



Democratic and Popular Republic of Algeria

Ministry of Higher Education and Scientific research



University of KASDI Merbah - Ouargla

Faculty of new information and communication technologies
Department of Computer Science and Information Technology

Academic Master Thesis

THEME

**Design and development of an
application for attendance monitoring
and management in the Department
of Computer Science and Information
Technology**

Presented by: Said Mansouri

Jury members

Dr. Benkhrourou CH President

Dr. Mahjoub M.B Examiner

Dr. Bachir Said Supervisor

Academic Year 2021/2022

Acknowledgement

- First of all, we thank ALLAH the Almighty for giving us the strength and courage to do this modest work.
- All thanks and appreciation to Dr. Bachir Said who gave us this project and his confidence in us, and all his advices and corrections and audit in the memoire.
- We would also to thank the Dr. Bachir Mahdjoub for all the efforts he made for us.
- I would like to thank my parents for all the support they have given me during my studies.

Abstract

The aim of this thesis is to facilitate and simplify the process of managing attendance and participation during the class, Therefore, through this project, we are trying to design and produce a program for attendance management and participation in the Department of Computers and Information Technology. To achieve this, we have used different programming languages, including JavaScript, html, css in different development environments. This program should facilitate the process of managing attendance and participation during the class, by recording the attendance and participation of students in a shorter time in an easier way.

Key words

website, Mobile app, attendance management, attendance management, UML modeling.

Résumé

Le but de cette thèse est de faciliter et de simplifier le processus de gestion de l'assiduité et de la participation pendant le cours. Par conséquent, à travers ce projet, nous essayons de concevoir et de produire un programme de gestion de l'assiduité et de la participation au Département d'informatique et de technologie de l'information. Pour y parvenir, nous avons utilisé différents langages de programmation, notamment JavaScript, html, css dans différents environnements de développement. Ce programme devrait faciliter le processus de gestion de l'assiduité et de la participation pendant le cours, en enregistrant l'assiduité et la participation des étudiants en un temps plus court et de manière plus simple.

Mots clés

site Internet, application mobile, Gestion de présence, gestion de la participation, Modélisation UML.

ملخص

الهدف من عمل هذه الاطروحة هو تسهيل وتبسيط عملية ادارة الحضور والمشاركة خلال الفصل، لذلك نحاول من خلال هذا المشروع تصميم وإنتاج برنامج لادارة الحضور والمشاركة في قسم الاعلام الالي وتكنولوجية المعلومات، ولتحقيق ذلك فقد استخدمنا لغات البرمجة المختلفة منها JavaScript ، html ، css في بيئات تطوير مختلفة. يجب ان يسهل هذا البرنامج عملية ادارة الحضور والمشاركة خلال الفصل وذلك من خلال تسجيل حضور ومشاركة الطلبة في وقت اقصر وبطريقة ايسر.

الكلمات المفتاحية

موقع ويب، تطبيق الهاتف، ادارة الحضور، ادارة المشاركة، نمذجة UML .

Contents

General Introduction	9
Chapter 1 Presentation General	10
1 Introduction	11
2 The UML modeling language	11
2.1 Definition:	11
2.2 History of UML	11
2.3 Types of UML diagrams	12
2.3.1 Structural Diagrams:	12
2.3.2 Behavioral Diagrams	12
3 Student Attendance Management System	14
3.1 ClassDojo	14
3.2 GoGuardian Teacher	14
3.3 SAP Litmos	15
3.4 Schoology	15
3.5 Blackboard Learn	15
4 Conclusion	15
Chapter 2 Requirements analysis and Design	16
1 Introduction	17
2 Students Evaluation in Current System	17
2.1 Specialization and level	17
2.2 Continuous Assessment	21
2.3 Presence and Participation	21
3 Suggest an alternative solution	22
4 capture of requirements	22
4.1 Non-functional requirements	22
4.2 functional requirements	22

4.2.1	Definition of actors	22
4.2.2	Functional requirements for each actor	23
5	The use case diagram	23
6	detailed explanation of use cases	23
6.1	The use case « Authentication »	23
6.2	The use case « Disconnect »	25
6.3	The use case « Import data »	26
6.4	The use case « Export data »	27
6.5	The use case «Assignment management»	28
6.6	The use case « Create session »	29
6.7	The use case « Participation acceptance »	30
6.8	The use case « Participation registration »	31
6.9	The use case «Manage evaluation»	32
6.10	The use case « Attendance registration»	33
6.11	The use case « Request to participation»	34
6.12	The use case « Show absences »	35
7	Conception phase	36
7.1	Class Diagram	37
7.2	Data Dictionary	38
7.2.1	The Attributes	38
7.2.2	The Operations	40
8	Conclusion	41
Chapter 3 Implementation and Realization		42
1	Introduction	43
2	General Architecture of System	43
3	The development environment and Tools used	44
3.1	Node JS	44
3.2	Visual Studio code	44
3.3	Postman	45

3.4	MongoDBCompass	45
3.5	React JS and React Native	46
3.6	Nest JS	46
3.7	Android Studio	47
3.8	Java SE Development Kit «JDK»	47
4	Programming languages used	48
4.1	language JavaScript	48
4.2	language HTML	48
4.3	language CSS	49
5	The functionalities of the realized application	49
5.1	Description of website interfaces	51
5.1.1	The functions of the Admin	51
5.1.2	The functions of the Teacher	52
5.2	Description of Application Mobile interfaces	55
5.2.1	The functions of the Student	55
6	Test phase	58
7	Conclusion	59
	General Conclusion	60

List of Figures

1.1	Example of Class Diagram.	13
1.2	Example of Use Case Diagram.	13
1.3	Example of Sequence Diagram.	14
2.1	Table Showing Semester 1.	18
2.2	Table Showing Semester 2.	18
2.3	Table Showing Semester 3.	18
2.4	Table Showing Semester 4.	19
2.5	Table Showing Semester 5.	19
2.6	Table Showing Semester 6.	19
2.7	Table Showing Semester 1 (fundamental).	20
2.8	Table Showing Semester 2 (fundamental).	20
2.9	Table Showing Semester 3 (fundamental).	21
2.10	use case diagram	24
2.11	sequence diagram for use case Authentication	25
2.12	sequence diagram for use case Disconnect	26
2.13	sequence diagram for use case Import data	27
2.14	sequence diagram for use case Export data	28
2.15	sequence diagram for use case Assignment management	29
2.16	sequence diagram for use case Create session	30
2.17	sequence diagram for use case Participation acceptance	31
2.18	sequence diagram for use case Participation registration	32
2.19	sequence diagram for use case Manage evaluation	33
2.20	sequence diagram for use case Attendance registration	34
2.21	sequence diagram for use case Request to participation	35
2.22	sequence diagram for use case Show absences	36
2.23	class diagram	37
3.1	general architecture of system	43
3.2	logo node js.	44
3.3	logo visual studio code.	45
3.4	logo Postman.	45
3.5	logo MongoDBCompass.	46
3.6	logo React	46

3.7	logo nest js.	47
3.8	logo Android Studio	47
3.9	JAVA Development Kit (JDK)	48
3.10	logo JavaScript.	48
3.11	logo HTML.	49
3.12	logo CSS.	49
3.13	Hierarchy of website.	50
3.14	Hierarchy of the application mobile.	50
3.15	login page.	51
3.16	session creation page.	52
3.17	QR Code Management page.	53
3.18	Participation Management page.	54
3.19	Attendance Management page.	54
3.20	interface of Login.	55
3.21	interface of Attendance Registration.	56
3.22	interface of Request to Participation.	57
5.23	interface of Show Absences.	57

List of Tables

1	List of actors with their functions	23
2	Data Dictionary: The Attributes.	39
3	Data Dictionary: The Operations.	41
4	Showing the purpose of the tests	58
5	Explains why some tests are not done	59

General Introduction

In recent years, technology has spread greatly and has become one of the manifestations of the modern era, and dispensing with it has become very difficult, as it includes every part of our life and in our day, as its impact has extended to many aspects of life, and among these aspects is the educational aspect, many lessons have become The lectures depend on technology, even the classrooms became virtual, that is, the educational process takes place remotely through the use of computers and phones, Recently, some departments of the Faculty of Computers and Information Technology at the University of Ouargla have witnessed some overcrowding, especially the first-year departments, the large number of students in one section makes it difficult for the teacher, as he works to record the attendance of students and this takes effort and time, the attendance of some students may not be recorded due to the mixing of students' names on it, Just as the process of registering attendance is done by calling, this may cause embarrassment for some teachers due to the difficulty of pronouncing some names correctly, and the participation process may cause chaos in the classroom, and recording students' participation It is not easy because it is difficult to remember the names of students, and for facilitate the process of registering attendance and participation, and to shorten the time and to avoid chaos during the class, we thought of developing and designing a program to manage attendance and participation during the class, this makes us avoid all these problems, through it, students' attendance is recorded in a shorter time in an easier way, and we can also manage the participation process and record student participation during the class.

The software goes through several stages during its development, including the stage of gathering requirements and the design stage, then the stage of writing the program and the stage of testing, and then the stage of delivering and installing the program, In our project, the most important of these stages have been addressed, and we have relied on some diagrams of the UML modeling language, the first chapter was devoted to presenting the UML modeling language and all its diagrams, and briefly touched upon each diagram separately, and in the second chapter, the stage of requirements analysis and design was addressed, through which the requirements of the program were collected and the use of some UML diagrams, and in the third chapter we devoted it to explain phase writing the program code and testing phase, where we touched on the languages, techniques, and tools used in the process of writing the code, and we provided a description of the main functions of the program.

Chapter 1

Presentation General

1 Introduction

The Unified Modeling Language (UML) is a way to visually show the behavior of a system or process or its structure. It helps to display potential errors in application structures, system behavior, and other processes. UML has a great role in the software development process. They are used to plan programming projects before starting them in practice, so in this introductory chapter, we will give an overview of UML and its diagrams, and we will use UML in the next chapter to model our program, so we have given its definition, history, and diagrams types, and A brief explanation of some of the best student attendance management systems has also been provided.

2 The UML modeling language

2.1 Definition:

”The Unified Modeling Language (UML) is a family of graphical notations, backed by single meta-model, that help in describing and designing software systems, particularly software systems built using the object-oriented (OO) style” [1]

Simply Unified Modeling Language is used to prototype applications and systems that depend on Object-Oriented Programming (oop), where these models define the structure and behavior of the system and are easy to understand and read.

2.2 History of UML

The development of UML began at the end of 1994 when the Booch and OMT (object modeling technology) methods were standardized by Grady Booch and Jim Rumbaugh and in 1995 they were joined by Ivar Jacobson In 1997 the first version of UML was introduced and adopted by OMG (object management group), and in the year 2000 the version UML 1.3 was introduced and this model was updated with the semantic, notations, meta-models of UML and in the year 2005, the version UML 2.0 was introduced which had some different features and aspects such as timing, object packages, Interaction, updated activity diagrams, and in 2017 the official version 2.5.1 UML was announced by OMG [2].

2.3 Types of UML diagrams

There are 14 types of UML diagrams that fall into two distinct groups structural diagrams and behavioral or interaction diagrams.

Note: The information in this section has been taken from references [3, 4, 5, 6].

2.3.1 Structural Diagrams:

The structural diagrams represent the static aspect of the system. i.e., how one object relates to another. It shows the things in the system – classes, objects, packages or modules, physical nodes, components, and interfaces, The structural diagrams are: Class Diagram, Object Diagram, Component Diagram, Composite Structure Diagram, Deployment Diagram, Package Diagram, Profile Diagram.

2.3.2 Behavioral Diagrams

The behavioral diagrams represent the dynamic aspect of the system. It shows how the system behaves and interacts with itself and other entities. They show how data moves through the system, how objects communicate with each other, how the passage of time affects the system, The behavioral diagrams are: Activity Diagram, Use Case Diagram, Interaction Overview Diagram, Timing Diagram, State Machine Diagram, Communication Diagram, Sequence Diagram.

In our project, 3 types of diagrams were used which are:

- **Class Diagram:** class diagrams showing the static structure of the system, including the attributes, operations, and classes, their properties and behaviors, and the relationships between each class. class is represented by a rectangle divided into three parts, the upper part contains the class name and is mandatory, and the lower two parts provide details about the class's attributes and class operations or behaviors.

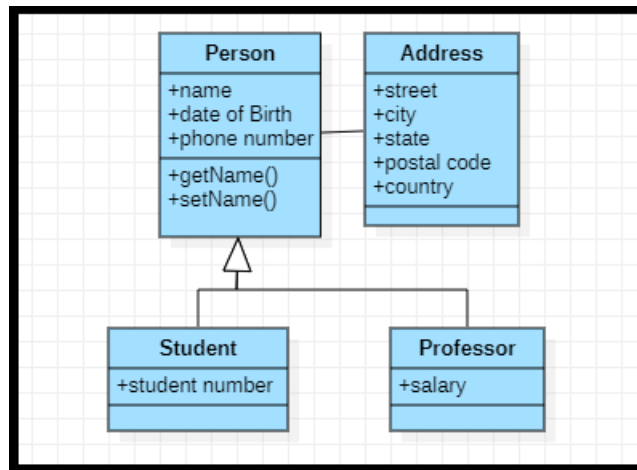


Figure 1.1: Example of Class Diagram.

- **Use Case Diagram:** Use case diagrams is a model that shows the interaction of users (actors) with the system, it gives a graphic overview of the actors and the different functions that actors need and how these different functions interact, that is, it summarizes the details of the system and the actors within this system.

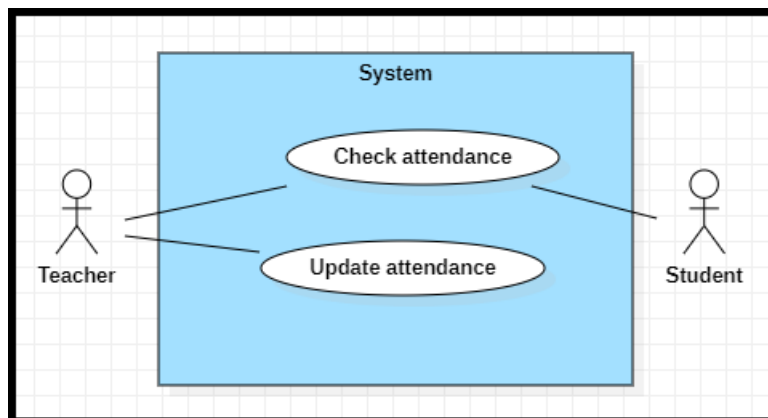


Figure 1.2: Example of Use Case Diagram.

