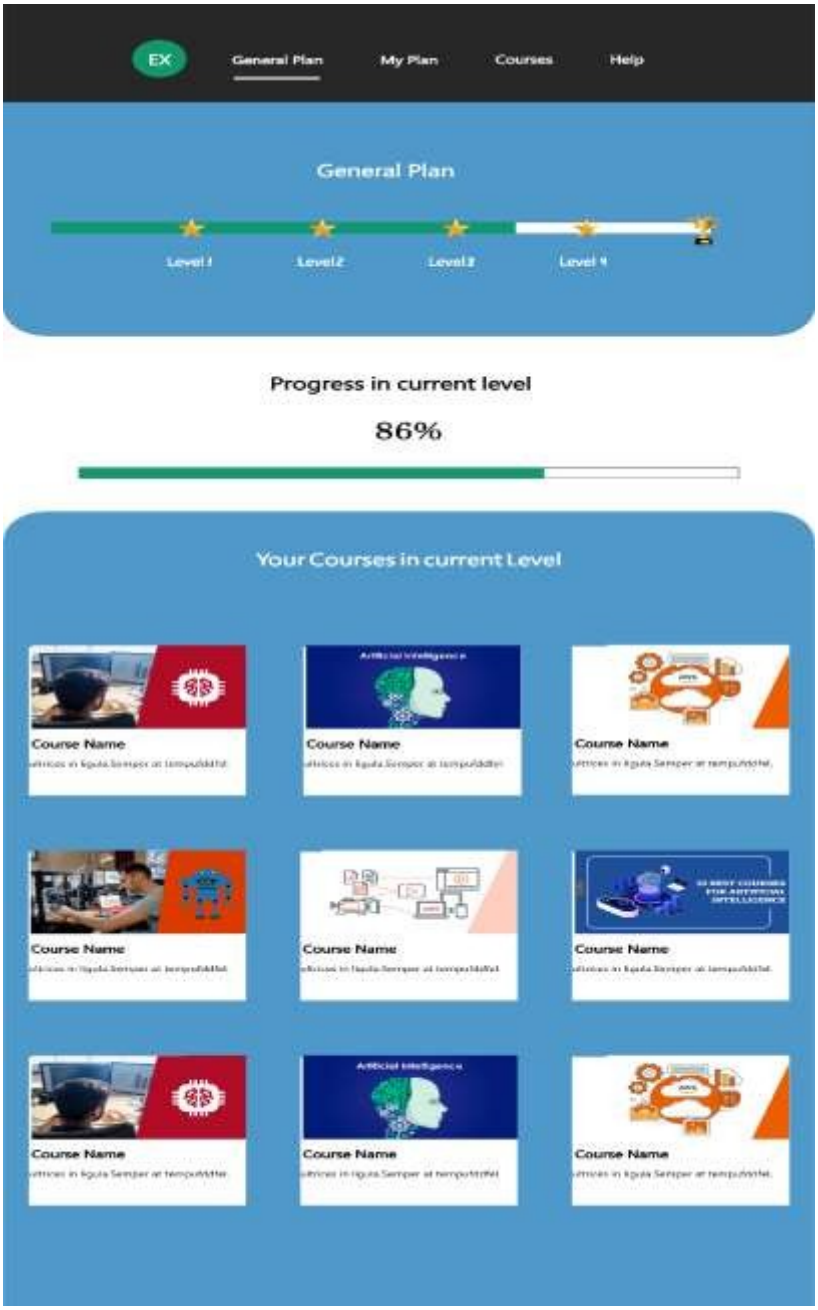


Figure 6: UI Design - a Student General Plan Page

5. Student Special Plans Page:

   It is a page dedicated to displaying the schedules that the student has created.

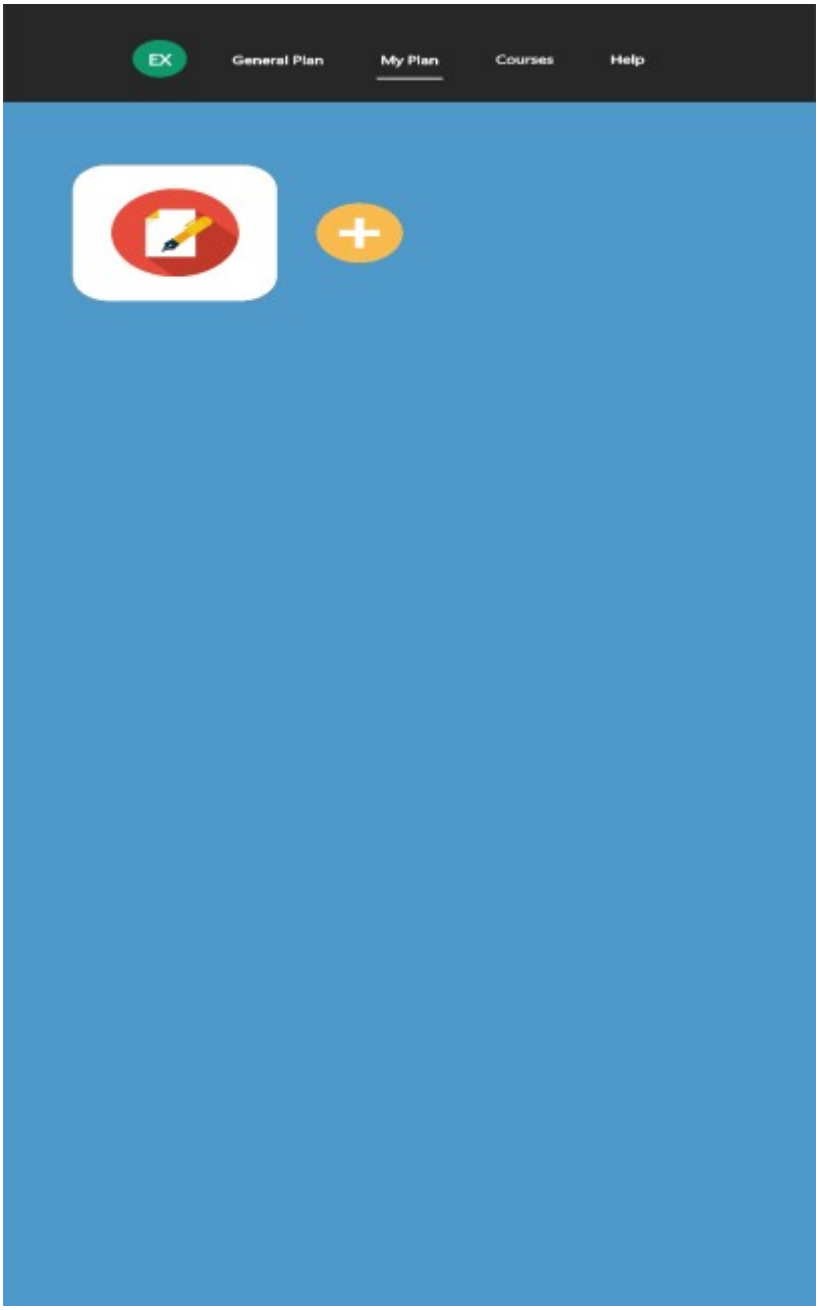


Figure 7: a Student Special Plans Page

6. Student Subjects:

   It is a page that displays all the materials included in the student own schedules.

Figure 8: UI Design - a Student Materials Page

7. Student Sign in Page:

   It is the student sign-in page where from it can a student enter his account in the web application.

Figure 9: UI Design - Student Sign in Page

8. Register Page:

   It is a page through which the user creates a new student account for him.

Figure 10: UI Design - Student Registration Page

Chapter Five

Functional Requirements Specification

Chapter Five: Functional Requirements Specification

1. Stakeholders

1) System owners: people that this application is made up from their side, and they will be responsible to manage the application content.

2) Users of the system: engineering students, especially students of SW Eng. and MecTr Eng. majors in this version of the application.

2. Actors and Goals

1) *Admin*: the actor who has the privileges of managing the content of the application.

2) *Super Admin*: have all privileges, and the only admin that can change student password and managing other admins accounts.

3) *User*: any person can use the application without the need to be registered or logged in.

4) *Student*: any person has a student account and has been signed in to the application.

3. Scenarios for System Use cases

System use cases are used to model the system functional requirements, and to demonstrates them in more precise way to better understand how the system works.

So, to extract system use cases, scenarios are needed since they used to illustrate some interactions between a specific actor or actors and the system, which helps to extract system use cases. The next lines will show a number of scenarios with the use cases that extracted from each one.

Note: while reading scenarios, the words with italic styles and underlined are use cases that extracted from the read scenario.

First Scenario: Scenario A

- ***Scenario A address:*** *a student forgets his password*
- ***Scenario A actors:*** *Ali is a student, and Rami as an admin*
- ***Scenario A story:***

  Ali enters the site's *login* page, enters his email and password, and presses the "*Login*" button. An error message appears in the password, Ali tries to *login* again and *re-enter the password, but all his attempts are unsuccessful*.

  Ali decides to communicate with the admin, so he goes to the home page of the site and *sends a message to the admin* regarding his problem logging in.

  Rami enters the *login page for the admin* and enters his email and password then confirms and enters the site, he notices a message from a user, reads it and finds that Ali has a problem with the password.

  Therefore, Rami chooses “the tables”, then “the students table”, and chooses the “*Change password*” button for Ali then changes the student's password and then confirms.

  Rami sends a message to this user's email containing the new password, then *log out* from website.

  Ali enters the email and his new password after reading it from his email and presses the "*Login*" button; he logs in and enters his account successfully.

- ***Use cases from Scenario A:***

  1) Student sign in.
  2) Student sign out.
  3) Student forget password.
  4) Contact application admins.
  5) Admin login.
  6) Admin logout.
  7) Admin contact student.
  8) Admin change student password.

Second Scenario: Scenario B

- ***Scenario B address:*** *Create a special plan for Android sub-major*
- ***Scenario B actors:*** *Ali is a SW Eng. student*
- ***Scenario B story:***

  Ali *logs into* the web site. Ali needs a special plan for Android to follow the path and learn it. That is why he decided to *create his own plan* to learn Android.

  Ali choses from his home page “My Plan” from the list at the top, then he will find the general plan and next to it a “+” button to create a special plan, so he will *press that button*.

  The system displays a form for creating a special plan, this form contains ready-made plans for sub-major such as the web and Android, in case the subjects were not chosen by himself.

  Ali chooses the special plan for Android and *adds it to his plans*, then begins to *display the special plan* and tracks its path.

  After Ali creating the special plan, the system allows him to interact with and study the subjects, *modifies* the plan if he needs to, and *deletes* it if he wants to.

- ***Use cases from Scenario B:***

  1) Student sign in.
  2) Student sign out.
  3) Create special plan.
  4) Update special plan.
  5) Delete special plan.
  6) Display special plan subjects.
  7) Interact in a subject.
  8) Track general plan.

4. System Use cases

4.1. Use cases

After defining the main scenarios in the application, and after analyzing the application functional requirements, system use cases have been appeared for this application project.

The system use cases for this application are divided into three groups according to the main actor of the use case.

First group describes the use cases that any user can do without the need for signing up in the application or signing in. the following table describes this group of use cases:

Table 13: Ordinary Actor Use cases

Use case	***Name***	***Definition***	***Relate to***
UC1	*Browse the application*	Accessing application.	FRQ2
UC2	*Browse SW Eng. section*	Open SW Eng. section in the application.	FRQ5
UC3	*Browse MecTr Eng. section*	Open MecTr Eng. section in the application.	FRQ6
UC4	*Contact application admins*	Send messages to admins.	FRQ3
UC5	*Contact application owners*	Reach Facebook, Telegram, Google, and WhatsApp of the application owners.	FRQ8
UC6	*Knowing about application owners*	Browse about us page in the application.	FRQ8
UC7	*Browse a subject*	Open a subject to see its contents.	FRQ12

Second group of system use cases appeared from the second scenario and the functional requirements of a user as a student. the table below describes this use cases:

Table 14: Student Actor Use cases

Use case	***Name***	***Definition***	***Relate to***
UC8	*Student sign in*	Sign in as a student to the application.	FRQ30
UC9	*Student register*	Register as a student to the application.	FRQ27
UC10	*Student reset password*	Reset password.	FRQ
UC11	*Student forget password*	Reset password when student forget it.	FRQ
UC12	*Student sign out*	Sign out as a student to the application.	FRQ
UC13	*Student change profile information*	Student updates his profile.	FRQ40
UC14	*Create special plan*	Student creates a particular plan.	FRQ34
UC15	*Update special plan*	Student updates a particular plan.	FRQ35
UC16	*Delete special plan*	Student deletes a particular plan.	FRQ36
UC17	*Display special plan subjects*	Student displays his plan subjects.	FRQ
UC18	*Track general plan*	Student mark what he finishes in the general plan.	FRQ33
UC19	*Interact in a subject*		FRQ

The last group of system use cases defines the main operations of the application admins (super and ordinary admins); a part of these use cases appears in the third scenario in the previous section. The table below shows the use cases of the third group:

Table 15: Admin Actor Use cases

Use case	***Name***	***Definition***	***Relate to***
UC20	*Admin login*	Log in as an admin in the application.	FRQ48
UC21	*Admin profile update*	Admin changes his profile information.	FRQ49
UC22	*Admin log out*	Admin logout from the application.	FRQ
UC23	*Add new admins*	A super admin adds new admins to the application.	FRQ
UC24	*Admin change student password*	Admin changes student password.	FRQ
UC25	*Change application content*	Admin changes general content in the application.	FRQ
UC26	*Change subject contents*	Admin adds and deletes contents to and from a subject.	FRQ
UC27	*Change SW Eng. department content*	Admin adds, updates and deletes content of SW Eng. department.	FRQ
UC28	*Change MecTr Eng. department content*	Admin adds, updates and deletes content of MecTr Eng. department.	FRQ
UC29	*Remove a student*	Removing specific student account from the application.	FRQ
UC30	*Contact user*	Admin reach and send messages to user email through the application.	FRQ

4.2. Use cases Description

After extracting and specifying the system use cases in the previous section, use cases need to be written in a detailed way to make the understanding of it easier, and the implementation of it known and clear. The next lines will show the description of each use case of the system.

The use case description form consists of four main parts, which are:

- **Name**: contains the name of the use case.
- **Description**: a brief description about the use case.
- **Actors**: who are participating in the use case.
- **Main flow**: contains the steps of doing the use case.

and three secondary parts, which are:

- **Preconditions**: actions that need to be done before the use case started.
- **Postconditions**: results of doing the use case.
- **Alternative flows**: any abnormal paths could be happened while doing the use case.

The next pages contain the use case description for all previous system use cases.

Description of UC 1

***Name*:** Browse the application
***Description*:** Accessing application.
***Actors*:** user
***Main Flow*:** 1. User opens a supported browser like Chrome for example. 2. User enters application name and search. 3. Browser gets results. 4. User chooses the application from results. 5. The home page of the application displayed to the user.

Description of UC 2

***Name*:** Browse SW Eng. section
***Description*:** Open SW Eng. section in the application.
***Actors*:** user
***Preconditions*:** User have entered the application.
***Main Flow*:** 1. User choose from the header “Departments” option. 2. System displays drop-down menu containing the departments. 3. User choose the option “SW Eng. Department” 4. System displays the SW Eng. Department page**.**

Description of UC 3

***Name*:** Browse MecTr Eng. section
***Description*:** Open MecTr Eng. section in the application.
***Actors*:** user
***Preconditions*:** User have entered the application.
***Main Flow*:** 1. User choose from the header “Departments” option. 2. System displays drop-down menu containing the departments. 3. User choose the option “MecTr Eng. Department” 4. System displays the MecTr Eng. Department page.

Description of UC 4

***Name*:** Contact application admin.
***Description*:** Users can communicate with the admin.
***Actors*:** user
***Preconditions*:** User have entered the application.
***Main Flow*:** 1. User selects "Contact Us" from the page header. 2. System displays "Contact Us" page for the user. 3. User enters his email and the message he wants to send. 4. User clicks "send message".
***Postconditions*:** The message is sent to the admin email.

Description of UC 5

***Name*:** Contact application owners.
***Description*:** The user can communicate with application owners.
***Actors*:** user
***Preconditions*:** User have entered the application.
***Main Flow*:** 1. User chooses any account of the owner from the bottom of any page. 2. System redirects the user to the chosen account.

Description of UC 6

***Name*:** knowing about application owners.
***Description*:** user knows the owner of the application.
***Actors*:** user
***Preconditions*:** User have entered the application.
***Main Flow*:** 1. User selects "About Us" from current page header. 2. System displays "About Us" page for the user. 3. User browses the "About Us" page.

Description of UC 7

***Name*:** Browse a subject
***Description*:** open a subject to see its contents.
***Actors*:** user
***Preconditions*:** Browse SW Eng. section or Browse MecTr Eng. section
***Main Flow*:** 1. Student choses “Subjects” choice. 2. System shows all subjects of the department. 3. Student choses a subject to browse it. 4. System shows the page of the selected subject.

Description of UC 8

***Name*:** Student sign in
***Description*:** student login to the system.
***Actors*:** student
***Preconditions*:** Student already registered to the application.
***Main Flow*:** 1. Student chooses "Sign In" button. 2. System shows an input dialog asking for email and password. 3. Student enters email and password, then confirm. 4. System checks if the entered information is in the system database and true. 5. System returns to the student's profile page with student login successfully if all previous entries is true.
***Postconditions*:** A session for the entered student start.
***Alternative Flows*:** If the email or password is not accepted, the system will ask to enter it again.

Description of UC 9

***Name*:** Student register
***Description*:** student creates an account on application.
***Actors*:** student
***Preconditions*:** Student enters the website and browsing it.
***Main Flow*:** 1. Student choses "Sign Up" button. 2. System shows an input dialog asking for email, password, password confirmation and major. 3. Student enters email, password, password confirmation and major, then confirm. 4. System verifies the input and creates an account for the student.
***Postconditions*:** System displays the registered student profile.
***Alternative Flows*:** If wrong email, short password, or empty fields are entered, the system will verify the entries and display an error message.

Description of UC 10

***Name*:** Student reset password
***Description*:** reset password.
***Actors*:** student, admin
***Preconditions*:** student had an account in the application.
***Main Flow*:** 1. Student clicks “Reset Password” in the sign in page. 2. System asks the student to enter his email to verify him. 3. System checks if the entered email has an account in the system database. 4. System sends a verification message to the email and wait the student to enter it. 5. Student enters the verification message. 6. System notifies the admin to change the password for the student account. 7. Admin changes the password of the student’s account. 8. System sends new password to the student email.
***Postconditions*:** student password reset successfully.
***Alternative Flows*:** If any entered information is wrong, system will repeat the process.

Description of UC 11

***Name*:** Student forget password
***Description*:** reset password when student forget it.
***Actors*:** student
***Preconditions*:** student had signed in.
***Main Flow*:** 1. Student choses “Change Password” from profile page. 2. System shows a form for the new password. 3. Student enters new password and rewrite it to confirm. 4. System asks student to confirm changing. 5. Student clicks “Confirm Change”. 6. System changes the student password to the new password.
***Postconditions*:** student password changed to the new password.
***Alternative Flows*:** If any entered information is wrong, system will repeat the process.

Description of UC 12

***Name*:** Student sign out
***Description*:** student session is end by the student.
***Actors*:** student
***Preconditions*:** Student has been signed in.
***Main Flow*:** 1. Student clicks on his account photo. 2. System shows some options. 3. Student chooses “Logout”.
***Postconditions*:** The system ends the student’s session and return to the home page.

Description of UC 13

***Name*:** Student change profile information
***Description*:** student updates any of his profile information.
***Actors*:** student
***Preconditions*:** Student sign in.
***Main Flow*:** 1. Student clicks on the cover photo in his page. 2. System shows the student profile information. 3. Student clicks on the information update. 4. The information field become editable. 5. Student edit the information and click “Done”. 6. The previous steps (3,4,5) are repeated to any field in the profile info to update.
***Postconditions*:** The profile information updated successfully in the system database.
***Alternative Flows*:** If any fields are written incorrectly depending on the system conditions, the system will ask the user to re-enter it.

Description of UC 14

***Name*:** Create special plan
***Description*:** student creates a particular plan.
***Actors*:** student
***Preconditions*:** Student sign in.
***Main Flow*:** 1. Student chooses “My Plans” from his home page. 2. System shows student’s particular plans, with the add (+) button after the plans. 3. Data fields of a new particular plan appears to be filled. 4. Students fill the information for the new plan and clicks “Create”.
***Postconditions*:** The new plan is created, stored in the system database, and appears in the “My Plans” section in the student web page.
***Alternative Flows*:** If any fields are written incorrectly depending on the system conditions, the system will ask the user to re-enter it.

Description of UC 15

***Name*:** Update special plan
***Description*:** student updates any information in a particular plan.
***Actors*:** student
***Preconditions*:** Student sign in, and enters “My Plans“ section in his web page.
***Main Flow*:** 1. Student clicks on the plan he wants to update. 2. Student chooses “Update”. 3. System shows the information fields of the chosen plan editable. 4. Student edit the information and click “Done”.
***Postconditions*:** The plan information updated in the plan page and the database.
***Alternative Flows*:** If any fields are written incorrectly depending on the system conditions, the system will ask the user to re-enter it.

Description of UC 16

***Name*:** Delete special plan
***Description*:** student deletes a particular plan.
***Actors*:** student
***Preconditions*:** Student sign in, and enters “My Plans“ section in his web page.
***Main Flow*:** 1. Student clicks on the plan he wants to update**.** 2. System opens plan form and its information. 3. Student clicks “Delete” button. 4. System asks the student to confirm the deletion. 5. Student clicks “Ok”.
***Postconditions*:** The deleted plan disappeared from “My Plans” section.

Description of UC 17

***Name*:** Display special plan subjects
***Description*:** student displays his plan subjects.
***Actors*:** student
***Preconditions*:** Student sign in, and enters “My Plans“ section in his web page.
***Main Flow*:** 1. Student clicks on the plan that he wants to see its contents. 2. System views plan view. 3. Student chooses “Subjects”. 4. System view plan subjects in the bottom of the same page.

Description of UC 18

***Name*:** Track general plan
***Description*:** student mark what he finishes in the general plan.
***Actors*:** student
***Preconditions*:** Student Sign in.
***Main Flow*:** 1. Student click on the general plan in his home page. 2. Student choses “current Level”. 3. System displays the student’s current level subjects. 4. To track a specific subject, student click in the subject name. 5. System shows the chosen subject sections to be marked. 6. Student mark what he finishes in the subject.
***Postconditions*:** The system computes the percentage of the new marked sections of the general plan subjects and update the progress in the general plan of the student.

Description of UC 19

***Name*: Interact in a subject**
***Description*: users move between subject sections and interact with them.**
***Actors*: student**
***Preconditions*:** Student sign in.
***Main Flow*:** 1. Student choses “My Subjects” from the menu bar in his web page. 2. A page of all subjects in the student’s plans appears. 3. User choses a subject. 4. System displays the web page of the chosen subject. 5. User browses different sections of the chosen subject.

