الجمهورية الشعبية الديمقراطية الجزائرية
People's Democratic Republic of Algeria
وزارة التعليم العالي و البحث العلمي
Ministry of Higher Education and Scientific Research
المدرسة العليا لأساتذة التعليم التكنولوجي سكيكدة
Higher School of Technological Teaching
ENSET-Skikda
قسم الرياضيات و الإعلام الآلي
Department of Mathematics and Computer Science

Dissertation

Field of Study: **Computer Science**

Theme

Design and Implementation of a Module under Odoo Framework for Managing Final Year Projects at ENSET-SKIKDA

Presented by
Ms. Maghlaoui Ikram
Ms. Zakkad Nesrine
Ms. Rjem Wiem

Jury:

Dr. Salah Halima	MCB	President
Dr. Sellami Samir	MCA	Supervisor
Ms. Taibi Sihem	Senior IT Engineer	Co-Supervisor
Dr. Abdoune Leila	MAA	Examiner
Dr. Guessoum Fatima	MCB	Examiner

Academic Year: 2025/2024

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

"In The Name of Allah,

The Most Gracious, The Most Merciful"

ACKNOWLEDGMENTS

First and foremost, we are deeply grateful to God Almighty for granting us the strength, patience and determination to undertake and complete this dissertation. Without His guidance and blessings, none of this would have been possible.

We would like to express our heartfelt gratitude to our parents for their unwavering encouragement and their moral and material support throughout these years.

We also extend our sincere appreciation to our supervisor, Dr. SELLAMI Samir and our co-supervisor, Ms. TAIBI Sihem, for their invaluable support, constructive feedback and continuous encouragement throughout the course of this work. Their guidance has been fundamental to the successful completion of this dissertation.

We are sincerely grateful to the President and the members of the jury for accepting to examine this work.

This dissertation stands as a testament to the collective support, patience and unwavering belief of all those who have stood by us throughout this journey. For this, we remain profoundly grateful.

Dedication

بسم الله الرحمان الرحيم

(قُلِ اعْمَلُوا فَسَيَرَى اللَّهُ عَمَلَكُمْ وَرَسُولُهُ وَالْمُؤْمِنُونَ)

To my beloved mother and father,
for their endless love, sacrifices and unwavering support throughout my life.

To my dear brother, Mohammad El-Amin,
for his steadfast support, encouragement and belief in me.

To my cherished sister, Ibtissem,
for her unconditional love and encouragement throughout this journey.
To her daughter, Leen, whose innocence and joy have always brought light to my days,
and to her husband, Aymen, for his kindness, support and positivity.

To my grandmother,
whose wisdom and prayers have accompanied me through every step.

To my dissertation team Nesrine and Wiem,
Thank you for your dedication, collaboration and unwavering belief in this project.
Your contributions have been instrumental in bringing it to life and I am truly proud of what we have accomplished together.

And to all the members of the Maghlaoui and Boudaoud families,
for their love, encouragement and belief in me.

Last but not least, I would like to thank myself
for never giving up, for persevering through every challenge,
and for believing in my own ability to accomplish this work.

M. IKRAM

Dedication

(وَآخِرُ دَعْوَاهُمْ أَنِ ٱلْحَمْدُ لِلَّهِ رَبِّ ٱلْعَٰلَمِينَ)

All praise is due to Allah, in a manner befitting His majestic countenance and His supreme sovereignty.

Through Him all good deeds are completed, and all goals are achieved. To Him belongs all gratitude, from beginning to end, for every step and every success.

To my dear father, my first supporter and constant pillar

may Allah reward you abundantly and protect you from all harm.

To my beloved mother, the source of tenderness and prayers

thank you for your endless patience and giving. I pray Allah grants you health and paradise.

To my dear sisters, Rayan and Dounia,

Your love and encouragement have been a tremendous blessing in my life. Your unwavering support has left a lasting mark on my heart.

To my dear brothers, Abdelmalek, Kheireddine, and Noufel,

You have always been my pride and strength. Thank you for your constant support.

To all my family

thank you for the love, prayers, and continued support.

To my Quran teacher, Nada,

Your prayers and kind words brought me peace. Thank you for your sincere guidance and encouragement.

To my precious friends: Nada, Hind, Nesrine, Jihan, Zineb, Baraa, Nour, and Asmaa,

You were my strength through every moment. Your presence was a great blessing in my life.

To my friends in the "Souqia Al-Rouh" Quran circle,

Your friendship was light and prayer. Thank you for your sincerity and sisterhood.

To Wiem and Ikram,

Thank you for the shared journey and mutual effort we were truly a complete team.

To my esteemed teachers,

To those who taught me a letter and guided me with care you have my deepest gratitude and respect.

And finally, to myself,

Thank you for your patience and perseverance.

Keep going forward with confidence and strength.

Z. NESRINE

Dedication

In the name of Allah, the Most Merciful.

Gratitude fills this page before any word of knowledge does. To those who stood by me, lifted me, inspired me, and believed in me this work carries your imprint in every line.

To my beloved mother, whose pure heart, unconditional love, and countless sacrifices shaped every step of my path. Even though you are no longer here, your strength and spirit remain with me, constantly guiding and inspiring me.

To my dear father, whose steady love and silent sacrifices have been the foundation of all my achievements. Your strength supported me through the most difficult moments, and your belief in me gave me the courage to keep going. I am deeply grateful more than words can express.

To my dear sisters, Nihed and Ibtissem, my lifelong companions. Through every moment of joy and difficulty, we grew side by side, supported one another through challenges, and shared dreams rooted in love and strength. This achievement is not mine alone it belongs to all of us.

To my nephews: Nazim, the gentle soul with eyes full of galaxies; Rassin, the fire bright, bold, and beautifully alive; and Lokman, the baby breeze soft and silent, a peace you can hold.

To my teachers, Oustad Badry and Oustada Ines, you did more than teach you inspired me. You believed in a version of me I had not yet seen. Thank you for your patience, your support, and your faith.

To the caring soul Chaima the heart that listens and the light that walks beside mine. Your friendship has been a refuge in every storm and a constant light in the darkest moments. More than a friend, you are chosen family a bond not defined by blood, but by loyalty, understanding, and unconditional support. Your presence has brought peace, strength, and a sense of belonging I will always cherish.

To my beloved circle of friends Lina, Malak, Hanadi, Wissal, Noussa, and Douaa who stayed even when things were difficult. Your presence meant a lot.

I also appreciate my colleagues, Kholoud and Afaf, whose kindness never went unnoticed. It was a privilege to share this path with them.

To my amazing dissertation teammates, Nesrine and Ikram this dissertation holds our shared journey, filled with effort, late nights, and determination. Thank you for being strong, committed, and always present. I'm proud we made it together.

To myself the one who stayed up too late, doubted too much, and still showed up.

This isn't a victory it's quiet survival.

I'm proud of the strength it took to keep going.

R. WIEM

ENGLISH ABSTRACT

Abstract

Managing final-year dissertation projects is a complex process that often suffers from inefficiencies, miscommunication and administrative burdens. This dissertation proposes the development of a web application using the Odoo framework to streamline the dissertation management process at the Higher School of Technological Education-Skikda (ENSET-Skikda). The web application module aims to provide a user-friendly platform for students, teachers and administrative staff, enhancing efficiency, transparency and collaboration throughout the dissertation lifecycle.

The methodology involves a systematic approach, including requirement analysis, system design, iterative development and validation. Key features of the system include: role-based access control and students groups formation, topics announcement, proposal submission and validation, dissertation assignments to groups and evaluation by assigned reviewers and final results publication. By leveraging the modular capabilities of the Odoo framework, the system will also ensure scalability and adaptability for future enhancements, while best practices in Agile development will guide the iterative refinement of the application.

Expected outcomes include reduced administrative overhead, improved communication among stakeholders and enhanced students and teachers satisfaction. This project represents a significant step toward modernizing dissertation management, fostering academic excellence and driving operational efficiency at ENSET-Skikda.

***Keywords*:** Odoo Framework; Dissertation Management System; Final-Year Projects; Web Application Development; Academic Workflow Automation; Higher Education Digitalization.

ARABIC ABSTRACT

الملخص

إدارة مشاريع التخرج للسنوات النهائية هي عملية معقدة غالبًا ما تعاني من عدم الكفاءة، وسوء التواصل، والأعباء الإدارية. تقترح هذه المذكرة تطوير تطبيق ويب باستخدام إطار العمل Odoo من أجل تحسين عملية إدارة مشاريع التخرج في المدرسة العليا للتعليم التكنولوجي بسكيكدة(ENSET-Skikda) . يهدف هذا التطبيق إلى توفيرمنصة سهلة الاستخدام للطلبة والأساتذة والطاقم الإداري، لتعزيز الكفاءة والشفافية والتعاون طوال دورة حياة مشروع التخرج.

المنهجية المعتمدة تتبع مقاربة تشمل تحليل المتطلبات، تصميم النظام، التطوير التكراري، والتحقق من صحة النتائج. من بين الخصائص الرئيسية للنظام: التحكم في الوصول حسب الأدوار و تشكيل افواج الطلبة،إعلان المواضيع وتقديم المقترحات، المصادقة على المواضيع ، تقييم المذكرات من قبل المقيمين المعينين، ونشر النتائج النهائية. ومن خلال الاستفادة من الطبيعة المعيارية لإطارالعمل Odoo ،سيضمن النظام أيضًا قابلية التوسع والتكيّف مع التحسينات المستقبلية، مع الالتزام بأفضل ممارسات التطوير التكراري المعتمدة في منهجية Agile .

تشمل النتائج المتوقعة تقليل العبء الإداري، وتحسين التواصل بين الأطراف المعنية، وتعزيز رضا الطلبة و الاساتذة. يمثل هذا المشروع خطوة مهمة نحو تحديث إدارة مشاريع التخرج، وتعزيز التميز الأكاديمي، وتحقيق الكفاءة التشغيلية داخل المدرسة العليا لاساتذة التعليم التكنولوجي بسكيكدة(ENSET-Skikda) .

الكلمات المفتاحية: إطار عمل أودو، نظام إدارة مذكرات التخرج، مشاريع نهاية الدراسة، تطوير تطبيقات الويب، أتمتة سير العمل الأكاديمي، رقمنة التعليم العالي.

CONTENTS

Acknowledgments ii

Abstract vi

List of Figures xi

List of Tables xiii

Listings xiii

General Introduction 1
- Introduction . . . 1
- Problem Statement . . . 1
- Objectives . . . 2
- Outline . . . 2

1 The State of Art (Fundamentals) 4
- 1.1 Introduction . . . 4
- 1.2 Study of the Current System . . . 4
  - 1.2.1 The Host Organizations ENSET-Skikda . . . 4
    - 1.2.1.1 School's Missions . . . 4
    - 1.2.1.2 Board of Directors . . . 5
    - 1.2.1.3 Educational Programs and Admission . . . 6
    - 1.2.1.4 Departments and Facilities . . . 6
    - 1.2.1.5 Strategic Location . . . 8
  - 1.2.2 Department of Mathematics and Computer Science . . . 8
    - 1.2.2.1 Mathematics Specialization . . . 8
    - 1.2.2.2 Computer Science Specialization . . . 8
    - 1.2.2.3 Role and Impact . . . 8
- 1.3 Challenges and Limitations . . . 9
- 1.4 Related Works . . . 14
  - 1.4.1 First Study 2012 . . . 14
  - 1.4.2 Second Study 2018 . . . 14
  - 1.4.3 Third Study 2021 . . . 15
- 1.5 Overview of Odoo Framework . . . 16

1.5.1 Definition of Odoo & History . . . 16
1.5.2 Key Features of the Odoo Framework . . . 16
1.5.3 Advantages of Odoo over Traditional Systems . . . 17
1.5.4 Comparing Odoo Framework to Traditional PHP Programming . . . 18
1.5.5 The Relevance of Odoo to the Proposed Platform . . . 19
1.6 Best Practices in Software Development . . . 19
1.6.1 What is Agile Methodology? . . . 19
1.6.2 The Agile Process: An Overview . . . 20
1.6.3 Why Agile is Ideal for our Project ? . . . 20
1.7 Conclusion . . . 21

2 Requirements Analysis & Design **23**
2.1 Introduction . . . 23
2.2 Analysis and Spesification . . . 23
2.2.1 Preliminary Analysis . . . 23
2.2.2 Understanding the Workflow . . . 25
2.2.2.1 Announcement for Student Group Formation . . . 25
2.2.2.2 Announcement to Teachers for Dissertation Topic Proposals 26
2.2.2.3 Monitoring the Progress of Work . . . 26
2.2.2.4 Dissertation Submission and Evaluation . . . 26
2.2.2.5 Dissertation Defense and Final Results . . . 26
2.2.3 Context Diagram . . . 29
2.2.4 Stakeholders Identification and Analysis . . . 29
2.2.5 Detailed Functional and Non-Functional Requirements . . . 31
2.2.5.1 Functional Requirements . . . 31
2.2.5.2 Non-Functional Requirements . . . 33
2.3 UML Diagrams Modeling and Design . . . 34
2.3.1 UML Diagrams Used . . . 34
2.3.1.1 Use Case Diagrams . . . 35
2.3.1.2 Activity Diagrams . . . 39
2.3.1.3 Automatic Topic Assignment Algorithm . . . 41
2.3.1.4 Sequence Diagrams . . . 42
2.3.1.5 Class Diagram . . . 48
2.3.1.6 Deployment Diagram . . . 50
2.4 Conclusion . . . 51

3 Implementation **53**
3.1 Introduction . . . 53
3.2 Technical Tools and Frameworks . . . 53
3.3 Odoo System Architecture . . . 55
3.3.1 Understanding Odoo Web Client Architecture . . . 57
3.3.2 Setting Up Odoo 16 Development Environment . . . 57
3.3.3 Odoo Module Structure . . . 59
3.3.4 Security and Access Control . . . 61
3.4 Workflow Implementation . . . 64
3.4.1 Data Model Implementation . . . 64
3.4.2 User Backend Interface Implementation (Views and Actions) . . . 66
3.4.3 Web Portal Implementation . . . 68

3.4.4 System Demonstration 68
3.5 Source Code and Repository 73
3.6 Deployment 73
3.7 Challenges Faced 75
3.8 Conclusion 75

General Conclusion and Perspectives **76**

List of Acronyms **78**

Bibliography **81**

LIST OF FIGURES

1.1 ENSET-Skikda Board of Directors. 6
1.2 Academic Departments of ENSET-Skikda. 7
1.3 Pedagogical Structures in ENSET-Skikda. 7
1.4 Sections of the Scientific Committee Report of the Mathematics and Computer Science Department. 11
1.5 Screenshot of the Published Dissertation Topics Listing without Detailed Descriptions. 12
1.6 Teacher's Facebook Comment Providing Clarification and Additional Details for his Proposed Dissertation Topic. 13
1.7 Agile Methodology Lifecycle. 20

2.1 Paper Form of Dissertation Project Topic Proposal. 25
2.2 Submission Form of Final-Year Dissertation. 27
2.3 Evaluation Defense Report of Final-Year Dissertation. 28
2.4 Context Diagram of the Dissertation Management System: Main Actors and Data Interactions. 29
2.5 Use Case Diagram of the Department Head. 37
2.6 Use Case Diagram of the Student. 37
2.7 Use Case Diagram of the Supervisor. 38
2.8 Use Case Diagram of the Scientific Committee. 38
2.9 Use Case Diagram of the Jury. 39
2.10 Activity Diagram of Topic Announcement and Validation Workflow. 40
2.11 Activity Diagram of Dissertation Review and Evaluation Workflow. 41
2.12 Automatic Topic Assignment Algorithm. 42
2.13 Sequence Diagram: User Authentication 43
2.14 Sequence Diagram: Student Group Formation 44
2.15 Sequence Diagram: Topic Proposal by Teacher (Supervisor) 45
2.16 Sequence Diagram: Validater Proposed Topics 46
2.17 Sequence Diagram: Review Dissertation Projects (Jury) 47
2.18 Sequence Diagram: View Final Dissertation Grade (Student) 48
2.19 Class Diagram. 49
2.20 Security-Related Data Model[1]. 50
2.21 Deployment Diagram of the Web-Based Dissertation Management System. . 51

3.1 The Three-Tier Architecture of Odoo. 55
3.2 The Odoo Architecture Layers. 56
3.3 The MVC (Model-View-Controller) Pattern. 56
3.4 Odoo Front-end/Website Back-end/Web-client Interaction Flow. [1] 57
3.5 Standard Directory Layout of an Odoo Module. 60
3.6 Successful installation of the PFE custom module. 60
3.7 Multi-Level Access Control in Odoo (Conceptual). 62
3.8 Implementation of Record Rules and Access Rights Using Security Groups. . 63
3.9 Screenshot: Login Page. 68
3.10 Screenshot: Scientific Committee Validates a Topic. 69
3.11 Screenshot: Department Head Assigns Dissertation Topics. 69
3.12 Screenshot: Website Main Landing Page. 70
3.13 Screenshot: Topic List in Frontend Website. 70
3.14 Screenshot: Portal Interface for Student. 71
3.15 Screenshot: Student Topic Choices. 71
3.16 Screenshot: Portal Interface for Teacher. 72
3.17 Screenshot: Teacher Topic Submission Form. 72
3.18 GitHub repository page of the *PFE Management ENSET-Skikda* module. . . 73
3.19 Screenshot: Proxmox VM (Ubuntu) Setup for Odoo. 74

LIST OF TABLES

1.1 The Distribution of Final-year Students and Supervisors by Specialization / Track in the Department of Mathematics and Computer Science. 9
1.2 Comparison of the 3 Studies Based on Criteria: User Authentication, Portal Features, Messaging, Notifications and Workflow Automation. 15
1.3 Comparison between Odoo Framework and Traditional PHP Programming. . 18

2.1 Stakeholder Roles and Goals. 30
2.2 Functional Requirements. 32
2.3 Non-Functional Requirements. 33
2.4 Description of UML Diagrams. 34
2.5 Presentation of Actors and Their Use Cases. 35

3.1 Installation Steps for Odoo 16 on Windows. 58

LISTINGS

3.1 Sample manifest .py File Snippet. 61
3.2 Access Control Example highlighting Teacher Group and ir.rule. 62
3.3 Access Control Example highlighting topic ir.model.access.csv. 63
3.4 Definition of the pfe.topic Model in Python. 65
3.5 Definition of the Topic Menus/ Actions /Views in XML. 67

GENERAL INTRODUCTION

Introduction

Managing final-year dissertation projects (PFE) is a critical academic process that significantly influences students' educational outcomes. However, traditional manual methods for managing these projects often face inefficiencies, miscommunication and administrative burdens, particularly at institutions like the Higher Normal School of Technological Education-Skikda (ENSET-Skikda). Manual dissertation management systems, heavily relying on paper-based processes and coordination, are prone to delays and errors.