- **Sequence Diagram:** The sequence diagram shows how objects interact with each other and the order in which these interactions occur, that is, it enables us to visually represent simple run-time scenarios. It describes the interactions between classes in terms of the exchange of messages over time, and the operations are represented in a vertical form and the interactions in the form of arrows.

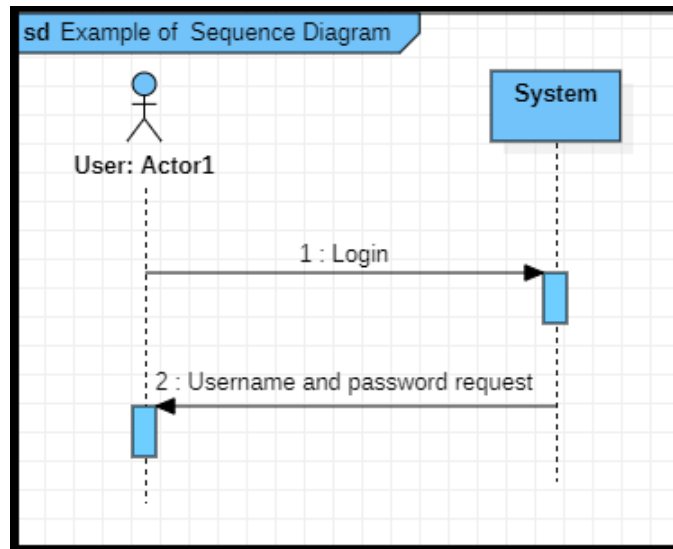


Figure 1.3: Example of Sequence Diagram.

3 Student Attendance Management System

Attendance management system is a program developed in order to manage the daily attendance of students in schools, colleges and institutes, it speeds up the performance of the task of attendance management and makes it easy, it relies on the use of barcode technology to register student attendance.

There are many student attendance management systems. In this section, we will only mention the top 5 student attendance management systems.

Note: The information in this section has been taken from reference [7].

3.1 ClassDojo

This program is very useful for the process of attendance management, it is the best program if you want a program that does more than register attendance, it sends real-time notes and communicates with parents.

3.2 GoGuardian Teacher

It is an online attendance tracking software that is useful for both teachers and students. It allows teachers to communicate with their students in many ways and with complete ease, Just as a teacher can use this program to take attendance, they can also use it to track online content and manage technology in the classroom.

3.3 SAP Litmos

This student attendance management system is used for taking attendance and can be used in classrooms and corporate training centers as well, it makes the learning process as smooth as possible, and it supports many languages to make the learning process more interactive.

3.4 Schoology

This student attendance system has many advanced features, unlike SAP Litmos software, this system is primarily designed for use in schools, it has many features that connect teachers with their students and provides different ways to customize learning. It is also useful for many institutions from schools and universities.

3.5 Blackboard Learn

Blackboard Learn is among the highly scalable software in terms of features. This student attendance management system is used in all types of classrooms from corporate classrooms to school environments.

4 Conclusion

In this chapter, we have introduced a definition of UML and touched on its history and all its types of diagrams, and A brief explanation of some of the best student attendance management systems has also been provided, In this project, we will use only the three most important main diagrams of UML, first the use case diagram in order to show the function of the system from the user's perspective, second, the sequence diagram In order shows the internal behavior of the system, and finally, the class diagram for shows the structure and basis of the system.

Chapter 2

Requirements analysis and Design

1 Introduction

In our project, we aim to design and develop an application for managing attendance and participation during the class, in order to facilitate the management of attendance and participation. During the development stage, the software goes through several different stages, so in our project, we touched on the stage requirements analysis, design stage, implementation stage, and testing stage. In this chapter, we will go through two stages, namely, requirements analysis and design. Therefore, we will describe the basic functional and non-functional requirements in a text way, summarize the functional requirements in the specifications, and then we will convert them into a set of UML diagrams, including: use case diagram, sequence diagram, class diagram Finally, we summarize this section with a conclusion at the end of the chapter.

2 Students Evaluation in Current System

2.1 Specialization and level

The LMD system is the teaching system approved in the Department of Computer Science and Information Technology, and this system consists of three stages, Bachelor, Master, and Doctorate where, a computer science student studying first-year majoring in Mathematics and Computer Science, then he moves to the second and third years specializing in Computer Science and then passes to the master phase, he studies the first year and the second year after choosing one of the four majors, fundamental, industrial, Network administration and security, or artificial intelligence, as for the doctorate, the certificate is obtained after submitting a research work lasting at least three years [8].

The following 6 tables show the semesters a student passes through during his/her studies at the stage of Licenses.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE Fondamentales									
UEF11(O/P)		4h30	4h30		6h	7	11		
UEF111 : Analyse 1	84h	3h00	3h00		3h	4	6	40%	60%
UEF112 : Algèbre 1	42h	1h30	1h30		3h	3	5	40%	60%
UEF12(O/P)		4h30	3h	3h	6h	7	11		
UEF121 : Algorithmique et structure de données 1	105h	3h00	1h30	3h	3h	4	6	40%	60%
UEF122 : Structure machine 1	42h	1h30	1h30		3h	3	5	40%	60%
UE Méthodologie									
UEM11(O/P)			3h		4h	2	4		
UEM111 : Terminologie Scientifique et expression écrite	21h		1h30		2h	1	2		100%
UEM112 : Langue Etrangère	21h		1h30		2h	1	2		100%
UE Découverte									
UED11(O/P) Choisir une Matière parmi :		1h30	1h30		2h	2	4		
- Physique 1									
- Electronique et composants des systèmes	42h	1h30	1h30		2h	2	4	40%	60%
Total Semestre 1	357h	10h30	12h	3h	18h	18	30		

Figure 2.1: Table Showing Semester 1.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF21(O/P)		4h30	3h		6h	6	10		
UEF211 : Analyse 2	63h	3h00	1h30		3h	4	6	40%	60%
UEF212 : Algèbre 2	42h	1h30	1h30		3h	2	4	40%	60%
UEF22(O/P)		3h	3h	1h30	6h	6	10		
UEF221 : Algorithmique et structure de données 2	63h	1h30	1h30	1h30	3h	4	6	40%	60%
UEF222 : Structure machine 2	42h	1h30	1h30		3h	2	4	40%	60%
UE méthodologie									
UEM21(O/P)		4h30	1h30	1h30	6h	4	7		
UEM211 : Introduction aux probabilités et statistique descriptive	42h	1h30	1h30		2h	2	3	40%	60%
UEM212 : Technologie de l'Information et de la Communication	21h	1h30			2h	1	2		100%
UEM213 : Outils de programmation pour les mathématiques	42h	1h30		1h30	2h	1	2	40%	60%
UE Transversale									
UET21(O/P)		1h30	1h30		2h	2	3		
UET211 : Physique 2 (électricité générale)	42h	1h30	1h30		2h	2	3	40%	60%
Total Semestre 2	357h	13h30	9h	4h30	20h	18	30		

Figure 2.2: Table Showing Semester 2.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1(O/P)		4h30	3h	3h00	6h00	6	11		
Architecture des ordinateurs	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
Algorithmique et structure de données 3	84h	3h00	1h30	1h30	3h00	3	6	40%	60%
UEF2(O/P)		3h00	3h00	1h30	6h00	5	9		
Système d'information	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
Théorie des graphes	42h	1h30	1h30		3h00	2	4	40%	60%
UE méthodologie									
UEM (O/P)		1h30		1h30	03h00	4	8		
Méthodes Numériques	42h	1h30		1h30	3h00	2	4	40%	60%
Logique Mathématique	42h	1h30	1h30		3h00	2	4	40%	60%
Unité Transversale									
Langue Etrangère	21h		1h30		2h00	1	2		100%
Total Semestre 3	357h	10h30	9h00	7h30	17h00	16	30		

Figure 2.3: Table Showing Semester 3.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1(O/P)		3h00	3h00	3h00	6h00	5	10		
Théorie des langages	63h	1h30	1h30	1h30	3h00	2	5	40%	60%
Système d'exploitation 1	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
UEF2(O/P)		4h30	1h30	3h00	6h00	6	10		
Base de données	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
Réseaux	63h	3h00		1h30	3h00	3	5	40%	60%
UE méthodologie									
UEM (O/P)		3h00		3h00	03h00	4	8		
Programmation orienté objet	42h	1h30		1h30	3h00	2	4	40%	60%
Développement d'Applications Web	42h	1h30		1h30	3h00	2	4	40%	60%
Unité Transversale									
Langue Etrangère	21h		1h30		2h00	1	2		100%
Total Semestre 4	357h	10h30	6h00	9h00	17h00	16	30		

Figure 2.4: Table Showing Semester 4.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1(O/P)		3h00	3h00	3h00	6h00	6	10		
Système d'exploitation 2	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
Compilation	63h	1h30	1h30	1h30	3h00	3	5	40%	60%
UEF2(O/P)		3h00	3h00	3h00	7h30	5	10		
Génie Logiciel	63h	1h30	1h30	1h30	3h00	2	5	40%	60%
Interface Homme Machine	63h	1h30	1h30	1h30	4h30	3	5	40%	60%
UE méthodologie									
UEM (O/P)		3h00	3h00		6h00	4	8		
Programmation Linéaire	42h	1h30	1h30		3h00	2	4	40%	60%
Probabilités et Statistiques	42h	1h30	1h30		3h00	2	4	40%	60%
Unité Transversale									
Economie numérique et veille stratégique	21h		1h30		2h00	1	2		100%
Total Semestre 5	357h	9h00	10h30	6h00	21h30	16	30		

Figure 2.5: Table Showing Semester 5.

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1 (O/P)		3h00	1h30	1h30	6h00	6	10		
Applications Mobiles	42h	1h30		1h30	3h00	3	5	40%	60%
Sécurité Informatique	42h	1h30	1h30		3h00	3	5	40%	60%
UE fondamentales									
UEF2 (O/P)		3h00		3h00	6h00	6	10		
Intelligence Artificielle	42h	1h30		1h30	3h00	3	5	40%	60%
Données semi-structurées	42h	1h30		1h30	3h00	3	5	40%	60%
Unité Méthodologie									
Projet					10h00	3	7		100%
Unité Transversale									
Créer et développer une startup	21h	1h30			3h00	1	3		100%
Total Semestre 6	189h	7h30	1h30	4h30	17h	16	30		

Figure 2.6: Table Showing Semester 6.

The following 3 tables show the semesters a student passes through while studying at the stage of Master.

Note: tables of semesters for one specialization has been shown only in the master's stage (fundamental).

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			Continu	Examen
UE Fondamentale									
UEF1(O/P) : Systèmes intelligents 1	202.5	4.5	4.5	4.5		9	18		
LF11- Logique et Fondements Informatique 1	67.5	1.5	1.5	1.5		3	6	50%	50%
SE- Systèmes Experts	67.5	1.5	1.5	1.5		3	6	50%	50%
PP- Paradigmes de Programmation	67.5	1.5	1.5	1.5		3	6	50%	50%
UE Méthodologie									
UEM1(O/P) : Modélisation et analyse	105	4.5	0	2.5		5	9		
MS- Modélisation et Simulation	45	1.5		1.5		2	4	50%	50%
AD- Analyse de Données	60	3		1		3	5	50%	50%
UE Transversale									
UET1(O/P) : Expression et déontologie1	67.5	3	1.5	0		3	3		
Anglais1	45	1.5	1.5			2	2	50%	50%
Déontologie et éthique	22.5	1.5				1	1	50%	50%
Total Semestre 1	375	12	6	7		17	30		

Figure 2.7: Table Showing Semester 1 (fundamental).

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1(O/P) : Outils pour la distribution	112.5	3	1.5	3	0	5	10		
PC-Protocoles de communication TCP/IP	67.5	1.5	1.5	1.5		3	6	50%	50%
Adis -Algorithmes distribués	45	1.5		1.5		2	4	50%	50%
UEF2(O/P) : Systèmes intelligents 2	90	3	3	0	0	4	8		
LF12 - Logique et Fondements Informatique 2	45	1.5	1.5			2	4	50%	50%
SFLP - Sémantique Formelle des Langages de Programmation	45	1.5	1.5			2	4	50%	50%
UE méthodologie									
UEM1(O/P) : Méthodologie de Gestion de projet	105	4.5	0	2.5	0	5	9		
TI- Théorie de l'Information	67.5	3		1.5		3	6	50%	50%
GP-Gestion de projet	37.5	1.5		1		2	3	50%	50%
UE transversales									
UET1(O/P) : Expression et déontologie2	67.5	3	1.5	0	0	3	3		
JLI-Juridiction et législation informatique	22.5	1.5				1	1	50%	50%
Anglais2	45	1.5	1.5			2	2	50%	50%
Total Semestre 2	375	13.5	6	5.5	0	17	30		

Figure 2.8: Table Showing Semester 2 (fundamental).

Unité d'Enseignement	VHS	V.H hebdomadaire				Coeff	Crédits	Mode d'évaluation	
	14-16 sem	C	TD	TP	Autres			Continu	Examen
UE fondamentales									
UEF1(O/P) : Concepts avancés	202.5	4.5	4.5	4.5	0	9	18		
ROA- Recherche Opérationnelle Avancée	67.5	1.5	1.5	1.5		3	6	50%	50%
GLA- Génie Logiciel Avancé	67.5	1.5	1.5	1.5		3	6	50%	50%
BDA- Base de Données Avancées	67.5	1.5	1.5	1.5		3	6	50%	50%
UE méthodologie									
UEM1(O/P) : Systèmes parallèles	105	4	1.5	1.5		5	9		
ArP- Architectures Parallèles	45	1.5		1.5		2	4	50%	50%
AlP- Algorithmes Parallèles	45	1.5	1.5			2	4	50%	50%
MFP- Méthodes Formelles pour le Parallélisme	15	1				1	1	50%	50%
UE Transversale									
UET1(O/P) : Recherche d'information	67.5	3	0	1.5		3	3		
MTS- Méthodologie de travail scientifique	45	1.5		1.5		2	2	50%	50%
Anglais technique	22.5	1.5				1	1	50%	50%
Total Semestre 3	375	11.5	6	7.5		17	30		

Figure 2.9: Table Showing Semester 3 (fundamental).