Description of UC 20

***Name*: Admin login**
***Description*:** Admin login to the application to control its content.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin enters to admin page.
***Main Flow*:** 1. Admin writes email and password in the text field that shown for it, then confirms. 2. If the password is true, admin enters application.
***Postconditions*:** Admin capabilities enabled in the website.
***Alternative Flows*:** If the entered password is wrong, the website will not open, and an error message will be shown.

Description of UC 21

***Name*: Admin profile update**
***Description*:** Admin changes his profile information.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses his profile page. 2. The Application displays the admin's profile page. 3. Admin clicks on the "Edit Profile" button. 4. Admin edits his profile.
***Postconditions*:** Profile was updated.
***Alternative Flows*:** If wrong data or empty fields are entered, an error message will be displayed and re-edit.

Description of UC 22

***Name*: Admin log out**
***Description*:** Admin logs out of the application.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses "log out "button.
***Postconditions*:** The admin is logged out of the application.
***Alternative Flows*:**

Description of UC 23

***Name*: Add new admins**
***Description*:** The admin adds a new admin to the application.
***Primary Actors*: Admin**
***Secondary Actors*: Admin**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then the admins' table. 2. The application displays the admin table. 3. Admin chooses the "Add admin" button, then enters the new admin's email and password, then confirms.
***Postconditions*:** A new admin is added in the application.
***Alternative Flows*:** If wrong data or empty fields are entered, an error message will be displayed and re-enter.

Description of UC 24

***Name*: Admin change student's password**
***Description*:** The admin changes the student's password because there is a problem with the student.
***Primary Actors*: Admin**
***Secondary Actors*: Student**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then the students table. 2. Admin chooses the "Change password" button for a specific student, then changes the student's password and then confirms.
***Postconditions*:** The student's password is changed.
***Alternative Flows*:** If wrong data or empty fields are entered, an error message will be displayed and re-edit.

Description of UC 25

***Name*: Change application contents**
***Description*:** Admin changes application content.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then the content table. 2. Admin chooses the "Change content" button, then changes application content.
***Postconditions*:** application content was changed.
***Alternative Flows*:** If wrong data or empty fields are entered, an error message will be displayed and re-edit.

Description of UC 26

***Name*: Change subject contents**
***Description*:** Admin changes subject contents.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then the subjects table. 2. Admin chooses the "Change subject contents" button, then changes subject contents.
***Postconditions*:** Course contents were changed.
***Alternative Flows*:** If wrong data or empty fields are entered, an error message will be displayed and re-edit.

Description of UC 27

***Name*: Change SW Eng. department content**
***Description*:** admin adds, updates and deletes content of SW Eng. department.
***Primary Actors*:** admin
***Preconditions*:** admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then SW Eng. department table. 2. Admin specifies content he wants to change. 3. System displays the chosen content information to change. 4. Admin change the content and submit. 5. System stores new changes.

Description of UC 28

***Name*: Change MecTr Eng. department content**
***Description*:** admin adds, updates and deletes content of MecTr Eng. department.
***Primary Actors*:** admin
***Preconditions*:** admin already login to the application.
***Main Flow*:** 1. Admin chooses the tables, then MecTr Eng. department table. 2. Admin specifies content he wants to change. 3. System displays the chosen content information to change. 4. Admin change the content and submit. 5. System stores new changes.

Description of UC 29

***Name*: Remove a student**
***Description*:** Admin removes student's account from application.
***Primary Actors*: Admin**
***Secondary Actors*:**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 6. Admin chooses the tables, then the students table. 7. Admin chooses "delete student" button for a specific student, then confirm.
***Postconditions*:** The student's account has been removed.

Description of UC 30

***Name*: Contact user**
***Description*:** Admin communicates with the user.
***Primary Actors*: Admin**
***Secondary Actors*: User and Student**
***Preconditions*:** Admin already login to the application.
***Main Flow*:** 1. Admin chooses the messages icon. 2. The Application displays users' messages. 3. Admin responds to users' messages.
***Postconditions*:** The messages were sent to the users.
***Alternative Flows*:**

4.3. Use cases Diagram

As the use cases of this system categorized into three groups, there are three diagrams that describes these three groups. The following photos shows the use case diagram for this system.

1. Use case diagram that describes the **general use cases** in the system:

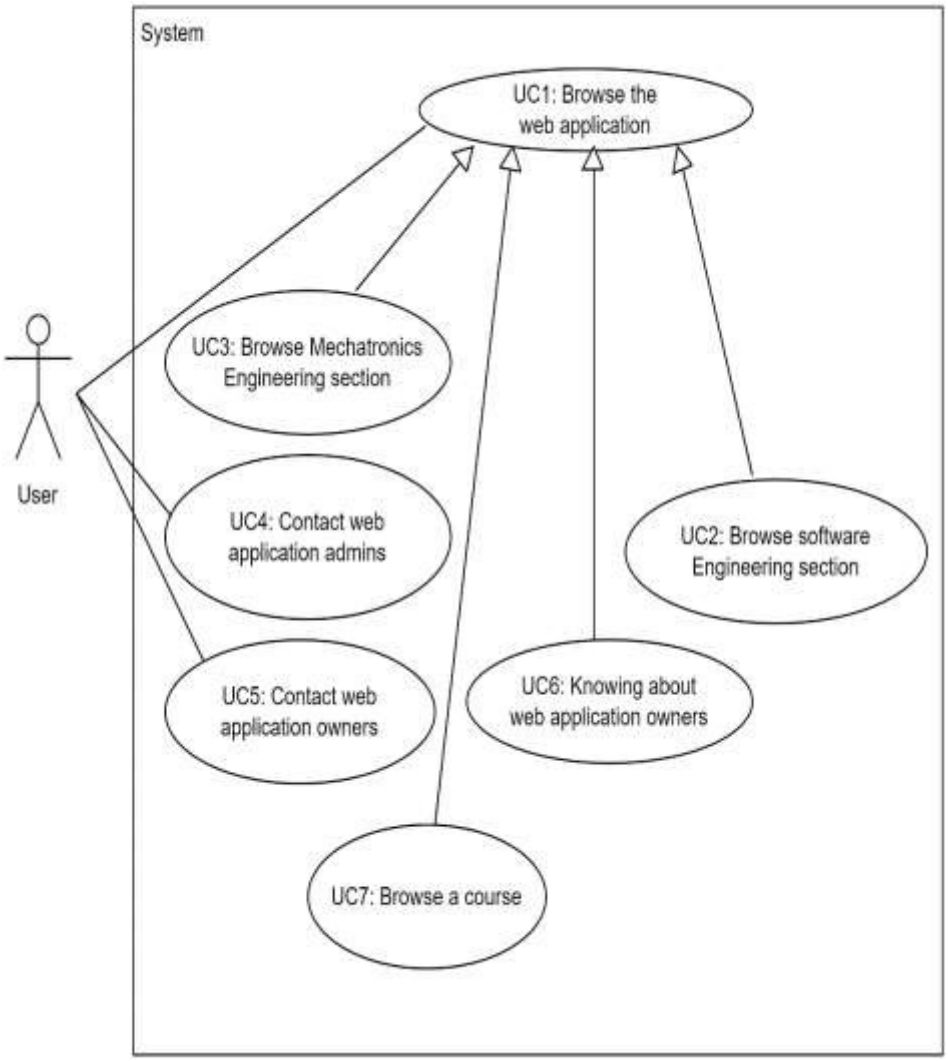


Figure 11: UC Diagram a

2. Use case diagram for the **students use cases**:

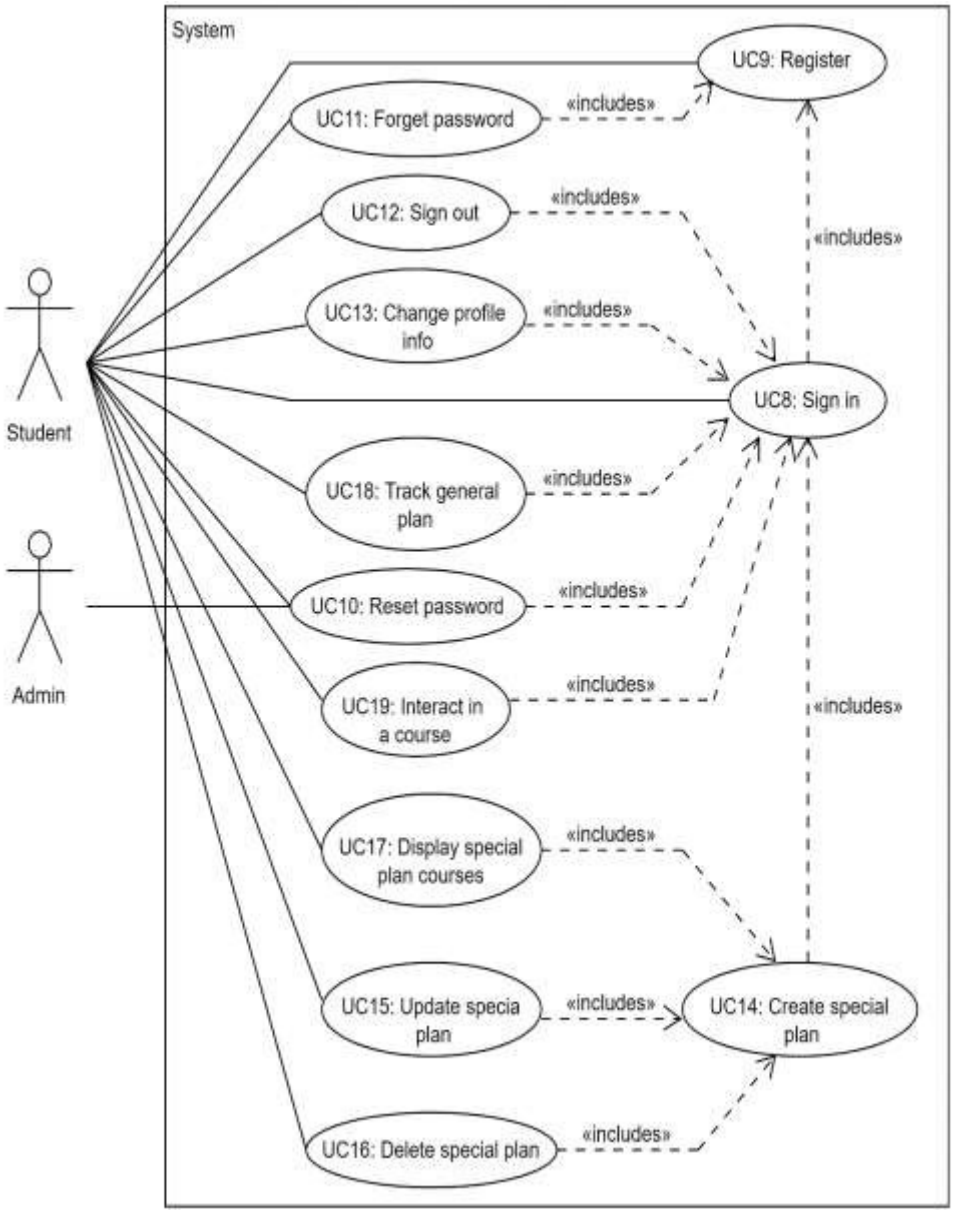


Figure 12: UC Diagram b

3. Use case diagram for the **admin use cases**:

Figure 13: UC Diagram c

Chapter Six

System Design

Chapter Six: System Design and UML Diagrams

This chapter introduces the design of the proposed application. It presents visual description of the application Architecture and design. The design is presented in this chapter using UML symbols and diagrams.

1. System Architecture

The system architecture is a top-level design that used to provide a knowledge about the overall structure of the system. This design is done using the main components that construct the system, providing how these components connected together to build the application.

1.1. Overview

PHP Laravel framework depends on the Model, View, Controller (MVC) architecture in building applications, and while the building of this application will be done using PHP Laravel framework, implementation section will answer why Laravel, the system architecture must be compatible with the implementation method, which follows the (MVC) architecture.

The Model, View, Controller (MVC) Architecture:
"Is a software architectural pattern commonly used for developing user interfaces that divide the related program logic into three interconnected elements. This is done to separate internal representations of information from the ways information is presented to and accepted from the user." (Wikipedia) [1]

The three main elements in the MVC model [2] are:

1. **Model**: is the code that reflects the real-word things, also the interface between the controllers and the database which the data and the data queries are sends through it.

2. **View**: the interface between the users and the system, and contains all the functions that directly interact with the users.

3. **Controller**: the code that responsible for the connection between Model section and View section, which consider the brain of the application. It ties the Model with the View by receiving user inputs,

processing it, send and get data using Model, and then return outputs.

1.2. System Architecture Diagram

After knowing the overview of the system architecture, and what is the model this application will be created on, this step will describe the components that struct this application, and will provide the system architecture diagram.

First of all, here are the main components that constructs the system architecture and the purpose of each component:

1. **Browser**: is the client run on a computer or mobile device that contacts the Web server, requests information and then displays results after retrieving them from server. [3]

2. **Web Server**: it's a computer that runs applications, which responsible for storing, processing and delivering web pages to the users. [4]

3. **Database**: a database application designed to be managed and accessed through the Internet. [5]

4. **PHP Laravel Framework Components**: [6]
   4.1. Models: sets and gets data to\from database.
   4.2. Views: takes the ORM objects which helps to get data from the database.
   4.3. Controllers: contacts with models and query builder to create, update, or delete data, and feeds views.
   4.4. Query Builder: provides a convenient, fluent interface to creating and running database queries. [7]
   4.5. Routing: handles the request based on the URL.

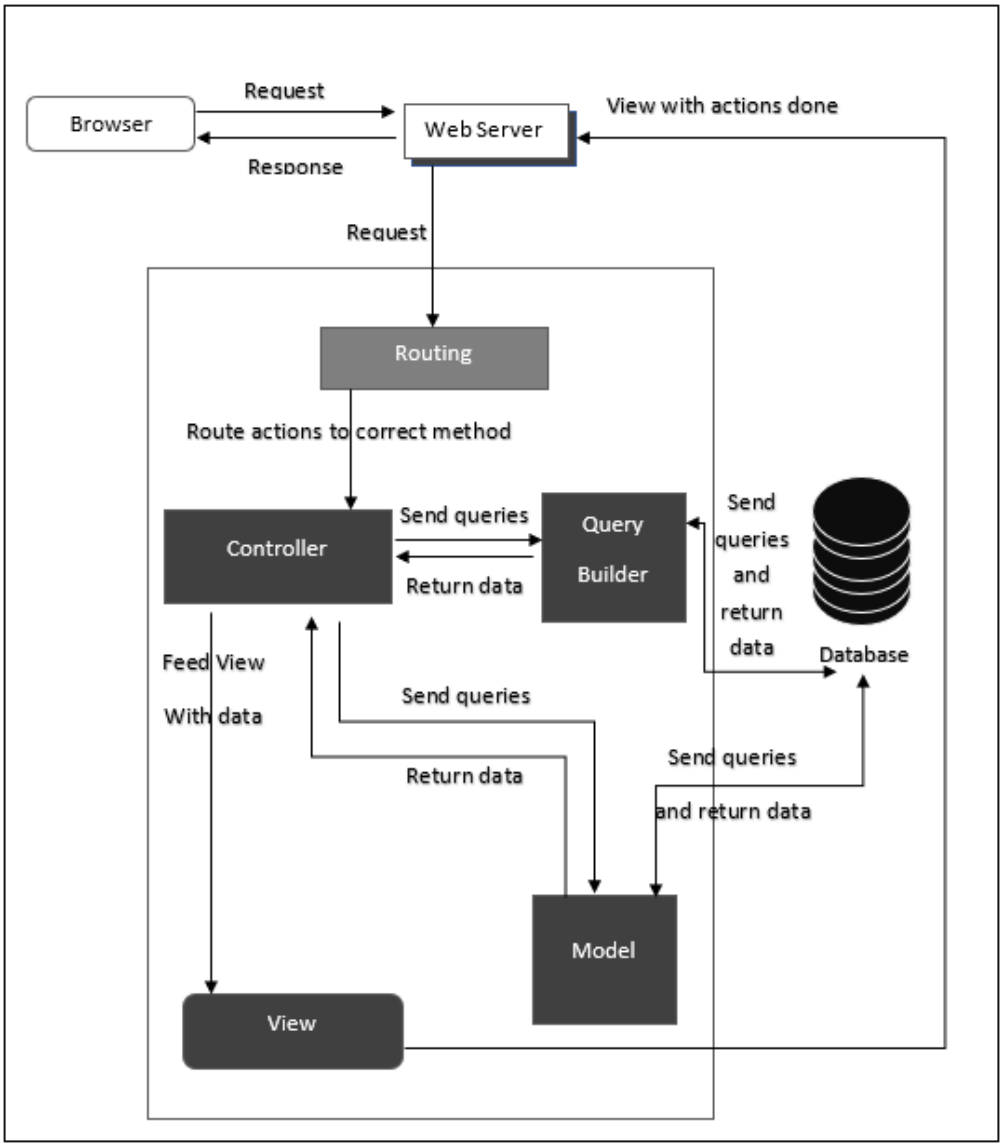


Figure 14: System Architecture Diagram

2. System UML Diagrams

This section talks about a lower-level design for this application, which illustrates the classes of this application, and the sequence diagram for the system use cases.

2.1. System Class Diagram

The system class diagram illustrates all classes that composed the system and shows how are these classes connected with the multiplicity between them.

The class diagram is made up in this phase to help programmers in the implementation phase of their mission while it contains the attributes of each class, the attributes types and visibility, and the methods and their types for each class.

The class diagram for this application considers the model of MVC when it was created, which means that it is made up of two types of classes, models and controllers. And it also illustrates the relationships between models using associations and multiplicity and illustrates the controllers in the system and each controller for which the model is associated.

- **Specification of the system class diagram:**

  As mentioned above, this system class diagram contains Model classes, and for each model, there is a Controller class that uses the Model class to make operations on the specified Model data.

  The majority of Model classes contain special attributes like [fillable, hidden, casts], and special methods for relations between models, which reflect the relations between tables in the database.
  While the majority of Controller classes contain just ordinary methods that do a specific work that is needed. The next lines demonstrate the methods and attributes used in the Models and the Controllers of this class diagram.

A. Models Classes:

- ***Attributes****:*
  1. fillable: an array that contains fields names that are in the table of the model, these fields names used in mass assignment.

     Mass Assignment: the process of automatically assign request value to the model attribute.

  2. hidden: an array that contains fields names that are in the table of the model, to limit the attributes, such as passwords, that are included in the model's array or JSON representation. [8]

  3. casts: an array where the key is the name of the filed in the table of the model, which want to be cast and the value is the type you wish to cast the column to.

     It provides a convenient method of converting attributes to common data types. [9]

- ***Methods****:*
  Methods in models' classes are for specifying relations between models.

  The name of the method will be the same name of the model class that the relation is between it and the model the method is written in, but the name starts in lowercase.

B. Controllers Classes:

The methods in controllers' classes can be distributed into two types:

1. CRUED methods:
   Which are six methods that do the main operations on the models' classes which are:

   1) index: get all instances of a models' data.
   2) show: get specific instance of a models' data.
   3) create: get create form to create new instance of a models' data.
   4) store: create new instance of a models' data.

5) edit: get edit form to update specific instance of a models' data.
6) update: update specific instance of a models' data.
7) destroy: delete specific instance of a models' data.

2. Custom methods:
   Any other methods added to do another work.

❖ System class diagram:

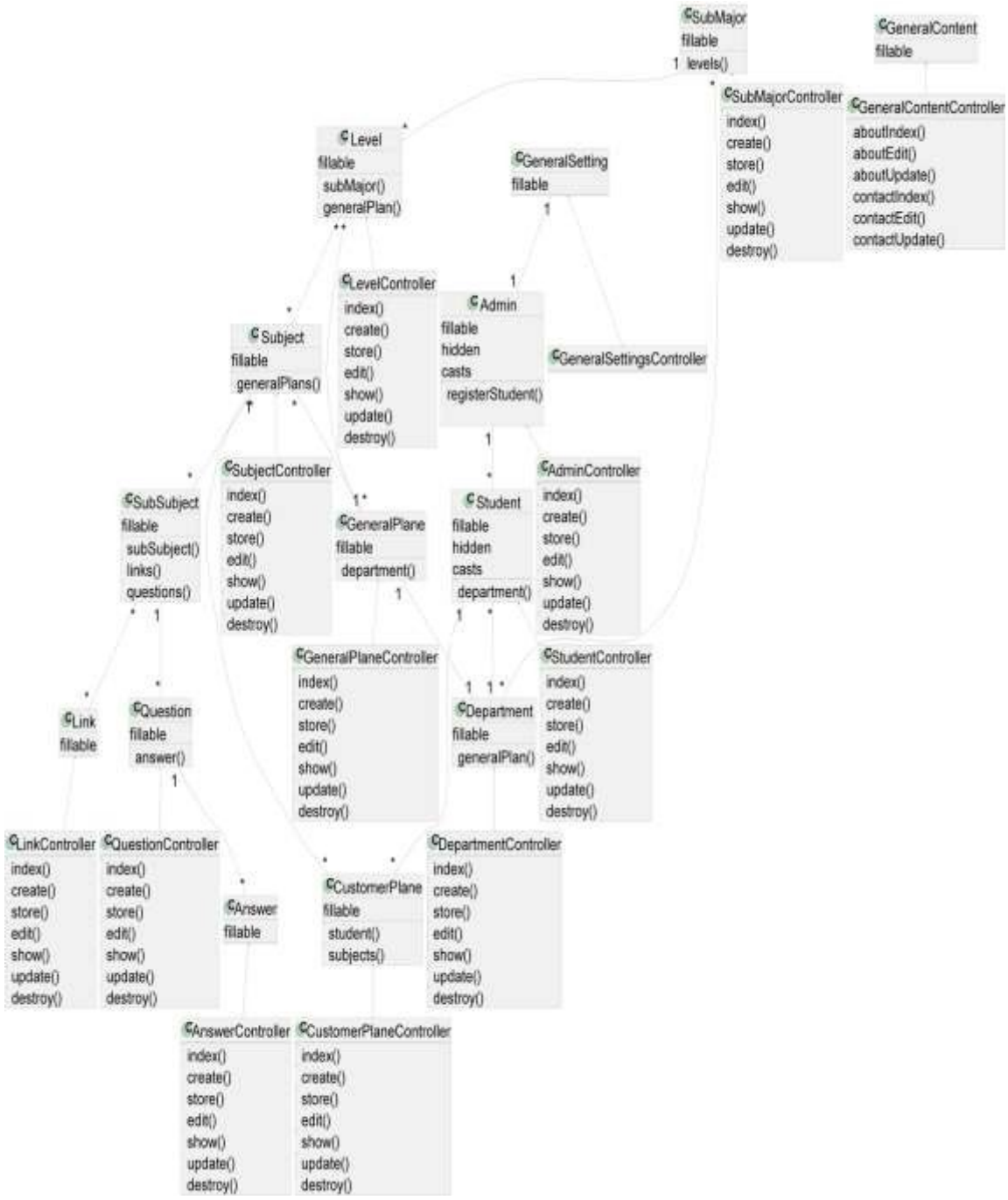


Figure 15: System Class Diagram

- **Classes and services:**

  For classes of this application, three groups of classes introduce the services of the application.