Web technologies and ERP frameworks, such as Odoo [1], have revolutionized organizational workflows. Odoo, a modular and scalable framework, provides robust tools for managing complex processes, including customizable workflows, role-based user management and real-time communication. By adopting such technologies, institutions can modernize their dissertation management systems, ensuring efficiency, accountability and an improved user experience.

Problem Statement

At ENSET-Skikda, the manual approach used to manage dissertations poses significant challenges to efficiency, transparency and collaboration. Validating, assigning and tracking dissertation projects is handled manually by the department head and scientific committee, leading to increased workloads, time consumption and heightened errors.

Additionally, the absence of a centralized system results in frequent miscommunication, delays in topic announcements and groups formation and challenges in resolving conflicts over students' topic choices. These issues contribute to dissatisfaction among students and place unnecessary burdens on teachers and administrative staff. A streamlined digital solution is essential to address these problems and enhance the dissertation management process at ENSET-Skikda.

Objectives

The primary objectives of this project are to:

1. **Automate the Dissertation Workflow:** Streamline the end-to-end management of final-year projects through process automation.
2. **Equip Department Heads & Scientific Committee:** Provide a dedicated web-client backend interface for Department Heads and the Scientific Committee to manage topic validation, group formation, jury assignment and evaluation tracking.
3. **Empower Students & Teachers:** Deliver user-friendly web portals for students and teachers that facilitate topic selection, proposal submission, progress monitoring and feedback exchange.
4. **Ensure Efficiency & Quality Control:** Enhance administrative efficiency and enforce standardized quality controls at every stage of the dissertation lifecycle.

This project will comprehensively address key stages of the dissertation lifecycle, including: formation of student groups, announcement, topic proposal submission, validation by the scientific committee, topic assignments, dissertation submission and evaluation and final results publication. Implementing this web application using the Odoo framework will significantly improve communication and reduce administrative overhead at ENSET-Skikda. Students will benefit from a transparent and streamlined workflow, resulting in greater satisfaction and improved outcomes. Additionally, teachers and administrators will gain efficiency through automated processes, enabling them to focus on academic and strategic priorities.

Outline

Our work is organized as follows: The introduction provides an overview of the project, outlining its purpose, significance and objectives.

Chapter 1: Explores the foundational aspects of the study. It begins with an overview of the host organizations ENSET-Skikda and the department of mathematics and computer science, including a review of related existing dissertation management systems. Next, it introduces the Odoo framework and supporting technologies, followed by a discussion of best practices in software development, such as Agile methodologies.

Chapter 2: Focuses on the requirements analysis and system design. Detailing functional and non-functional requirements gathered from stakeholders. Design techniques, including UML diagrams and database schemas, are used to illustrate the proposed system design.

Chapter 3: Describes the implementation and results. It provides insights into the technical realization of the web application, system demonstrations and highlighting the outcomes achieved.

Conclusion: The dissertation concludes with a summary of the project's achievements, a discussion of its limitations and suggestions for future enhancements, including potential integration and scalability improvements.

The State of Art (Fundamentals)

CHAPTER 1
THE STATE OF ART (FUNDAMENTALS)

1.1 Introduction

This chapter lays the foundation for the development of a web application to manage final-year dissertation projects by examining relevant systems, technologies and methodologies. We begin with introducing the host organization ENSET-Skikda and the analysis of the current manual dissertation management process, identifying key inefficiencies and challenges. This is followed by a review of some existing dissertation management systems, highlighting their strengths, limitations and gaps that the proposed solution will address. The adoption of the Odoo framework is central to this project and we discuss its modular, collaborative and scalable capabilities in streamlining workflows. The chapter concludes with a focus on best practices in software development, particularly Agile methodologies, which emphasize iterative development and stakeholder involvement. These insights provide a strong foundation for designing an efficient and user-centric system.

1.2 Study of the Current System

1.2.1 The Host Organizations ENSET-Skikda

ENSET-Skikda is a scientific, cultural and professional institution established under Executive Decree N°09-254 dated August 10, 2009[2]. This decree authorized the creation of a school for technological education in Skikda. Located in the city of Skikda. The ENSET School employs a dedicated team of teachers, administrative staff and workers to fulfill its mission of providing high-quality teacher training for students in technological education[3].

1.2.1.1 School's Missions

The primary missions of ENSET-Skikda, as outlined in Article 7 of Executive Decree N°05/500[4], include:

- Contributing to the national effort in scientific research and technological development.
- Promoting sciences and technologies.

- Supporting national technical capacity.
- Enhancing the results of scientific research and disseminating scientific and technical information.
- Participating in the international scientific community by exchanging and enriching knowledge.

These missions emphasize the institution's role as a cornerstone for fostering innovation and advancing technological education in Algeria[1].

1.2.1.2 Board of Directors

The organizational structure of ENSET-Skikda is headed by a General Director. The General Director oversees several key areas, including:

- **General Secretary:** Supervises the Deputy Director of Personnel, Training and Cultural & Sports Activities and the Deputy Director of Finance and Resources.
- **Deputy Director in Charge of Information Systems, Communication and External Relations:** Oversees the Information and Communication Service, Monitoring, Statistics and Foresight Service and External Relations Service.
- **Deputy Director in Charge of Teaching, Diplomas and Continuing Education:** Manages Teaching, Internships and Evaluation Service, Continuing Education Service and Diplomas Service.
- **Deputy Director in Charge of Doctoral Training, Scientific Research, Innovation and Entrepreneurship Promotion:** Monitoring Research Activities, leads the Foreign Internships Service and Innovation and Entrepreneurship Promotion Service.
- **Library Director:** Responsible for the Bibliographic Research Service, Acquisition and Processing Service and Reception and Orientation Service.

The structure of the Board of Directors ensures diverse perspectives in decision-making and strategic planning (See Figure 1.1).

[1] https://www.enset-skikda.dz/

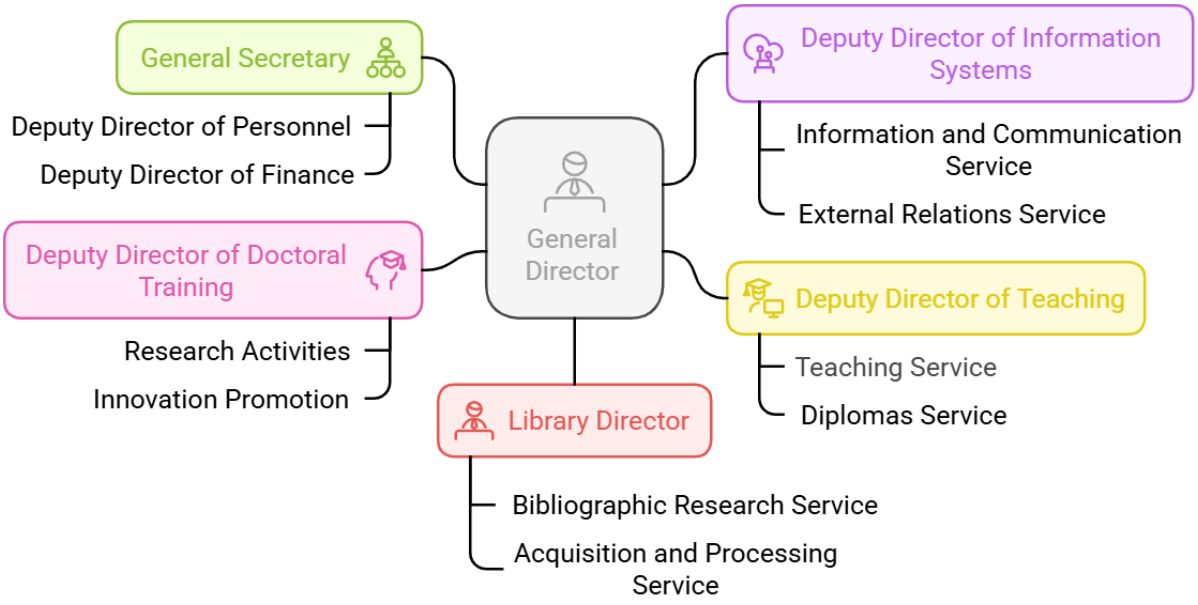


Figure 1.1: ENSET-Skikda Board of Directors.

1.2.1.3 Educational Programs and Admission

ENSET-Skikda is dedicated to preparing qualified educators for middle and secondary education. It offers 11 specialized training programs in scientific and technical disciplines, addressing the needs of the national education sector. Admission is highly competitive, requiring students to achieve a minimum overall average of 14/20 in the baccalaureate examination, with strong performance in core subjects like mathematics and sciences.[5]

The duration of study is four years for middle school educators (PEM) and five years for secondary school educators (PES). During this period, students sign a commitment contract with the Ministry of Education, guaranteeing employment upon successful completion of their program.

1.2.1.4 Departments and Facilities

The school is organized into four departments: the Department of Technology, the Department of Physics and Chemistry, the Department of Mathematics and Computer Science and the Department of Natural Sciences (See Figure 1.2).

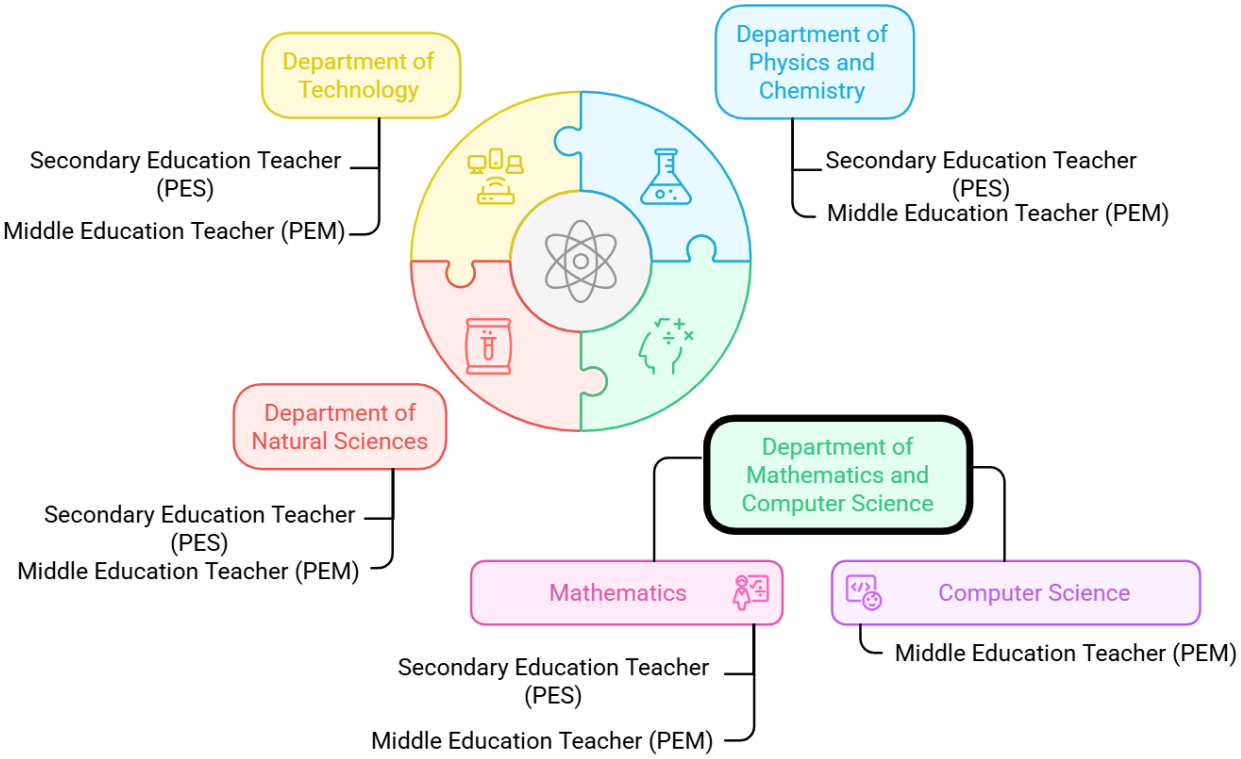


Figure 1.2: Academic Departments of ENSET-Skikda.

Each department offers specialized training for Secondary Education Teachers (PES) and Middle Education Teachers (PEM) [6].

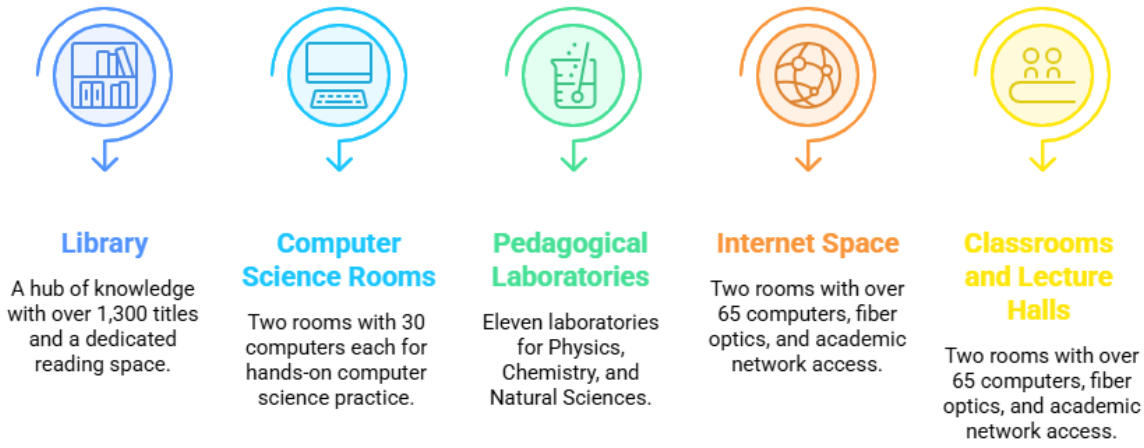


Figure 1.3: Pedagogical Structures in ENSET-Skikda.

ENSET-Skikda is equipped with pedagogical, administrative and recreational facilities to support the educational process (See Figure 1.3). These include classrooms, laboratories, a library, administrative offices, health services and recreational areas. The institution also organizes seminars, conferences and cultural activities to promote research and innovation, creating opportunities for faculty and students to engage and contribute to knowledge dissemination.

1.2.1.5 Strategic Location

The institution's strategic location enhances its accessibility and practical benefits[6]. It is situated near numerous educational institutions for student internships and essential facilities such as the girls' university residence, hospital, municipal headquarters and East-West Highway (1 km away). ENSET-Skikda is also well connected to nearby cities: 40 km from the provincial capital, 67 km from Annaba city and 77 km from Constantine city.[5]

1.2.2 Department of Mathematics and Computer Science

The Department of Mathematics and Computer Science plays a vital role in advancing education and research by offering comprehensive training programs in applied mathematics and computer science. The department has a diverse team of teachers specializing in fields such as web and data science, software engineering and mathematical modeling, ensuring high-quality education and mentorship [7]. This study primarily focuses on improving dissertation management within the Department of Mathematics and Computer Science as a pilot implementation. The proposed system will address existing challenges in project coordination, evaluation and communication within this department. In the future, the solution will be expanded and adapted to serve other departments within ENSET-Skikda, ensuring a unified and efficient dissertation management process across the institution.