2.2 Continuous Assessment

The method of evaluating students during the semester is up to the teacher, and this means that there are several methods for the evaluation process. The method that most teachers follow depends on calculating the participation point, attendance point, interrogation point, and homework point, the method of calculating the evaluation is done by collecting the attendance point with the participation point and then Adding to it either the interrogation point, or the homework point, or the combination of the interrogation point and homework point together [8].

2.3 Presence and Participation

The process of registering attendance is done by calling, and this takes time, During this process, some teachers may find it difficult to pronounce some names, Which may cause embarrassment for some of them, as for the participation process most teachers do not record participation during each session they may be satisfied with registering the participation during some sessions only, and some of them may be satisfied with registering the attendance only due to the lack of time and to avoid chaos.

Therefore, developing and designing an application that manages attendance and participation during the class makes us avoid all these problems.

3 Suggest an alternative solution

Some departments of the college contain a large number of students, especially in the first year, and this makes it difficult for professors to assess students during the semester, which in turn depends on recording the attendance and participation of students, to avoid all these problems, we suggest creating a program (website and application mobile) that works to manage attendance and participation during the semester Through it, we can record the attendance of the student in an easier and faster way, and through it, we can record some of the student's participation during the class, and the administration can also view the student's absences during the semester, and the student allows me to be aware of the number of his absences during the semester, It can also calculate the continuous assessment at the end of the semester.

4 capture of requirements

4.1 Non-functional requirements

Non-functional requirements are quality constraints that must be met in the system, so we must make sure of the following requirements:

Performance: The application must be able to complete the operations requested by the user at a high speed.

Security: The application must be secure to protect users' data, and therefore requires that usernames and passwords be specified for each user.

Ease of use: The application should be easy to deal with his, and its interface should be simple and easy to use.

Extensibility: The application must be capable of adding new functions, modifying or removing existing functions.

4.2 functional requirements

4.2.1 Definition of actors

definition: "an actor is an outside entity that interacts with the system" [9] It can be said that the actor is an entity physique (person) or abstract (software) interacting directly with the system In our project we have 3 actors: admin, teacher, student.

4.2.2 Functional requirements for each actor

The following table shows the functions of each actor

Actors	functions of actors
Admin	<ul style="list-style-type: none">-Import data into the database-Export data from the database-Assignments management (modules for teachers, groups for teachers, students for groups)-Accounts Management (teachers-student)
Teacher	<ul style="list-style-type: none">-Manages the class session-Manages attendance-Manages participation-Manages evaluation
Student	<ul style="list-style-type: none">-Registers attendance during the class-Participates during the class-Sees his absences

Table 1: List of actors with their functions

5 The use case diagram

The use case diagram gives an overview of the functionality of the system and the interaction of users (actors) with that system.

Figure 2.1 represents the use case diagram of our system

6 detailed explanation of use cases

Upon completion of the creation of the use case diagram, we will move to a textual description of each use case its supported bysequence diagram for a detailed explanation of each use case. We will also follow a model inspired by the model described in the following references [10, 11, 12].

6.1 The use case « Authentication »

Title: Authentication.

Actor: Admin/Teacher/Student.

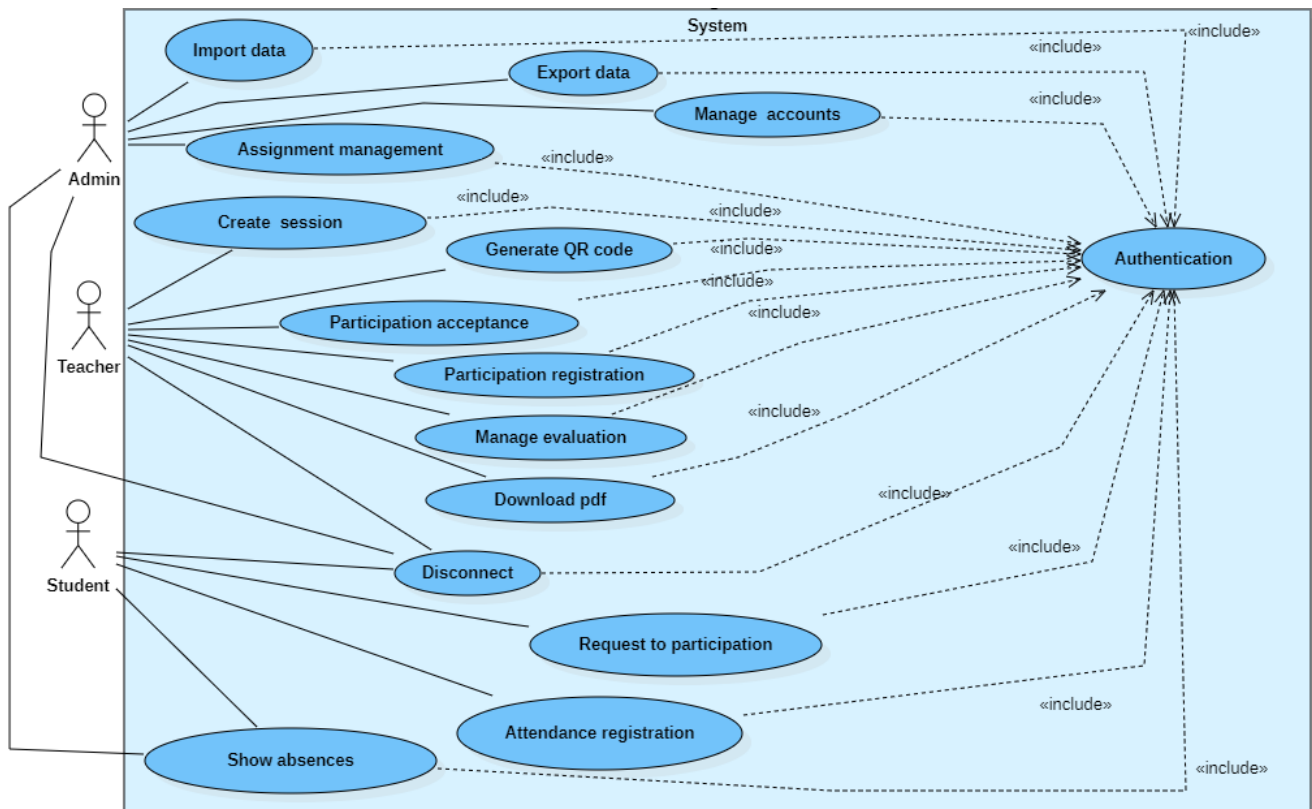


Figure 2.10: use case diagram

Abstract: This use case allows the administrator, teacher, and student to open a session

Pre-condition: Internet connection

Nominal scenario:

1. The admin/teacher/student requests authentication from the system
2. The system asks the admin/teacher/student to enter the username and password
3. The admin/teacher/student enters the username and password then validated the action
4. The system verifies the entered information (to make sure that there is information)
5. The system sends a request to the server to open a session
6. The server checks the username and password
7. The server opens a session according to each user's role by displaying their functions

The alternative sequences

1. Data entry is not completed.
2. Starts at point 5 of the nominal scenario.
3. The system informs the user that the data entry process has not been completed.
4. The system goes back to point 3 of the nominal scenario.

The sequences of errors

1. "User name" or "password" is incorrect.
2. It starts at point 7 of the nominal scenario.

3. The system tells the user that their “username” or “password” is wrong.
4. The system goes back to point 2 of the nominal scenario.

Post-condition: The session is open

Figure 2.2 shows the sequence diagram for this use case.

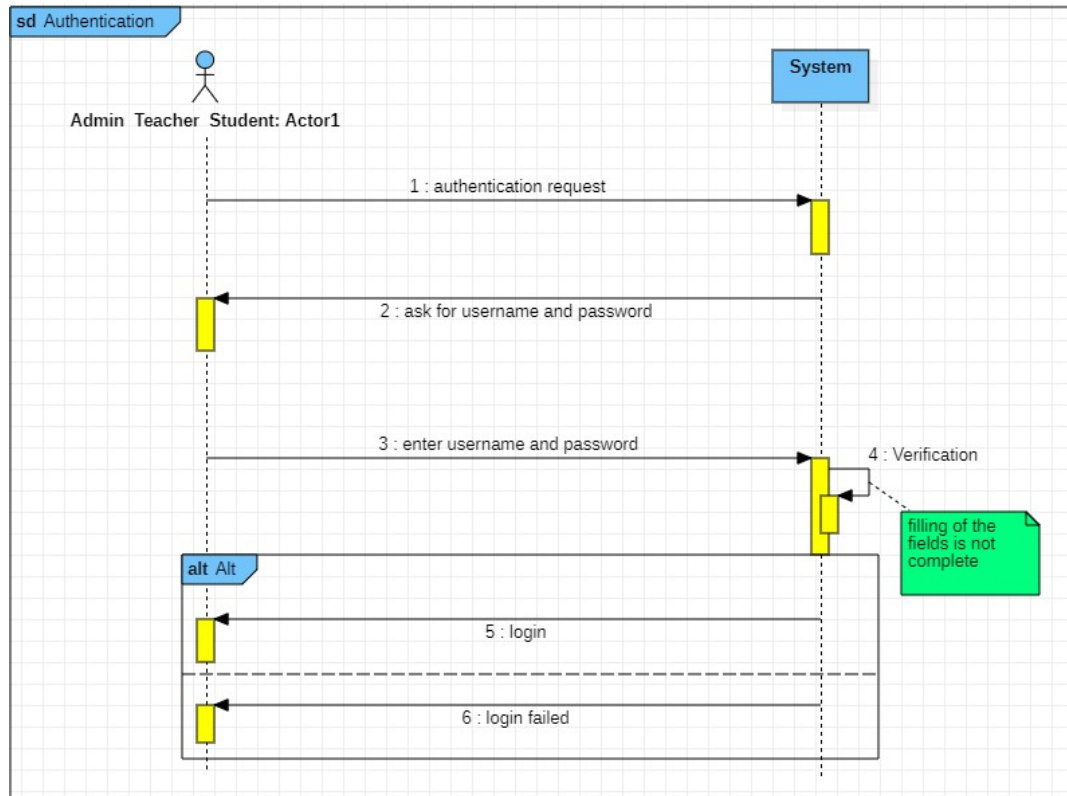


Figure 2.11: sequence diagram for use case Authentication

6.2 The use case « Disconnect »

Title: disconnect.

Actor: Admin/Teacher/Student.

Abstract: This use case allows the administrator, teacher, and student to close the open session

Nominal scenario:

1. The admin/teacher/student asks the system to close the session
2. The system sends the request to the server
3. The server is processing the request sent from the system
4. The admin/teacher/student receives a notification from the system (session closed)

Post-condition: The session is closed

Figure 2.3 shows the sequence diagram for this use case.

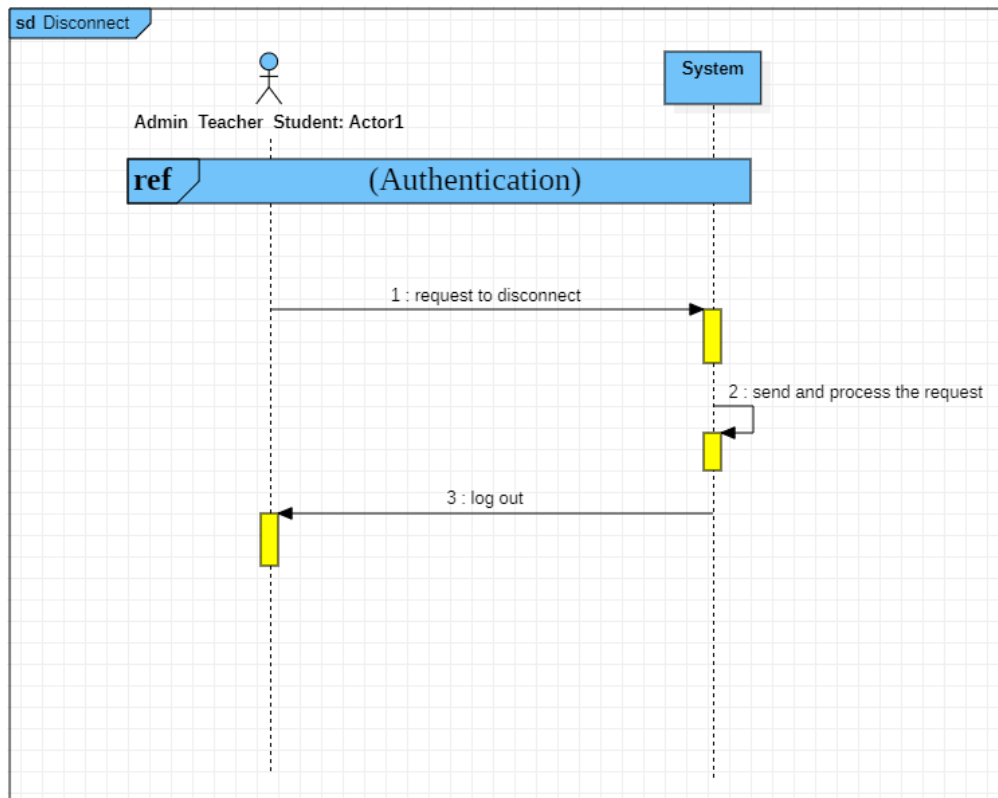


Figure 2.12: sequence diagram for use case Disconnect

6.3 The use case « Import data »

Title: import data.

Actor: Admin.

Abstract: This use case allows the administrator to move data from external files to the database

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The admin asks the system to transfer data from an external file to the database
3. The system sends a request to the server
4. The server transfers data from the external file to the database
5. The admin receives a notification from the system (the file was imported successfully)

Post-condition: Data has been imported

Figure 2.4 shows the sequence diagram for this use case.