  The first group is responsible for providing the scientific content for the users. While the second and the third group are responsible for admins' accounts management. The next pages demonstrate these groups and the class diagram for each group.

1) **First group:** this group of classes contains the classes that are responsible for dealing with the scientific content of the application, which are departments, subjects, sub-subjects, questions, answers, and sub-majors. The following figure shows the class diagram for this group of classes.

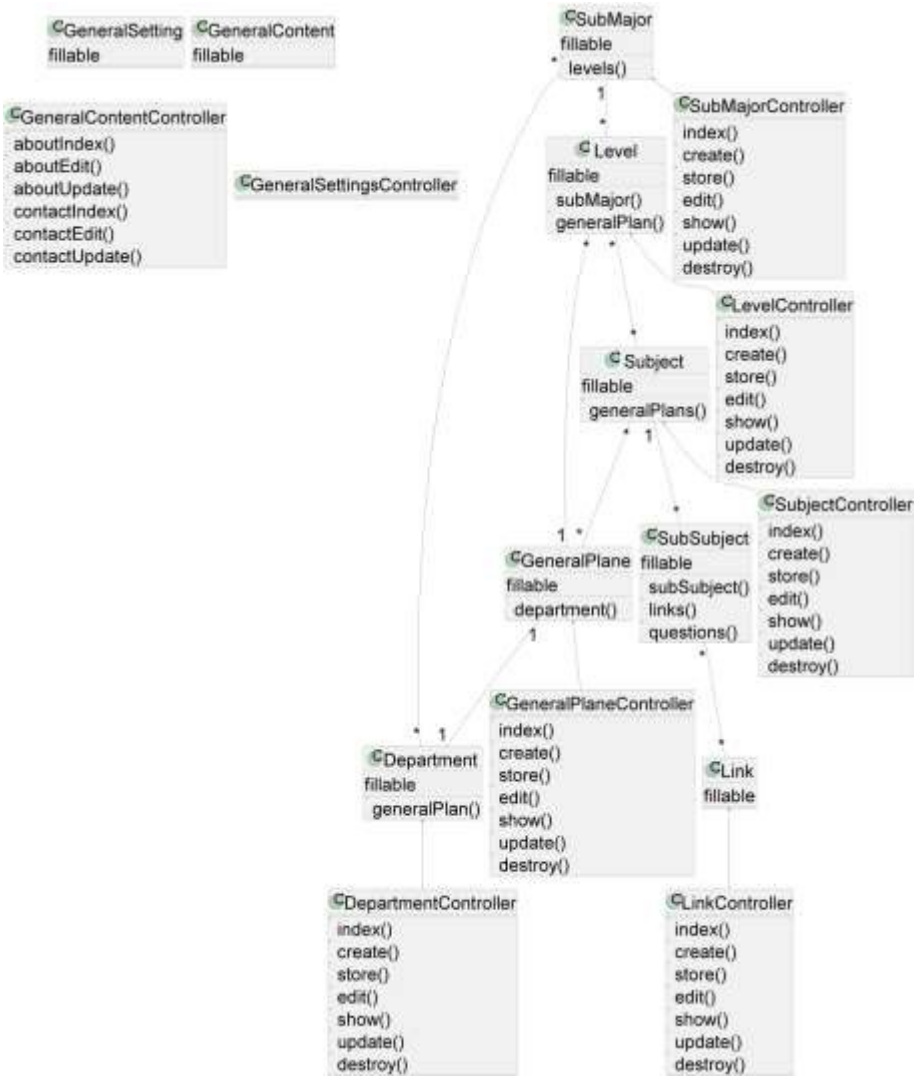


Figure 16: First Group of Classes

2) **Second group:** this group contains all the classes that are responsible for providing and managing the registered students' services just as authentication, tracking plans, and making and managing schedules. The following figure shows the class diagram for this group of classes.

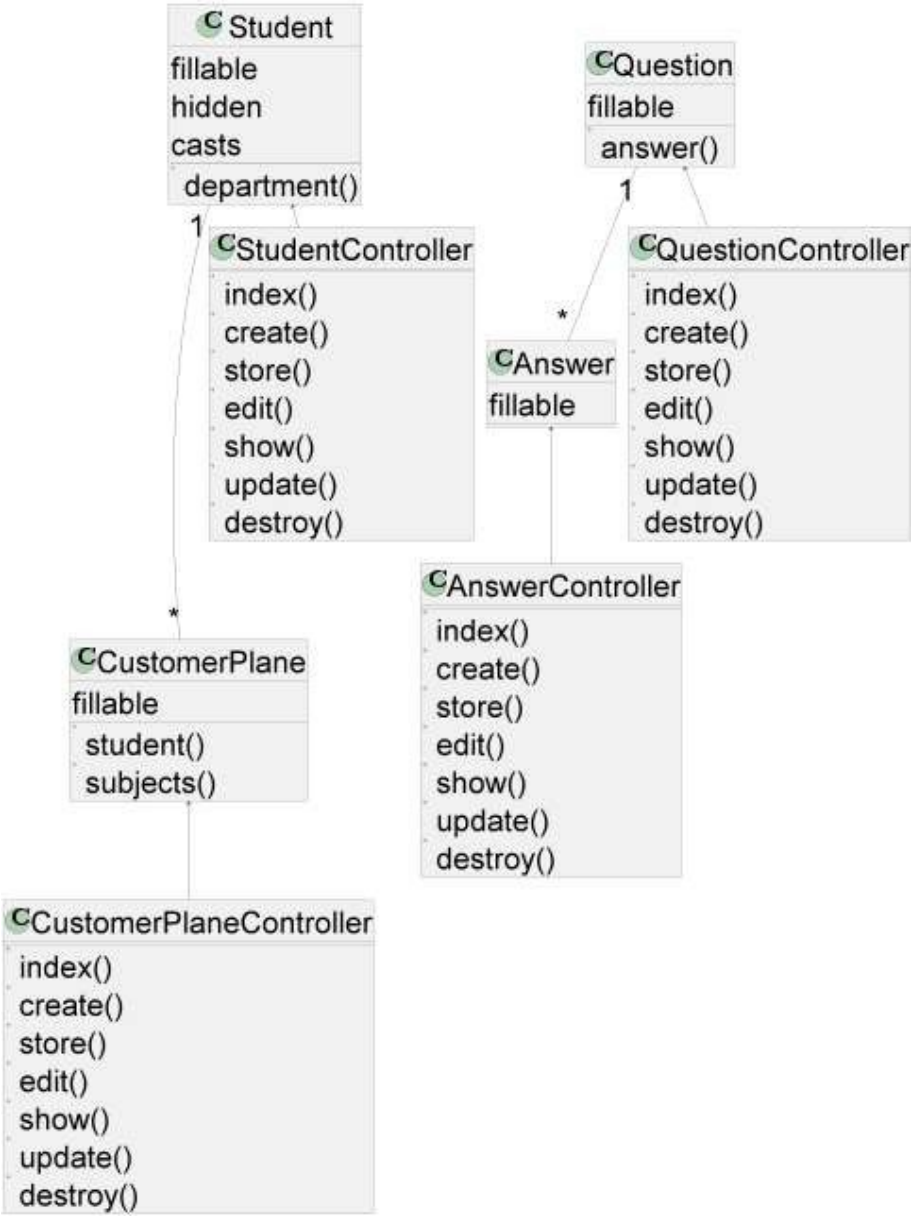


Figure 17: Second Group of Classes

3) **Third group:** this group of classes contains the admin model and admin controller, which are responsible for application admins' accounts. The following figure shows the class diagram for this group of classes.

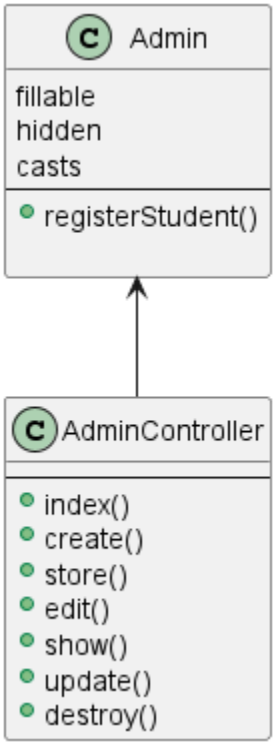


Figure 18: Third Group of Classes

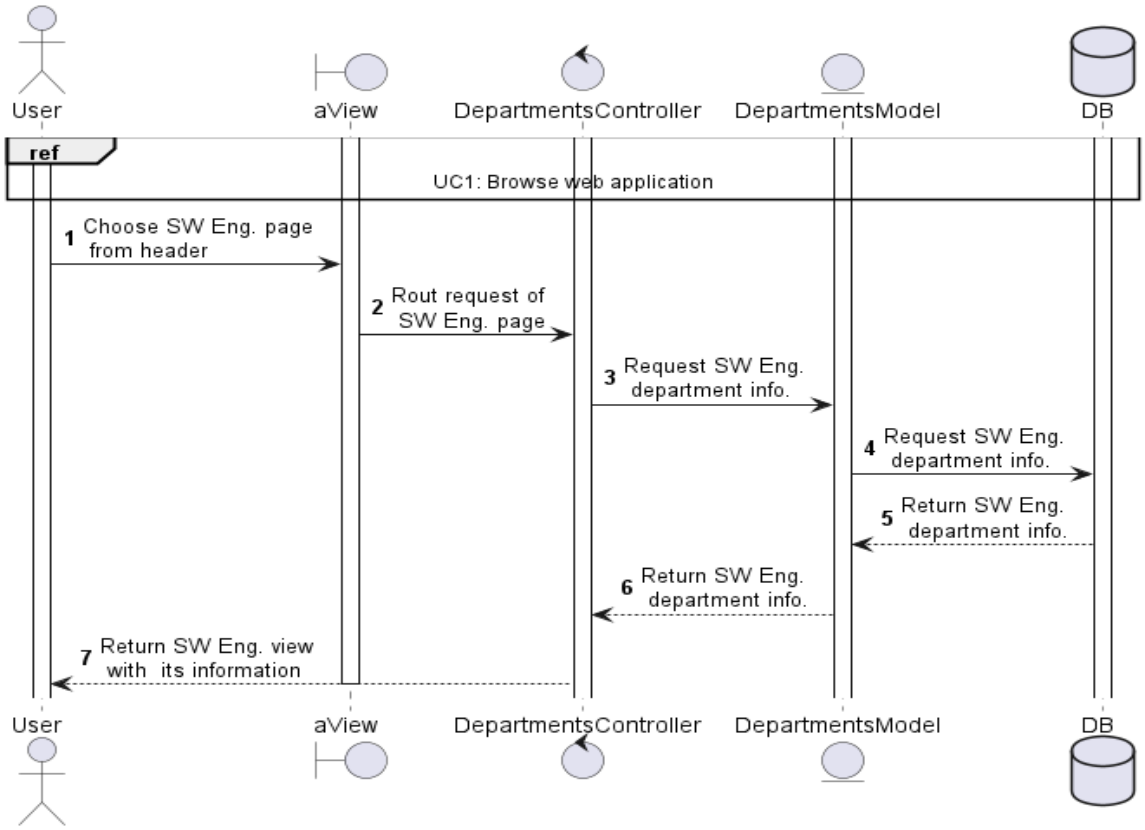
Note that all the three groups of classes are connected to construct the full class diagram of the application as illustrated in ***figure 15***.

2.2. Sequence Diagrams

Sequence diagrams are used to demonstrate how a specific operation is done, it shows the steps and operations of doing the operation and the time sequence of those steps and operations.

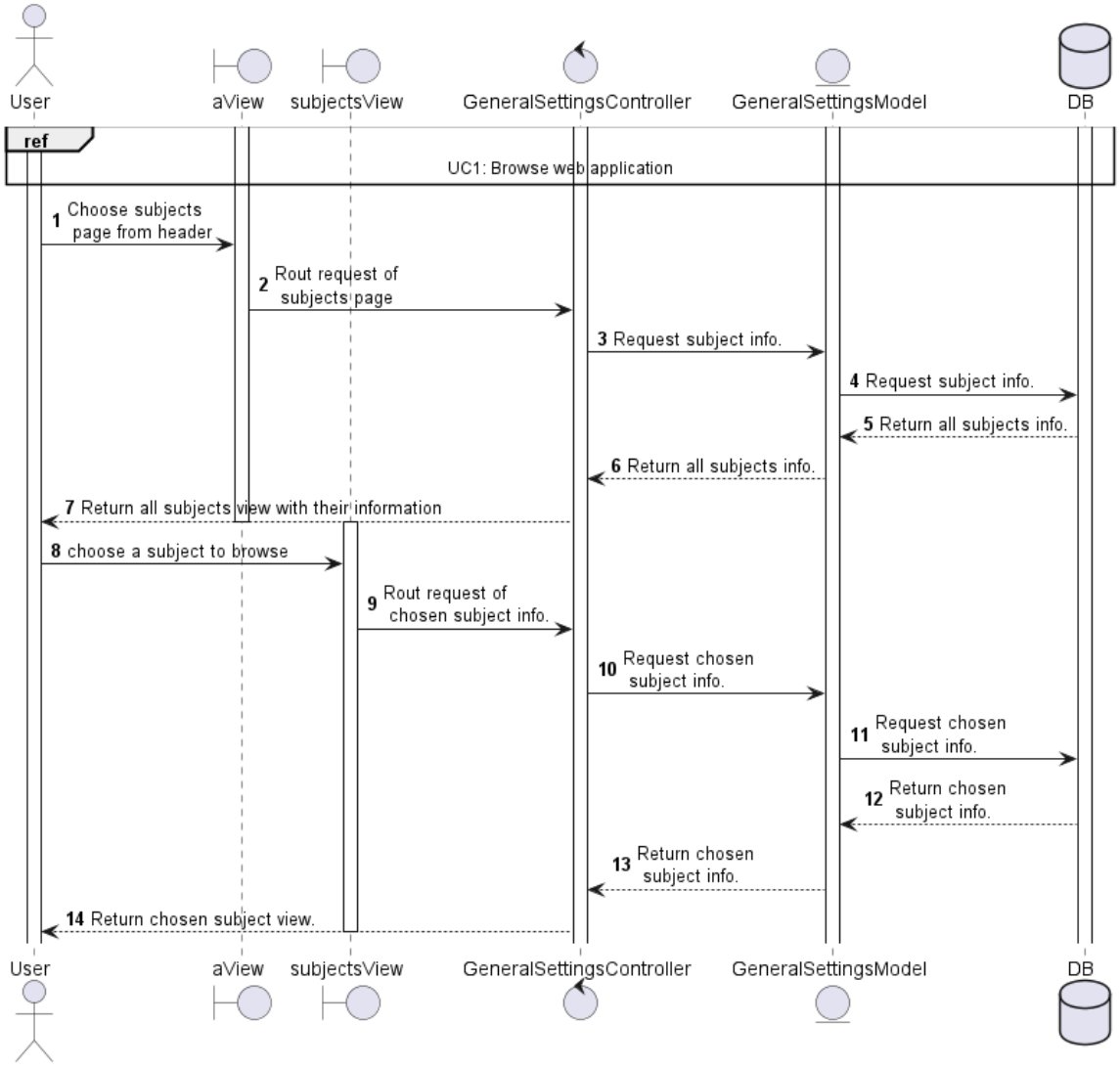
In this section, the most important and complicated sequence diagrams are shown and illustrated with a brief description. For all sequence diagrams of the application see *Appendix C*.

1) **UC 2**(user browsing SW Eng. page): a **user** can browse the SW Eng. section in the application easily by **choosing the SW Eng. page from the header** of any page of the application.



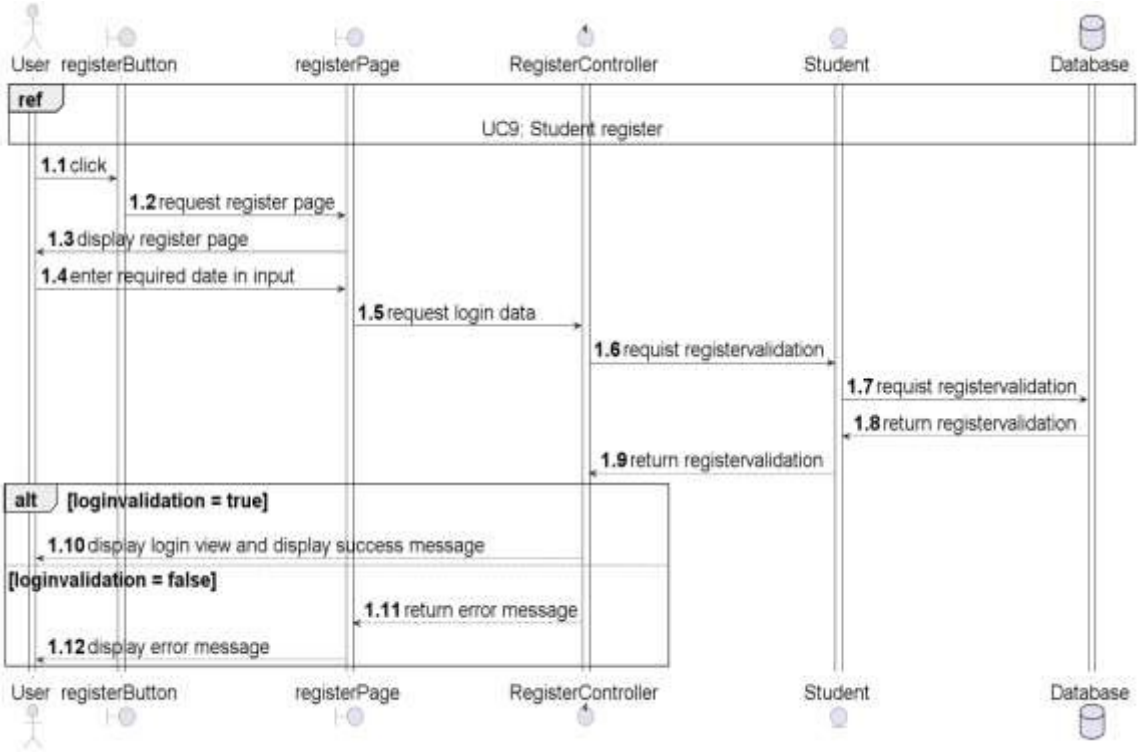
SQD 1 for UC 2

2) **UC 7**(browse a subject): a **user** browses subjects in the application by choosing **the subjects page from the header** of any page, the **system replays** with a **page of all subjects** then, the **user chooses** which **subject** to browse.



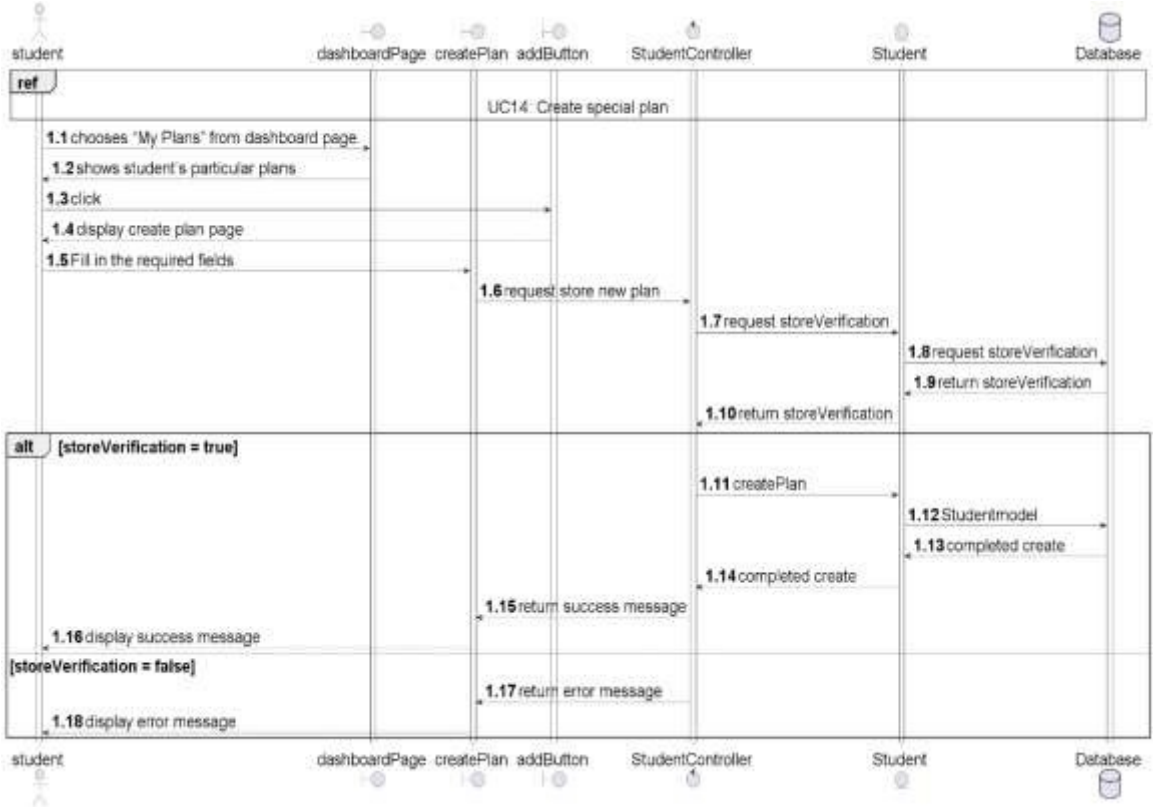
SQD 2 for UC 7

3) **UC 9**(student registration): when an ordinary user wants to create a student account in the application he **clicks on the register button** on the main page, the system will show him the **registration page**, and he **fills** in the **required information** and confirms. The **system check** entered information, **if it is true it creates an account** for the new student.