1.2.2.1 Mathematics Specialization

The mathematics program focuses on preparing students for teaching careers in middle schools (PEM) and high schools (PES). This program emphasizes the study of mathematical theories and their practical applications. It aims to develop students' analytical and logical thinking skills, enabling them to solve complex problems and deliver knowledge effectively using modern pedagogical methods. This scholar year (2024-2025), approximately 43 final-year students are enrolled in the mathematics-PEM program and around 25 final-year students are pursuing the mathematics-PES program.

1.2.2.2 Computer Science Specialization

The computer science program trains students in programming, networking, data analysis and software design. The goal is to develop specialist teachers in IT and modern technology who can deliver technical knowledge across various sectors. The program emphasizes practical and applied skills, creativity and innovation to prepare students for the digital teaching job market. Approximately 32 final-year students are enrolled this scholar year (2024-2025) in the computer science PEM program. The program seeks to equip students with the ability to design software systems, analyze data and adapt to rapidly evolving technological landscapes.[7]

The department's distribution of final-year students and supervisors across its specializations and programs for the academic year 2024-2025 is shown in Table 1.1.

1.2.2.3 Role and Impact

With its dual focus on mathematics and computer science, the department plays a central role in advancing education and technology at ENSET-Skikda. It produces highly qualified graduates who contribute to scientific research, education and the digital industries and

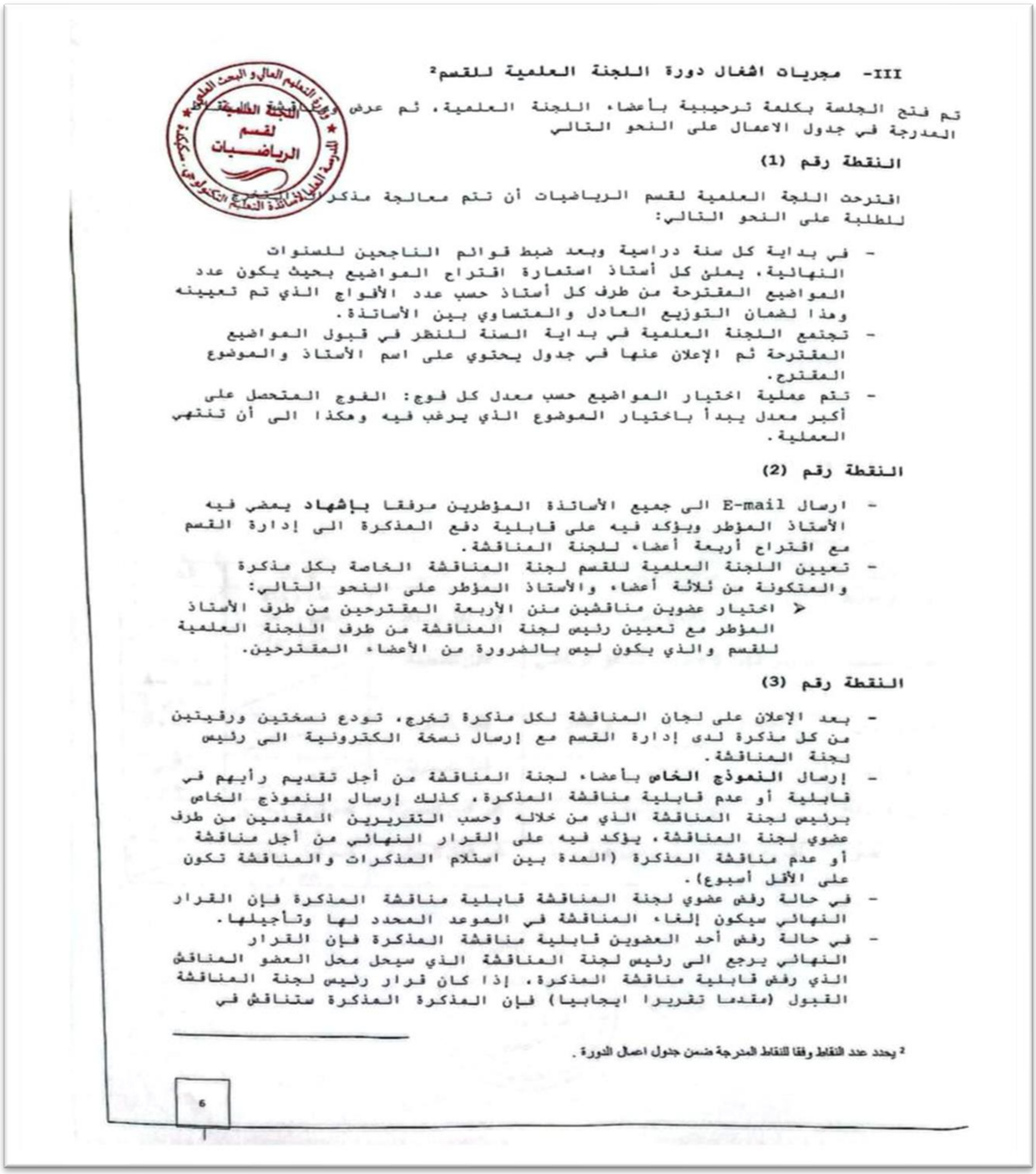
Table 1.1: The Distribution of Final-year Students and Supervisors by Specialization / Track in the Department of Mathematics and Computer Science.

Specialization	**Track**	**Number of Students**	**Number of Supervisors**
Mathematics	PEM	43	16
	PES	25	13
Computer Science	PEM	32	10
Total		100	39

job market. This alignment ensures that students gain expertise in both educational and technological fields, opening up broad career opportunities.

1.3 Challenges and Limitations

To better understand the specific challenges related to managing final-year projects (PFE) in our department, we conducted an interview with the department head[8] and the president of the scientific committee[9]. During these interviews, we were provided with a scientific committee report of the Mathematics and Computer Science department (See Figure 1.4).



III- مجريات اشغال دورة اللجنة العلمية للقسم[2]

تم فتح الجلسة بكلمة ترحيبية بأعضاء اللجنة العلمية، ثم عرض و مناقشة النقاط المدرجة في جدول الاعمال على النحو التالي

النقطة رقم (1)

اقترحت اللجة العلمية لقسم الرياضيات أن تتم معالجة مذكرات التخرج للطلبة على النحو التالي:

- في بداية كل سنة دراسية وبعد ضبط قوائم الناجحين للسنوات النهائية، يعلن كل أستاذ استمارة اقتراح المواضيع بحيث يكون عدد المواضيع المقترحة من طرف كل أستاذ حسب عدد الأفواج الذي تم تعيينه وهذا لضمان التوزيع العادل والمتساوي بين الأساتذة.
- تجتمع اللجنة العلمية في بداية السنة للنظر في قبول المواضيع المقترحة ثم الإعلان عنها في جدول يحتوي على اسم الأستاذ والموضوع المقترح.
- تتم عملية اختيار المواضيع حسب معدل كل فوج: الفوج المتحصل على أكبر معدل يبدأ باختيار الموضوع الذي يرغب فيه وهكذا الى أن تنتهي العملية.

النقطة رقم (2)

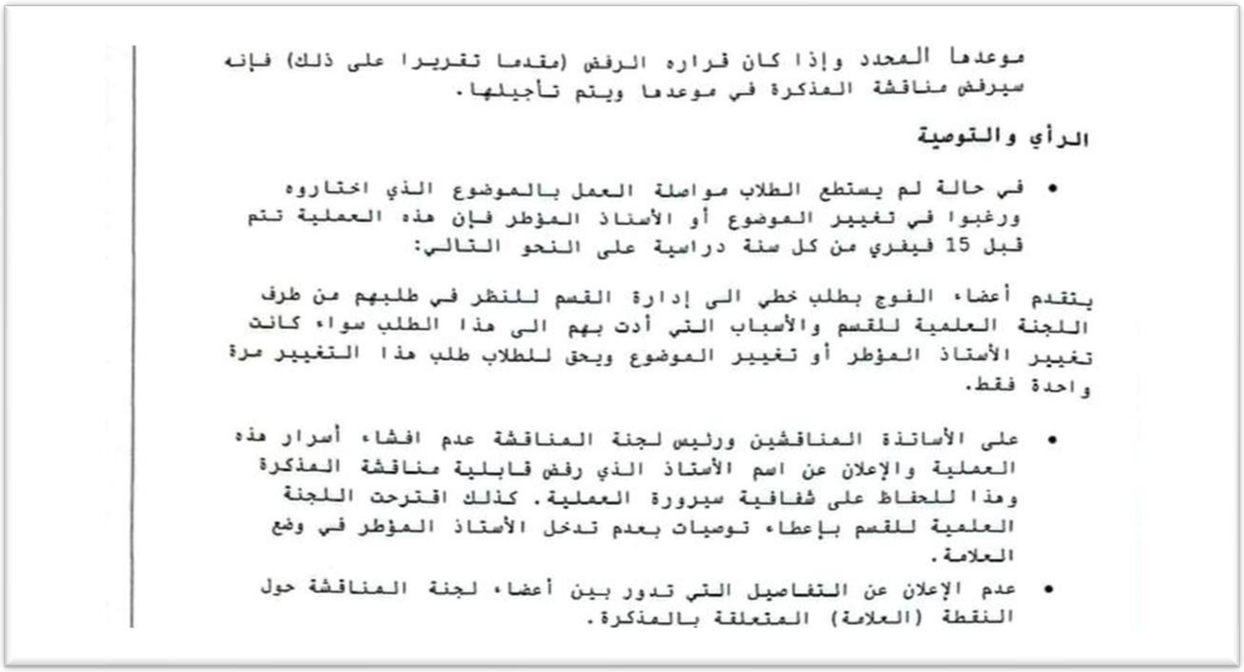
- ارسال E-mail الى جميع الأساتذة المؤطرين مرفقا **بإشهاد** يمضي فيه الأستاذ المؤطر ويؤكد فيه على قابلية دفع المذكرة الى إدارة القسم مع اقتراح أربعة أعضاء للجنة المناقشة.
- تعيين اللجنة العلمية للقسم لجنة المناقشة الخاصة بكل مذكرة والمتكونة من ثلاثة أعضاء والأستاذ المؤطر على النحو التالي:
  - اختيار عضوين مناقشين من الأربعة المقترحين من طرف الأستاذ المؤطر مع تعيين رئيس لجنة المناقشة من طرف اللجنة العلمية للقسم والذي يكون ليس بالضرورة من الأعضاء المقترحين.

النقطة رقم (3)

- بعد الإعلان على لجان المناقشة لكل مذكرة تخرج، تودع نسختين ورقيتين من كل مذكرة لدى إدارة القسم مع إرسال نسخة الكترونية الى رئيس لجنة المناقشة.
- إرسال **النموذج الخاص** بأعضاء لجنة المناقشة من أجل تقديم رأيهم في قابلية أو عدم قابلية مناقشة المذكرة، كذلك إرسال النموذج الخاص برئيس لجنة المناقشة الذي من خلاله وحسب التقريرين المقدمين من طرف عضوي لجنة المناقشة، يؤكد فيه على القرار النهائي من أجل مناقشة أو عدم مناقشة المذكرة (المدة بين استلام المذكرات والمناقشة تكون على الأقل أسبوع).
- في حالة رفض عضوي لجنة المناقشة قابلية مناقشة المذكرة فإن القرار النهائي سيكون إلغاء المناقشة في الموعد المحدد لها وتأجيلها.
- في حالة رفض أحد العضوين قابلية مناقشة المذكرة فإن القرار النهائي يرجع الى رئيس لجنة المناقشة الذي سيحل محل العضو المناقش الذي رفض قابلية مناقشة المذكرة، إذا كان قرار رئيس لجنة المناقشة القبول (مقدما تقريرا ايجابيا) فإن المذكرة المذكرة ستناقش في

[2] يحدد عدد النقاط وفقا للنقاط المدرجة ضمن جدول اعمال الدورة.

6



موعدها المحدد وإذا كان قراره الرفض (مقدما تقريرا على ذلك) فإنه سيرفض مناقشة المذكرة في موعدها ويتم تأجيلها.

الرأي والتوصية

- في حالة لم يستطع الطلاب مواصلة العمل بالموضوع الذي اختاروه ورغبوا في تغيير الموضوع أو الأستاذ المؤطر فإن هذه العملية تتم قبل 15 فيفري من كل سنة دراسية على النحو التالي:

يتقدم أعضاء الفوج بطلب خطي الى إدارة القسم للنظر في طلبهم من طرف اللجنة العلمية للقسم والأسباب التي أدت بهم الى هذا الطلب سواء كانت تغيير الأستاذ المؤطر أو تغيير الموضوع ويحق للطلاب طلب هذا التغيير مرة واحدة فقط.

- على الأساتذة المناقشين ورئيس لجنة المناقشة عدم افشاء أسرار هذه العملية والإعلان عن اسم الأستاذ الذي رفض قابلية مناقشة المذكرة وهذا للحفاظ على شفافية سيرورة العملية. كذلك اقترحت اللجنة العلمية للقسم بإعطاء توصيات بعدم تدخل الأستاذ المؤطر في وضع العلامة.
- عدم الإعلان عن التفاصيل التي تدور بين أعضاء لجنة المناقشة حول النقطة (العلامة) المتعلقة بالمذكرة.

Figure 1.4: Sections of the Scientific Committee Report of the Mathematics and Computer Science Department.

The key steps in the current system, as we have noticed, include:

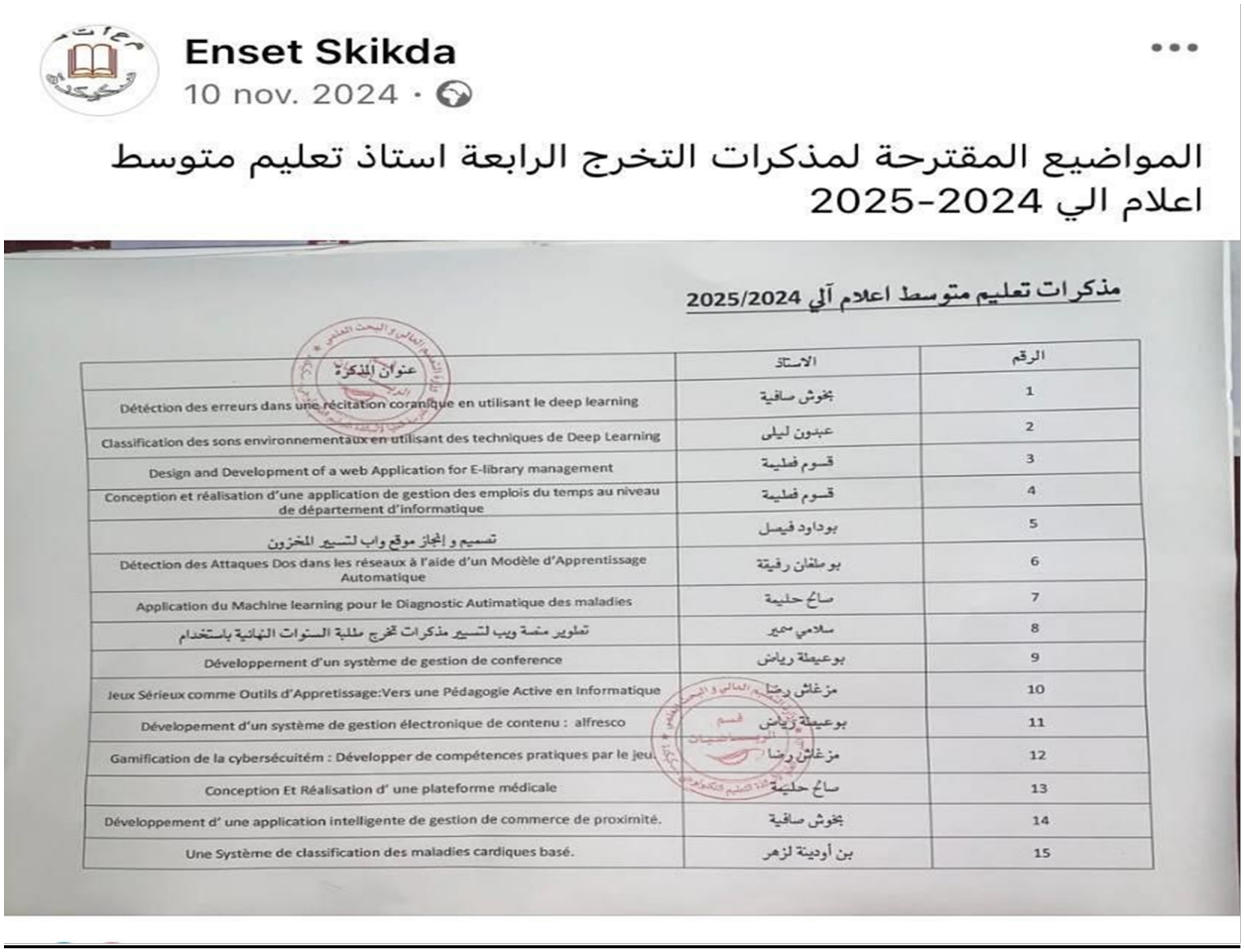
The announcement for students to start selecting dissertation topics is on the school's official Facebook page: It has been observed that the validated dissertation topics were announced through a post on the school's official Facebook page. While social media can be an effective communication tool, this method lacks the structure and reliability needed for such an important academic announcement. In our view, it would be more efficient and professional to publish first the list of approved dissertation topics on the school's official website or a dedicated portal. This approach would ensure that the information is delivered in a more organized and transparent manner, making it easier for students to access, review and reference the topics accurately. Additionally, using the official website or a dedicated portal would reinforce the seriousness and credibility required for managing academic affairs.