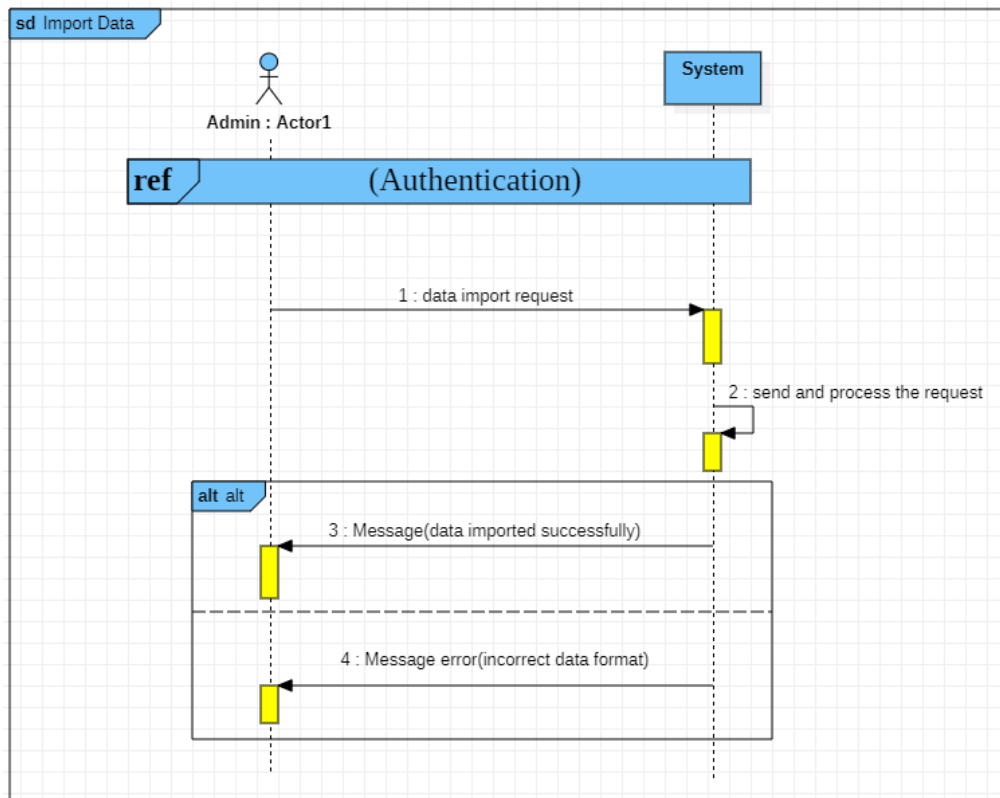


Figure 2.13: sequence diagram for use case Import data

6.4 The use case « Export data »

Title: export data.

Actor: Admin.

Abstract: This use case allows the administrator to move data from the database to external files

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The admin asks the system to export data from the database
3. The server presents the list of downloadable files
4. Admin exports data by downloading files

Post-condition: The data has been exported as files

Figure 2.5 shows the sequence diagram for this use case.

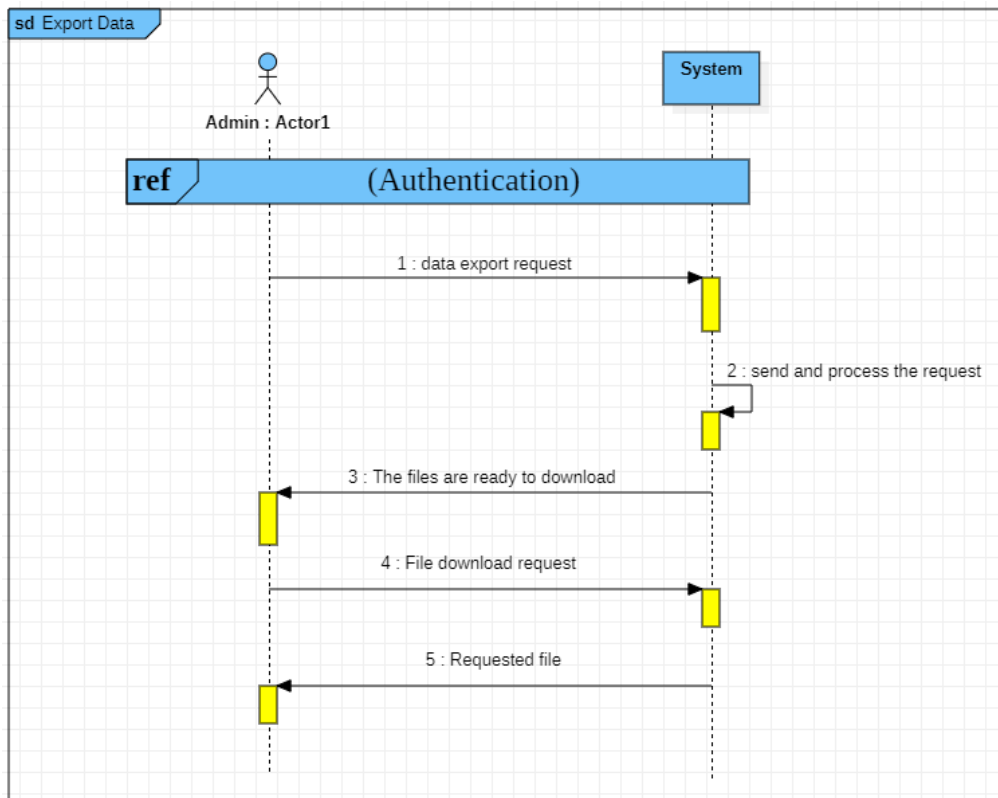


Figure 2.14: sequence diagram for use case Export data

6.5 The use case «Assignment management»

Title: assignment management.

Actor: Admin.

Abstract: This use case allows the admin to make assignments for (modules to teachers) and (groups to teachers) and (students to groups).

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The admin asks the system to assign (modules to teachers) and (groups to teachers) and (students to groups)
3. The system asks the admin to enter the information
4. The admin enters the information
5. The system verifies the information entered.
6. The system sends the request to the server
7. The server processes the request and saves the data
8. The admin receives a notification from the system that the operation was completed

The alternative sequences

1. Data entry is not completed.
2. Starts at point 6 of the nominal scenario.
3. The system informs the admin that the data entry process has not been completed.
4. The system goes back to point 3 of the nominal scenario.

Post-condition: make assignment

Figure 2.6 shows the sequence diagram for this use case.

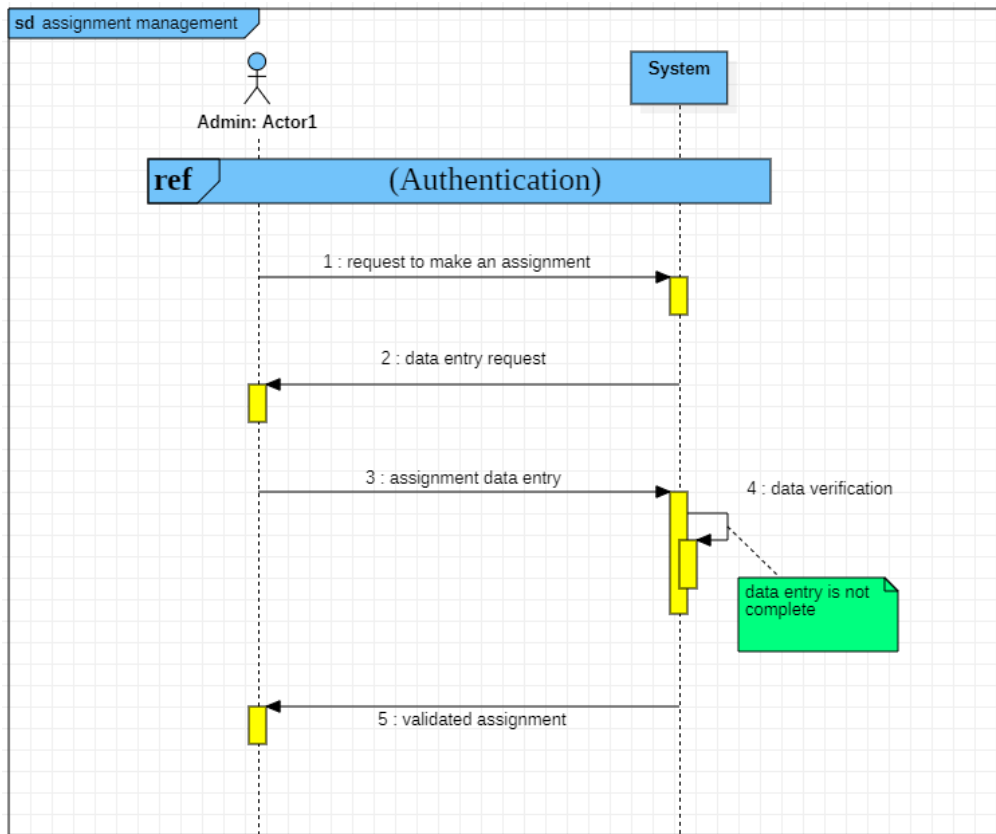


Figure 2.15: sequence diagram for use case Assignment management

6.6 The use case « Create session »

Title: create session.

Actor: Teacher.

Abstract: This use case allows the teacher to create a session to record (attendance and participation)

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The teacher asks the system to create a session
3. The system asks the teacher to specify the information required to create the session

4. The teacher selects the information
5. The system verifies the information specified by the teacher
6. The system sends the request to the server
7. The server creates the session
8. The teacher receives a notification from the system that the process is complete

The alternative sequences

1. Not all data is specified.
2. Starts at point 6 of the nominal scenario.
3. The system informs the teacher that the data selection process has not been completed.
4. The system goes back to point 3 of the nominal scenario.

Post-condition: the session created

Figure 2.7 shows the sequence diagram for this use case.

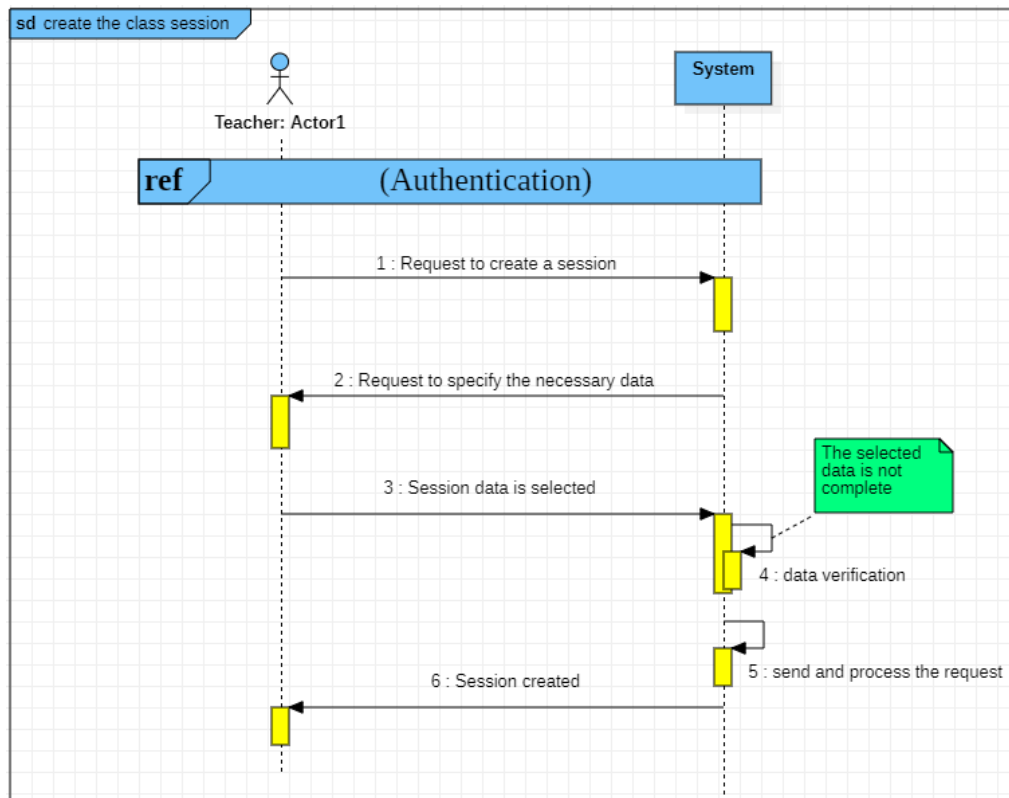


Figure 2.16: sequence diagram for use case Create session

6.7 The use case « Participation acceptance »

Title: Participation Acceptance.

Actor: Teacher.

Abstract: This use case allows the teacher to enable participation and select participation from students' participation

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The teacher asks the system to enable participation
3. The system sends the request to the server
4. The teacher receives a notification from the system that participation is enabled
5. The teacher asks the system to accept the student's participation
6. The system sends the request to the server
7. The teacher receives a notification from the system that the process is complete

Post-condition: Student participation accepted

Figure 2.8 shows the sequence diagram for this use case.

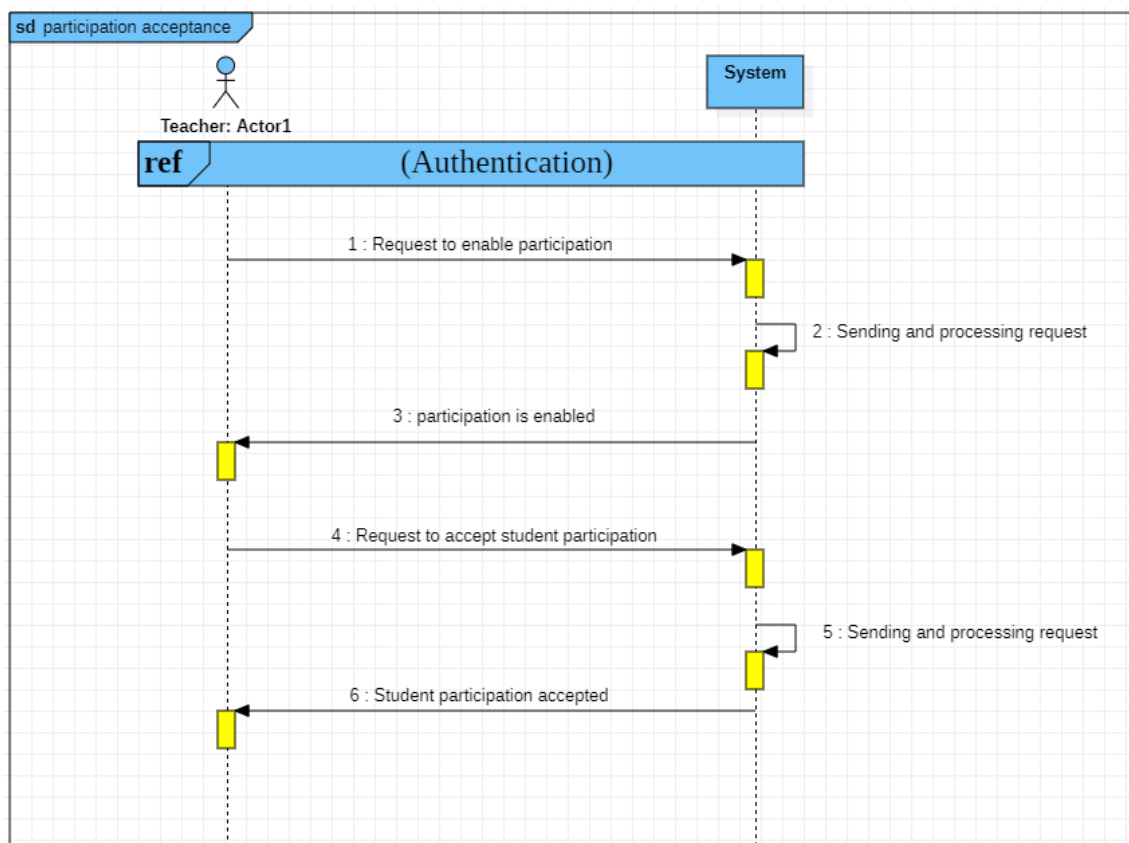


Figure 2.17: sequence diagram for use case Participation acceptance

6.8 The use case « Participation registration »

Title: Participation Registration.