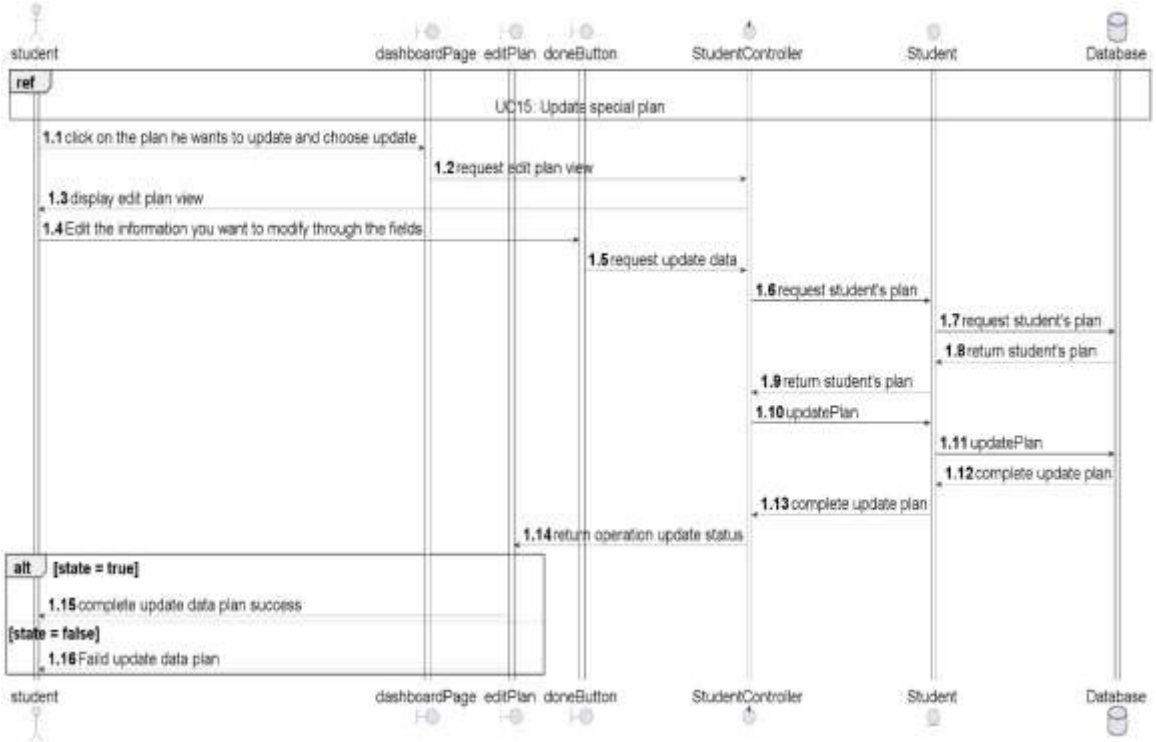
SQD 3 for UC 9

4) **UC 14**(create special plan): a **logged in student** creates a schedule for his study by choosing **My Plans** from his dashboard and choosing to **add a new plan**, the system shows a **plan form** to be filled, the student **fills** the plan and **confirms**, and the system **creates** the new plan for the student.



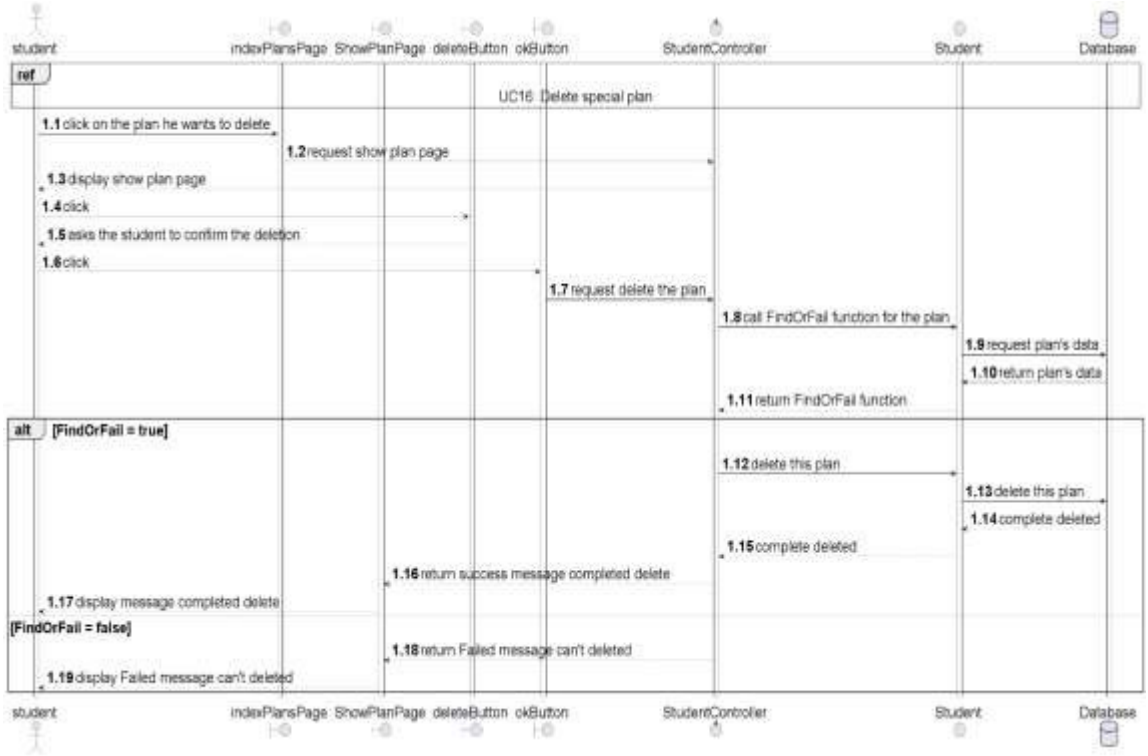
SQD 4 for UC 14

5) **UC 15**(update special plan): a **logged in student** chooses a **plan** from his plans page and chooses **Edit Plan**, system shows **the edit form** then, the student **edits** the plan and **confirms**. If the new information for the plan is **acceptable** the editing process **done** successfully.



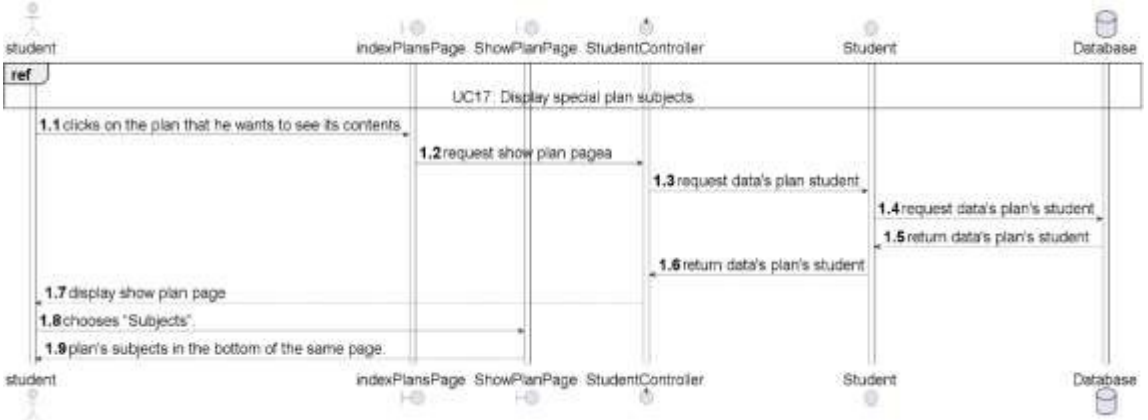
SQD 5 for UC 15

6) **UC 16**(delete special plan): a **logged in student** deletes a special plan by choosing the plan from the plans page and then clicks on **delete**, the system will ask to **confirm** deleting, the **student confirms**, the system **deletes** the plan.



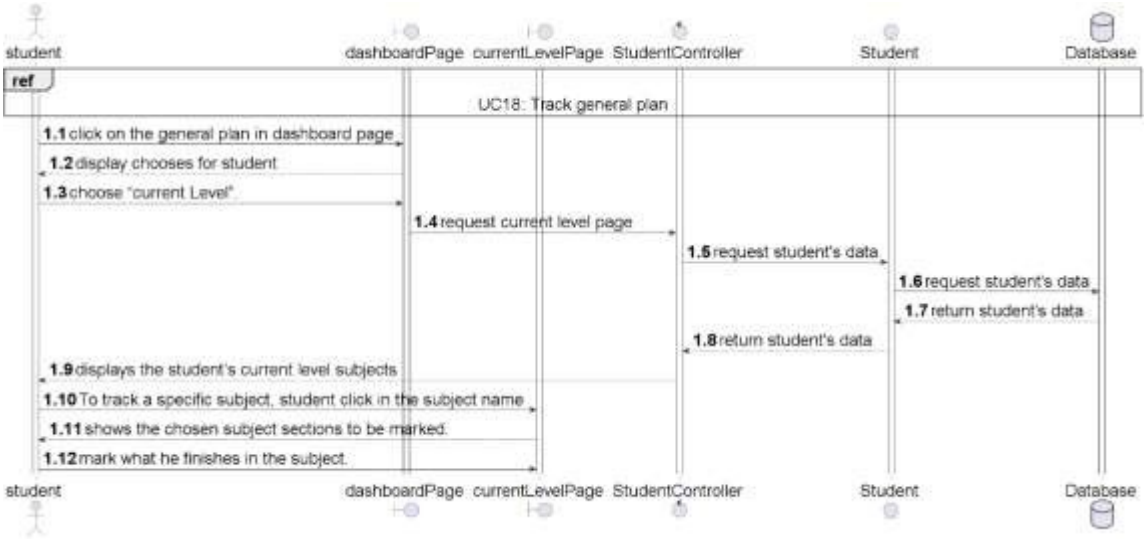
SQD 6 for UC 16

7) **UC 17**(display special plan subjects): a **logged in student** chooses a special plan to see its subjects by choosing **subjects** choice from the plan page.



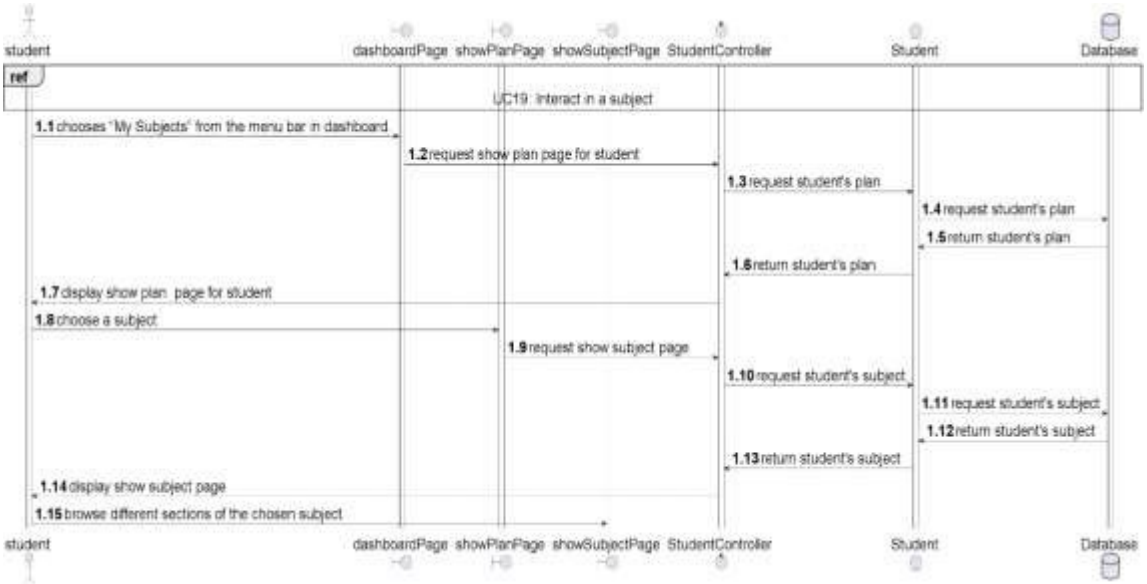
SQD 7 for UC 17

8) **UC 18**(track general plan): a **logged in student** clicks on **the general plan** from his dashboard, and the system displays the student's general plan. The student chooses **his current level** and then chooses the **subject** to **mark** his new progress in.



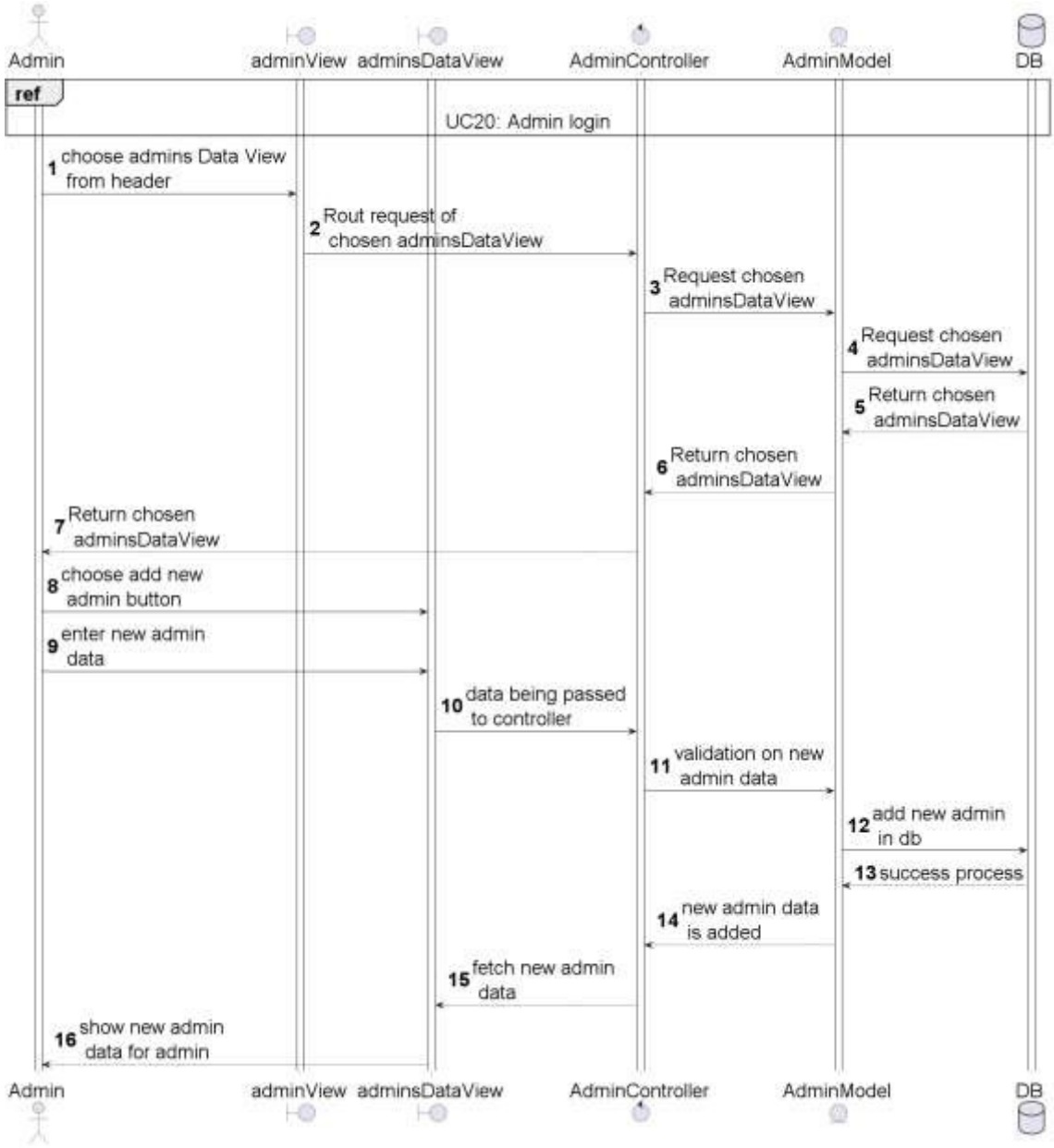
SQD 8 for UC 18

9) **UC 19**(interact in a subject): a **logged in student** access a specific subject from the **My Subjects** option on his dashboard, the system **opens** the chosen subject page then the student **interacts** inside the subject.



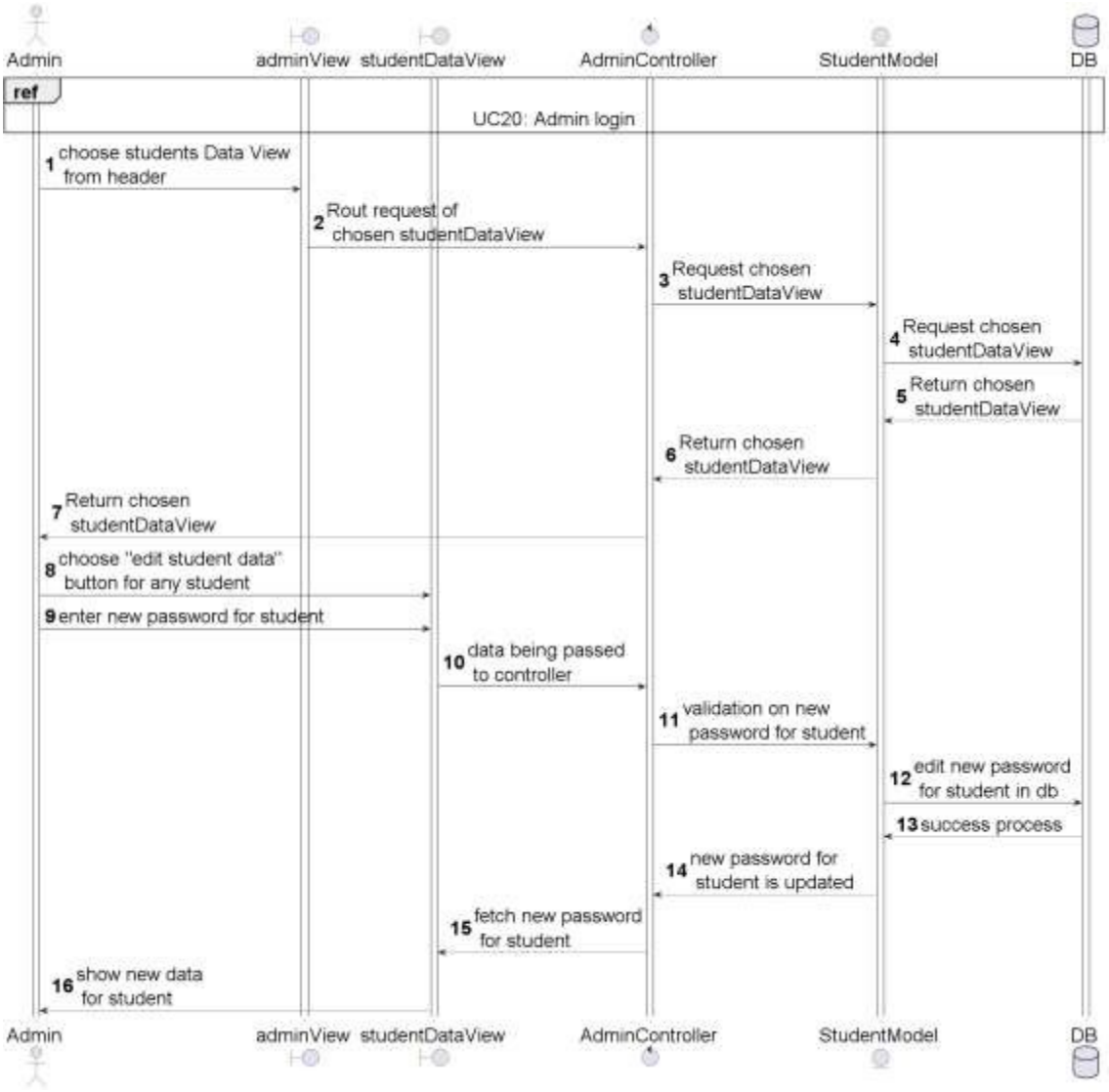
SQD 9 for UC 19

10) **UC 23**(add new admin): a **logged in super admin** chooses **admins data**, the system shows **the admins data page**, the admin chooses to **add admin**, the system asks for new admin's information, then the admin **fills** the information and **confirms** adding.



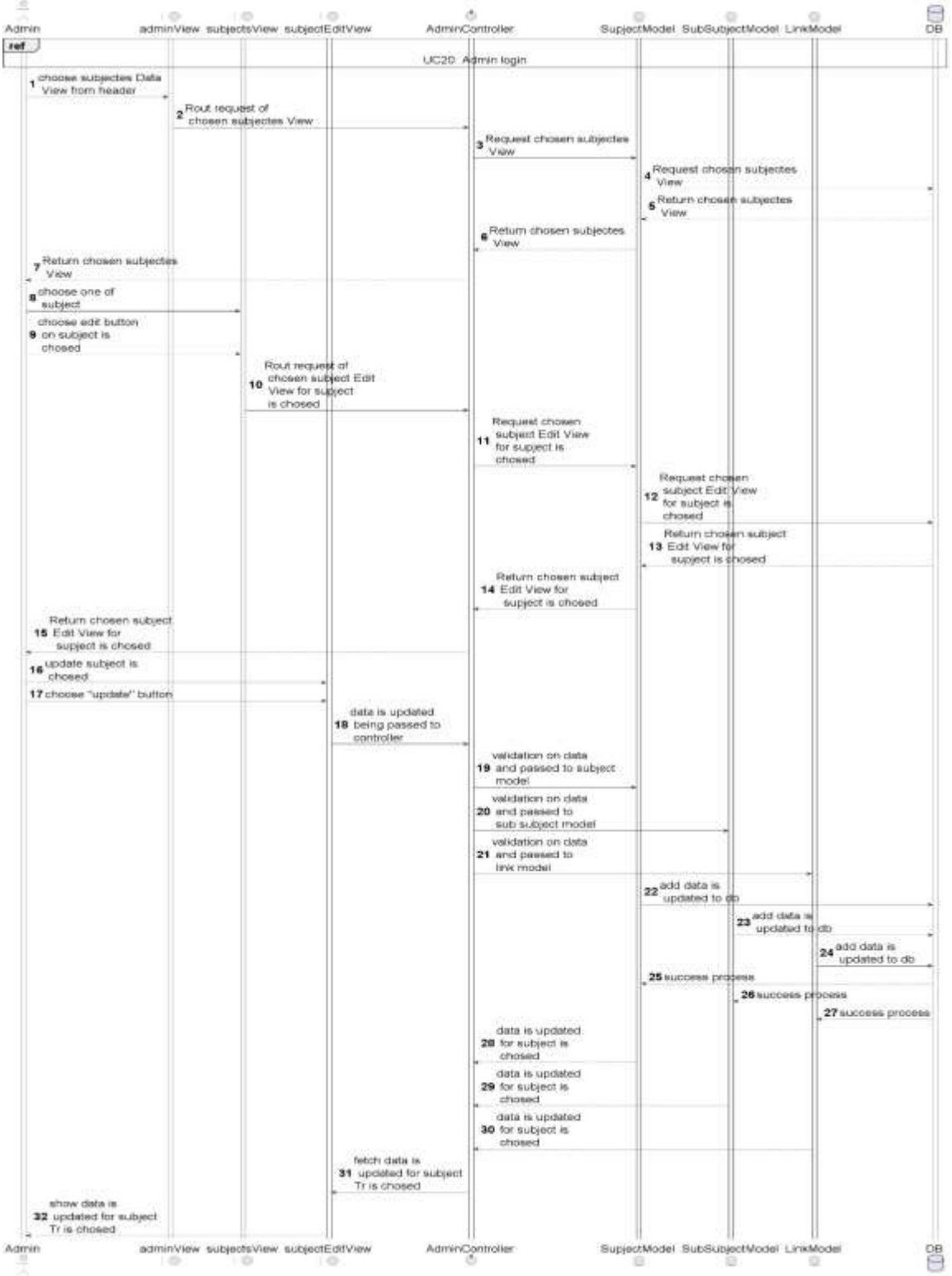
SQD 10 for UC 23

11) UC24(admin change student password): a **logged in admin** chooses **students data**, the system shows **the students data page**, the admin chooses **to edit student data and enters a new student's password**, and the system **stores** the new password for the student.