Manual Processing of Large Volumes of Information Leading to Errors: The heavy reliance on paper-based formatting poses a significant challenge to the accuracy and integrity of information. This manual approach not only increases the likelihood of errors but also results in inefficiencies, particularly when handling many students. Furthermore, the forms used often lack sufficient details to fully clarify project requirements, making them impractical. To enhance efficiency and ensure better information management, adopting a more structured and comprehensive digital solution is necessary.

The challenge of two equally ranked student groups selecting the same topic: Arises when multiple student groups with identical academic average marks compete for the same topic. This situation can create difficulties in fairly allocating topics, as the demand exceeds availability. To address this issue, a structured prioritization or redistribution mechanism (algorithm) may need to be implemented. However, this process can introduce challenges in ensuring transparency, fairness and student satisfaction in topic selection [10].

Issues with the Method of Publishing Dissertation Topics: The current method of publishing dissertation topics does not effectively serve its intended purpose, as it only includes the professor's name and the topic title without providing a detailed explanation.

This lack of clarity can lead to confusion among students when selecting their research topics.



الرقم	الاستاذ	عنوان المذكرة
1	بخوش صافية	Détection des erreurs dans une récitation coranique en utilisant le deep learning
2	عبدون ليلى	Classification des sons environnementaux en utilisant des techniques de Deep Learning
3	قسوم فطيمة	Design and Development of a web Application for E-library management
4	قسوم فطيمة	Conception et réalisation d'une application de gestion des emplois du temps au niveau de département d'informatique
5	بوداود فيصل	تصميم و إنجاز موقع واب لتسيير المخزون
6	بو ملفان رقية	Détection des Attaques Dos dans les réseaux à l'aide d'un Modèle d'Apprentissage Automatique
7	صائغ حليمة	Application du Machine learning pour le Diagnostic Autimatique des maladies
8	سلامي سمير	تطوير منصة ويب لتسيير مذكرات تخرج طلبة السنوات النهائية باستخدام
9	بوعيطة رياض	Développement d'un système de gestion de conference
10	مزغاش رضا	Jeux Sérieux comme Outils d'Appretissage:Vers une Pédagogie Active en Informatique
11	بوعيطة رياض	Dévelopement d'un système de gestion électronique de contenu : alfresco
12	مزغاش رضا	Gamification de la cybersécuitém : Développer de compétences pratiques par le jeu
13	صائغ حليمة	Conception Et Réalisation d' une plateforme médicale
14	بخوش صافية	Développement d' une application intelligente de gestion de commerce de proximité.
15	بن أودينة لزهر	Une Système de classification des maladies cardiques basé.

Figure 1.5: Screenshot of the Published Dissertation Topics Listing without Detailed Descriptions.

As illustrated in Figure1.5 a screenshot showing how dissertation topics are announced—an issue arise when one topic was published with an incomplete title. This forced the supervising professor to add a comment clarifying the full title and providing additional details to ensure students understood the topic properly (See Figure 1.6). Such an approach creates unnecessary ambiguity and complicates the decision-making process for students.

Figure 1.6: Teacher's Facebook Comment Providing Clarification and Additional Details for his Proposed Dissertation Topic.

To improve the clarity and accessibility of information, it would be more effective to pub-

lish the complete topic title along with a comprehensive description and necessary information (objectives, tools and references) from the outset. This would help students make informed decisions and prevent misunderstandings regarding the research scope and expectations.

Delays in Dissertation Selection Due to the Lack of a Defined Timeframe: Absence of a specified duration for students to choose a dissertation topic leads to unnecessary delays in the overall process. Without a clear time limit, students may take longer than necessary to make their decisions, postponing the start of their work and disrupting the workflow. Implementing a structured timeframe for topic selection would ensure a smoother and more efficient process, helping students manage their time effectively and avoid unnecessary setbacks.

Issues in Organizing Topic Selection Due to the Absence of a Predetermined Number of Groups: Before presenting dissertation topics for selection, it is crucial to first establish the student groups and determine their exact number. The number of available topics should match the number of groups to prevent selection issues. For instance, this year mathematics students faced a challenge when the number of proposed topics was insufficient, leaving some groups without a topic after the selection process. To resolve this, new topics were added; however, this led to another issue—students with higher averages grades, who had already selected topics, requested to change them in accordance with their priority rights. This created conflicts among students and caused unnecessary delays. A structured approach that ensures an equal number of topics per student groups from the outset would prevent such disruptions and improve the efficiency of the selection process.

1.4 Related Works

In this section, we present and analyze several previous studies related to the management of final-year projects, highlighting their strengths and limitations. Based on this analysis, we identify the weaknesses of these studies, thereby establishing the need for a more comprehensive and effective solution.

1.4.1 First Study 2012

The study "Conception et réalisation d'une application pour la gestion des soutenances", conducted by Souichi Leila at Mouloud Mammeri University of Tizi-Ouzou, aims to develop an automated system for managing final-year projects. This system is designed to accelerate administrative tasks, reduce manual errors and ensure data security. The implementation utilizes PHP, JavaScript, HTML and CSS, with MySQL serving as the database[11].

1.4.2 Second Study 2018

The study "Conception et réalisation d'une application web pour la gestion des projets de fin d'études" by Bensadi Aissa at Mohamed Boudiaf University aims to replace the traditional manual system with a web-based platform for project submission and selection. This approach enhances efficiency and simplifies interactions between students and professors. The system utilizes HTML, CSS, JavaScript and PHP, with PostgreSQL for data management and incorporates HTTP and SSL for secure communication[12].

1.4.3 Third Study 2021

The study "Conception et réalisation d'une application web pour la gestion de projets de fin d'études" by Meddah Mohamed Majid at Abdelhamid Ben Badis University of Mostaganem focuses on automating project management and allocation. It enables supervisors to propose topics and allows students to select or modify their choices online, thereby simplifying project distribution. This system is built using HTML, CSS, PHP, JavaScript and Apache as the web server, with MySQL as the database and it employs both PHP and Node.js for session management[13].

Table 1.2: Comparison of the 3 Studies Based on Criteria: User Authentication, Portal Features, Messaging, Notifications and Workflow Automation.

Criterion	**Studies**		
	Souichi Leila [11] (2011/2012)	**Meddah Mohamed Majid** [13] (2020/2021)	**Bensadi Aissa** [12] (2017/2018)
User authentication and access control	Basic login, no role-based access.	Role-based authentication (Admin, Teacher, Student)	Basic login
User portal	Portal only for admin	Fully integrated for students, teachers and administrators	Portal includes access for students and supervisors
Messaging system	Not available	Not available	Not available between students and supervisors
Notifications system	Not available	Available but limited to basic alerts (Registration confirmation, password reset)	Not available
Workflow Automation	Doesn't integrate all project stages	Partially automated process for project assignment	Automates only two steps of the workflow: addition of a final year project and its validation.

As shown in Table 1.2 Comparison of 3 Studies Based on Criteria: User Authentication, Portal Features, Messaging, Notifications and Workflow Automation, our analysis indicates that existing dissertation management systems exhibit notable limitations. These studies often suffer from an incomplete internal messaging system and workflows that fail to encompass all essential processes. Moreover, the absence of dedicated user portals restricts the personalization of the user experience, while the access control mechanisms are not sufficiently flexible to set precise permission levels. These deficiencies highlight the need for a more advanced solution. In contrast, the Odoo framework can be customaized to offers a robust suite of features—including comprehensive messaging, automated workflows, dedicated user portals and flexible access control—that make it an ideal choice for enhancing final-year project management.

1.5 Overview of Odoo Framework

1.5.1 Definition of Odoo & History

Odoo is an open-source enterprise resource planning (ERP) framework. One of the key features of Odoo is its friendly user interface, which makes it easy for users to navigate and use the system. The system also offers a high degree of customization, allowing users to tailor the system to their specific requirements. Odoo follows a multitier architecture, meaning that the presentation, business logic and data storage are separated [14].

The Odoo framework, previously known as OpenERP, was developed by a **Belgian company** in 2005. Initially, it was called **Tiny ERP**; three years later, the name was changed to **OpenERP** and in 2014, version 8 was released, leading to the final rebranding as **Odoo** (On Demand Open Objects) [15].

1.5.2 Key Features of the Odoo Framework

Odoo is renowned for its robust, modular design and comprehensive suite of features, which together provide significant benefits for organizations. Its key features and advantages include:

- **Modular Architecture:** Odoo's structure is modular, allowing users to install only the components relevant to their operations. This approach ensures flexibility and prevents unnecessary system complexity [16].
- **Deployment Options:** Odoo supports both cloud-based and on-premises deployment, giving institutions the flexibility to choose infrastructure options that match their technical and security needs.
- **Open-Source Foundation:** As an open-source platform, Odoo offers extensive customization capabilities. Organizations can tailor functionalities to meet specific requirements, making it suitable for diverse industries and academic institutions.
- **Integration Capabilities:** All modules in Odoo are inherently interconnected, enabling seamless data exchange across departments. This reduces redundancy, improves efficiency and facilitates consolidated reporting. For instance, an event created in the Event module can automatically update the Website module and generate announcements.
- **Inheritance Mechanism:** Odoo provides a powerful inheritance system that enables developers to extend or modify existing models, views and business logic without altering the original source code. Through both classical (Python-based) and prototypical (XML-based) inheritance, this feature promotes reusability, modularity and clean customization—ideal for adapting default modules to the specific needs of educational institutions.
- **Workflow Automation:** Odoo enables automation of repetitive and administrative tasks, streamlining critical processes such as dissertation submission, validation and evaluation.

- **Scalability:** The framework is scalable and adapts well to growing organizational needs, including increasing numbers of users, projects and data. This makes it ideal for educational institutions managing large student cohorts and research projects.
- **User Management:** The platform includes built-in role-based access control. It allows institutions to assign precise permissions to users such as students, teachers and scientific committee members, ensuring data confidentiality and operational clarity [17].
- **Customizable Website and Portals:** Odoo enables institutions to build and tailor dynamic websites and user portals using drag-and-drop tools and XML customization. This flexibility supports the creation of student and teacher portals, event pages and project dashboards, all aligned with the institution's branding and workflow requirements.

These features make Odoo an ideal choice for organizations looking to modernize and optimize their processes through a customizable, scalable and integrated digital solution.

1.5.3 Advantages of Odoo over Traditional Systems

Odoo stands out due it's user-friendly approach compared to other proprietary ERP systems or traditional programming choices such as PHP. It offers a fully integrated suite of modules covering HR, website builder, event manager and more, with both free open-source and affordable enterprise options. Unlike rigid proprietary systems, Odoo is highly customizable and scalable, suitable for small businesses and medium size enterprises. Its modern interface, strong community support and automation features make it a versatile and efficient business management solution.

Table 1.3: Comparison between Odoo Framework and Traditional PHP Programming.

Feature	**Odoo (Python-Based ERP)**	**Traditional PHP Programming**
Architecture	Modular, scalable, app-based structure	Custom-built, requires manual feature integration
Built-in Features	Pre-built business apps (event, HR, website, discuss, etc.)	Requires third-party integrations or custom development
Database & ORM	Uses PostgreSQL with Object-Relational Mapping (ORM)	Uses mainly MySQL with custom database structures
User Interface (UI)	Modern, responsive web UI with dashboards & drag-and-drop features	Basic HTML/PHP pages, requires extra work for interactivity
Automation	Built-in workflow automation	Requires custom scripts for automation.
Development & Customization	Uses Python with XML	Requires extensive manual coding in PHP
Security & Maintenance	Regular updates, role-based access control and multi-company support	Requires manual security patches and maintenance efforts. Manel coding of identity management.
Integration & Ecosystem	Seamless integration across apps with an extensive Odoo App Store	Requires additional work to connect different modules
Scalability	Highly scalable, suitable for small to large enterprises	Can become complex and difficult to scale without custom coding
Inheritance Support	Supports both classical (Python) and prototypical (XML) inheritance to extend and customize models and views without modifying core code	Lacks structured inheritance; customization often involves code duplication or altering base files, increasing maintenance risks
Overall Efficiency	Faster development, lower maintenance cost and better user experience	Time-consuming development with higher maintenance costs

1.5.4 Comparing Odoo Framework to Traditional PHP Programming

Table 1.3 compares the Odoo framework with traditional PHP-based development approaches to highlight their respective advantages and limitations in the context of academic software solutions.

1.5.5 The Relevance of Odoo to the Proposed Platform

Odoo offers a comprehensive suite of features designed to streamline the management of final-year dissertation projects. Its capabilities not only facilitate effective collaboration but also automate workflows and provide robust reporting functions, all of which contribute to a more efficient and transparent project management process.
- Internal Messaging System: Enables direct, real-time communication among student groups, supervisors and administrators through instant messaging and group discussions.
- Email Integration: Allows users to send and receive emails within Odoo, ensuring seamless communication without needing to switch platforms.
- Notifications & Alerts: Provides real-time updates on tasks, approvals and messages, ensuring that all stakeholders remain informed and can respond promptly.
Activity Scheduler: Lets users set reminders for meetings, calls, or follow-ups, ensuring that critical project milestones are not overlooked.
- Document Sharing: Facilitates the uploading and sharing of dissertation materials, drafts and feedback, thereby promoting effective collaboration.
- Task Management: Helps track the progress of the project by assigning tasks, setting deadlines and monitoring progress at each stage of the dissertation process.
- Workflow Automation: Odoo automates routine processes—such as sending deadline reminders, tracking work hours and coordinating communications via an integrated email gateway—thereby reducing manual workload and improving overall process efficiency.
- Evaluation & Reporting: The framework provides powerful reporting tools that allow for the generation of custom reports, monitoring of project status through customizable dashboards and data export for presentations or final submissions. These features support effective evaluation and continuous monitoring of key performance indicators throughout the dissertation lifecycle.

Together, these integrated capabilities and advantages make Odoo an ideal solution for managing final-year projects, offering a flexible, scalable and user-centric platform that meets the diverse needs of educational institutions.

1.6 Best Practices in Software Development

This section outlines the software development methodology adopted for our project, with a focus on Agile practices. Unlike traditional models such as Waterfall, which follow a rigid, sequential process, Agile's flexibility and iterative approach make it better suited for dynamic academic environments where continuous feedback is essential.

1.6.1 What is Agile Methodology?

Agile is a software development and project management methodology that emphasizes individuals and interactions, working solutions, customer collaboration and responsiveness to change over rigid processes and exhaustive documentation. It focuses on delivering small, incremental improvements in short cycles called sprints, encouraging frequent inspections and adaptations to meet evolving user needs and requirements effectively [18].

1.6.2 The Agile Process: An Overview

The Agile process involves several iterative steps that are repeated throughout the development lifecycle (See Figure 1.7):
- **Requirements:** Gather and prioritize customer requirements to establish the project's direction.
- **Planning:** Develop a comprehensive plan outlining the timeline for feature development and delivery in successive iterations.
- **Development:** Execute the software development in rapid, iterative cycles.
- **Testing:** Perform rigorous testing to ensure that the product meets quality standards and user expectations.
- **Deployment:** Roll out the software for end-user access.
- **Maintenance:** Continuously update and maintain the software to address new requirements and resolve issues [19].

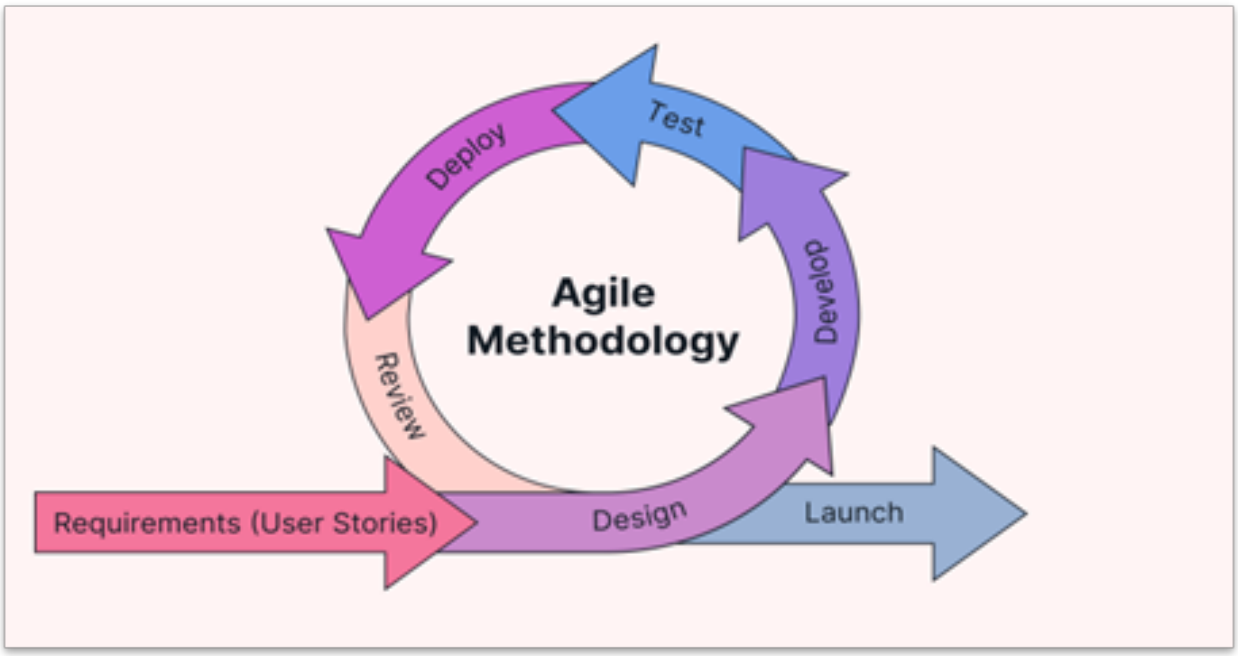


Figure 1.7: Agile Methodology Lifecycle.