Actor: Teacher.

Abstract: This use case allows the teacher to register student's participation

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication and participation acceptance process
2. The teacher asks the system to register student's Participation
3. The system sends the request to the server
4. The teacher receives a notification from the system that the process is complete

Post-condition: participation has been registered

Figure 2.9 shows the sequence diagram for this use case.

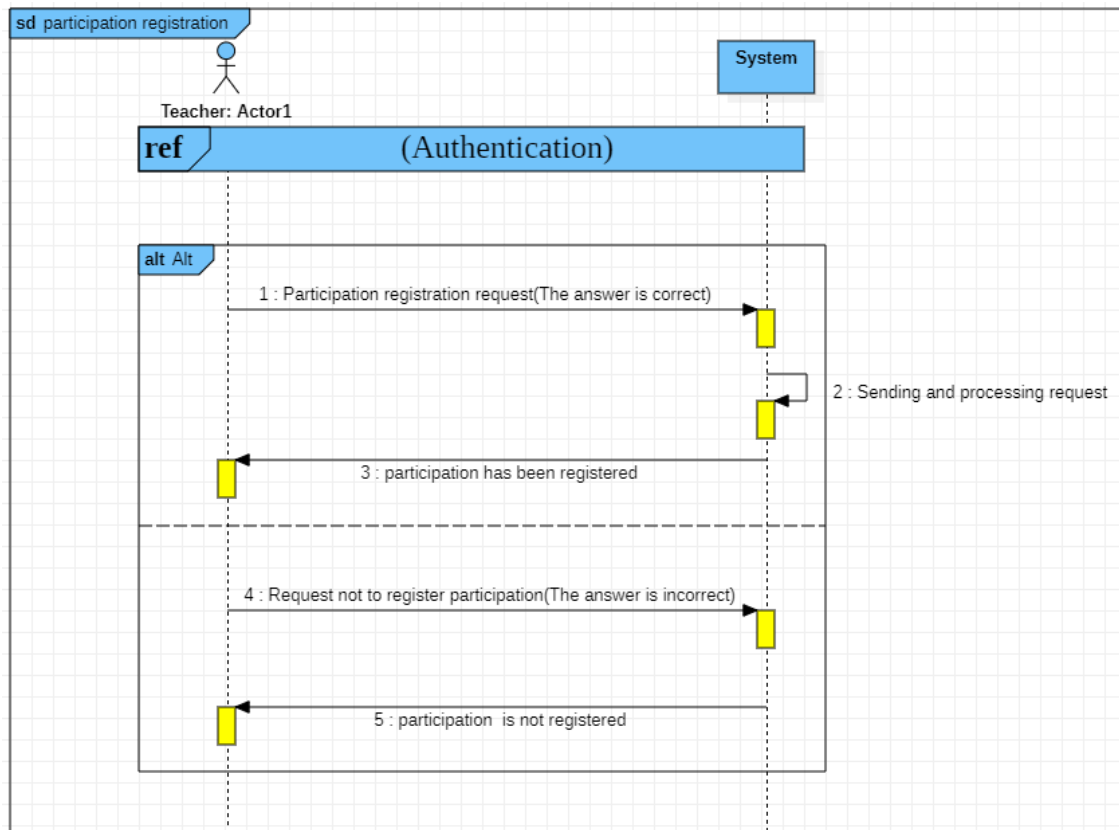


Figure 2.18: sequence diagram for use case Participation registration

6.9 The use case «Manage evaluation»

Title: Manage Evaluation.

Actor: Teacher.

Abstract: This use case allows the teacher to calculate student assessments during the semester

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The teacher asks the system to calculate student assessments

3. The system asks the teacher to enter the data
4. The teacher enters the data
5. The system verifies the data entered.
6. The system sends the request to the server
7. The teacher receives a notification from the system that the operation was completed

The alternative sequences

1. Data entry is not completed.
2. Starts at point 6 of the nominal scenario.
3. The system informs the teacher that the data entry process has not been completed.
4. The system goes back to point 3 of the nominal scenario.

Post-condition: displayed student assessments

Figure 2.10 shows the sequence diagram for this use case.

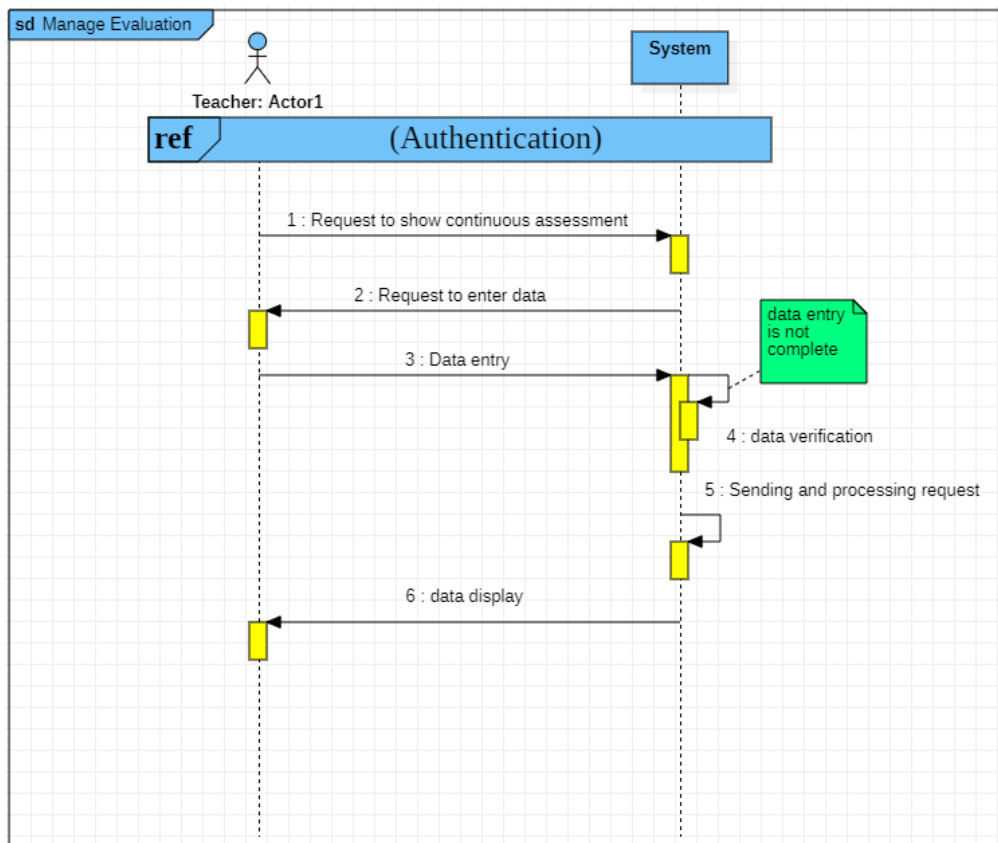


Figure 2.19: sequence diagram for use case Manage evaluation

6.10 The use case « Attendance registration »

Title: Attendance registration.

Actor: Student.

Abstract: This use case allows the student to record attendance during class

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The student asks the system to register his attendance
3. The system asks the student for a QR code for authentication
4. The student submits the QR code to the system
5. The system sends the request to the server
6. The server processes the request and registers the student's attendance
7. The student receives a notification from the system that the attendance registration process has been completed

Post-condition: Attendance is registered

Figure 2.11 shows the sequence diagram for this use case.

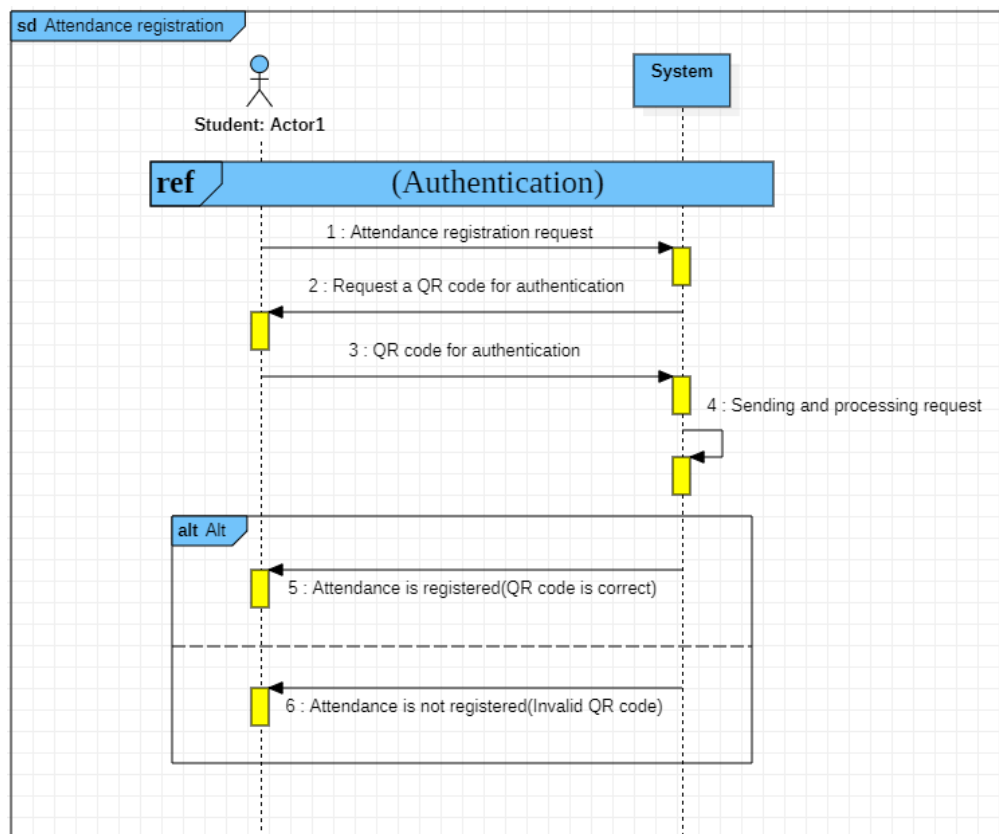


Figure 2.20: sequence diagram for use case Attendance registration

6.11 The use case « Request to participation »

Title: Request to Participation.

Actor: Student.

Abstract: This use case allows the student to participate during the class and knows if (his

participation is acceptable or not)

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The student asks the system to allow him to participate
3. The system sends the request to the server
4. The server processes the request (adding it to the list of participants)
5. The student receives a notification from the system stating that his participation has been accepted

Post-condition: Acceptance of participation

Figure 2.12 shows the sequence diagram for this use case.

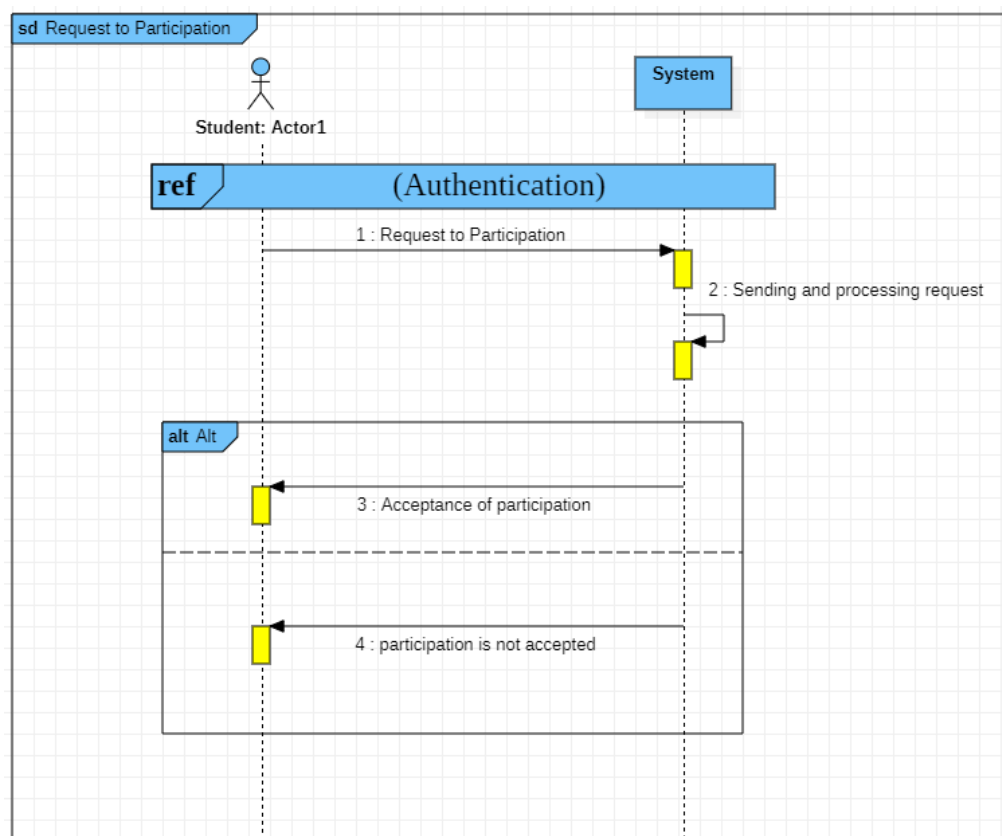


Figure 2.21: sequence diagram for use case Request to participation

6.12 The use case « Show absences »

Title: Show absences.