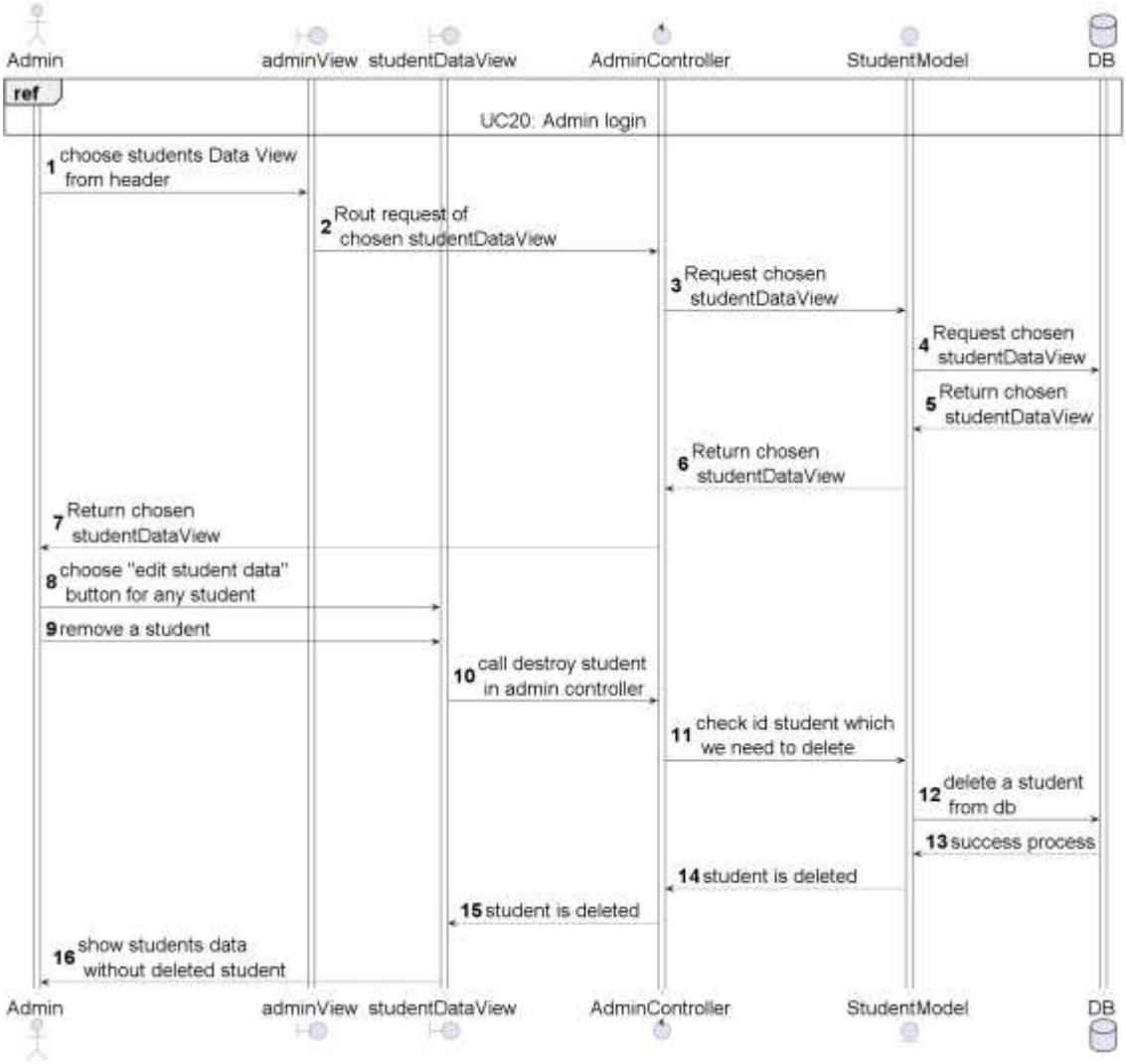
SQD 11 for UC 24

12) **UC 26**(change subject content): a **logged in admin** chooses **subjects data**, the system shows **the subjects data page**, the admin chooses **to edit subject data and enters new changes**, and the system **stores** new data for the subject.



SQD 12for UC 26

13) UC 29(remove a student): a **logged in admin** chooses **students data**, the system shows **the students data page**, the admin chooses to **delete a student** for a specific student, the system **asks to confirm**, the admin **confirms** deleting then the system **deletes** the selected student.



SQD 13 for UC 29

3. Database Design

3.1. Overview

An essential part for any project in design phase is that the design of the database of the system.

While the designed system is a application, the database needed for the system must be compatible with application structure, and the server structure.

As mentioned before, the implementation method for implementing this application is the PHP Laravel framework, and because of that, the type of the system database will be a MySQL database.

3.2. Database Schema

There were many types of database schemas, in this section, the logical database schema design will be used to illustrate tables, datatypes, relations and constraints of the system database schema.

The database schema for this system will be discussed in the three following points:

1) Database tables:
   The forms below demonstrate each table schema which includes table name, table attributes, and attributes types and constraints.

   The database schemas bellow is divided into two sections, the first is for tables that containing the essential content for the application. While the second is tables for translating the many-to-many relations between tables.

- First section:

Admin		
Attribute	***Type***	***Constraints***
id	UUID	
username	STRING	
email	STRING	Email form
password	STRING	
name	STRING	
image	TEXT	

Table Schema 1: Admin Table

Students		
Attribute	***Type***	***Constraints***
id	UUID	
username	STRING	
email	STRING	Email form
password	STRING	
name	STRING	
image	TEXT	

Table Schema 2: Students Table

Departments		
Attribute	***Type***	***Constraints***
id	UUID	
name	STRING	
definition	STRING	
description	STRING	
general_plan_id	UUID	Id in general plans table

Table Schema 3: Departments Table

Subjects		
Attribute	***Type***	***Constraints***
id	UUID	
name	STRING	
definition	STRING	
department_id	UUID	Id in departments table

Table Schema 4: Subjects Table

Sub subjects		
Attribute	***Type***	***Constraints***
id	UUID	
title	STRING	
description	STRING	
Subject_id	UUID	Id in subjects table

Table Schema 5: Sub subjects Table

Levels		
Attribute	***Type***	***Constraints***
id	UUID	
description	STRING	
sub_major_id	UUID	Id in sub majors table
general_plan_id	UUID	Id in general plans table

Table Schema 6: Levels Table

General plans		
Attribute	***Type***	***Constraints***
id	UUID	
department_id	UUID	Id in departments table

Table Schema 7: General Plans Table

Custom Plans		
Attribute	***Type***	***Constraints***
id	UUID	
student_id	UUID	Id in students table

Table Schema 8: Custom Plans Table

Links		
Attribute	**Type**	**Constraints**
id	UUID	
type	LinkType	Type is enum type (video, document, website)
link	TEXT	

Table Schema 9: Links Table

Questions		
Attribute	***Type***	***Constraints***
id	UUID	
title	STRING	
textOfQuestion	TEXT	
sub_subject_id	UUID	Id in sub subjects table

Table Schema 10: Questions Table

Answers		
Attribute	***Type***	***Constraints***
id	UUID	
textOfAnswer	TEXT	
question_id	UUID	Id in questions table

Table Schema 11: Answers Table

Sub majors		
Attribute	***Type***	***Constraints***
id	UUID	
name	STRING	
definition	STRING	
description	TEXT	

Table Schema 12: Sub majors Table

General Settings		
Attribute	***Type***	***Constraints***
websiteName	STRING	
about	TEXT	
phone	STRING	Numbers
email	STRING	Email form
logo	TEXT	
location	TEXT	
section	TEXT	Html, CSS, and JS code

Table Schema 13: General Settings Table

- Second section:

Name	***Relation many_to_many between:***	
subject_general_plan	subjects	general_plans
Attribute	***Type***	***Constraints***
subject_id	UUID	Id in subjects table
general_plan_id	UUID	Id in general plans table

Table Schema 14

Name	***Relation many_to_many between:***	
levels_subjects	levels	subjects
Attribute	***Type***	***Constraints***
level_id	UUID	Id in levels table
subject_id	UUID	Id in subjects table

Table Schema 15

Name	***Relation many_to_many between:***	
subject_custom_plan	subjects	custom_plans
Attribute	***Type***	***Constraints***
custom_plan_id	UUID	Id in custom plans table
subject_id	UUID	Id in subjects table

Table Schema 16

Name	***Relation many_to_many between:***	
department_sub_major	departments	sub_majors
Attribute	***Type***	***Constraints***
department_id	UUID	Id in departments table
sub_major_id	UUID	Id in sub majors table

Table Schema 17

Name	***Relation many_to_many between:***	
sub_subject_link	sub_subjects	links
Attribute	***Type***	***Constraints***
sub_subject_id	UUID	Id in sub subjects table
link_id	UUID	Id in links table

Table Schema 18

2) Primary keys:
In a system that implementing in PHP Laravel framework, it allows to specify the primary key for the table when creating it. Most tables specifying an id attribute with the type of UUID to be a primary key, which is a unique number that are defined automatically when adding a new record to the table.

The *id primary key* used in *first section of tables* in the database schemas of this application. But in *many_to_many relation tables*, the *primary key* is specified to be *the combination of the two foreign keys* from the tables that the relation between.

3) Relations:
The following table shows the relations between tables that mentioned in the *database tables* point in this section:

Table 16: Database Tables Relations

Relation Type	Between
One_to_one (1 to 1)	general_palns to departments
	general_palns to admin
One_to_many (1 to *)	sub_majors to levels
	general_plans to levels
	general_plans to subjects
	admin to subjects
	subjects to sub_subjects
	sub_subjects to questions
	questions to answers
	students to custom_plans
	departments to students
Many_to_many (* to *)	sub_majors with links through sub_subject_link
	departments with sub_majors through department_sub_major
	subjects with custom_plans through subject_custom_plan
	subjects with general_plans through subject_general_plan
	levels with subjects through level_subject
Ordinary association	Admin with general_settings

3.3. System Database Diagram

A system database diagram is a diagram that shows all tables in a system database with their attributes and attributes types, also it illustrates relations and relations types between tables.

The next page shows the diagram of the database for this application system.

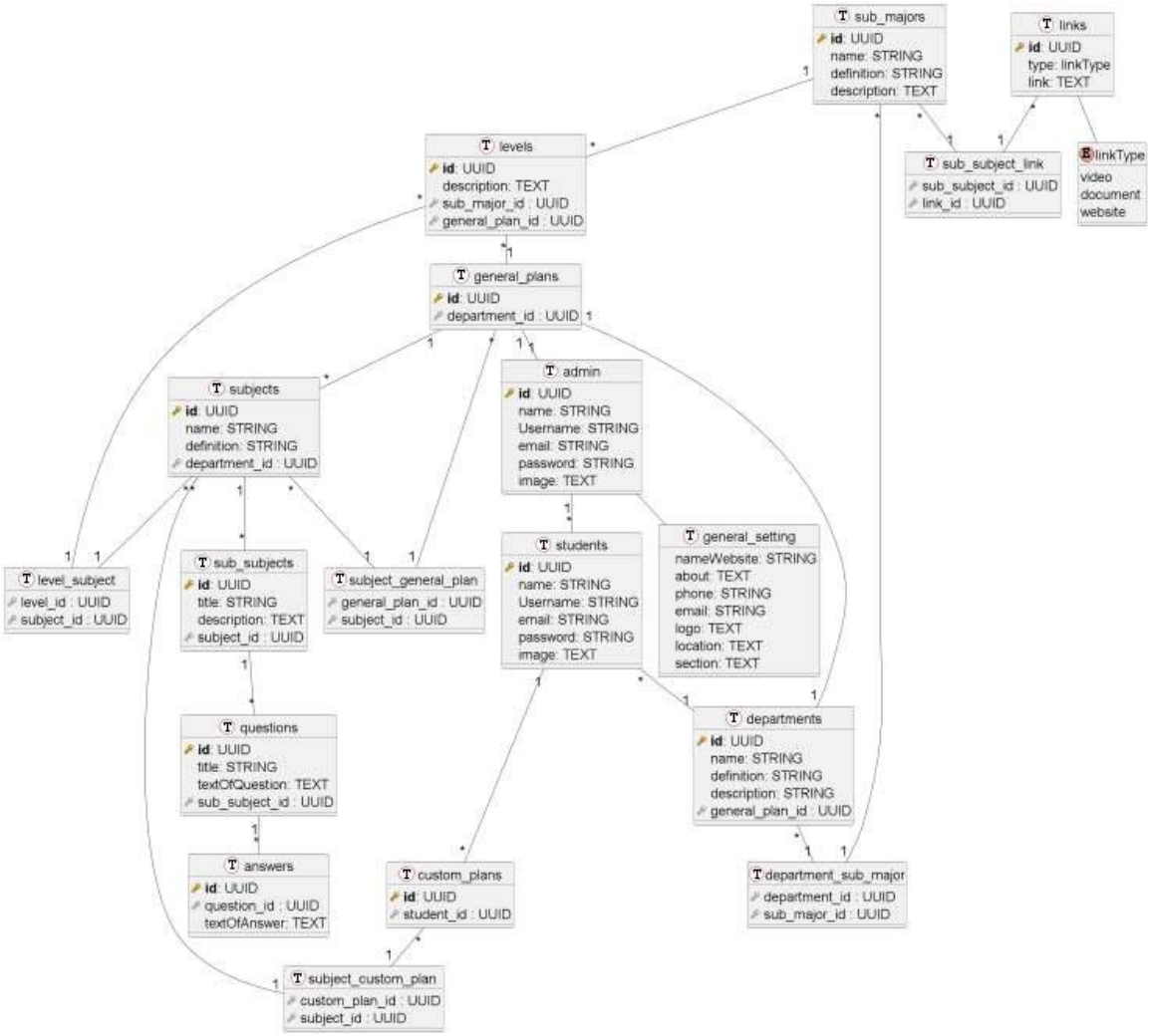


Figure 19: System Database Diagram

Chapter Seven

System Implementation

Chapter Seven: System Implementation

This chapter demonstrates how implementation process of the application is done by illustrating the implementation process phases which are the backend development, the frontend development, and the application execution.

1. Implementation Methodology

1.1. Overview

As known, any web application development consists of two main development processes, frontend development which considers the design of the application web pages, and backend development which is responsible for implementing application algorithms and managing application work.

1.2. Development Methods

1. **Backend development method:**

   There were many implementation methods used in backend development, but as mentioned in previous sections of this document the PHP Laravel framework used for implementing the backend section for this application. Here are the reasons why the Laravel framework has been chosen:

   1) It is the project team's field of work.
   2) It supports the **MVC model**.
   3) It provides **modularity** future for the application which is needed for future evolutions for the application.
   4) It contains a **full built-in authentication system**, which makes it easier to make authentication in the application.
   5) It serves as an API backend to a JavaScript single-page application or mobile application [10].

   And there are many other features, but the ones mentioned above are the most important features for the application.

2. **Frontend development method:**

   The implementation method used for implementing the frontend section in the application is the React JS framework.

It has been used because it works perfectly with the Laravel framework. And also, because it provides flexibility to the work and improves the total rendered pages from the application server [11].

2. Backend Development

2.1. Setup Laravel Project

Setup the Laravel project is the first step in the development of the backend section for the application. This step is responsible for downloading all essential files and the needed libraries for the Laravel application.

2.2. Preparing System Database

The system database is an essential component of the application. It is needed for storing the application data. Preparing the system database includes three main steps:

1. **Creating the database:**

   This is the first step in preparing the application database, it done by creating a *MySQL* database using *Xampp* program.

   In this step, the main characteristics of the application database are specified. These characteristics are:
   1) **Name** of the database.
   2) **Type** of the database.
   3) **Language** of the data in the database.
   4) **Encryption type** used in the database.

2. **Database configuration:**

   Database configuration responsible for binding the application with its database through setting a number of configurations. The following lines demonstrates how these configurations are setting.

   Database configuration is done through the .env file in the Laravel project, which contains many configuration options for the whole application.

The following table shows the database configuration options with their appropriate values for the application:

Table 17: Database Configurations

Option	Value	Meaning
DB_CONNECTION=	mysql	Database connection type.
DB_HOST=	127.0.0.1	Database host address.
DB_PORT=	3306	Database port to access it from.
DB_DATABASE=	laravel	Database name.
DB_USERNAME=	root	Username of the database user.
DB_PASSWORD=		User account password.

3. Creating data tables:

Creating data tables is the last step in the database preparation for the application. This operation is done in two steps, creating tables schema and migrating these schemas to the application database.

The creation of tables' schemas is done by creating a **PHP source file** for each table containing the functions that are responsible for creating, deleting, and updating the table schema. This file is stored in the ***migrations*** folder in the ***database*** folder in the Laravel application.

After creating the migrations files which contain tables schemas as mentioned in the design phase in *Chapter Six, Database Design, Database Schema* in this document, these files are migrated to apply the tables' schema to the application database using the command line role: ***php artisan migrate.***

2.3. Implementing Models

In a Laravel application, Models are classes that reflect data tables in the application database, and through models' data objects in the database can be retrieved, added, updated and deleted.

Implementing models in the Laravel application is done by creating PHP source files containing model classes with the same name as their data table in the database, but starting with uppercase. These files are stored in the ***Models*** folder in the ***app*** folder in the application project.

Models' classes contain special attributes, custom methods, and relations methods which are responsible for specifying relations between models, these relations reflect the relations between the opposite tables in the database.

The implemented models in this application are the models discussed in *Chapter Six, System UML Diagram, Class Diagram* section of this document.

2.4. Implementing Controllers

Controllers are PHP classes which contain several methods that control the processing of the application. Controllers receive requests, process requests and manipulate data through models, and return views to the client.

Each controller class is implemented in a PHP source file which is stored in the ***Controllers*** folder in the ***Http*** folder in the ***app*** folder in the application project.

The implemented controllers in this application are the controllers discussed in *Chapter Six, System UML Diagram, Class Diagram* section of this document.

2.5. API Routing

The routing technique is a process used for executing a specific method of a specific controller when a call of a specific URL is done by the browser.

API Routing is a type of routing in the Laravel application used for returning JSON data, which is the data that the application process. Also, this JSON data is required for React JS in the frontend development process.

There are many methods used for the routing process, but in this application, there are four methods that are used to route the application's URL which are Get, Post, Put, and Delete.

3. Frontend Development

Frontend development is the second phase in developing the application after the backend development. This phase is implemented using the React JS framework because React JS works perfectly with the Laravel framework. And also, it provides flexibility to the work and improves the total rendered pages from the application server.
The following demonstrates how this phase was implemented and developed as a process in the application development process.

3.1. Setup React JS

The first step of starting work is setting up the React project, this step is done by downloading React library as the begin of the frontend process.

After starting the React project, several of libraries and packages have been installed within the React environment. These packages and libraries are:

1) **React-Bootstrap library**: this library replaces the Bootstrap JavaScript. Each component has been built from scratch as a true React component, without unneeded dependencies like jQuery. [12]

2) **React-Router library**: this library is responsible for declaring routes for the react application.

3) **Axios library**: this library is used for the API requesting process in the react application.

4) **Redux library**: this library responsible for the process of state management in the application.

3.2. Construct Routes

After setting up the react project and installing the needed libraries for the work, the process of constructing routes is started.

Before creating the routes, routes are logically categorized into five groups according to the user of the application. These groups are **user** group, **auth** group, **student** group, **admin** group and **not found** group.

After categorizing the routes, the process of creating URL's done by the **React-Router** library.

3.3. Components Design

In this step, applications' web pages are designed and constructed using HTML, CSS, and React-Bootstrap library.

Before starting the design, pages of the application are logically categorized as the routes in five groups, which are: **user** group, **auth** group, **student** group, **admin** group and **not found** group.

The next pages show the final actual design of the most important pages in the application.

1) **Home page**: Home page: it is the application's main interface which from it can reach other pages in the application.

Figure 20: Home Page

2) **A course page**: this page contains all the contents and the supporting methods of a particular subject.

Figure 21: A Course Page

3) **Student dashboard**: the dashboard that is used by logged in students to help them accomplish their work, just like editing profiles, tracking plans and managing study schedules.

4) **Admin dashboard**: the dashboard that is used by logged in admins to help them manage the applications' work.

3.4. REST API's

REST API is an API that conforms to the design principles of the REST, or *representational state transfer* architectural style. It provides a flexible, lightweight way to integrate applications, and have emerged as the most common method for connecting components in microservices architectures. [13]

To display the coming data from the API, APIs requests are sent by Axios library in a get method that returns the response, which is a JSON data displayed in the components.

Also, to create, update or delete data from the Database, APIs requests are sent by Axios library in a post, put, or delete method that returns the response, which sure process success or failure.

3.5. State Management

Providing state management is one of the most important features managing data in the application.

In React JS, state management provided using the Redux library which creates a data store for the data coming from the APIs requests. After that, components can access the data using the created store.

References

References

[1] Wikipedia, "Model–view–controller - Wikipedia," [Online]. Available: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller#:~:text=Model%E2%80%93view%E2%80%93controller%20%28usually%20known%20as%20MVC%29%20is%20a%20software,is%20presented%20to%20and%20accepted%20from%20the%20user..

[2] C. Team, "MVC: Model, View, Controller | Codecademy," [Online]. Available: https://www.codecademy.com/article/mvc.

[3] V. Beal, "What is a Web Browser? | Webopedia," [Online]. Available: https://www.webopedia.com/definitions/browser/.

[4] "What is Web Server? Definition of Web Server, Web Server Meaning - The Economic Times (indiatimes.com)," [Online]. Available: https://economictimes.indiatimes.com/definition/web-server.

[5] M. North, "What Is a Web Database? | Techwalla," [Online]. Available: https://www.techwalla.com/articles/what-is-a-web-database.

[6] "yD0eH.jpg (1024×616) (imgur.com)," [Online]. Available: https://i.stack.imgur.com/yD0eH.jpg.

[7] "Database: Query Builder - Laravel - The PHP Framework For Web Artisans," [Online]. Available: https://laravel.com/docs/9.x/queries.

[8] "Eloquent: Serialization - Laravel - The PHP Framework For Web Artisans," [Online]. Available: https://laravel.com/docs/9.x/eloquent-serialization.

[9] "Eloquent: Mutators & Casting - Laravel - The PHP Framework For Web Artisans," [Online]. Available: https://laravel.com/docs/9.x/eloquent-mutators#attribute-casting.

[10] Laravel.com. [Online]. Available: https://laravel.com/docs/9.x/installation#laravel-the-fullstack-framework.