1.6.3 Why Agile is Ideal for our Project ?

The development of our web platform for managing final-year dissertation projects using the Odoo framework greatly benefits from Agile methodology due to its adaptability and emphasis on continuous improvement. Here's why Agile is a great fit:

- **Flexibility:** Agile's iterative process allows the platform to quickly adapt to changing requirements and emerging challenges, ensuring alignment with evolving academic standards. Unlike Waterfall, which imposes a rigid step-by-step approach, Agile provides the flexibility needed in dynamic environments.

- **Frequent Feedback:** Agile promotes ongoing collaboration with key stakeholders—students, supervisors and department heads—allowing the platform to evolve in response to real-time feedback from those directly involved in the dissertation process.
- **Iterative Development:** By working in short sprints, Agile enables incremental feature development (e.g., groups formation, topics selection, evaluation), which facilitates quick adjustments and faster delivery of functional components.
- **Clear Progress Tracking:** Agile allows progress to be monitored in a transparent, step-by-step manner, making it easier to track the development of features and prioritize tasks effectively.
- **Enhanced Collaboration:** Agile fosters continuous communication among all stakeholders, including department heads, the scientific committee, teachers and students, ensuring that the platform meets everyone's needs throughout the dissertation process.

By adopting Agile practices, our project aims to deliver a highly adaptable, efficient and user-centric solution for managing final-year dissertations, ultimately ensuring a smoother workflow and better outcomes for all participants.

1.7 Conclusion

In this chapter, we provided a comprehensive overview of the current landscape in dissertation management, starting with an examination of the host organization, ENSET-Skikda and a detailed look at the Mathematics and Computer Science department. We reviewed the training tracks offered (PEM and PES) and analyzed existing dissertation management systems to identify their strengths and limitations. Next, we examined the Odoo framework, exploring its modular design, integration capabilities and collaborative tools that align with our project goals. Additionally, we discussed the Agile methodology and its suitability for developing a dynamic, responsive platform that can adapt to evolving user needs.

These insights highlight the gaps in current practices and underscore the need for a comprehensive, digital solution to manage final-year projects efficiently. In the next chapter, we will present the requirements analysis and design, which will establish the foundation for the technical implementation of our proposed platform.

Requirements Analysis & Design

CHAPTER 2

REQUIREMENTS ANALYSIS & DESIGN

2.1 Introduction

This chapter aims to translate the needs identified during the preliminary analysis into a clear set of technical and functional specifications, while establishing the foundations of the system design.We begin with a detailed analysis of the requirements, emphasizing stakeholder identification, feasibility study and an understanding of the current workflow. This phase is crucial for formalizing both functional and non-functional requirements, ensuring that the solution optimally addresses the user's needs and the organization's objectives. Next, we move on to the design phase, leveraging UML through the following diagrams: the use-case diagram, sequence diagram, activity diagram, class diagram, deployment diagram and automatic topic assignment algorithm flowchart. These diagrams graphically represent the structure, interactions and operational flow of the system, facilitating clear communication and understanding among the various project stakeholders.Together, these tools aim to establish a robust, modular and scalable architecture capable of optimizing the management processes of final-year dissertation projects. Thus, this chapter provides a comprehensive methodological framework that links the analysis of needs to detailed design, ensuring a coherent and structured transition to the development phase.

2.2 Analysis and Spesification

2.2.1 Preliminary Analysis

The development process of an information system must go through several rigorous stages to ensure that the final product meets the specifications and constraints defined in the established requirements, following a clear and structured approach. Adopting a goal-oriented approach to requirements engineering offers several advantages:

- Requirements frequently evolve, making it preferable to link them directly to the organization's objectives.
- This approach is promising and merits further exploration.
- Few similar studies have been conducted, according to our research.

- Conducting a study in this area could inspire interest among other students and introduce them to this field of research.

During this phase, we aimed to understand the context of our study by focusing on the management of final-year dissertation projects within the Mathematics and Computer Science Department at ENSET Skikda.Our primary objective was to gather all relevant information and data related to the dissertation project lifecycle—from the announcement of student group formation and the proposal of topics by teachers to the final defense, the publication of defense results and the submission of corrected dissertations.

This analysis aimed to clarify existing functionalities and elicit business, system and user requirements in order to better understand the needs and processes in place. This work allowed us to identify key workflow steps, such as coordination between the department head and students for topic selection, supervisor assignment, project tracking and final evaluation.

To conduct this analysis, we used the following requirement elicitation techniques throughout our study:

- **Startup Questions:** We gathered data by asking questions in various ways to understand requirements from different perspectives.Non-technical individuals often have different viewpoints from IT professionals.Questions were directed at various stakeholders involved in graduation project management, as well as final-year students from both the current and previous years.

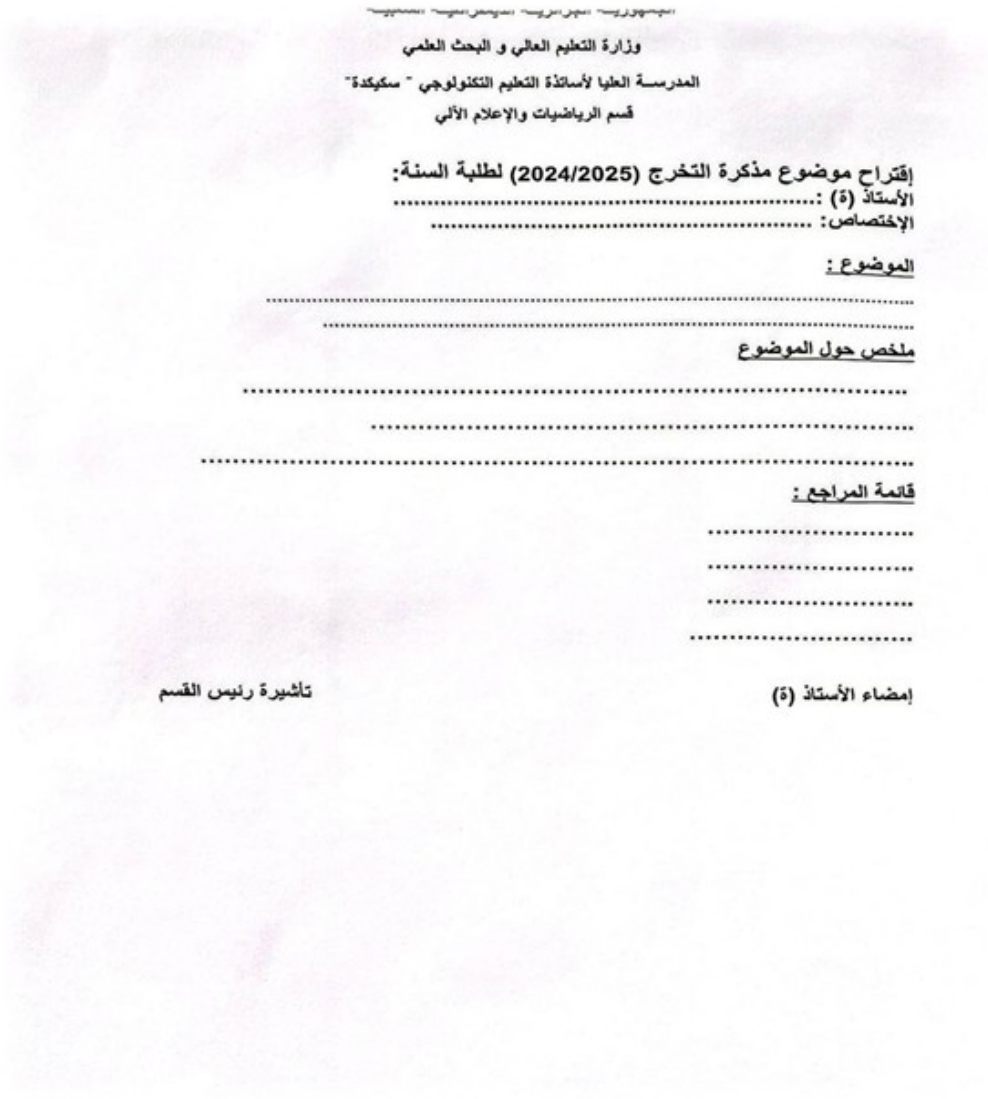
- **Document Analysis:** We consulted archives and regulatory documents. For example, we reviewed part of the scientific committee's report (See Figure 1.4) and the paper form provided to supervisors for proposing (See Figure 2.1). Additionally, we examined the dissertation project management workflow to better understand the various stages of the process from student group formation and topic proposals to final evaluation and result announcements.

- **Meetings:** We organized meetings with the scientific committee head to discuss the fairness of group average scores in order to facilitate project topic selection.

- **Interviews:** We conducted interviews with direct and indirect questioning to accurately capture functional, non-functional and evaluation criteria requirements. Interviews were also held with multiple stakeholders in the administration department to gather diverse perspectives and encourage discussions with other parties, including department head and the head of the scientific committee.

- **Observation:** We observed how tasks are performed within the department.

- **Prototyping:** The objective was to demonstrate to stakeholders whether the data collected by the application meets their needs.



وزارة التعليم العالي و البحث العلمي

المدرسة العليا لأساتذة التعليم التكنولوجي " سكيكدة"

قسم الرياضيات والإعلام الآلي

إقتراح موضوع مذكرة التخرج (2024/2025) لطلبة السنة:

الأستاذ (ة) :.....

الإختصاص: .....

الموضوع :

.....

.....

ملخص حول الموضوع

.....

.....

.....

قائمة المراجع :

.....

.....

.....

.....

إمضاء الأستاذ (ة)

تأشيرة رئيس القسم

Figure 2.1: Paper Form of Dissertation Project Topic Proposal.

2.2.2 Understanding the Workflow

In the implementation of the dissertation management process for final-year students, the Department Head plays a crucial role in overseeing and organizing the process. He ensures administrative coordination and overall organization while supervising the various stages of thesis management in a structured and coordinated manner, which includes the following steps:

2.2.2.1 Announcement for Student Group Formation

- At the beginning of each academic year, students are informed about the group formation process. This announcement provides essential details regarding how groups should be created and the timeline condition.
- Groups must consist of either a single student, two students or a maximum of three students.

2.2.2.2 Announcement to Teachers for Dissertation Topic Proposals

- Once the number of student groups is determined, teachers are invited to propose dissertations topics.
- A balance is maintained between the number of student's groups and proposed topics.
- After topics are validated, students are assigned to projects and supervisors accordingly.

2.2.2.3 Monitoring the Progress of Work

- Throughout the dissertation preparation period, students' progress is monitored to ensure that each phase of their work is conducted properly, including state of art study, data collection, design and implementation.

2.2.2.4 Dissertation Submission and Evaluation

- Once the dissertation is completed, students are required to submit their work using a designated submission form (See Figure2.2). This form ensures that all necessary information is provided and that the submission process is standardized for evaluation and record-keeping purposes.
- The submitted dissertation is reviewed to ensure compliance with academic standards before the final evaluation.

2.2.2.5 Dissertation Defense and Final Results

- After submission, students participate in a defense session, where they present their work against a specialized jury.
- The final evaluation is based on the quality of the dissertation and the student's defense performance.
- At the end of the defense, the final results are announced and grades are assigned based on predefined academic criteria (See Figure2.3).
- Following the defense, students must correct any mistakes identified by the jury in their dissertations and submit a revised version.

الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
المدرسة العليا لأساتذة التعليم التكنولوجي سكيكدة

قسم الرياضيات و الإعلام الآلي

سكيكدة في:

.....:

محضر إيداع مذكرة التخرج

يسمح الأستاذ المؤطر :.....

لطلبة:

1- .....
2- .....
3- .....

عنوان المذكرة المنجزة : .....

التخصص: .....

بإيداع مذكرة التخرج لسنة الجامعية 2024/2023 على مستوى المكتبة للحفظ بعد الإطلاع و تصحيح النقائص
رفقة قرص مضغوط.

إمضاء المؤطر

رئيس القسم .

Figure 2.2: Submission Form of Final-Year Dissertation.

The People's Democratic Republic of Algeria
وزارة التعليم العالي و البحث العلمي
Ministry of Higher Education and Scientific Research
المدرسة العليا لاساتذة التعليم التكنولوجي-سكيكدة
Higher Normal School of Technological Teaching
قسم الرياضيات والاعلام الالي
Department of Mathematics & Computer Sciences

محضر مناقشة مذكرة تخرج استاذ
تعليم ثانوي-متوسط

في يوم: السبت بتاريخ: 22 من شهر: جوان من سنة: 2024 ناقش(ت) علنيا الطالب(ة) (الطلبة)

- [redacted]
- [redacted]
- [redacted]

مذكرة التخرج الموسومة بـ: تحويل لابلاس و بعض تطبيقاته

تخصص: رياضيات

أمام لجنة المناقشة المعينة من طرف اللجنة العلمية للقسم في اجتماعها المنعقد في 05 جوان 2024.

وقد تشكلت اللجنة من:

الرقم	الاسم و اللقب	الرتبة	مؤسسة الانتماء	الصفة	التوقيع
01	[redacted]	MAA	ENSET Skikda	رئيسا	
02	[redacted]	MCB	-	مشرفا	
03	[redacted]	MCB	-	مناقشا	
04	[redacted]	MCB	-	مناقشا	

و بعد المناقشة العلنية و المداولة القانونية قررت اللجنة منح الطالب(ة) (الطلبة المناقشين):

العلامة: 15,50

تقدير: مشرف في دورة: جوان من سنة: 2024

ملاحظة: يؤخذ بعين الاعتبار في التنقيط، المخطوط و العرض الشفهي، و الإجابة عن الأسئلة و تمنح التقديرات الآتية: ممتاز (الدرجة أ)، جيد جدا (الدرجة ب)، جيد (الدرجة ج) ، قريب من الجيد (الدرجة د)، مقبول (الدرجة هـ).

مصادقة رئيس القسم

Figure 2.3: Evaluation Defense Report of Final-Year Dissertation.

2.2.3 Context Diagram

A context diagram is a high-level visual representation that shows how a software system interacts with external entities such as users, other systems, or hardware components. It helps stakeholders quickly understand where the system fits within its broader environment. In a typical context diagram, the system being developed is placed at the center, while all external actors and systems that interact with it are positioned around it. These interactions are represented as labeled arrows that indicate the direction and nature of data or communication flow[20].

In our context diagram, illustrated in Figure 2.4, we focus on the primary internal actors relevant to the management of final-year dissertation projects—namely department heads,students, supervisors, the scientific committee, mail server and jury members. Each actor is connected to the system via labeled interactions that represent specific actions, requests, or data exchanges. For instance, students submit their dissertations to the system, while supervisors provide feedback and propose topics.

To read the diagram, start by identifying each actor and following the arrows that link them to the central system. Each arrow is labeled with a step number and brief description of the action, making it easy to trace the workflow and understand the system's inputs and outputs.

This diagram serves as an overview of the operational boundaries of the proposed system and the responsibilities of each actor in the dissertation process.