Actor: Student.

Abstract: This use case allows the student to see his absence

Pre-condition: Internet connection

Nominal scenario:

1. After the authentication process
2. The student asks the system to display his absences
3. The system sends the request to the server and then the server processes the request
4. The system display to the student his absences

Post-condition: absences displayed.

Figure 2.13 shows the sequence diagram for this use case.

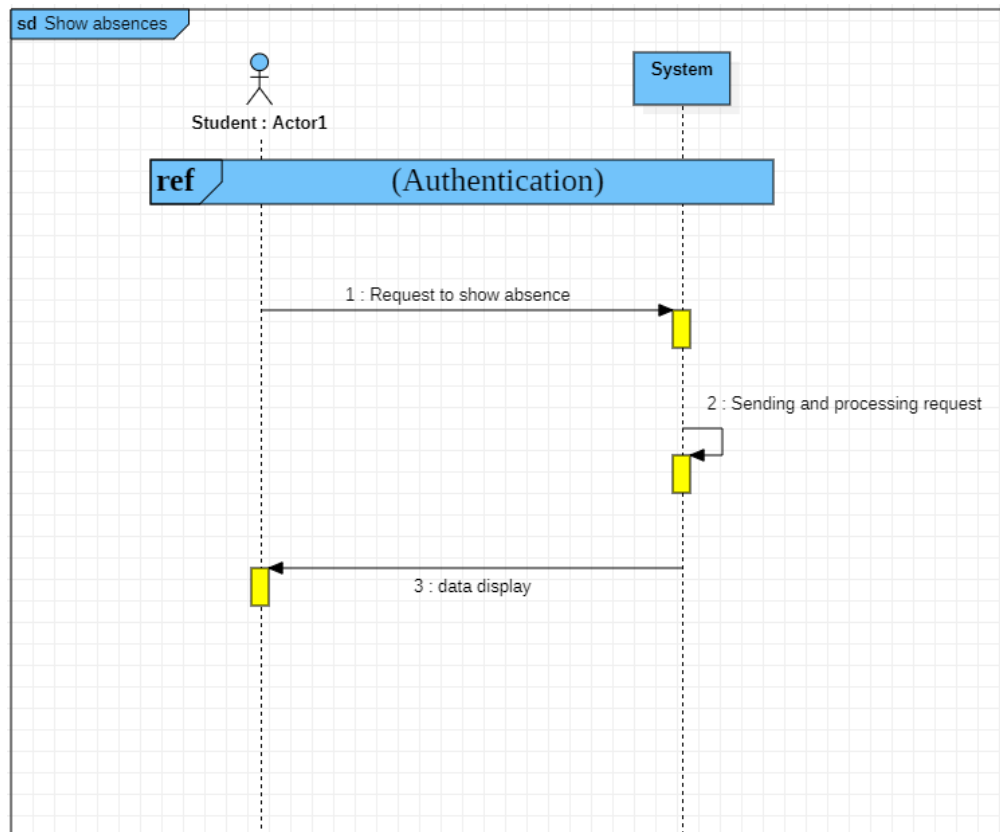


Figure 2.22: sequence diagram for use case Show absences

7 Conception phase

It is the stage through which the initial design of the project is done. It also defines the structure of the system and its basic parts, through the creation of system diagrams, and to know out the structure of the product we used a class diagram, which describes the fixed objects that will be stored in the database, and we also provided a detailed explanation of the parts of this diagram .

7.1 Class Diagram

Through the class diagram, we can know the static objects of the project, which will be stored in the database later.

Figure 2.14 shows the class diagram of this project.

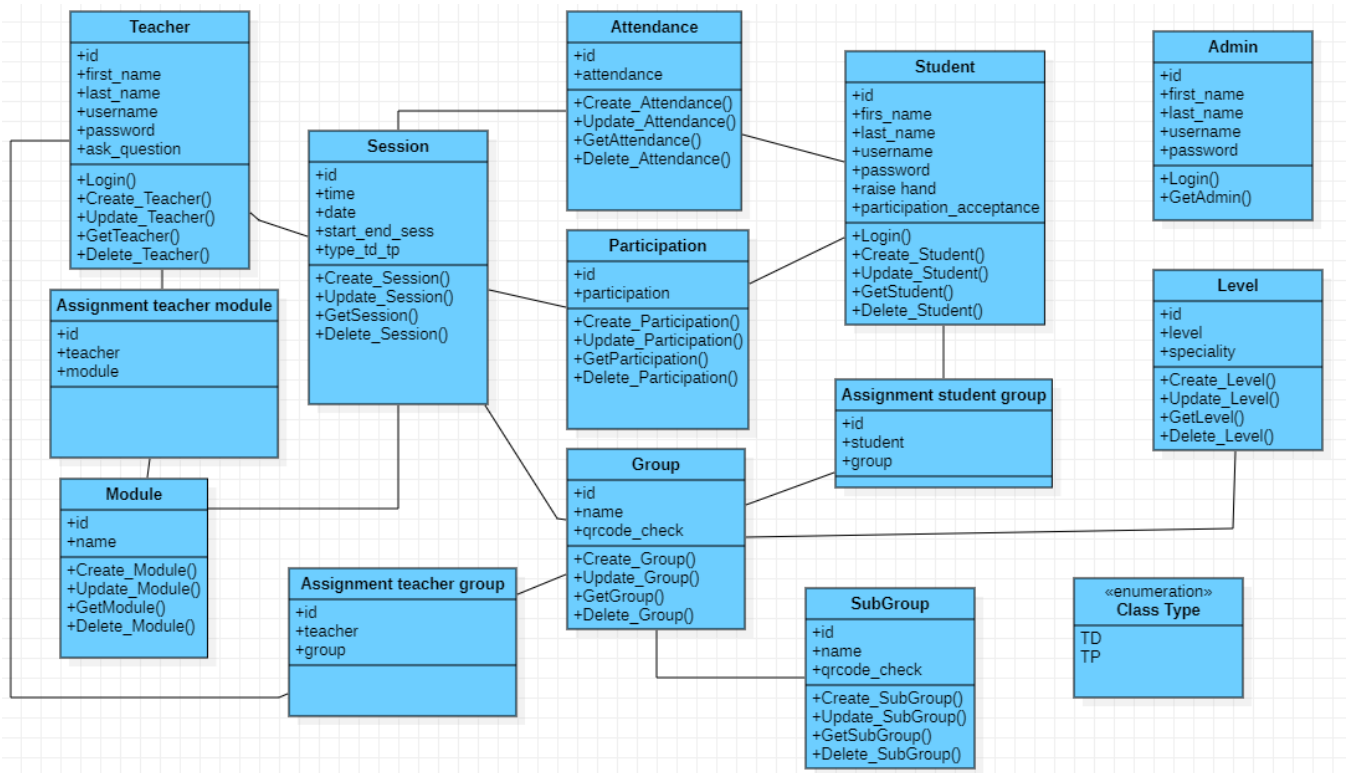


Figure 2.23: class diagram

7.2 Data Dictionary

7.2.1 The Attributes

Class	Attribute	Description	Type
Admin	id	Admin identifier	String
	first_name	Admin's first name	String
	last_name	Admin's last name	String
	username	Username for the admin account	String
	password	Password for the admin account	String
Teacher	id	Teacher identifier	String
	first_name	Teacher's first name	String
	last_name	Teacher's last name	String
	username	Username for the Teacher account	String
	password	Password for the Teacher account	String
	ask_question	Variable to activate and deactivate participation	Boolean
Student	id	Student identifier	String
	first_name	Student's first name	String
	last_name	Student's last name	String
	username	Username for the Student account	String
	password	Password for the Student account	String
	raise_hand	Variable to request participation or cancel the participation request	Boolean
	par_acceptance	Variable to acceptance or non-acceptance of student participation	Boolean
Module	id	Module identifier	String
	name	The Module's name	String
Group	id	Group identifier	String
	name	The Group's name	String
	qrcode_check	A variable to store the modified QR-code value	String
Attendance	id	Attendance identifier	String
	attendance	Variable to calculate the number of attendance times	number

Participation	id	Participation identifier	String
	participation	Variable to calculate the number of participation times	number
Session	id	Session identifier	String
	time	the time when the session began	Time
	date	The date of the completed session	Date
	start_end_sess	Variable to know the start and end of the session	Boolean
	type_td_tp	Class session type TD or TP	ClassType
Level	id	Level identifier	String
	level	Student's educational level	String
	speciality	Student's speciality	String

Table 2: Data Dictionary: The Attributes.

7.2.2 The Operations

Class	Operations	Description
Admin	Login()	The Admin login
	GetAdmin()	Get The Admin
Teacher	Login()	The Teacher login
	Create_Teacher()	Create The Teacher
	Update_Teacher()	Update The Teacher
	GetTeacher()	Get The Teacher
	Delete_Teacher()	Delete The Teacher
Student	Login()	The Student login
	Create_Student()	Create The Student
	Update_Student()	Update The Student
	GetStudent()	Get The Student
	Delete_Student()	Delete The Student
Module	Create_Module()	Create The Module
	Update_Module()	Update The Module
	GetModule()	Get The Module
	Delete_Module()	Delete The Module
Group	Create_Group()	Create The Group
	Update_Group()	Update The Group
	GetGroup()	Get The Group
	Delete_Group()	Delete The Group
Attendance	Create_Attendance()	Create The Attendance
	Update_Attendance()	Update The Attendance
	GetAttendance()	Get The Attendance
	Delete_Attendance()	Delete The Attendance
Participation	Create_Participation()	Create The Participation
	Update_Participation()	Update The Participation
	GetParticipation()	Get The Participation
	Delete_Participation()	Delete The Participation

Session	Create_Session()	Create The Session
	Update_Session()	Update The Session
	GetSession()	Get The Session
	Delete_Session()	Delete The Session
Level	Create_Level()	Create The Level
	Update_Level()	Update The Level
	GetLevel()	Get The Level
	Delete_Level()	Delete The Level

Table 3: Data Dictionary: The Operations.

8 Conclusion

In this chapter, we finished the needs analysis and design phase, and for that we used some UML diagrams, now everything is ready to move on to the implementation and investigation phase that we will discuss in the next chapter.

Chapter 3

Implementation and Realization

1 Introduction

The implementation phase and the investigation phase are the last two phases of our project, In the implementation phase, we will convert all the information and diagrams that we touched upon in the analysis and design phases into programming commands (writing code) and in investigation phase we will divide the program into several parts and then make sure that All parts of the program are working correctly, and they are consistent and integrated with each other and give us the required results.

In the first part of this chapter, we will give a brief presentation about the work environment, programs and programming languages that were used to create the application, and in the second part, we will give an overview of the most important application interfaces and explain how it works.

2 General Architecture of System

Our system consists of a website and a mobile application, with an architecture of a type The client/server, where the web server handles the HTTP requests received from the clients, and the database server also handles the requests received from the (mobile application, database server).

Figure 3.1 illustrates the overall architecture of our system.

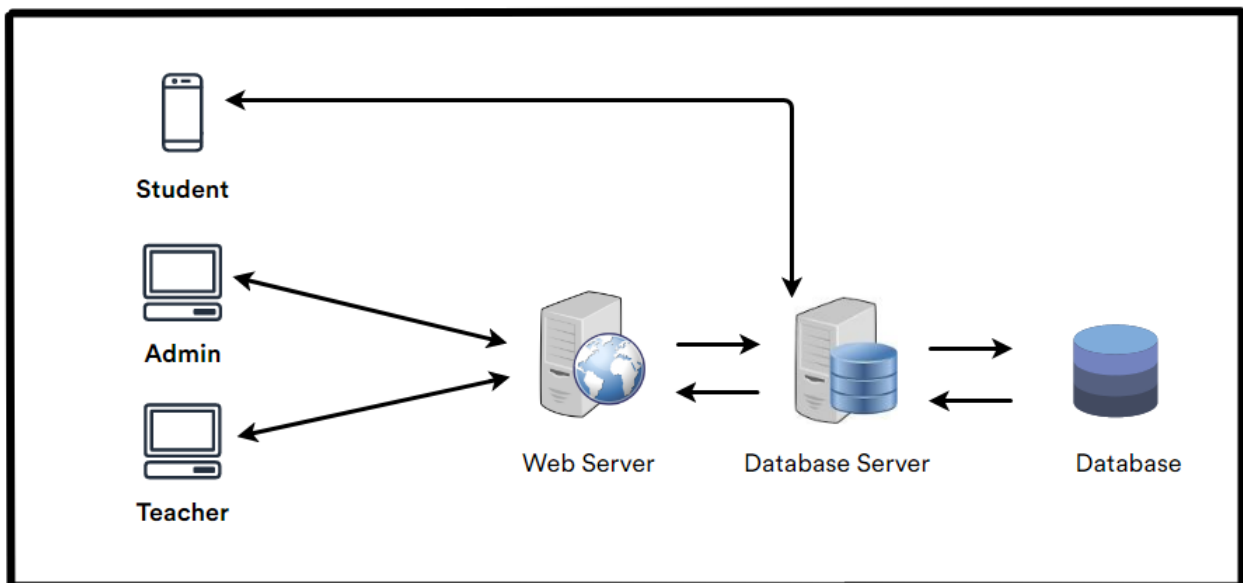


Figure 3.1: general architecture of system

3 The development environment and Tools used

In our project, React js library was used to design the front-end of the website, for the back-end, NestJS was used as a framework and Postman for testing and development, and for the design of the mobile application, React Native was used as the framework and also used both JDK and Android Studio as a group Of the tools and libraries needed to develop Android applications, Node JS was used as a running environment and as a program to create local servers on the device, and Visual Studio Code was used as a code editor, and MongoDBCompass was used to facilitate and simplify dealing with the MongoDB database, in this part we will provide a brief explanation of each A tool used in the development process.

3.1 Node JS

nodejs is an open source environment for JavaScript, this technology was introduced for the first time in 2009 by Ryan Dahl, and its are fast performance. Nodejs uses the V8 JavaScript engine that was developed by google chrome programmers and was first introduced in 2008. Figure 3.2 represents the logo of node js [13].



Figure 3.2: logo node js.