[11] Thinkwik. [Online]. Available: https://medium.com/@thinkwik/why-reactjs-is-gaining-so-much-popularity-these-days-c3aa686ec0b3#:~:text=Because%20ReactJS%20helps%20to%20prevent%20updating%20of%20DOM%2C,the%20total%20rendered%20pages%20from%20the%20website%20server..

[12] R. Bootstrap. [Online]. Available: https://react-bootstrap.github.io/.

[13] I. C. education. [Online]. Available: https://www.ibm.com/cloud/learn/rest-apis.

Appendices

Appendices

1. Appendix A: Simple Analysis for The Application

Appendix A: Quick Analysis for The Application

This section contains simple analysis of the application that being developed, this simple analysis talks about main pages in the application and how moving between these pages. This quick analysis provides the project team with some information for functional requirements analysis, also helps the team to find the overall structure of the system.

1. Main pages in the application:

    1) *Home Page*: the main page in the application that appears directly when someone inters the application.
    2) *SW Eng. Page*: contains all information about SW Eng. major that can help students of this major.
    3) *MecTr Eng. Page*: contains all information about MecTr Eng. major that can help students of this major.
    4) *About Page*: the page that enables users of this application to know about who developed this application.
    5) *Student Sign in Page*.
    6) *Student Sign up Page*.
    7) *Student Profile Page*: contains the student's main information, and student plans for his major.
    8) *Admin Login Page*.
    9) *Admin Page*: which enables site admins to control and manage this application.

2. Essential content for the application essential pages:

    1) *Home Page*: Logo and Pages Menu, Welcoming and Site Definition, Majors Definitions, Contact Us, Copy Rights.

    2) *Software and MecTr Eng. Pages*: Logo and Pages Menu, side: Definition, side: General Static Plan [Levels and Materials], side: Particular Major Plans [Sub Majors [in: Levels and Materials]], Contact us, Copy Rights.

3) *About us*: Logo and Pages Menu, About the Developers, Contact Us, Copy Rights.

3. Application sitemap:

The application sitemap describes the application pages and the path of transmitting between these pages. The following shape describes the sitemap of the application that was analyzed before in this section:

Sitemap keys to best understand the following sitemap:

1. Oval Shape: For pages that accessed directly when opening the application.
2. Rounded Rectangle: main pages that are accessed by any user.
3. Regular Rectangle: admins pages after entering the application as an admin.
4. Circle: students pages after entering the application as a student.
5. Filled Shapes: just connectors between arrows and other arrows.
6. Arrows: connectors between pages showing how can transfer between pages.

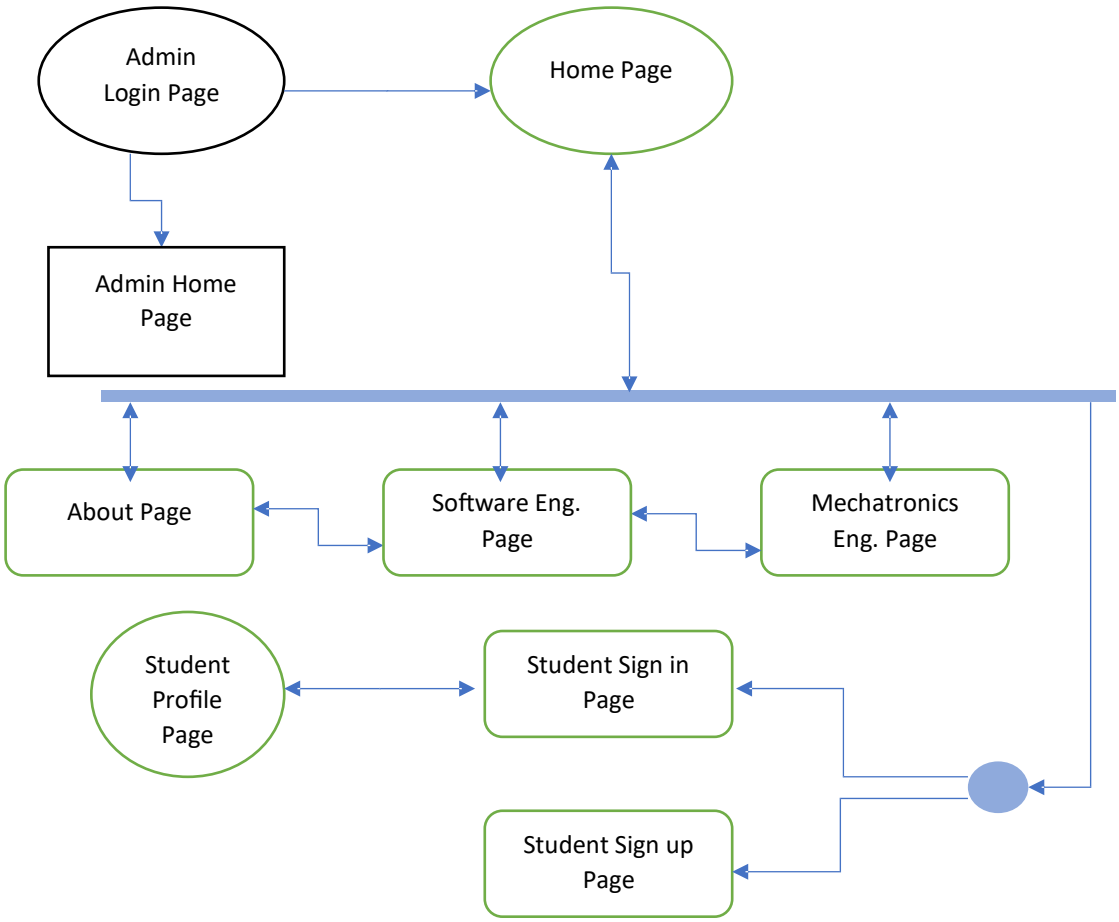


Figure 22: First Sitemap - Quick Analysis

4. Functional requirements from this analysis:

   Here are some functional requirements that appear from this analysis:
   1. Application must describe a general plan for each engineering major (SW Eng. and MecTr Eng.).
   2. Application must allow the user to contact application owners from any application page.
   3. Each engineering department must contain an overview of its science.
   4. The application must enable users to be students in the application (registration).
   5. The application must have at least one admin to manage its contents.
   6. Students have a profile page in the application.
   7. Only when a student signs in can go to his profile page.
   8. The application should allow the student to create a study plan.
   9. The application should allow the student to update his study plan.

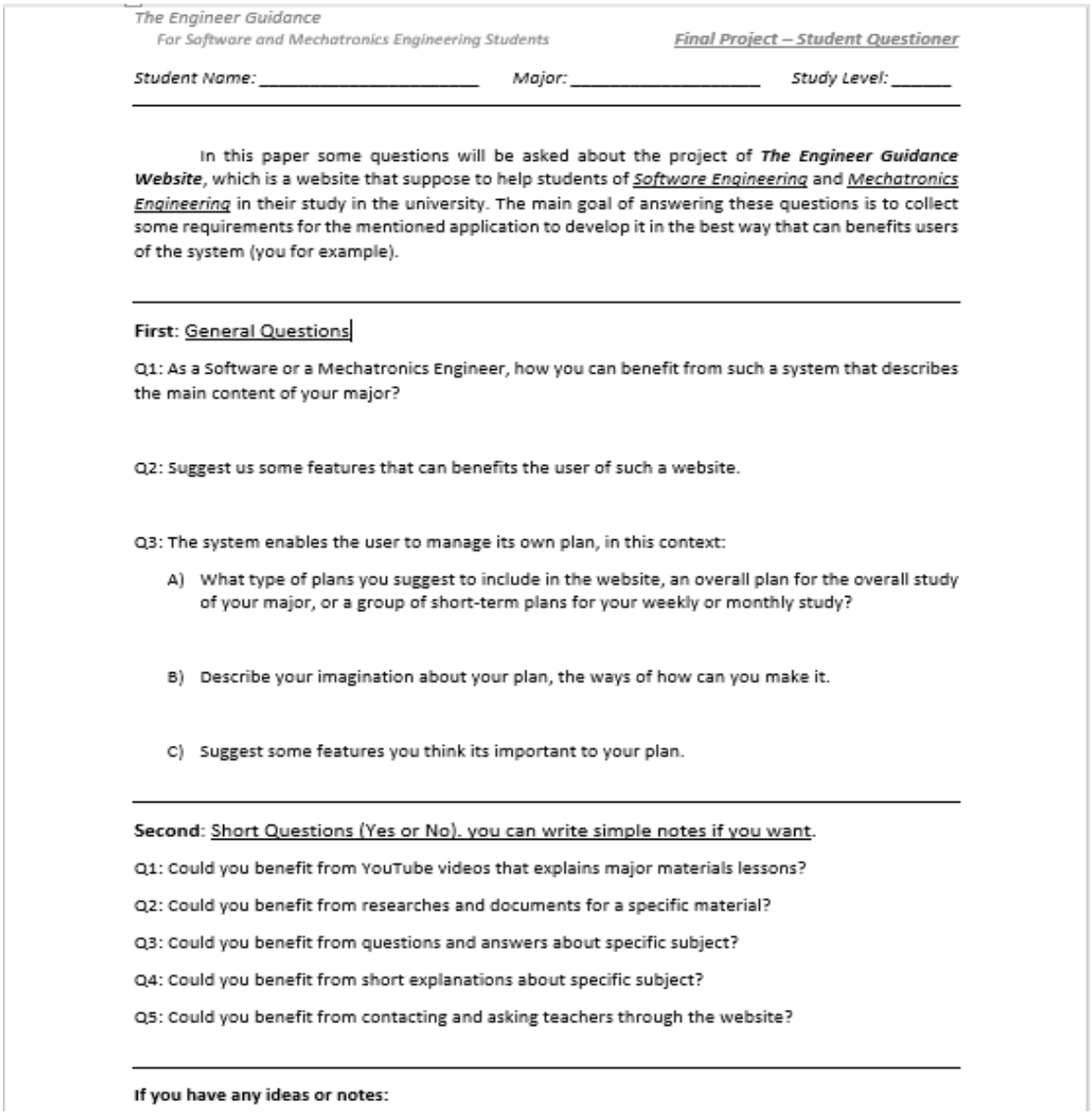
2. Appendix B: Student Questioner

Appendix B: Some Questions for Students for Collecting Functional Requirements

This section includes the answers to the questions that have been asked to some students to collect some functional requirements for the application to be developed.

1. The asked questions:

   The below picture shows the questionnaire document that has been sent to students to answer:



The Engineer Guidance
For Software and Mechatronics Engineering Students — *Final Project – Student Questioner*

Student Name: _____ *Major:* _____ *Study Level:* _____

In this paper some questions will be asked about the project of ***The Engineer Guidance Website***, which is a website that suppose to help students of *Software Engineering* and *Mechatronics Engineering* in their study in the university. The main goal of answering these questions is to collect some requirements for the mentioned application to develop it in the best way that can benefits users of the system (you for example).

First: General Questions

Q1: As a Software or a Mechatronics Engineer, how you can benefit from such a system that describes the main content of your major?

Q2: Suggest us some features that can benefits the user of such a website.

Q3: The system enables the user to manage its own plan, in this context:

A) What type of plans you suggest to include in the website, an overall plan for the overall study of your major, or a group of short-term plans for your weekly or monthly study?

B) Describe your imagination about your plan, the ways of how can you make it.

C) Suggest some features you think its important to your plan.

Second: Short Questions (Yes or No). you can write simple notes if you want.

Q1: Could you benefit from YouTube videos that explains major materials lessons?

Q2: Could you benefit from researches and documents for a specific material?

Q3: Could you benefit from questions and answers about specific subject?

Q4: Could you benefit from short explanations about specific subject?

Q5: Could you benefit from contacting and asking teachers through the website?

If you have any ideas or notes:

Figure 23: Students' Questions Document

2. Participating students:

   Before viewing the opinions and answers of the students who answering the questionnaire here are the basic information about those students. Note that all those students studying at (Al Azhar University – Gaza), and all of them are at the fifth level which is the last two years and that because they are approximately finishing their studies so that they have the best understanding of the problem trying to solve and could have the best suggestions for the solution.

Table 18: Participating Students in Answering Students' Questions

Name in This Doc	Origin Name	Major	Study Level
Std1	Ahmed Abo Ghali	SW Eng.	Fifth Level (5th)
Std2	Mohammad Zakout	SW Eng.	Fifth Level (5th)
Std3	Mohammad Abo Qaoud	SW Eng.	Fifth Level (5th)
Std4	Emad Eddin Raafat Ali Abu Radwan	SW Eng.	Forth Level (4th)
Std5	Ahmed Atif Al-Bashiti	SW Eng.	Fifth Level (5th)

3. Students' answers:

   Finally, here is the conclusion of the answers and the opinions of the participating students in the requirements questioner that mentioned in this document:

Table 19: Answers on Students Questions - Std1, Std2, Std3

First: Questions Group 1			
Question	**Std1**	**Std2**	**Std3**
Q1	Arranging the study plan in proportion to my field of work at the present time	The system that describes the main content of my major can helps me in different way such as 1.Taking the references, paper and books for each semester course. 2.Saving my time and effort because I don't have to ask	It will save my time by providing the resources I will need in one place

		the former student about my major courses. 3.Prepare my self to take the semester by knowing what I'm going to study in the upcoming semester also to take courses or watching helping materials before taking the semester to get high marks and benefit of semester.	
Q2	1- Consultants from among those in charge or developing appropriate plans for the field 2- Put books and references on the requirement page	1.Give the academic plan for each major. 2.Make section for each semester courses with available resources. 3.Suggest online courses in each major. 4.Make section for each student to track his achieved courses and display what is remain for graduation.	Allow students to share and upload files to the platform Adding a feature that allows the student to discuss and exchange experiences
Q3.A	Plans to study the field in a divided manner for management and programming, where there are engineering and administrative subjects	Short term plan is better , it is easier to follow and commit	comprehensive plan
Q3.B	Plans for different programming fields (web, mobile, etc.) by placing materials related to the field in one chapter to be interconnected.	My plan for each semester come with stages Stage 1 : plan for each semester by researching for each course and what I'm going to take and what is the benefit of it . Stage 2 : plan for study each semester courses by distributing my week among those courses by condition that at most one course by a day. Stage 3 : plan for midterm , I divide the days between the day of announce of midterm schedule and the day of midterm between courses , also , in these days I divide every course into some tasks	Plan A, it will be comprehensive and in-depth to study the course completely within a period of time to fulfill its right Plan B, which is for crises so that the maximum amount of the course is completed in a record time

		and distribute it in consecutive days . Stage 4 : plan for final , same as midterm plan.	
Q3.C	The paid programs are distinguished as we obtain prepaid licenses in order to keep pace with development and not work with outdated programs	1-Know the content of each course so I can divide it into task. 2-Determine the suitable time for each course to study.	The feature to communicate with the student will help greatly
First: Questions Group 2			
Question	**Std1**	**Std2**	**Std3**
Q1	Yes	Yes	Yes
Q2	Yes (but in less percentage)	Yes	Yes
Q3	Yes	Yes	Yes
Q4	No	Yes	Yes
Q5	Yes	No	Yes

Table 20: Answers on Students Questions - Std4, Std5

First: Questions Group 1		
Question	**Std4**	**Std5**
Q1	You can benefit from focused courses that develop skills and exercises that test your abilities	As a Software Engineer, I see that the website will be more powerful and more beneficial if was contain annotated practical applications for use in projects in studied subjects supported by video explanations
Q2	Develop methodological paths and organize them into levels for each test level	i have no idea
Q3.A	A comprehensive plan that explains your progress and progress in the field, and short plans that add new skills to you, tests that ensure your success in them, and you will continue to learn more	- Weekly revisions plans - Monthly revisions plans - Plan revisions for midterm exam. - Plan revisions for final exam.

Q3.B	Program and methodological plans divided into levels that qualify the student for the work environment and provide him with all the necessary skills and experience	For example, revision plans for the final exam The system will collect the study material required for the exam, collect the summative videos of the study material and the existing written summaries, collect the questions asked on the platform during the course period, collect the previous final exams for the study material and put them in a separate box for the final exam
Q3.C	Pass the tests to pass the levels	Conduct weekly short tests to track the student's academic status
Second: Questions Group 2		
Question	**Std1**	**Std2**
Q1	Yes	Yes
Q2	Yes	Yes, but in less percentage than other
Q3	Yes	Yes
Q4	Yes	Yes
Q5	Yes	Yes

4. Functional requirements from this section:

   In the conclusion, and after analyzing the answers on these questions, here are a number of functional requirements from this section:

   FRQ1 The application should enable students to create shore-term schedule.

   FRQ2 The application must provide a comprehensive, general plan for the overall department.

   FRQ3 The application must enable students to mark their progress in the general plan.

   FRQ4 Students should be able to update their short-term schedules.

   FRQ5 The application should provide users with videos links for a specific subject.

FRQ6 The application should enable users to ask questions about specific subject throw the application.

FRQ7 The application should enable students to answer on others' questions about specific subject.

FRQ8 The application should provide links of other websites that describing specific subject.

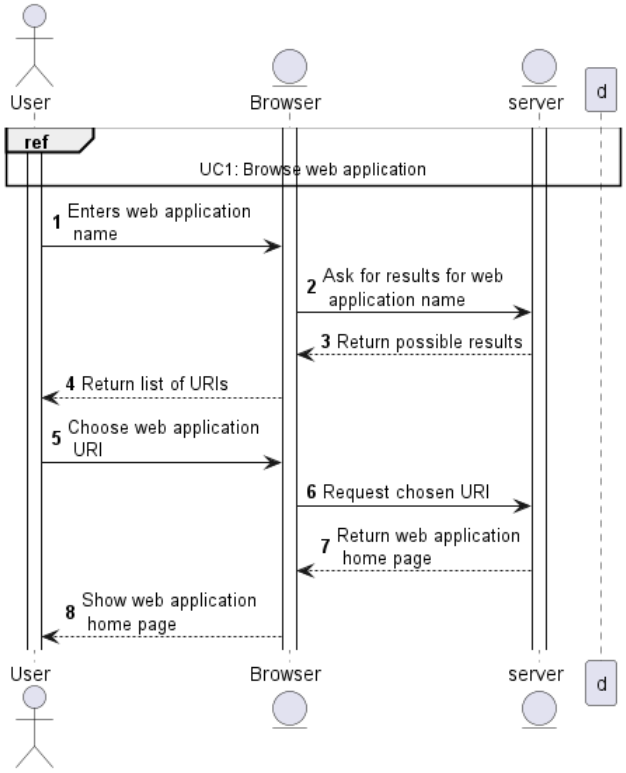
FRQ9 The application should provide supporting documents for specific subject.

3. Appendix C: Sequence Diagrams

Appendix C: Rest of Sequence Diagrams

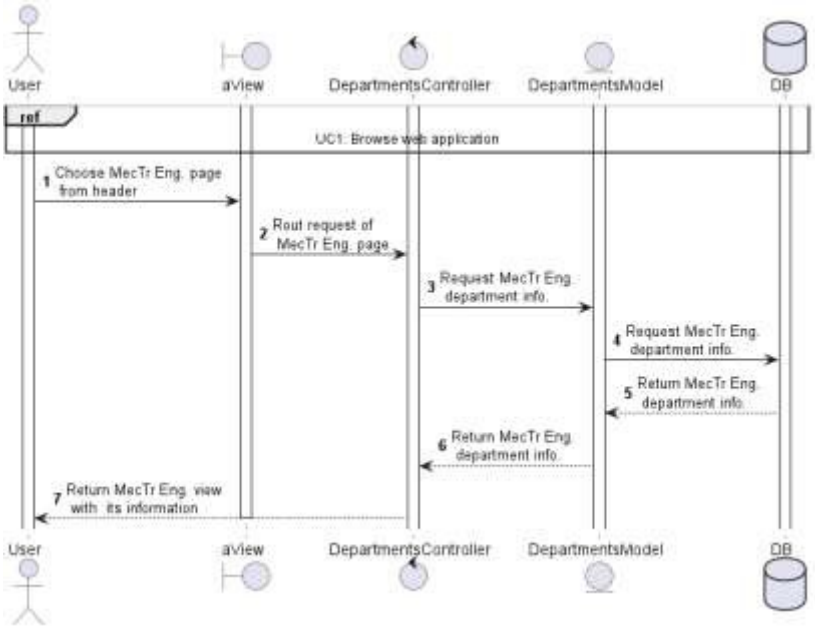
This section mentioned the rest of sequence diagrams that are not mentioned in *Chapter Six* in this document. The next figures illustrate those sequence diagrams.

1) **UC1**)user browsing the application): through a **web browser**, the user can **search** for the application and **reach** it.



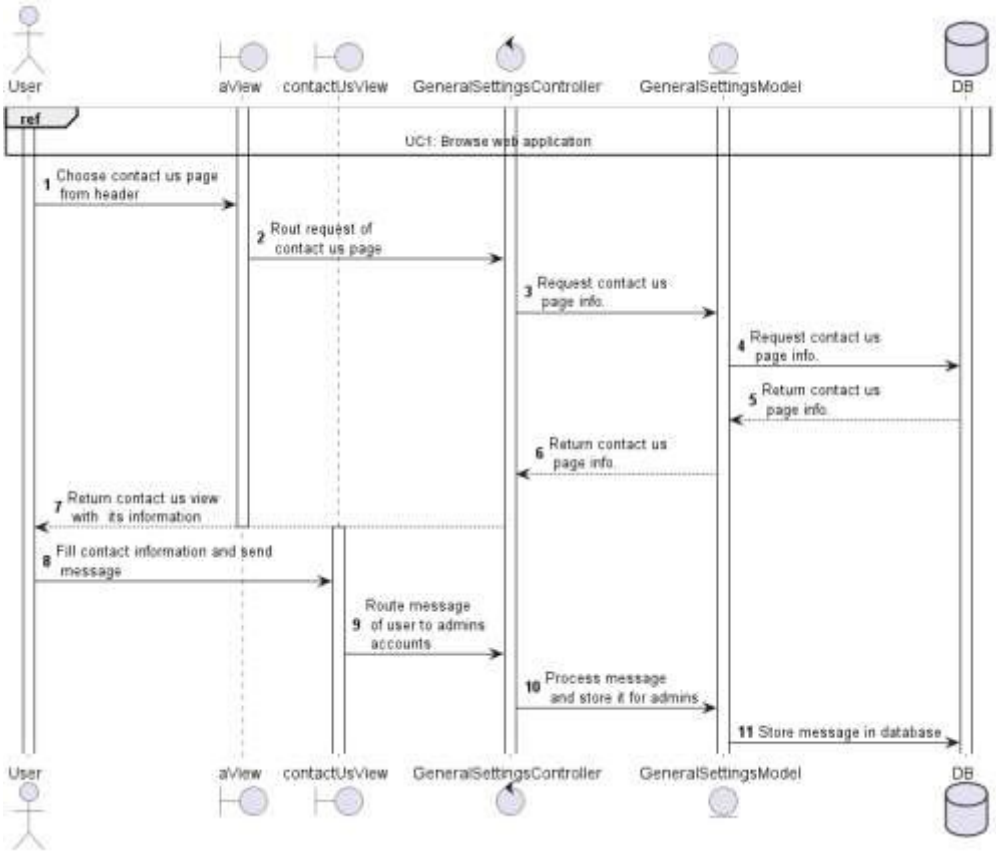
SQD 14 for UC 1

2) **UC3**(user browse MecTr department): a user **chooses** MecTr Eng. department from the header of the application, the system **returns** the department page, and then the user **browses** inside the department.