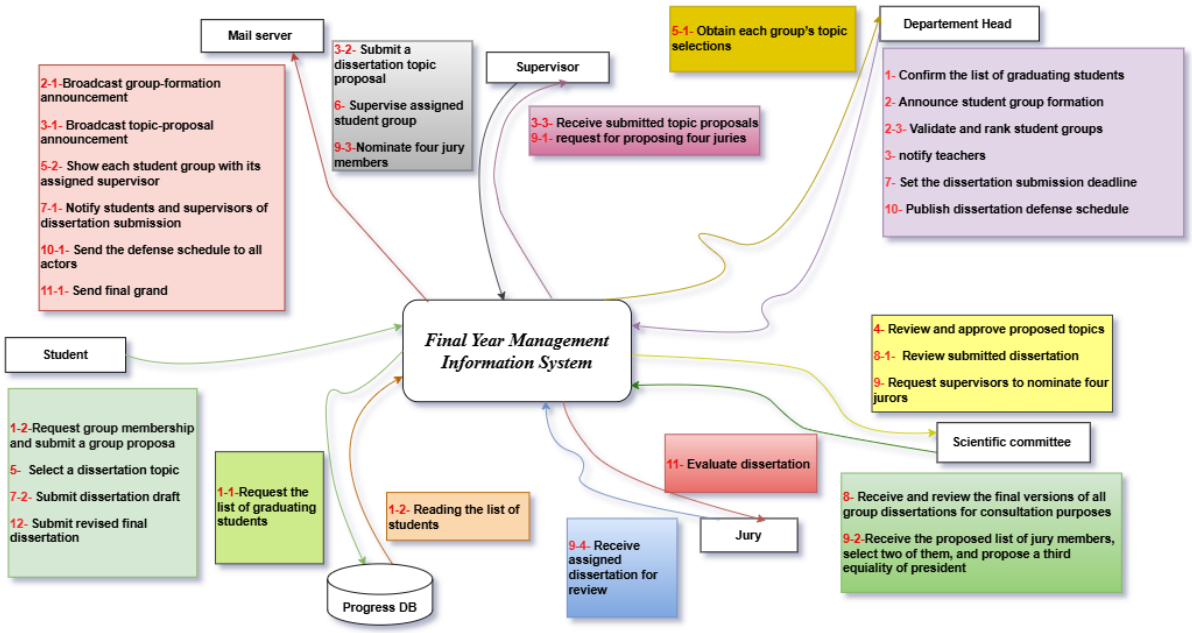


Figure 2.4: Context Diagram of the Dissertation Management System: Main Actors and Data Interactions.

2.2.4 Stakeholders Identification and Analysis

Stakeholders involved in the dissertation management system can be categorized into primary and secondary stakeholders. Table 2.1 below provides a comprehensive overview of their roles and goals.

Table 2.1: Stakeholder Roles and Goals.

Stakeholder	**Role**	**Goals**
Primary Stakeholders		
Students	Direct participants conducting projects and submitting dissertations.	• Transparency: Clear guidelines, assessment criteria and deadlines. • Support: Access to supervisors, resources (e.g., libraries, labs) and feedback.
Teachers / Supervisors	Mentors guiding students through project design, execution and feedback.	• Resource Availability: Tools to monitor progress. • Academic Integrity: Ensuring originality and rigor in student work.
Department Head	Oversees program quality and alignment with departmental objectives.	• Compliance: Adherence to institutional policies and accreditation requirements. • Reputation: High-quality outputs that enhance the department's academic standing.
Scientific Committee	Sets academic guidelines, approves dissertation topics, ensures alignment with research standards.	• Quality Assurance: Validation of project methodologies and outcomes. • Ethical Compliance: Approval of topics. • Innovation: Encouraging cutting-edge projects that align with institutional priorities.

Stakeholder Category	**Role**	**Goals**
Juries	Evaluate dissertations for rigor, originality and presentation quality.	• Clarity: Well-defined rubrics and submission formats. • Timeliness: Access to materials before defense dates. • Objectivity: Provide quality review.
Secondary Stakeholders		
System Administrators	Maintain digital platforms (e.g., submission portals, databases, access rights).	• Efficiency: Automated workflows for submission, grading and feedback. • Security: Protection of student data and intellectual property. • Uptime: Minimal system downtime during critical periods.
Institution Management	Strategic oversight of resources and institutional reputation.	• Resource Optimization: Cost-effective use of staff, tools and infrastructure. • Accountability: Data tracking for accreditation and funding purposes.

2.2.5 Detailed Functional and Non-Functional Requirements

2.2.5.1 Functional Requirements

Functional requirements define what the system must do to support end-to-end dissertation processes for students, teachers and department head.Table 2.2 presents a structured breakdown of the core functionalities.

Table 2.2: Functional Requirements.

Category	**Requirements**
Student Group Formation Management	**Automated Announcements:** • Send automated notifications to students at the start of the academic year (via email) • Provide backend management interface for department head and portal for students **Group Validation:** • Enforce maximum group size (3 students) through system-level validation **Deadline Management:** • Define a fixed period for group formation and automatically lock the system afterward
Dissertation Topic Proposal Management	**Teachers Proposal Submission:** • Teachers and supervisors are provided with a dedicated portal to propose dissertation topics (title, description, prerequisites, required expertise, tools and references) **Topic Validation Workflow:** • Implement a multi-step approval process: Teachers → Department Head → Scientific Committee • Send automated alerts for topic approval/rejection **Project Assignment:** • Algorithm to balance student numbers and average grads with available supervisors • Student portal to select/rank preferred topics (priority-based assignment)
Progress Monitoring	**Progress Reporting:** • Provide supervisors and department head with dashboards to view progress, add feedback and flag issues **Communication Tools:** • Built-in messaging system for student-supervisor communication (comments/chat)
Dissertation Submission and Evaluation	**Submission Portal:** • Upload final dissertation in PDF format with validation checks (file size, structure) **Pre-Defense Review:** • Supervisors and department heads review the dissertation for academic standards (clarity, methodology, citations) • Reject submissions failing validation criteria **Evaluation Workflow:** • Route approved dissertations to the jury members for review

Category	Requirements
Dissertation Defense and Final Results	**Defense Scheduling:** • Publish and send of defense sessions with notifications (date, time) **Performance Evaluation:** • Jury members input grades directly into the system based on predefined criteria (research quality, presentation, Q&A) **Result Publication:** • Publish results on student portal with an option to request reviews/appeals

2.2.5.2 Non-Functional Requirements

Non-functional requirements (NFRs) define how the system should perform rather than what it does. They ensure usability, scalability and security.Table 2.3 lists the key non-functional requirements (NFRs) for the dissertation management system.

Table 2.3: Non-Functional Requirements.

Category	Requirements
Usability	**User Experience:** • Intuitive UI with role-specific portal (students see deadlines, supervisors see progress reports) **Ease of Use:** • Minimal learning curve with user-friendly design • Consistent navigation across all modules **Multilingual Support:** • Support for English and Arabic interfaces
Performance and Scalability	**Concurrent Users:** • Support high number of concurrent users during peak times (e.g., submission deadlines) **Mobile Responsiveness:** • Ensure seamless responsive access on mobile devices • Built on Odoo (uses Bootstrap framework) for full responsiveness across screen sizes
Security	**Role-Based Access Control (RBAC):** • Authorized users can access/modify data based on roles (students submit, supervisors review, department head manages) **Data Encryption:** • Sensitive dissertation data (submissions, feedback, grades) must be securely encrypted

2.3 UML Diagrams Modeling and Design

Unified Modeling Language (UML) is a standardized visual modeling language employed in the analysis and design of software systems. While it is referred to as a "language," it should not be confused with natural or programming languages. Rather, UML is a modeling and specification language that provides a well-defined set of rules and graphical notations used to represent software architecture and behavior [21].

UML is widely recognized and adopted in the software engineering industry for its ability to describe, visualize, construct and document the components of a software system. Its standardized diagrams facilitate communication among stakeholders and serve as a blueprint during various stages of the software development lifecycle [22].

2.3.1 UML Diagrams Used

Table 2.4 presents a description of the UML diagrams employed in this study. These diagrams were selected based on their relevance to the modeling needs of the proposed system and their effectiveness in representing structural and behavioral aspects of the software architecture.

Table 2.4: Description of UML Diagrams.

Diagram	**Objectives**	**Type**
Use Case Diagram	Use cases are used to specify the behaviour of some entity such as a system or subsystem, known as the subject. Use cases do not specify the detail of how that behaviour is carried out.[21]	Functional
Activity Diagram	It allows modeling the triggering of events based on the system's states, representing parallel behaviors and describing the workflow and the internal control flow of operations.[23]	Behavioral
Flowchart Diagram	Flowcharts are used to represent the logic and control flow of algorithms. In our system, it illustrates the automatic topic assignment process based on student group averages and ranked preferences.	Behavioral
Sequence Diagram	Sequence diagrams are used to model the interaction between object instances by showing the sequence of messages that are exchanged by the objects.[21]	Dynamic
Class Diagram	It helps everyone involved in a project—like developers and designers—understand how the system is organized and how its components interact and also represents the logical structure of the database tables and relations. [24]	Static
Deployment Diagram	Deployment diagram shows execution architecture of systems that represent the assignment (deployment) of software artifacts to deployment targets.[25]	Physical

2.3.1.1 Use Case Diagrams

The following table 2.5 presents each actor along with their respective use cases:

Table 2.5: Presentation of Actors and Their Use Cases.

Actor	Use Cases
Department Head (See Figure2.5)	• Authenticate • Access to The backend web client • Announce groups formation • Receive the student's groups • Formation the groups • Send the proposed topics for review • Publish the topics • Set deadline for submitting the dissertation • Publish the schedule of defenses • Publish the defenses grades
Student (See Figure 2.6)	• Authenticate • Access to the student portal • Consult group/members • Consult the topic/dissertation • Choose a topic • Send a message • Submit dissertations • Receive the final defense's grade

Actor	Use Cases
Supervisor (See Figure2.7)	• Authenticate • Access to the teacher portal • Propose the topic (title, description, references, tools) • Give feedback and percentage on student's progress or completion • Suggest four juries
Scientific Committee (See Figure2.8)	• Authenticate • Access to The backend web client • Consult the proposed topics • Contact the supervisors for choosing juries • Select three juries • Contact juries for reviewing the dissertations
Jury (See Figure2.9)	• Authenticate • Access to The backend web client • Receive the dissertation • Consult the dissertations (accept or reject) • Review the dissertation and provide feedback

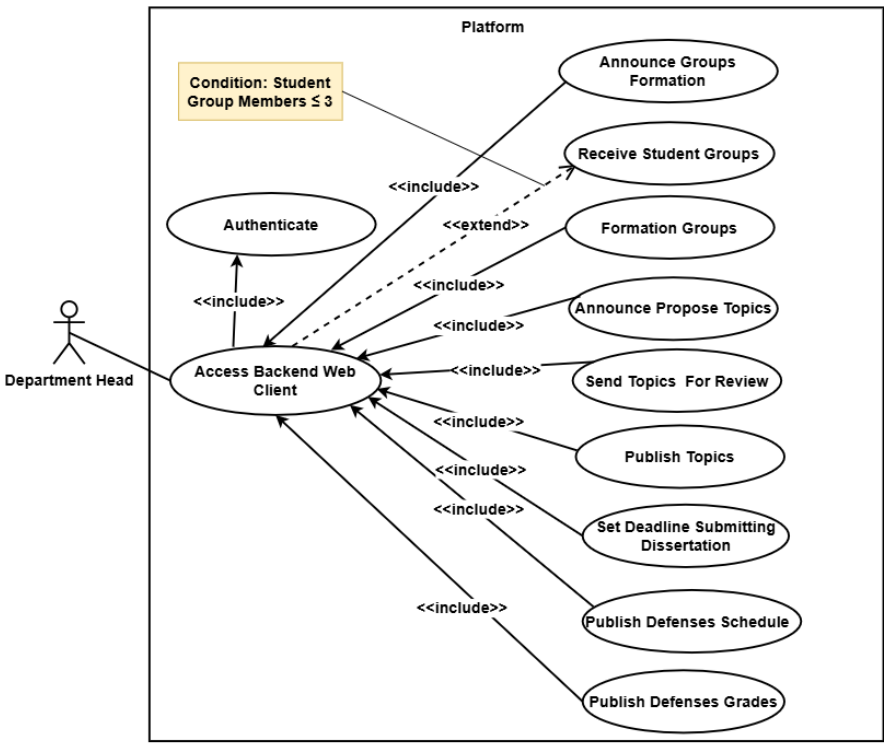


Figure 2.5: Use Case Diagram of the Department Head.

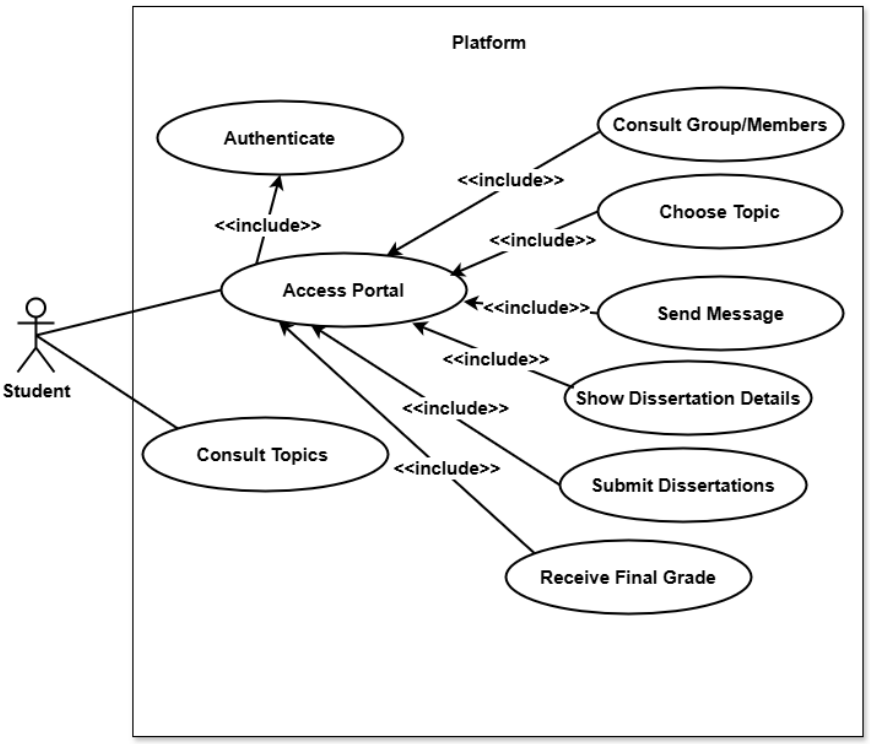


Figure 2.6: Use Case Diagram of the Student.

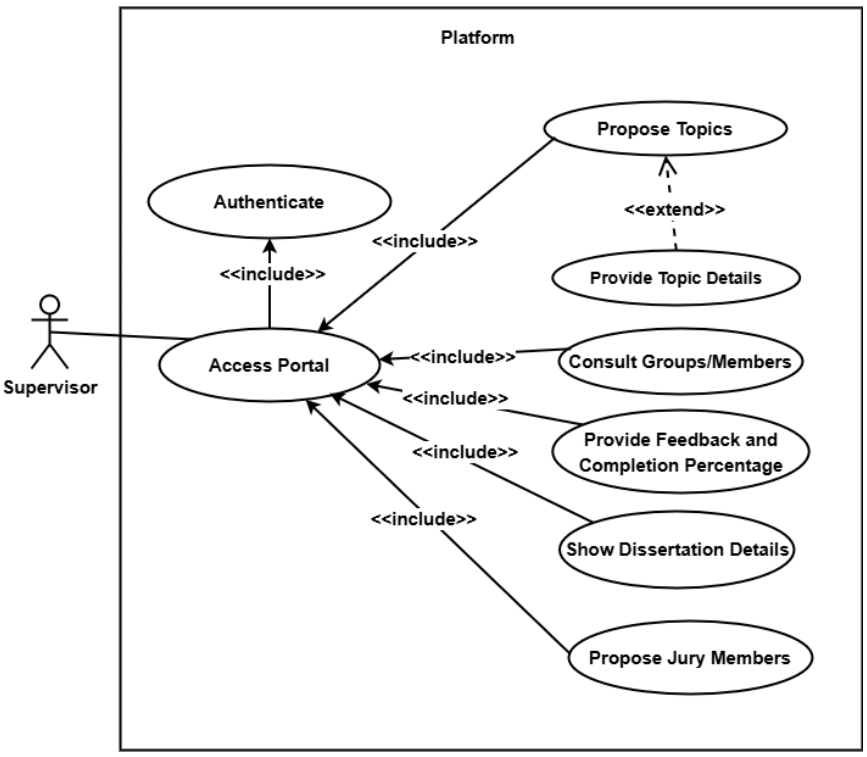


Figure 2.7: Use Case Diagram of the Supervisor.