3.2 Visual Studio code

visual studio code It is a free and open source code editor designed by Microsoft Corporation. This editor supports many programming languages such as JavaScript, TypeScript, CSS, HTML and many other languages. It is also available for Windows, macOS and Linux systems. It is lightweight and includes some powerful features that has made it one of the most popular development environment tools in recent times.

Figure 3.3 represents the logo of the visual studio code [14].

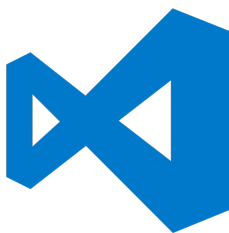


Figure 3.3: logo visual studio code.

3.3 Postman

postman is a tool that sends requests from the server to the client as it is used in the development of APIs, testing and sharing of resources between different applications, and through which we can connect a single database between many applications and sites through an API, appeared in 2012 by Abhinav Asthana to simplify the workflow API work in testing and development.

Figure 3.4 represents the logo of Postman [15].



Figure 3.4: logo Postman.

3.4 MongoDB Compass

”MongoDB Compass is a powerful GUI for querying, aggregating, and analyzing your MongoDB data in a visual environment”. [16]

MongoDB Compass can be run on macOS, Windows and Linux and is free and open source software

Figure 3.5 represents the logo of MongoDB Compass [17].



Figure 3.5: logo MongoDBCompass.

3.5 React JS and React Native

React JS:

Reactjs is an open source library of javascript libraries used to build websites and web applications. This library was created by Facebook and is specialized in building user interfaces (UI) and this library is built on the basis of the concept of Component where ReactJs provides an easy, simple and powerful way For building interfaces, the library first appeared in 2013 and is considered one of the most used front-end libraries for web development.

React Native:

React Native is a framework based on javascript and the React library for programming mobile applications that can run locally on both Android and iOS. React Native was first introduced in 2015 by Facebook. As an open-source project, React Native allows us to build an iOS and Android app with a single code.

Figure 3.6 represents the logo of React [18].

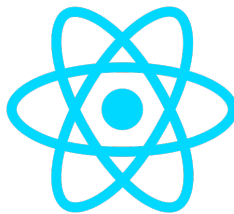


Figure 3.6: logo React

3.6 Nest JS

Nest.js is an open-source and extensible Node.js framework. Nest also relies on Javascript and Typescript, in addition to other libraries and frameworks. he uses Express framework is used as an HTTP Server, that specializes in the backend aspect only, leaving you to choose

the frontend right for you.

Figure 3.7 represents the logo of nest js [19].



Figure 3.7: logo nest js.

3.7 Android Studio

Android Studio is an integrated development environment for the development of Android applications it appeared for the first time in 2013 and that was in conjunction with the event of the Google I/O conference, where it received full support from Google, and Android Studio offers many features, the most important of which are:

- 1- Flexible build system based on Gradle
- 2- A fast and feature-rich emulator
- 3- A unified environment in which you can develop for all android devices

Figure 3.8 represents the Android Studio logo [20].



Figure 3.8: logo Android Studio

3.8 Java SE Development Kit «JDK»

JDK stands for Java Development Kit which is a set of tools and libraries needed to develop Java-based software applications and applications and is a core package in java, along with the Java Runtime Environment (JRE) and the Java Virtual Machine (JVM).

Figure 3.9 represents the components and structure of Java SE Development Kit [21]

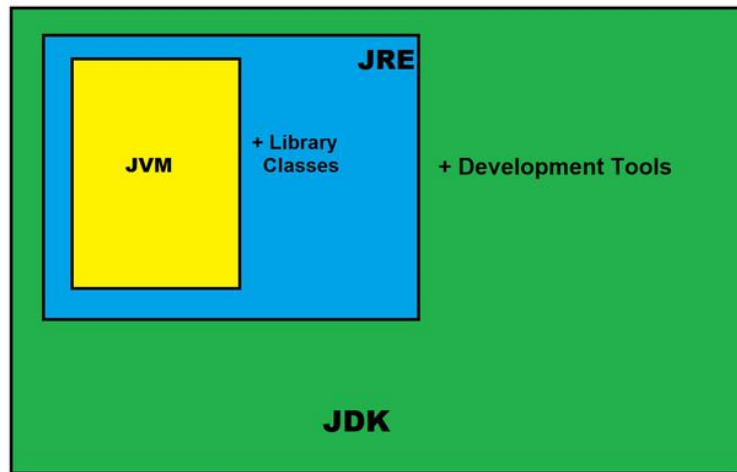


Figure 3.9: JAVA Development Kit (JDK)

4 Programming languages used

4.1 language JavaScript

JavaScript is a dynamic programming language used for web development, web applications, and others, It is used by the client for Front-end development, and It has also evolved into a language for back-end development using Node.js, this makes it competitive with PHP as it does not need a compiler, It appeared in 1995 and was called Previously Live Script, JavaScript is widely used with the design language HTML and CSS.

Figure 3.10 represents the logo of JavaScript.



Figure 3.10: logo JavaScript.

4.2 language HTML

HTML is an abbreviation for HyperText Markup Language and is known as Hypertext Markup Language, which is a special markup language used to design and build web pages and websites.

HTML is not considered a programming language and is the main structure of any page or

website , and the history of the emergence of HTML dates back to 1991 by Tim Berners-Lee. Figure 3.11 represents the logo of the HTML.



Figure 3.11: logo HTML.

4.3 language CSS

CSS is an abbreviation for Cascading Style Sheet. It is one of the main techniques used to build web pages and is friendly to the HTML language. It accompanies it and beside it in the design and creation of web pages. CSS appeared in 1996 by the World Wide Web Consortium W3C.

Figure 3.12 represents the logo of CSS.



Figure 3.12: logo CSS.

5 The functionalities of the realized application

We can divide our program into three categories, which are the administrator, the teacher and the student, the functions of the admin and the teacher are implemented through the website and the student's functions are implemented through the mobile application.

Figure 3.13 The hierarchical chart presents the content of the website according to the roles of the admin and teacher

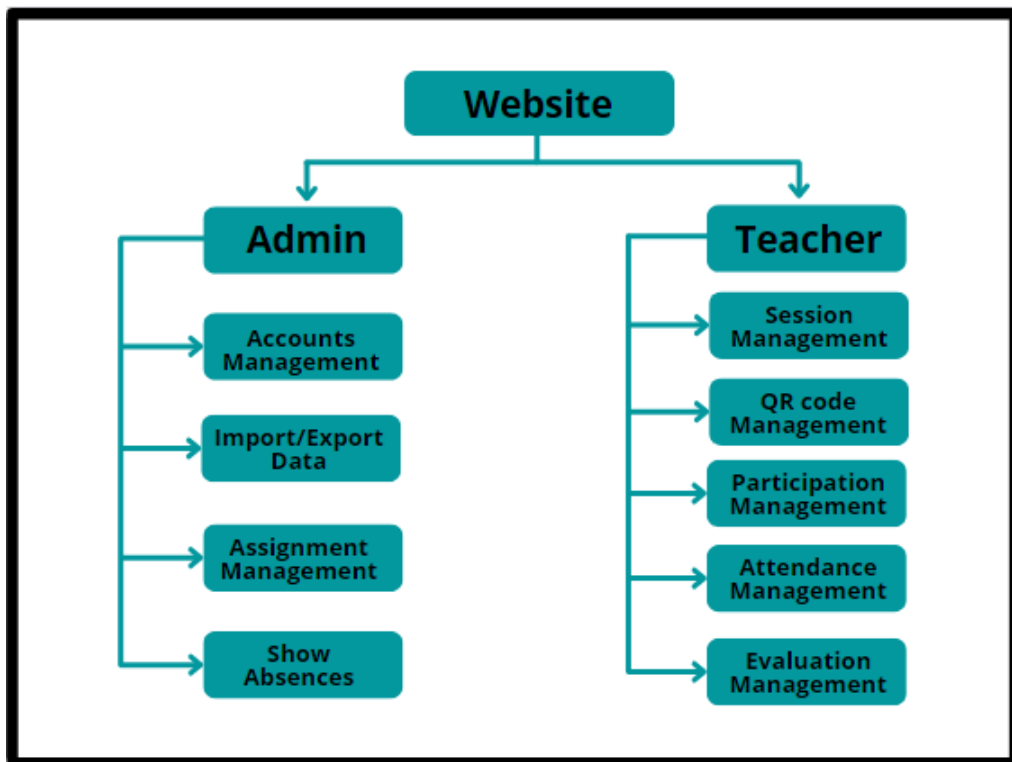


Figure 3.13: Hierarchy of website.

Figure 3.14 The hierarchical chart presents the content of the application mobile according to the role of the student

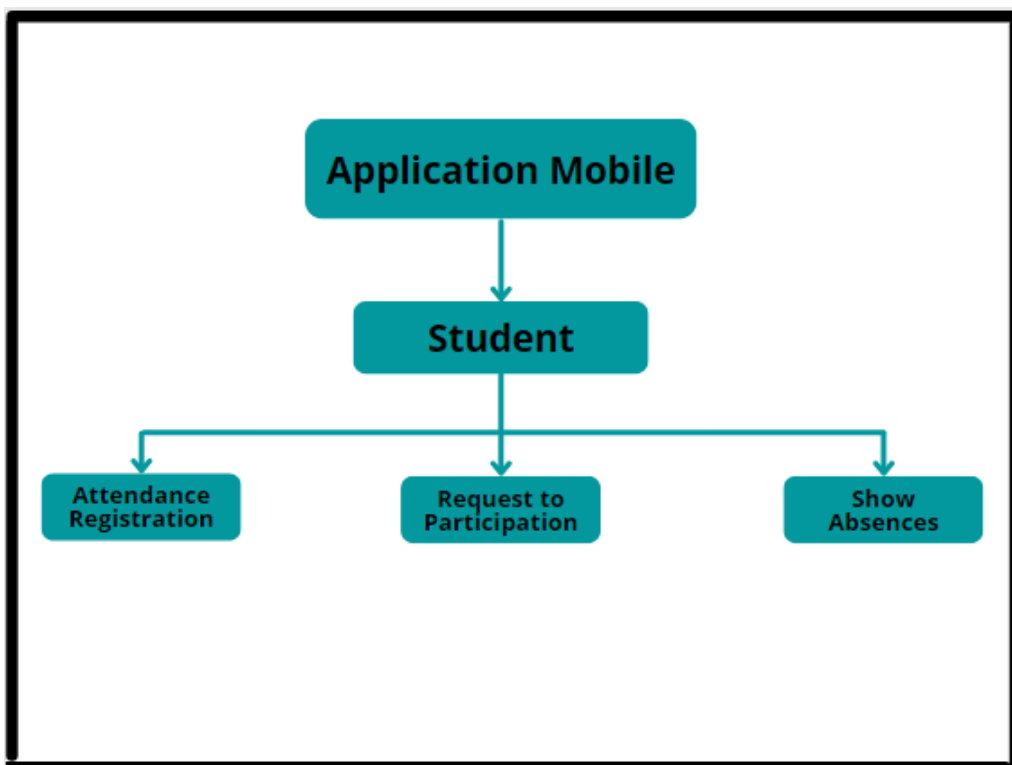


Figure 3.14: Hierarchy of the application mobile.

5.1 Description of website interfaces

by website, The admin and teacher can access their workspace through the login page. Figure 3.15 shows the teacher's login page.

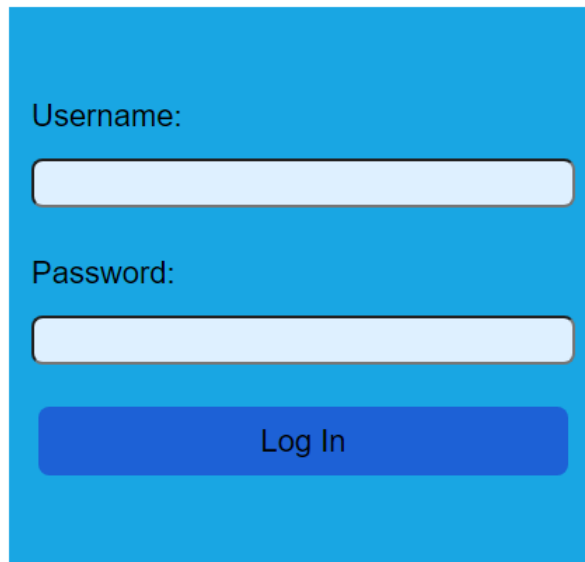
The image shows a login form on a blue background. It consists of two text input fields: the first is labeled 'Username:' and the second is labeled 'Password:'. Below these fields is a blue button with the text 'Log In' in white.

Figure 3.15: login page.

After the login process, which is done by entering the user name and password, both the teacher and the admin can access his workspace on the site.

5.1.1 The functions of the Admin

An administrator can perform several functions after the login process, so we have briefly explained its functions in this part.

Accounts Management:

The administrator can manage user accounts (teachers and students), for example, he can see the account information of any user, and he can also add, modify or delete the account of any user, the administrator chooses one of the operations (viewing account information, adding, modifying, deleting) and from Then he enters the necessary information and presses the show account button to display the account information or presses the add account button to add an account or presses the update account button To modify account information or press the delete account button to delete the account.

Import/Export Data:

The admin can import/export the data, to import the data into the database the admin specifies the file to import the data from and select the collection of database, and then

press the Import button to receive a message stating the success or failure of the task, As for the data export process, the admin presses Export button so that the system displays all collections of the database, then the admin chooses one of the files to download.

Assignment Management:

Through the task management process, the admin can assign modules to teachers and assign students to groups and assign groups to teachers.