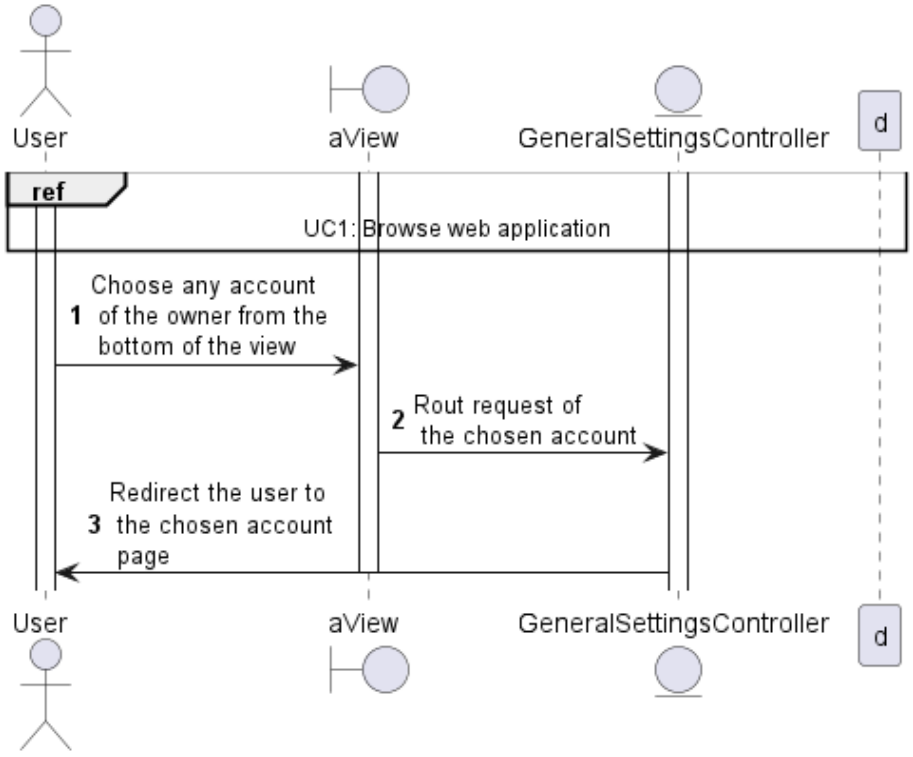
SQD 15 for UC 3

3) **UC4**(user contact application admins): a user **chooses** the contact us page from the application header, then **fills** the contact us form that the system **displayed** and **confirm**.



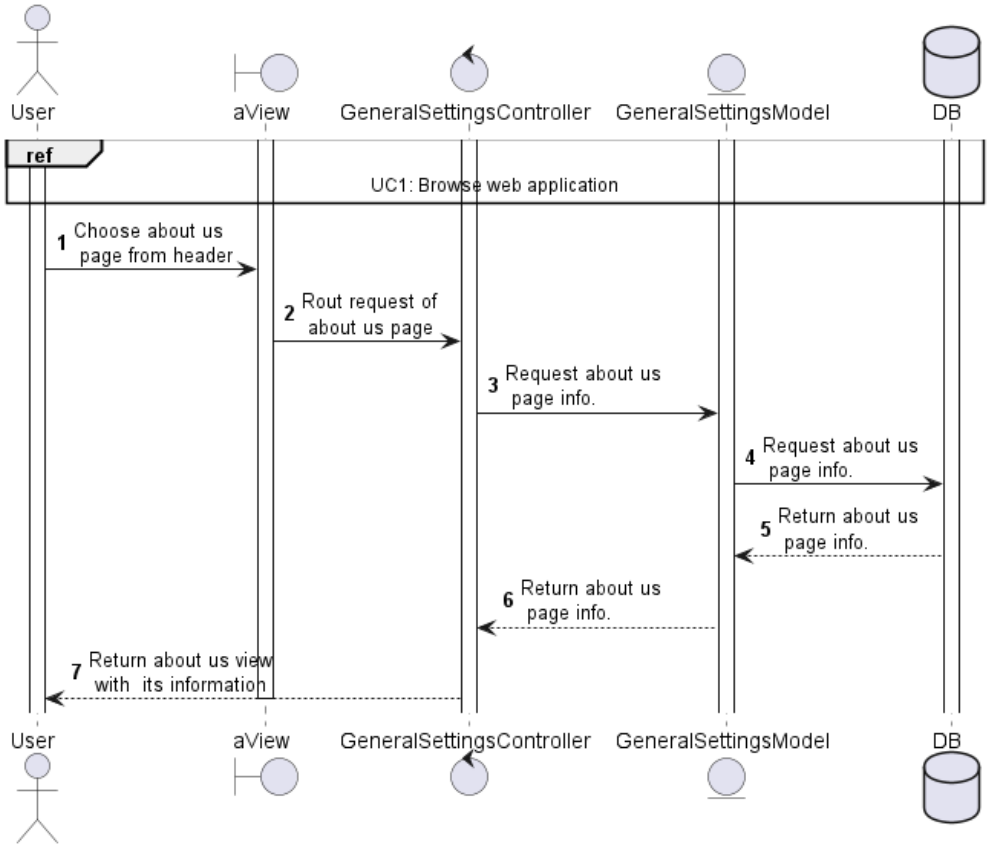
SQD 16 for UC 4

4) **UC5**(contact application owners): user **chooses** any contact account of the application's owner from the button of application's pages.



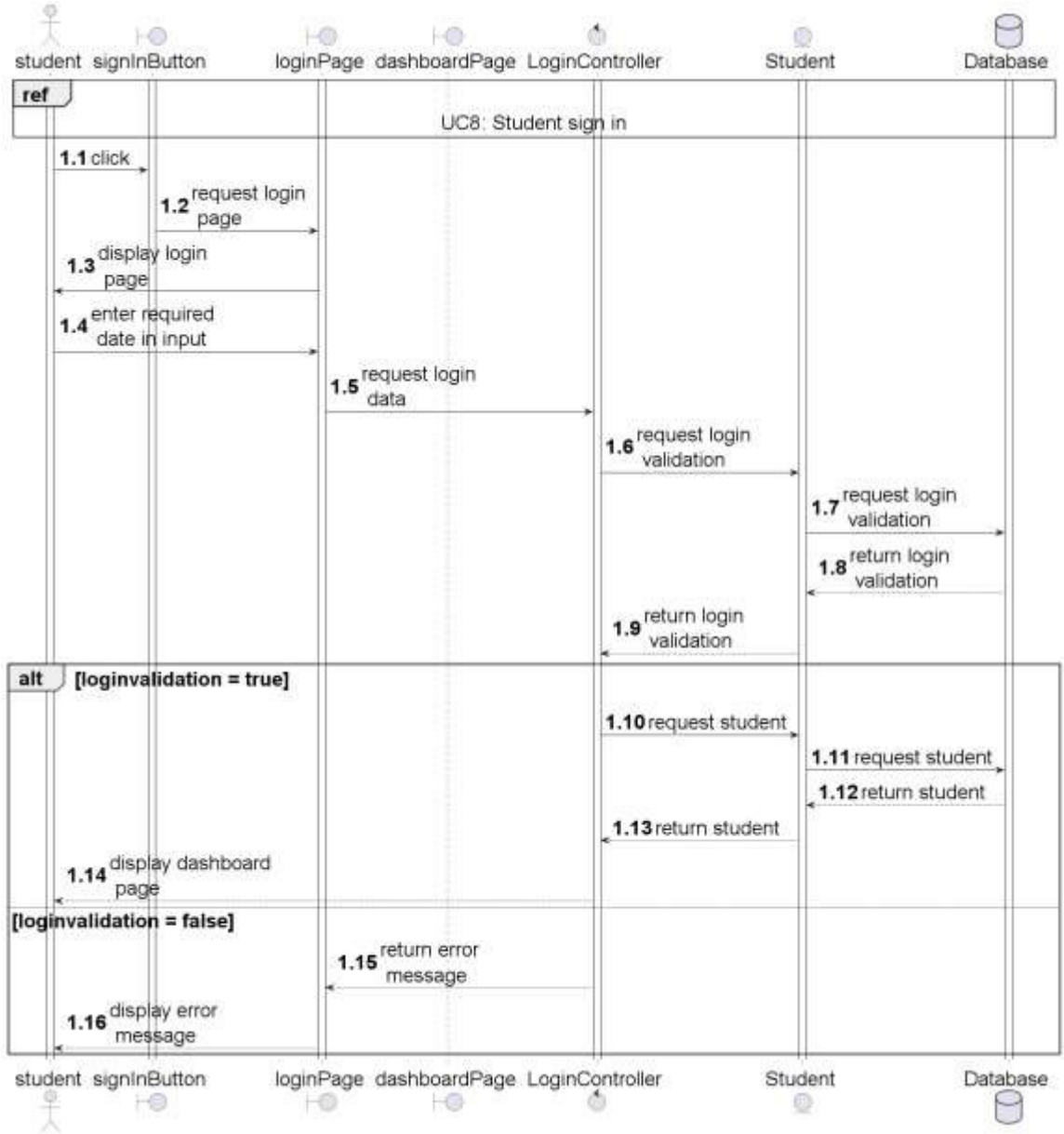
SQD 17 for UC 5

5) UC6(knowing about application owners): user **enters** about us page.



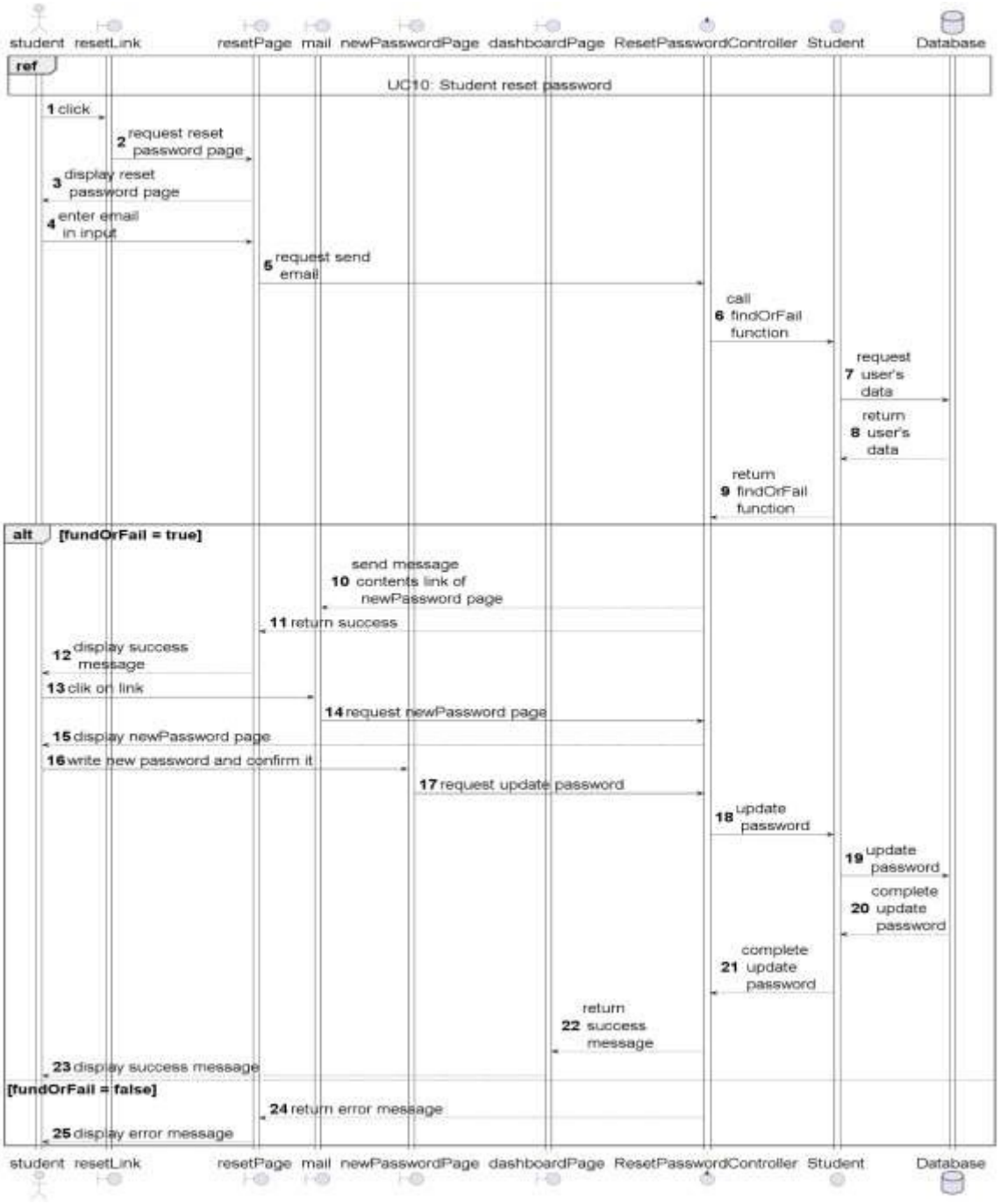
SQD 18 for UC 6

6) **UC8**(student sign in): a registered student **enters** his account through **the students' sign in page**.



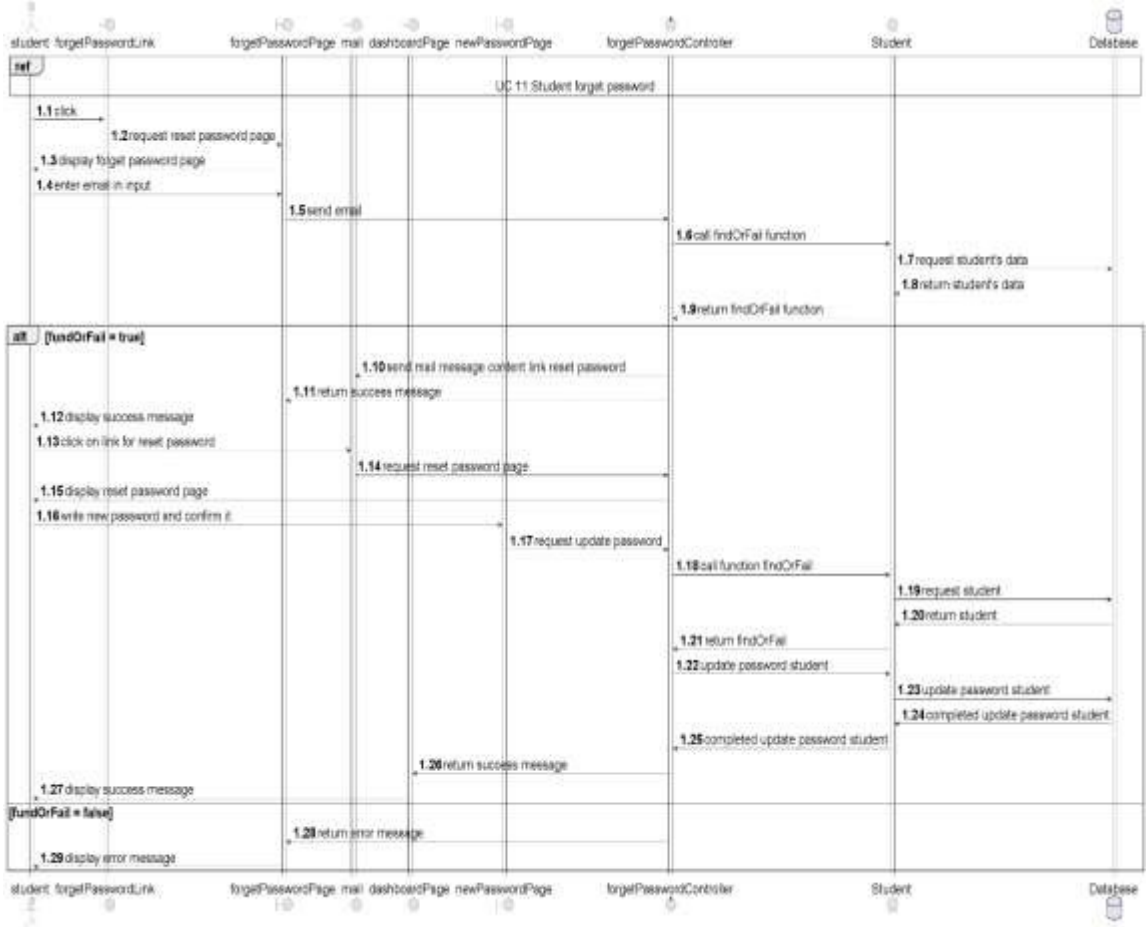
SQD 19 for UC 8

7) **UC10**(student reset password): a **logged in student chooses** the reset password link, the system **shows** the reset password form, the student **fills** required data, and if the data **is true** then, the reset password process **is done** successfully.



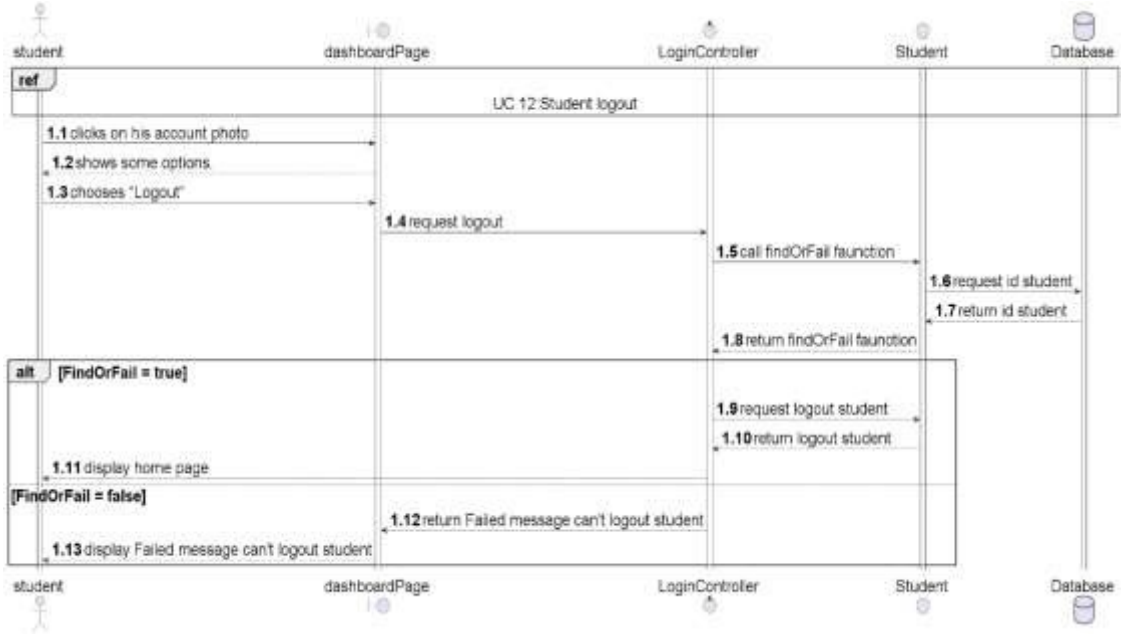
SQD 20 for UC 10

8) **UC11**(student forget password): a **logged in student chooses** forgot password link, the system **shows** forgot password form, and the student **fills** the required data if the data **is true** then, the system **shows** the reset password form, and the student **fills** the required data if data **true** then the reset password process **done** successfully.



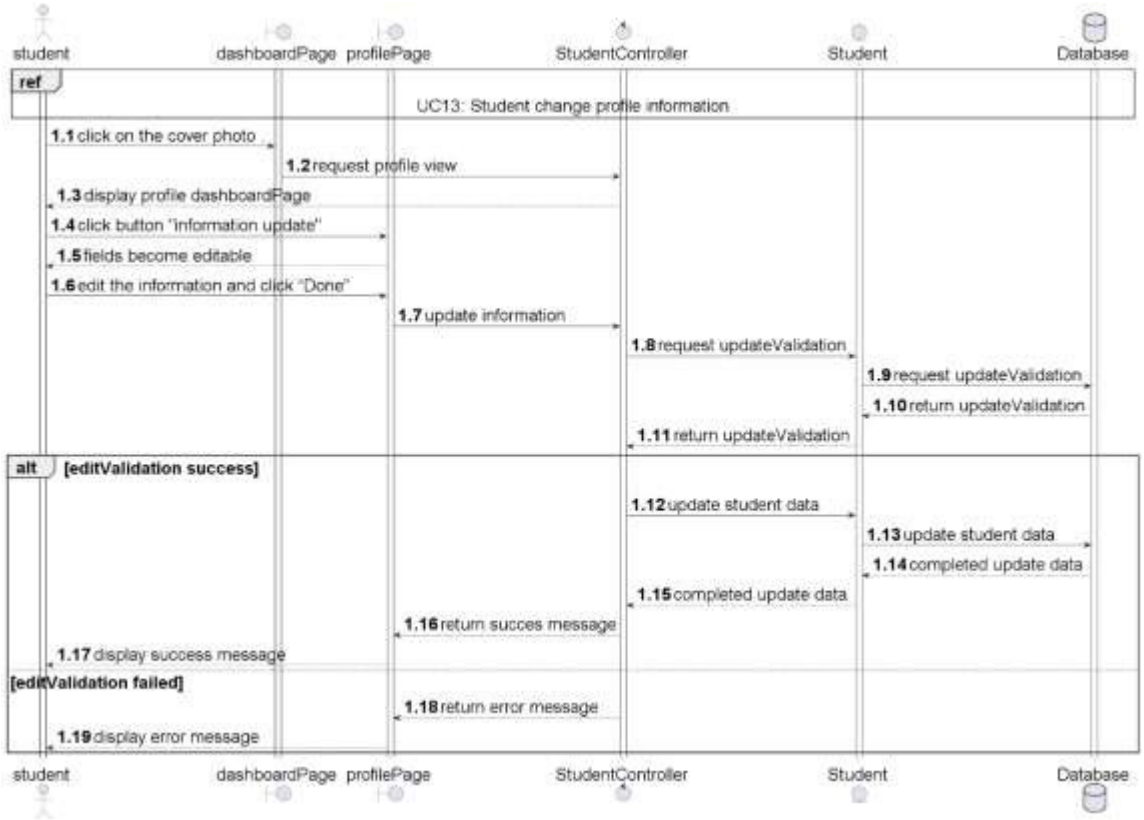
SQD 21 for UC 11

9) UC12(student logout): a **logged in student chooses logout** from the page header, then the student's session **ends**.