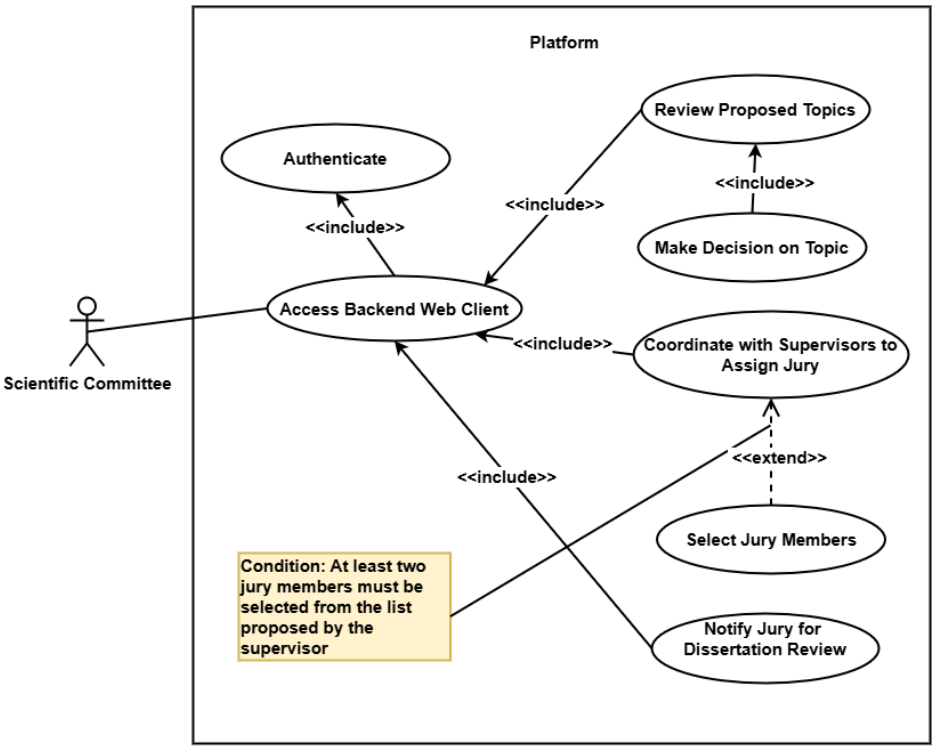


Figure 2.8: Use Case Diagram of the Scientific Committee.

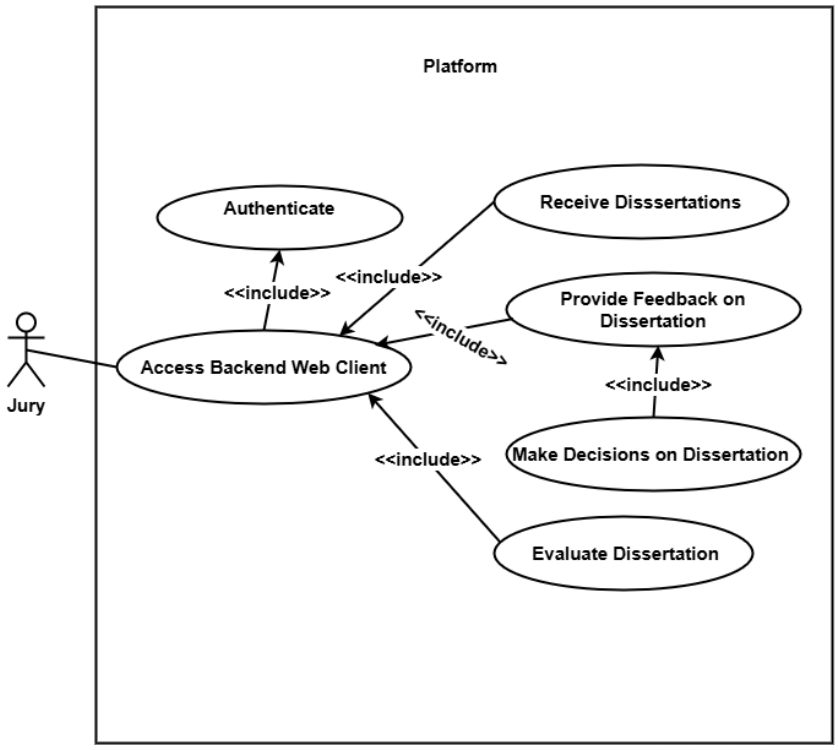


Figure 2.9: Use Case Diagram of the Jury.

2.3.1.2 Activity Diagrams

In this study, two distinct scenarios are modeled using UML activity diagrams:

1. **Topic Announcement and Validation**: This scenario outlines the sequence of activities required for supervisors (teachers) to propose dissertation topics and for the Scientific Committee to validate them (See Figure 2.10).

2. **Dissertation Validation and Review**: This scenario depicts the process by which submitted dissertations are reviewed by jury members, checked against predefined criteria and either approved for defense or returned for revision (See Figure 2.11).

Each diagram illustrates the flow of control, decision points and parallel or sequential actions that occur during these key phases of the dissertation lifecycle.

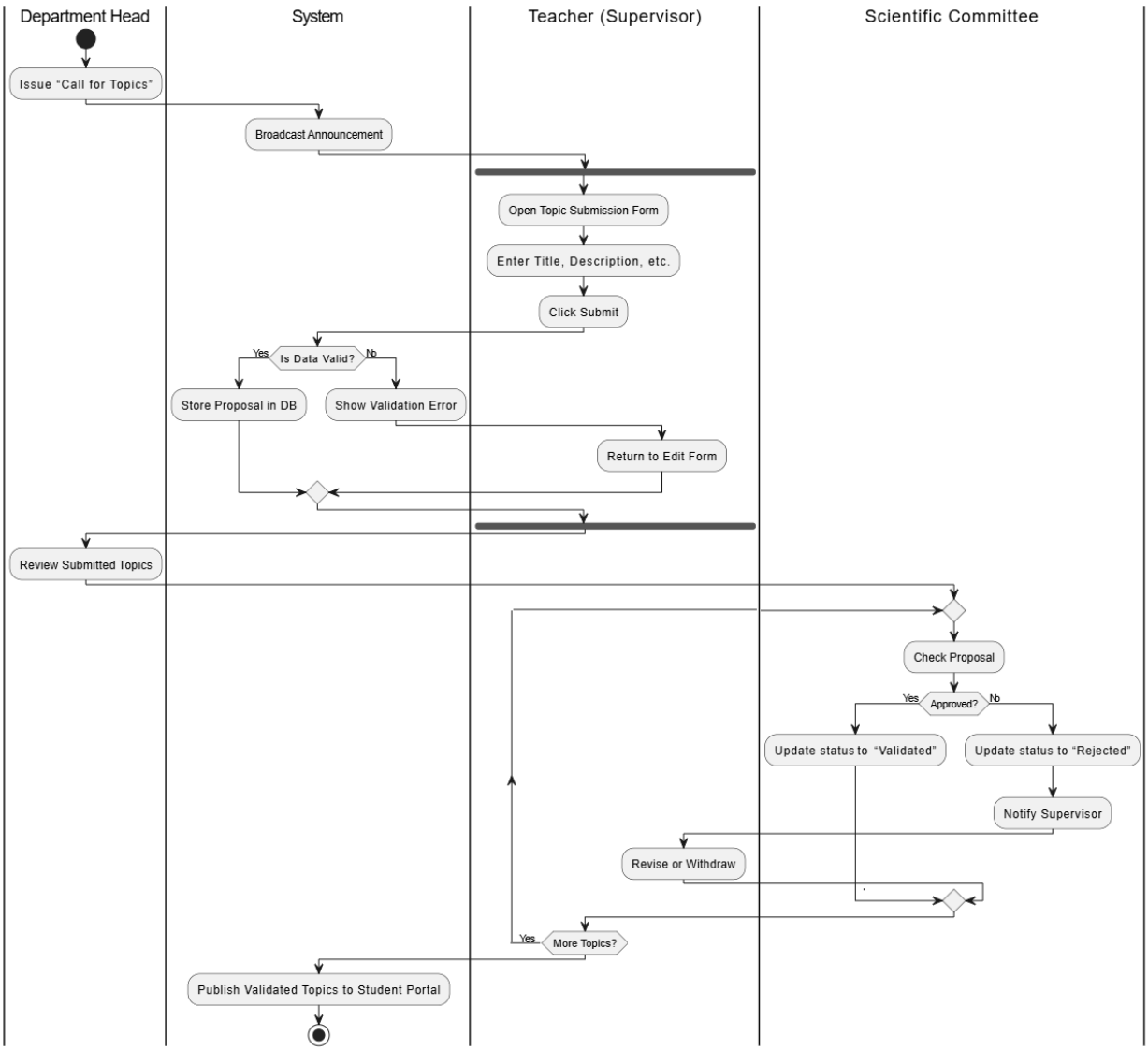


Figure 2.10: Activity Diagram of Topic Announcement and Validation Workflow.

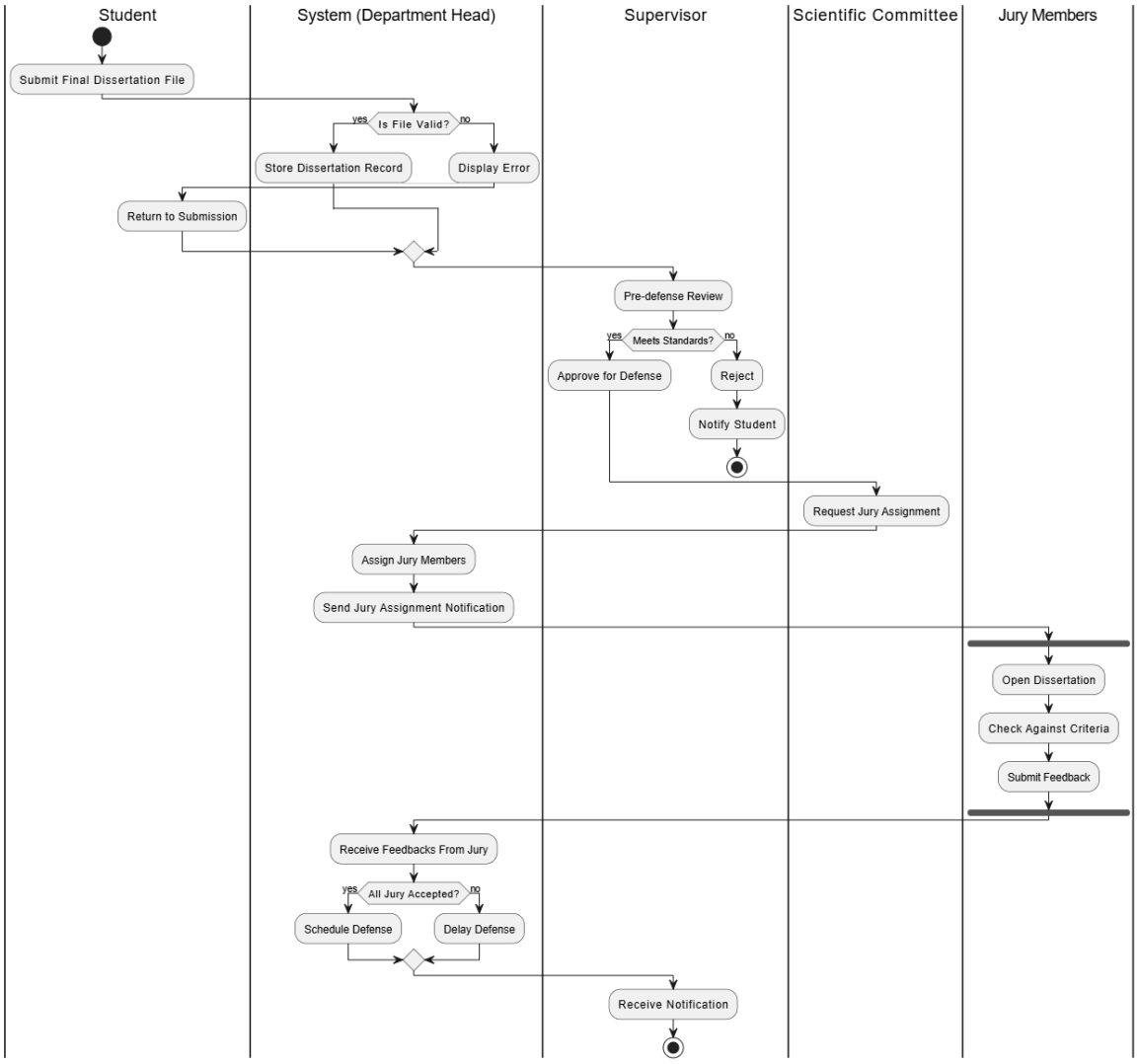


Figure 2.11: Activity Diagram of Dissertation Review and Evaluation Workflow.

2.3.1.3 Automatic Topic Assignment Algorithm

To ensure fairness, efficiency and transparency in the dissertations assignment process, our system incorporates an automatic topic assignment algorithm. This algorithm eliminates the need for manual intervention by assigning topics to student groups based on objective criteria.

Each student group is allowed to select a number of preferred topics equal to the total number of validated topics proposed by the supervisors. Within this list, groups must rank their topic preferences in order of priority, from highest to lowest.

The assignment algorithm (See Figure 2.12) operates according to the following logic:

- First, all student groups are sorted in descending order based on their academic average (GPA), establishing a merit-based priority.
- Then, beginning with the highest-ranking group, the algorithm attempts to assign the first available topic from that group's list of ranked choices.
- If a topic has already been assigned to a higher-ranking group, the algorithm proceeds to the next preference in the list.

- This process continues until each group is assigned one topic or all options in their preference list have been exhausted.

This method not only ensures an equitable and merit-based distribution of dissertation topics, but also reduces administrative workload and mitigates the risk of conflicts or perceptions of favoritism.

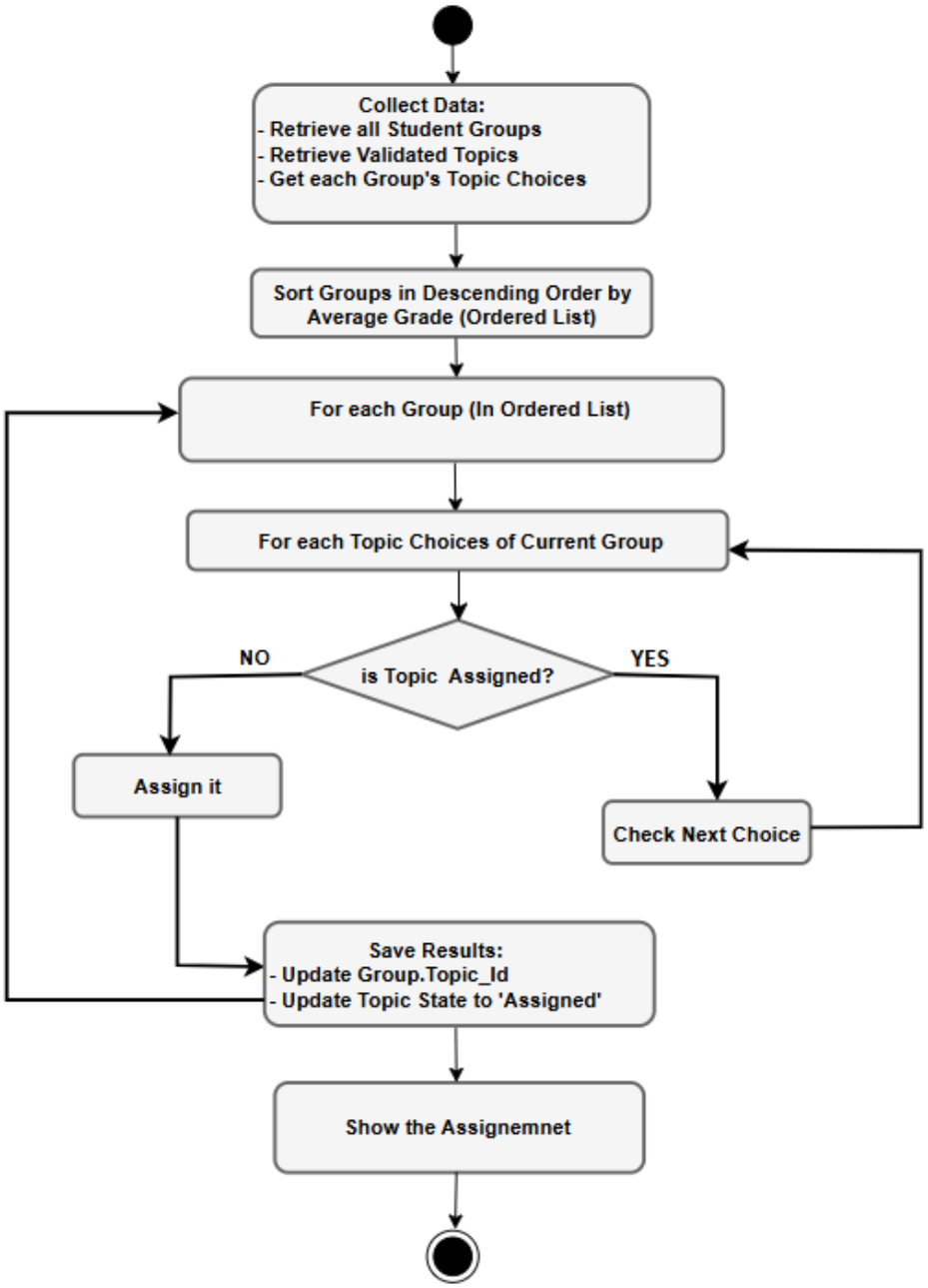


Figure 2.12: Automatic Topic Assignment Algorithm.

2.3.1.4 Sequence Diagrams

In the context of our dissertation management platform, sequence diagrams illustrate the communication flow between actors (such as students, supervisors and department heads) and the system components during various core processes.