Show Absences:

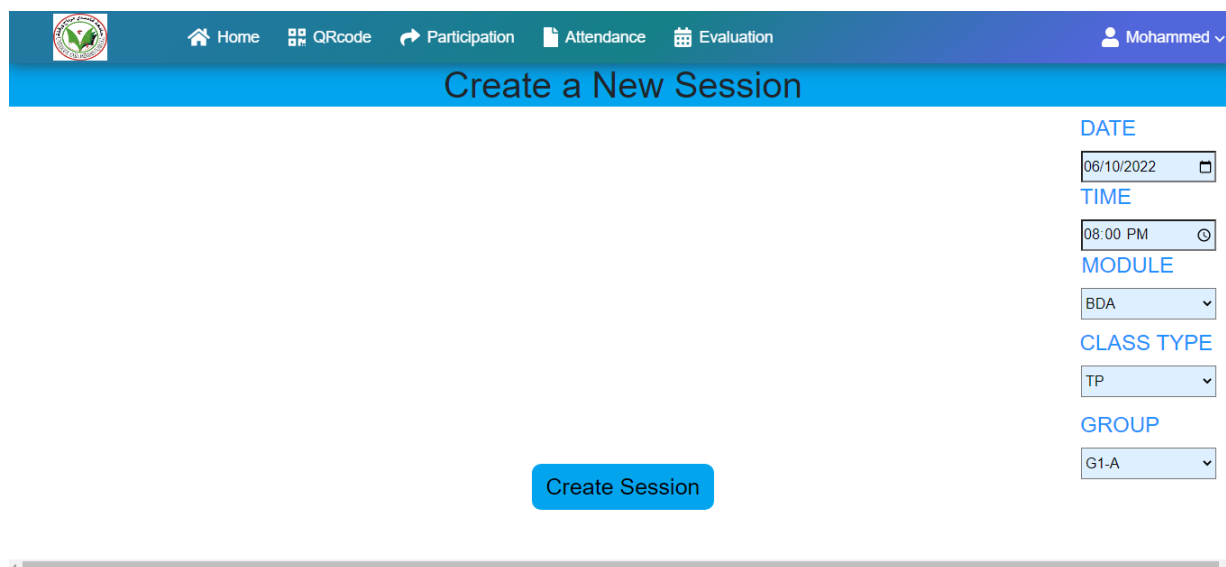
Through the process of displaying absences, the admin can see the absences of students during the semester.

5.1.2 The functions of the Teacher

A Teacher can perform several functions after the login process, so we have briefly explained its functions in this part.

Session Management:

To create a session, the teacher must specify the following information, which is the time, date, module, “class type” and group, and then press the Create Session button to create the class session through which the teacher records attendance and participation during the class (See Figure 3.16).



The screenshot shows a web interface for creating a new session. At the top, there is a navigation bar with icons for Home, QRcode, Participation, Attendance, and Evaluation, and a user profile for Mohammed. Below the navigation bar, the page title is "Create a New Session". The form includes the following fields:

- DATE:** 06/10/2022
- TIME:** 08:00 PM
- MODULE:** BDA
- CLASS TYPE:** TP
- GROUP:** G1-A

A blue "Create Session" button is positioned below the form fields.

Figure 3.16: session creation page.

QR Code Management:

The teacher can create a QR code by pressing the Generate QR Code button to create a QR code, and can also change it several times during the class then students can register

their attendance (See Figure 3.17).



Figure 3.17: QR Code Management page.

Participation Management:

The following points explain how the teacher manages the participation process:

The teacher asks a question and then presses the Ask a Question button - Then an alert is sent to the student's mobile application to tell him there is a question if he wants to participate - the student participates - the professor receives a notification informing him that this student wants to participate - the teacher agrees to the student's participation - A notification is sent to the student telling them that their participation has been accepted and they must answer the question - When the student has finished answering the question, the professor can register a student's participation if he wants - To end the participation process on this question, the professor presses the "end the question" button - The teacher can edit the student participate

Figure 3.18 shows the Participation Management page.

Participation

End the Question

List of Participation | Group: G1-A | Module: BDA

N°	First name	Last name	Registered Participation	Participation	Edit Participation
1	Brian	last_name_Brian	1		<input type="button" value="Edit"/>
2	steven	last_name_steven	0		<input type="button" value="Edit"/>
3	Richard	last_name_Richard	1	<input type="button" value="ACCEPT"/>	<input type="button" value="Edit"/>
4	lucy	last_name_lucy	0		<input type="button" value="Edit"/>
5	Samuel	last_name_Samuel	3		<input type="button" value="Edit"/>
6	Alex	last_name_Alex	2		<input type="button" value="Edit"/>
7	david	last_name_david	0		<input type="button" value="Edit"/>
8	Sarah	last_name_Sarah	0		<input type="button" value="Edit"/>

Figure 3.18: Participation Management page.

Attendance Management:

Through the attendance management process, the teacher can modify the attendance list of students, and in cases of exception, he can record the attendance of some students without the QR code, and he can also download the attendance list in pdf format.

Figure 3.19 shows the attendance management page.

Attendance Management

Download

List of Attendance | group: G1-A | module: BDA

N°	First name	Last name	Attendance	Actions
1	Brian	last_name_Brian	0	<input type="button" value="Edit"/>
2	steven	last_name_steven	1	<input type="button" value="Edit"/>
3	Richard	last_name_Richard	1	<input type="button" value="Edit"/>
4	lucy	last_name_lucy	1	<input type="button" value="Edit"/>
5	Samuel	last_name_Samuel	0	<input type="button" value="Edit"/>
6	Alex	last_name_Alex	1	<input type="button" value="Edit"/>
7	david	last_name_david	1	<input type="button" value="Edit"/>
8	Sarah	last_name_Sarah	1	<input type="button" value="Edit"/>
9	Michael	last_name_Michael	0	<input type="button" value="Edit"/>

Figure 3.19: Attendance Management page.

Evaluation Management:

Through the evaluation management process, the teacher can calculate the student's participation point and attendance point during the semester, after the teacher enters the necessary information (how many points for attendance, how many points for participation, how much is decreased for each absence) and then selects the group, and then presses the

Calculate button to show the students' attendance and participation point.

5.2 Description of Application Mobile interfaces

The student can access his workspace through the mobile application, after the login process.

Figure 3.20 shows the student login interface.

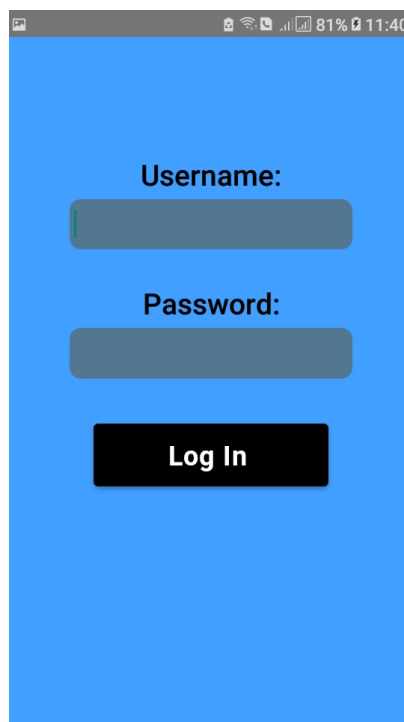


Figure 3.20: interface of Login.

After the student enters the user name and password, he can perform his own functions.

5.2.1 The functions of the Student

A Student can perform several functions after the login process, so we have briefly explained its functions in this part.

Attendance Registration:

After the teacher displays the QR code during the class, the student records the attendance through the attendance registration process, by pressing the Scan QR button and then directing the camera towards the QR code.

Figure 3.21 shows the attendance registration interface of the application.

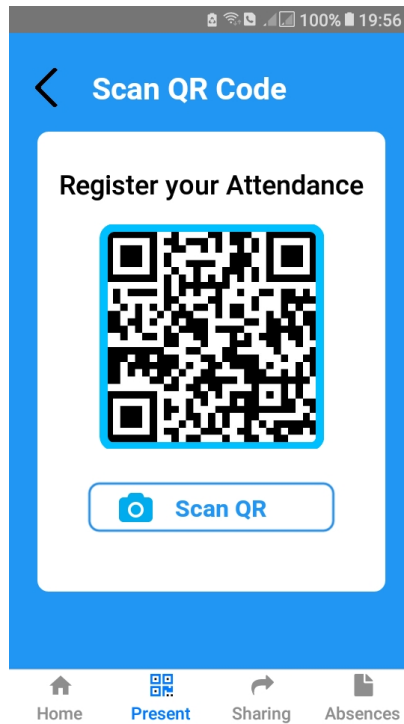


Figure 3.21: interface of Attendance Registration.

Request to Participation:

Through the process of requesting participation, the student can request participation through the active participation button, or cancel the participation request through the deactivate participation button in the event that there is a question that requires participation, and he can also know whether there is a question or not to participate in it, In the event that his participation is approved, he will receive a notification informing him that his participation has been accepted, and he must answer the question.

Figure 3.22 shows the request to participation interface of the application.

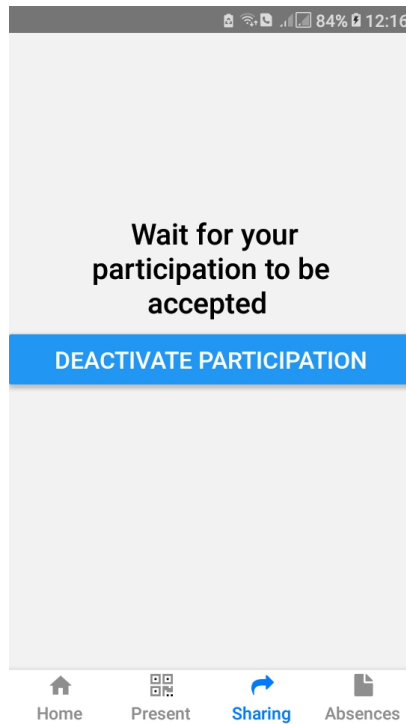


Figure 3.22: interface of Request to Participation.

Show Absences:

Through the process of showing absence, the student can directly see his absence and the modules in which he was absent with the date of absence (See Figure 3.23).



Figure 5.23: interface of Show Absences.

6 Test phase

After the implementation phase is completed, the testing phase comes. In this phase the software is tested, and it is verified that it works well. This phase allows us to evaluate the quality of the developed software. Thus, we consider the following tests: unit tests, integration tests, functional tests, configuration tests, and performance tests. And finally the installation test.

Test	Objective	Was the Test done?
Unit Test	Test each unit of the program individually	Yes
Integration Test	Testing the work of all program units with each other, i.e. testing the integration between program units	Yes
Functional Test	Verify that the software complies with the functional requirements	Yes
Configuration Test	Testing the program with a group of different programs and devices, i.e. testing the ability of the program to adapt to various programs and devices	No
Performance Test	Verify that the software meets specified performance requirements when running on servers	No
Installation Test	the software is verified during installation in the target environment, i.e. that it is installed correctly and works according to expectations	No

Table 4: Showing the purpose of the tests

From the table we see that some tests were not carried out.

The next table explains some of the justifications for which Because of which some tests were not done.

Test	Raison
Configuration Test	Due to time constraints, we were unable to provide the necessary hardware and software to run the program for a trial period
Performance Test	Due to time constraints, and no prior knowledge of how websites run on servers, we were unable to complete this test
Installation Test	We were unable to provide the necessary equipment, and due to time constraints, we also did not complete this test

Table 5: Explains why some tests are not done

7 Conclusion

This chapter has been devoted to the stage of implementing and realizing the application, while respecting the modeling that was developed in the previous chapter, At the beginning, we acquainted with the techniques, tools and programming languages that were used during the creation of the application, and at the end of the chapter we presented some of the most important interfaces that compose the application and clarify The way it works.

General Conclusion

This project of ours was implemented as part of the end-of-study project, and its aim was to design and develop an application that works on managing attendance and participation at the level of the Department of Computers and Information Technology, First, we briefly describe the UML modeling language and its diagrams, Then we extracted the requirements, created their UML diagrams (use case diagrams, sequence diagrams, class diagrams), and finally set up the environment and tools we need for the process of application development, code writing, and testing.

Where this application allows us to:

- Attendance registration
- Managing the participation process and recording student participation during the class
- A student can see his absences during the semester

That would be good If it is completed Incomplete features such as:

- The possibility of calculating the continuous assessment of students during the semester by collecting the attendance and participation point with the homework point or the interrogation point
- The possibility of adding the data (teachers, students, modules, groups) to the database and extracting them by the administrator and the possibility of assigning tasks (teachers, students, modules, groups)
- The possibility of the admin seeing the absences of students during the semester

We also believe that it is useful to modify some parts of the code for the mobile application to work on iOS devices instead of working on Android devices only.

References

- [1] M Fowler. *UML distilled: a brief guide to the standard object modeling language*. Addison-Wesley Professional, 2004.
- [2] <https://www.omg.org/spec/uml/about-uml/>, (consulted on 07/ 04/ 2022).
- [3] <https://www.lucidchart.com/blog/types-of-uml-diagrams>, (consulted on 08/ 04/ 2022).
- [4] <https://creately.com/blog/diagrams/uml-diagram-types-examples/>, (consulted on 13/ 04/ 2022).
- [5] <https://www.smartdraw.com/uml-diagram>, (consulted on 14/ 04/ 2022).
- [6] Michael Jesse Chonoles and James A Schardt. *UML 2 for Dummies*. John Wiley & Sons, 2011.
- [7] <https://pdf.wondershare.com/student-software/online-attendance-system.html>, (consulted on 14/ 06/ 2022).
- [8] <https://fntic.univ-ouargla.dz/index.php/ar/2013-05-05-09-31-44/2013-04-25-10-34-30.html>, (consulted on 10/ 05/ 2022).
- [9] E Kulak, D. Guiney. *Use cases: requirements in context*. Addison-Wesley, 2012.
- [10] Pascal Roques. *UML in practice: the art of modeling software systems demonstrated through worked examples and solutions*. John Wiley & Sons, 2006.
- [11] Doug Rosenberg and Kendall Scott. *Applying use case driven object modeling with UML: an anotated e-commerce example*. Addison-Wesley Professional, 2001.
- [12] Hassan Gomaa. *Software modeling and design: UML, use cases, patterns, and software architectures*. Cambridge University Press, 2011.
- [13] <https://nodejs.org/en/>, (consulted on 05/ 05/ 2022).
- [14] <https://code.visualstudio.com/>, (consulted on 06/ 05/ 2022).
- [15] <https://www.postman.com/>, (consulted on 06/ 05/ 2022).
- [16] <https://www.mongodb.com/>, (consulted on 10/ 05/ 2022).
- [17] <https://www.mongodb.com/products/compass>, (consulted on 06/ 05/ 2022).

[18] <https://www.mongodb.com/products/compass>, (consulted on 17/ 05/ 2022).

[19] <https://nestjs.com/>, (consulted on 18/ 05/ 2022).

[20] <https://developer.android.com/studio/intro>, (consulted on 07/ 05/ 2022).

[21] <https://www.geeksforgeeks.org/jdk-in-java/> , (consulted on 03/ 05/ 2022).