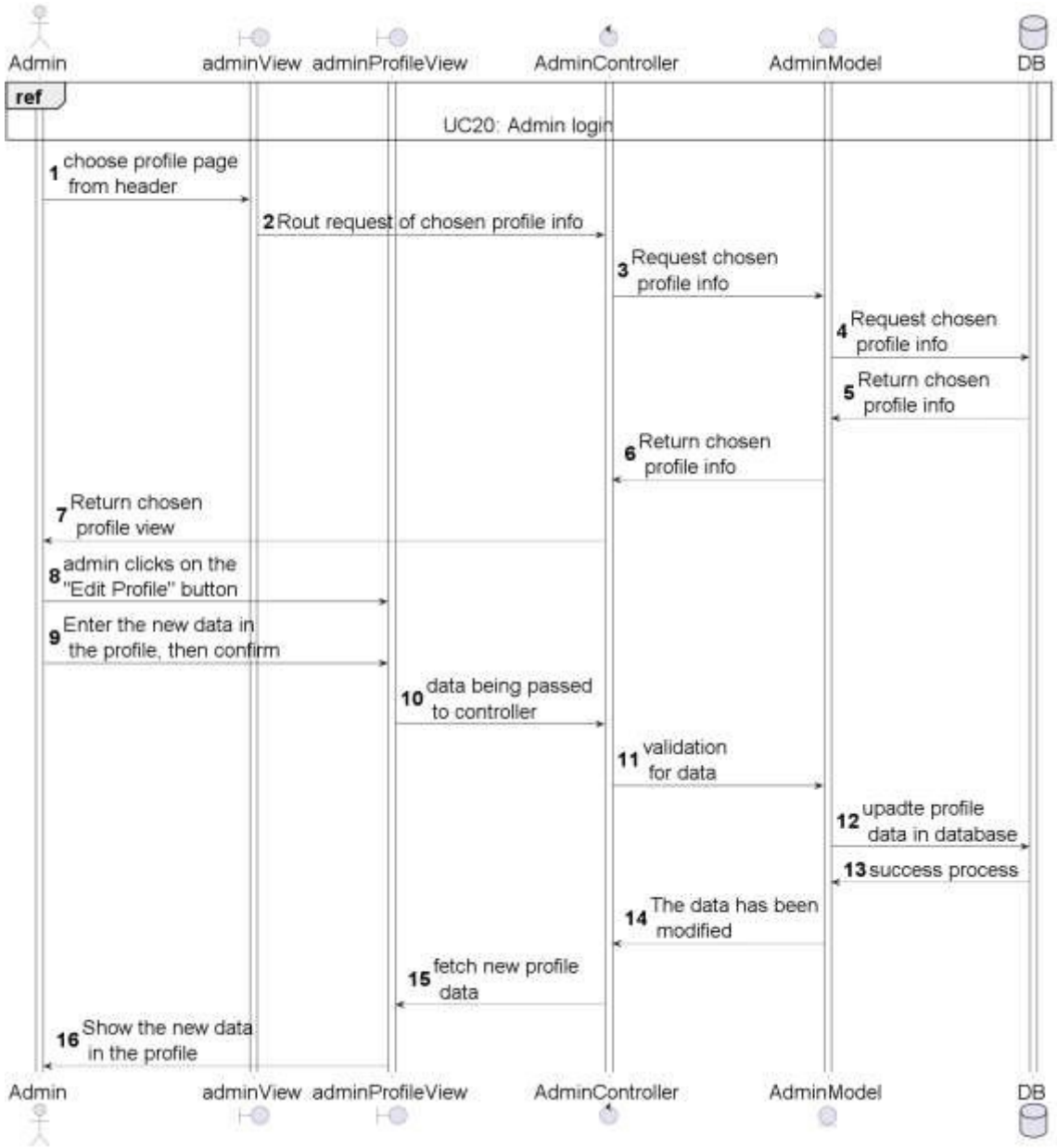
SQD 22 for UC 12

10) UC13(student change profile information):a **logged in student enters** his profile page by **clicking** on the profile image, he **edits** what he wants, and if the edits **are acceptable**, the system will **save** changes.



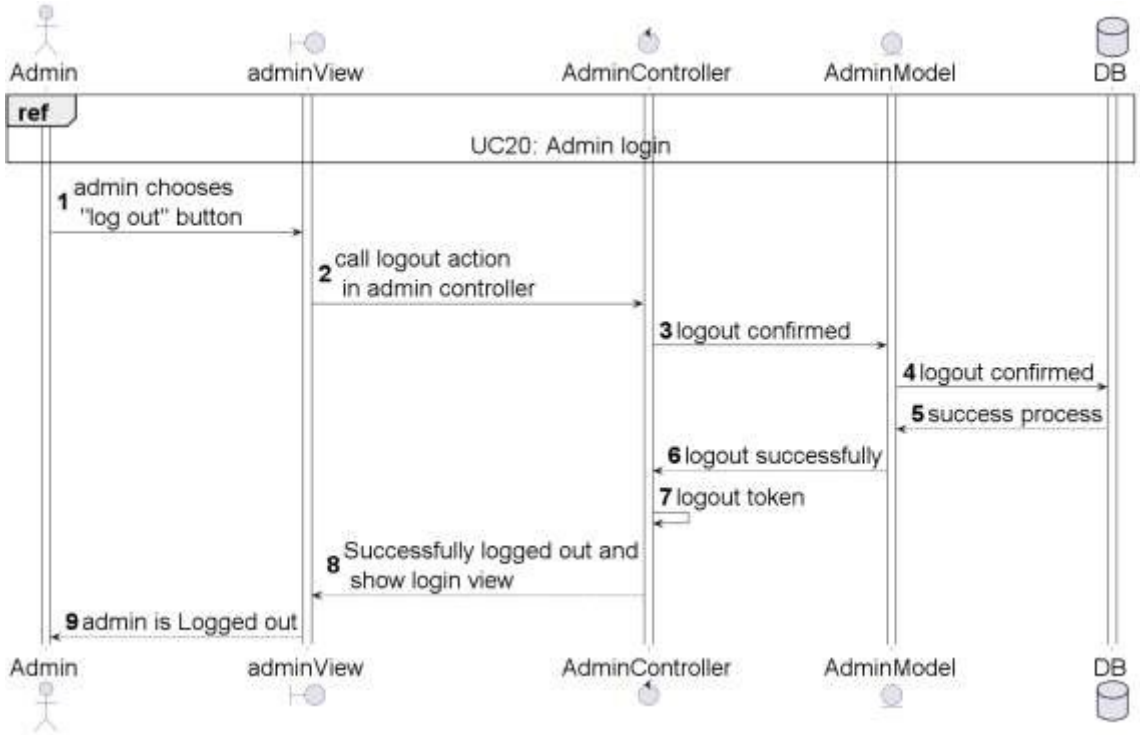
SQD 23 for UC 13

11) UC21(admin change his profile): a **logged in admin enters** his profile by choosing it from the header, he **edits** what he wants, and if the edits **are acceptable**, the system will **save** changes.



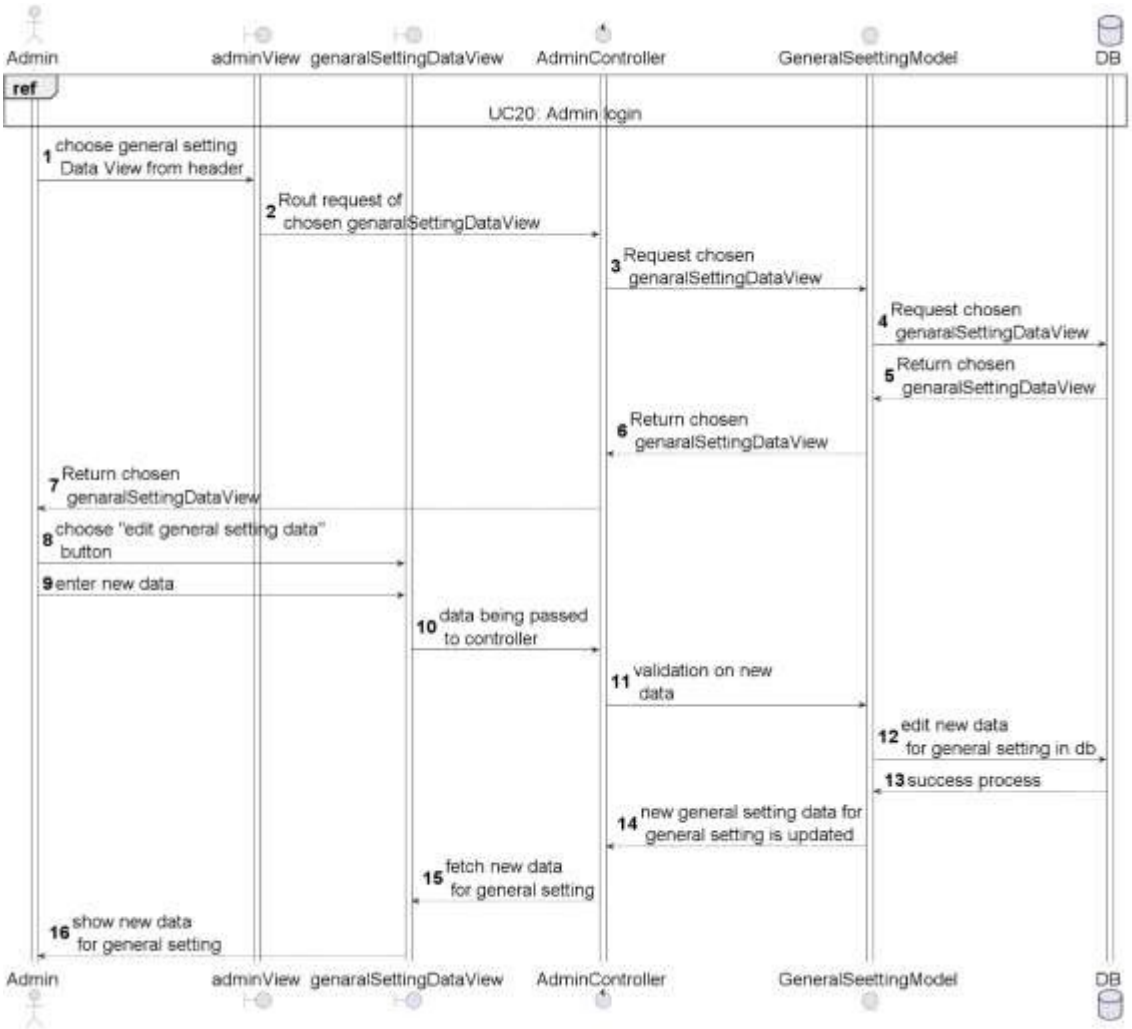
SQD 24 for UC 21

12) UC22(admin logout): a **logged in admin chooses logout** from the page header, and then the admin's session **ends**.



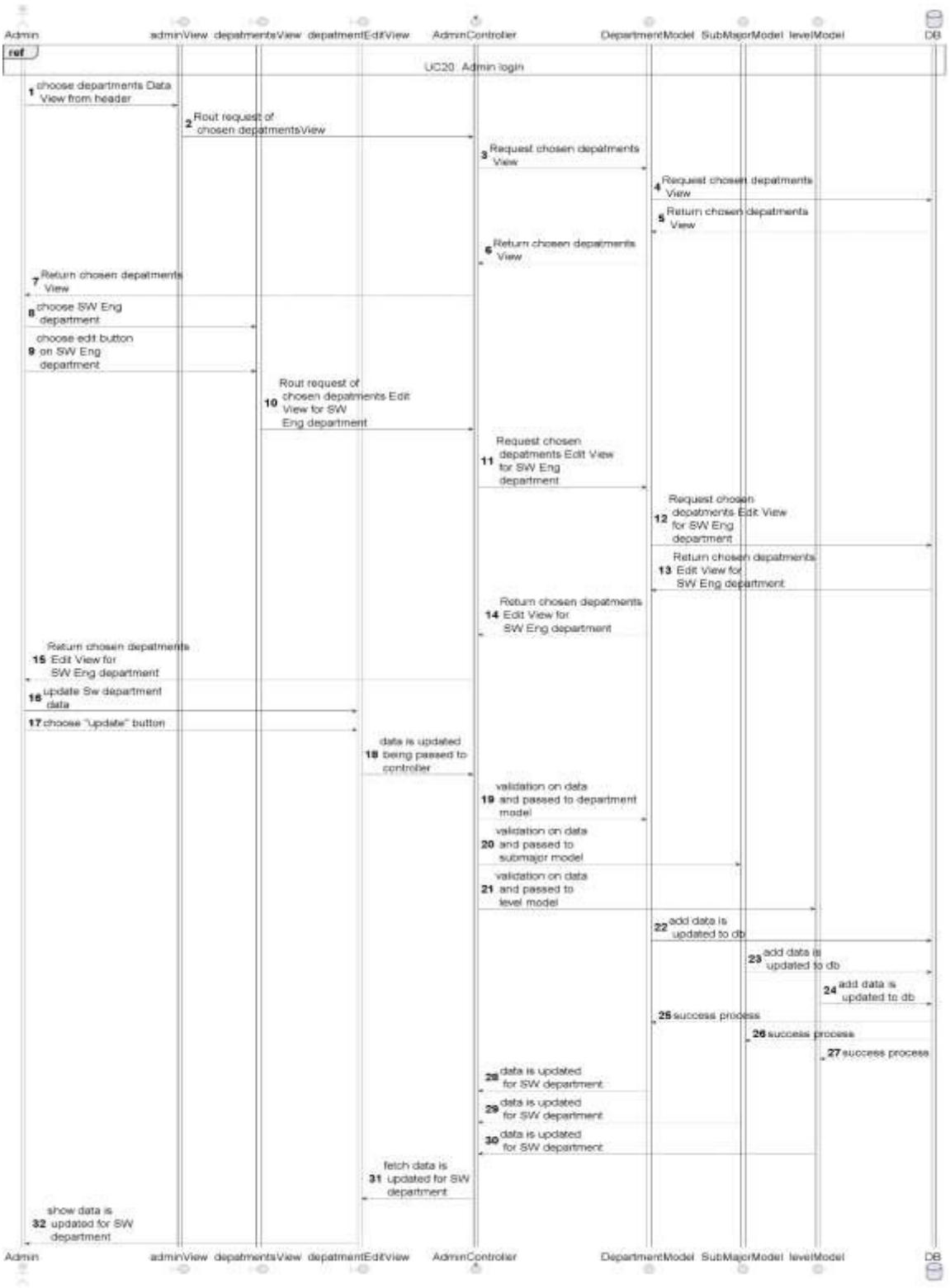
SQD 25 for UC 22

13) **UC25**(change application content):a **logged in admin** chooses **general settings data**, the system **shows** the general settings data page, the admin chooses to **edit general settings** for a specific setting and **enters** new data, the system **asks to confirm**, the admin **confirms** editing, and then the system **saves** the new changes.



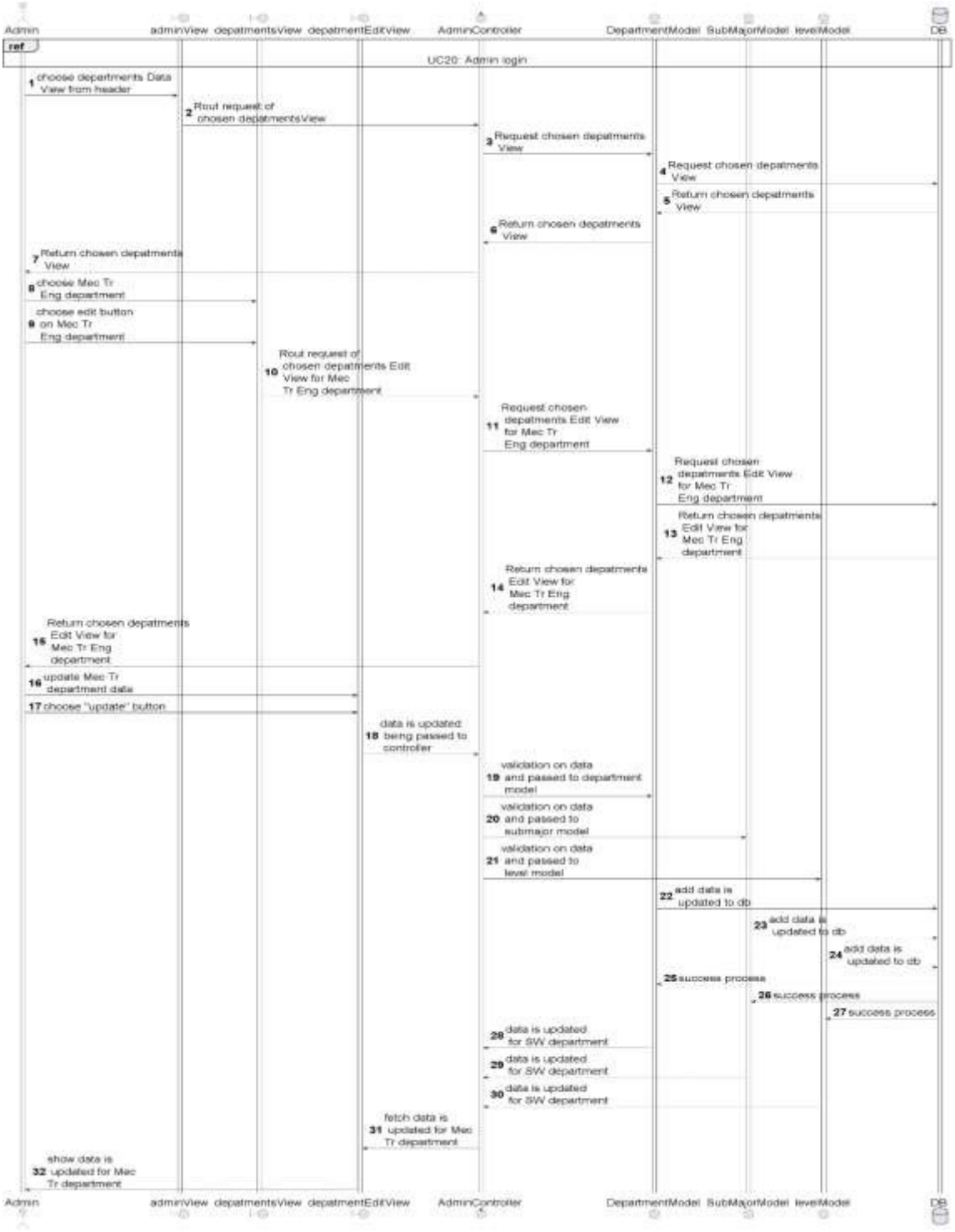
SQD 26 for UC 25

14) **UC27**(change SW Eng. department content): a **logged in admin** chooses **the departments data**, the system **shows** the departments data page, the admin chooses **SW Eng. Department** and chooses **edit** choice, he enters new data, the system **asks to confirm**, the admin **confirms** editing, and then the system **saves** the new changes.



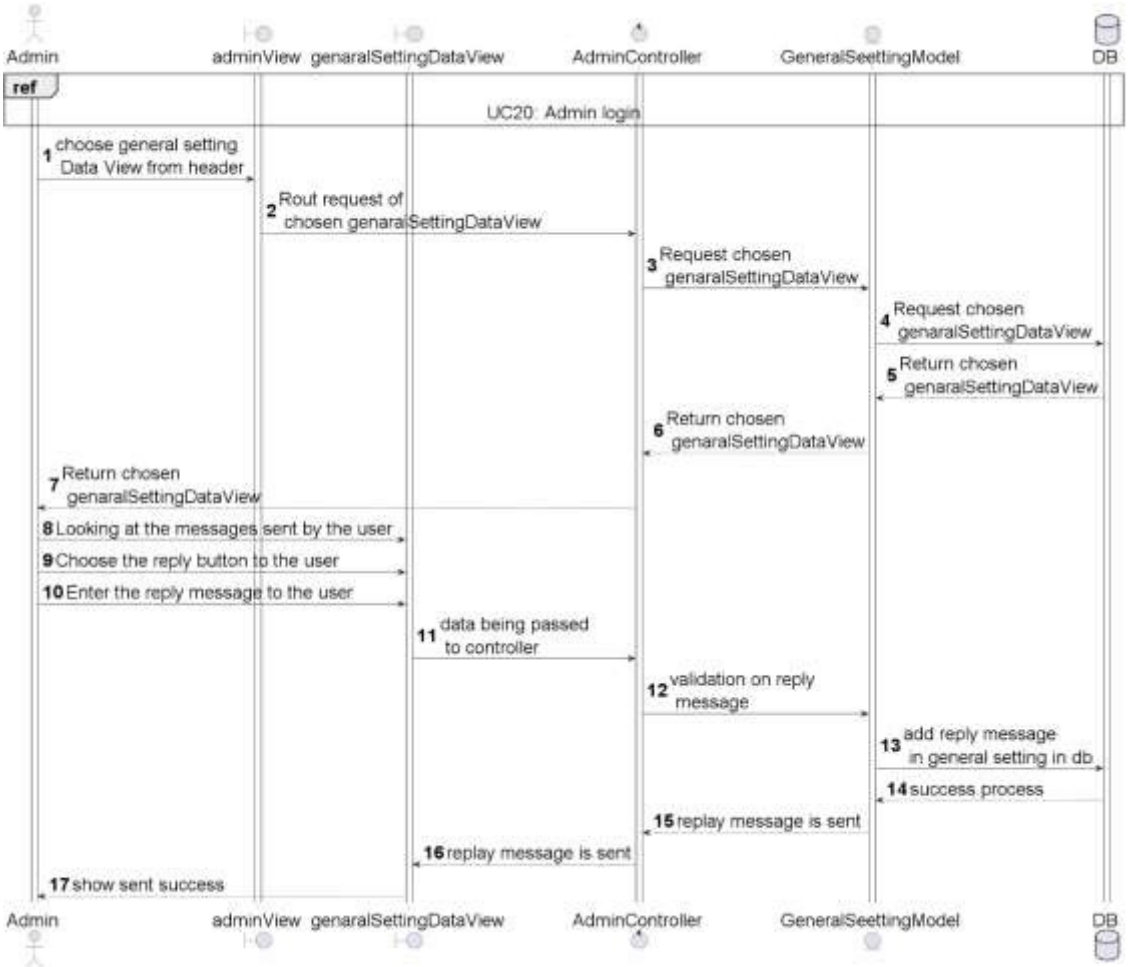
SQD 27 for UC27

15) UC28(change MecTr Eng. department content): a **logged in admin** chooses **the departments data**, the system **shows** the departments data page, the admin chooses **MecTr Eng. Department** and chooses **edit** choice, he enters new data, the system **asks to confirm**, the admin **confirms** editing, and then the system **saves** the new changes.



SQD 28 for UC 28

16) **UC30**(contact users): a **logged in admin** chooses **general settings data** and from it, he **chooses** users' messages, he chooses to **replay** a specific message, then he **enters** the replay and **sends** it.



SQD 29 for UC 30

End of The Final Report