Given the large number of use cases defined throughout the system, we have selected seven representative scenarios. These diagrams provide insight into both user interactions and the internal processing logic of the system. The selected sequence diagrams are organized in a logical order corresponding to the typical user journey and administrative workflow:

- **Sequence Diagram 1:** User Authentication
- **Sequence Diagram 2:** Student Group Formation
- **Sequence Diagram 3:** Topic Proposal by Supervisor
- **Sequence Diagram 4:** Scientific Committee Consult and Validate Proposed Topics
- **Sequence Diagram 5:** Viewing Assigned Projects
- **Sequence Diagram 6:** Receiving Final Dissertation Grade

Each of these diagrams is presented in the following sections to detail the sequence of messages exchanged between system actors and components.

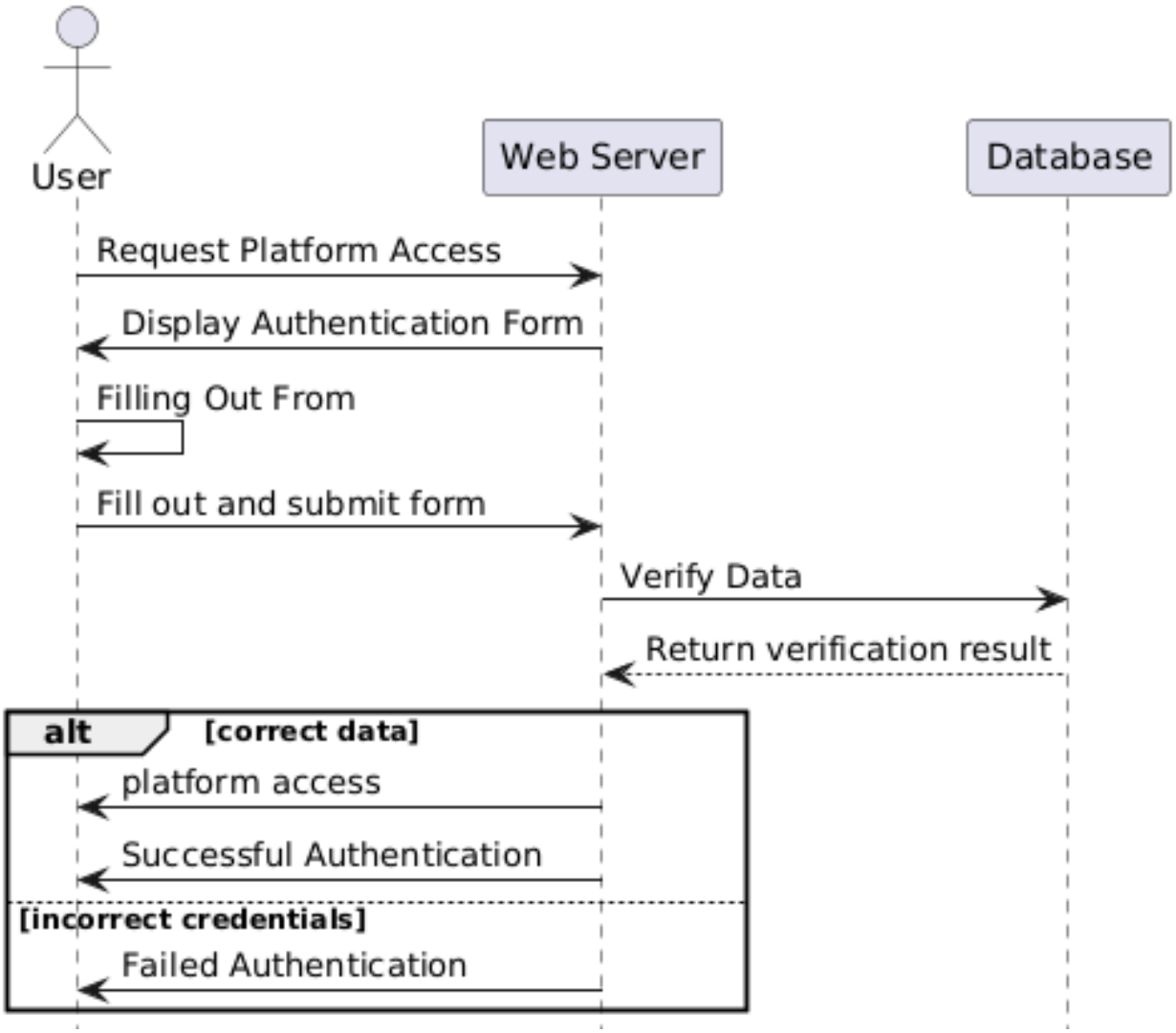


Figure 2.13: Sequence Diagram: User Authentication

Figure 2.13 illustrates the sequence of interactions during the user authentication process. When a user (Student, Teacher, or Department Head) submits their login credentials, the system forwards the request to the web server. The server validates the credentials against the database. If verified, the user gains access to their session; otherwise, an error message is displayed and the user is prompted to try again.

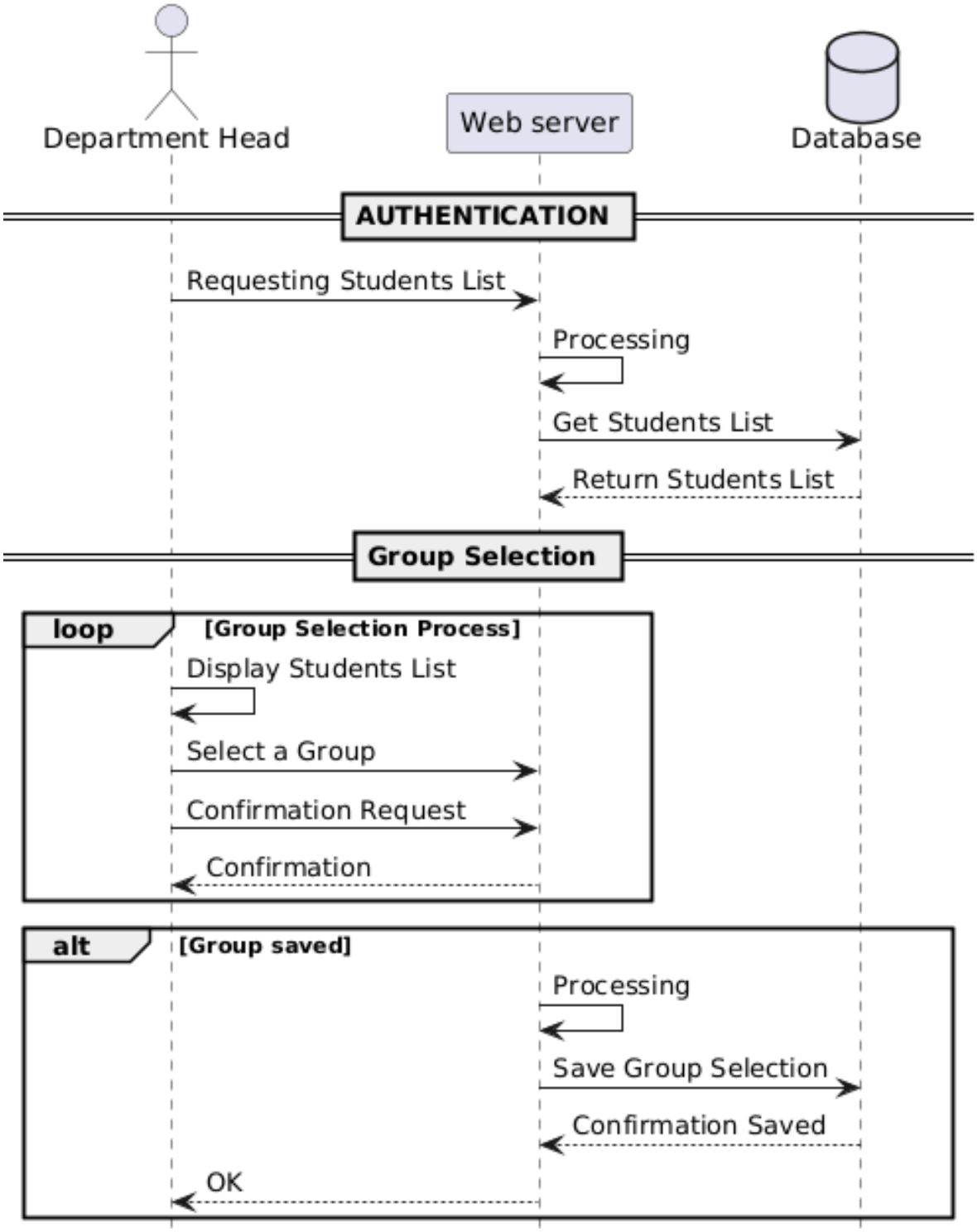


Figure 2.14: Sequence Diagram: Student Group Formation

Figure 2.14 illustrates the sequence of interactions for student group creation. After

successful authentication, the Department Head requests a list of available peers. The system retrieves and displays this list from the Database. The Department Head selects one or more members to form a group. The system then validates and stores the group in the Database and sends a confirmation response, completing the group creation process.

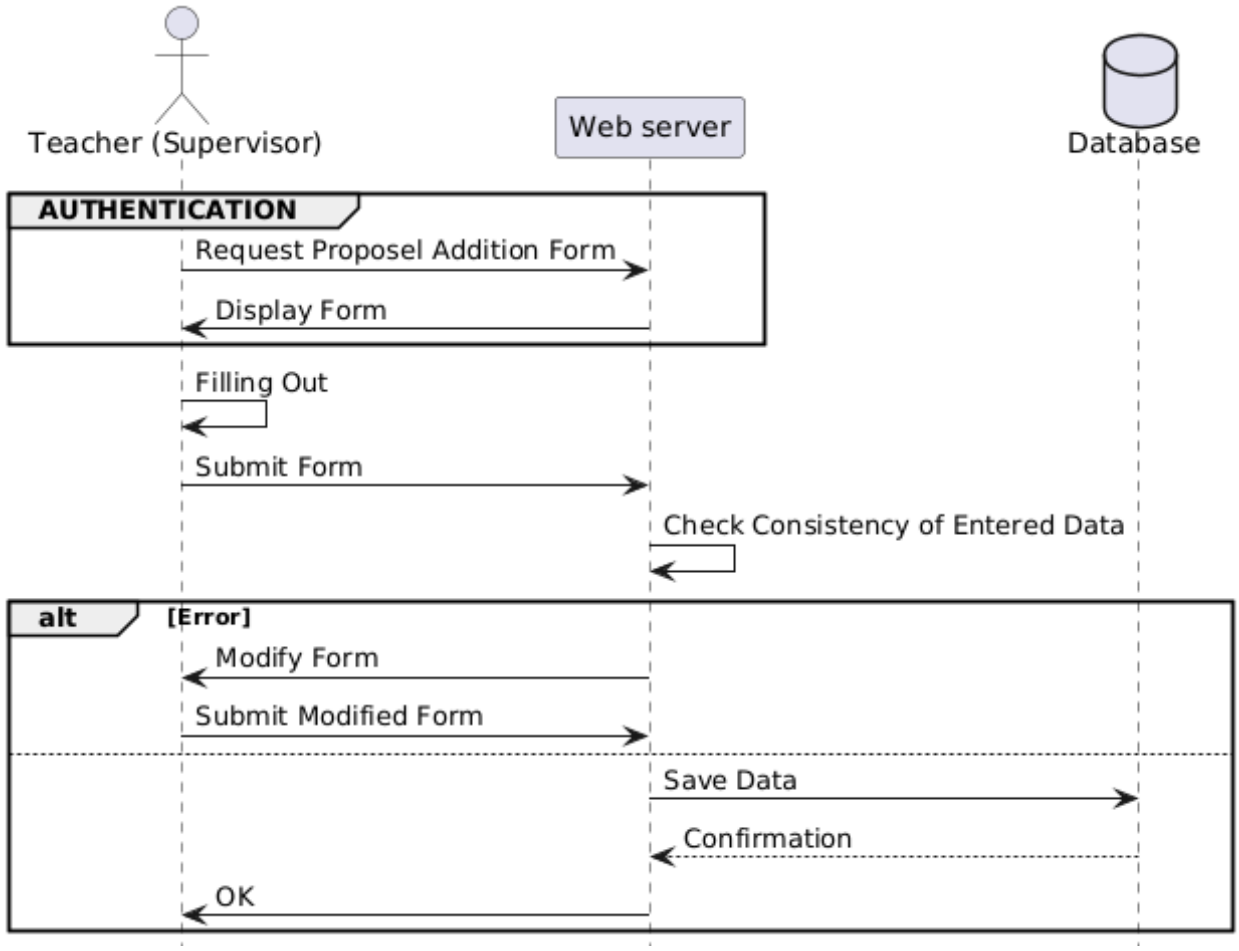


Figure 2.15: Sequence Diagram: Topic Proposal by Teacher (Supervisor)

Figure 2.15 illustrates the sequence of interactions involved in the topic proposal process initiated by a Teacher (Supervisor). Following authentication, the Teacher accesses the topic submission form, which is rendered by the system. After completing the form and submitting it, the system validates the input data. If any inconsistencies are detected, the Teacher is prompted to correct them. Upon successful validation, the data is saved to the database, a confirmation message is displayed and the interface is updated to reflect the changes.

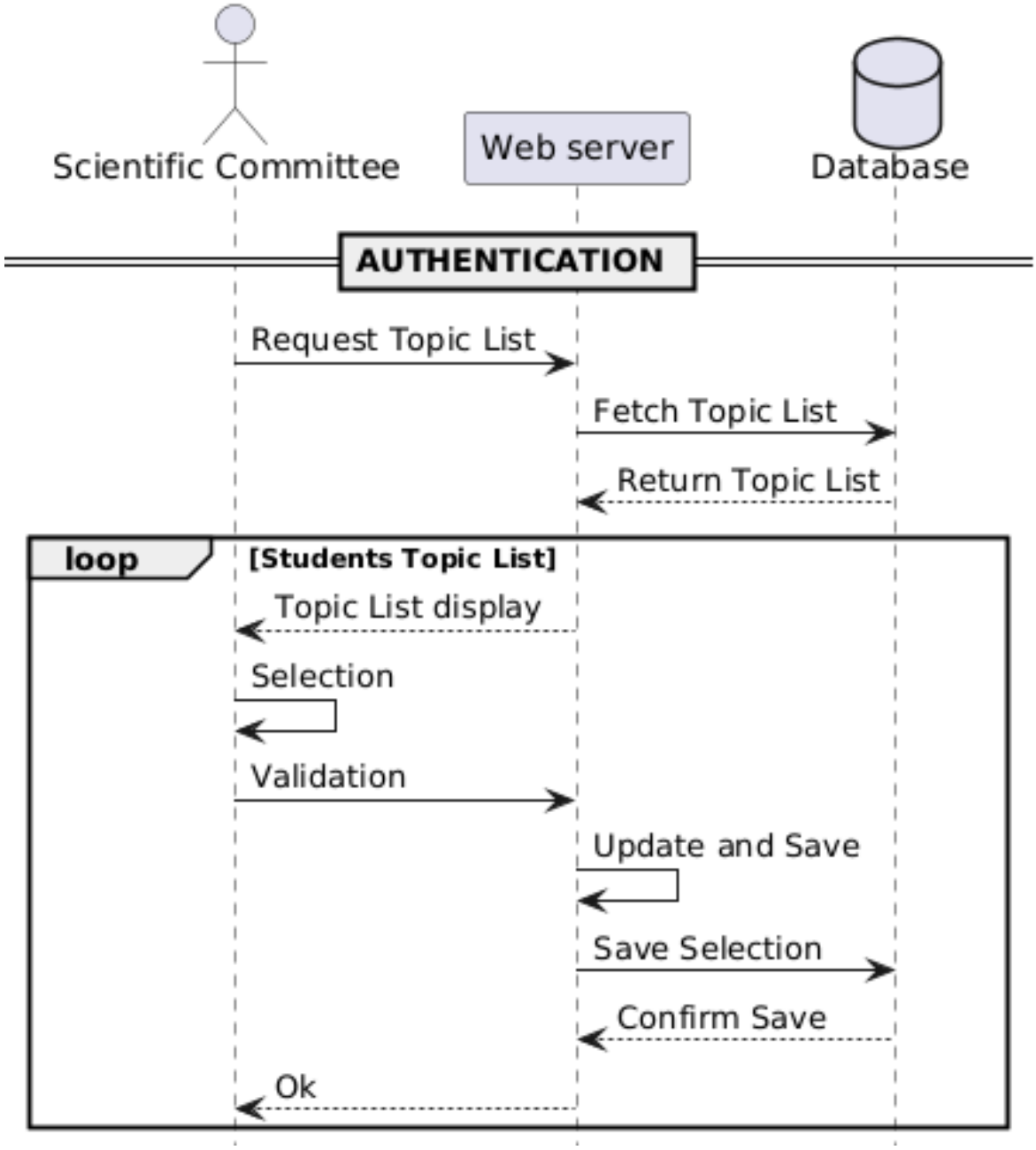


Figure 2.16: Sequence Diagram: Validater Proposed Topics

Figure 2.16 depicts the interaction where a User (Scientific Committee) consults the proposed dissertation topics. After authentication, the user requests the list of topics. Upon selecting and validating a topic, the system updates the status in the database and confirms the action with feedback. In case of an error, a corresponding message is displayed and the list is refreshed to reflect any changes.

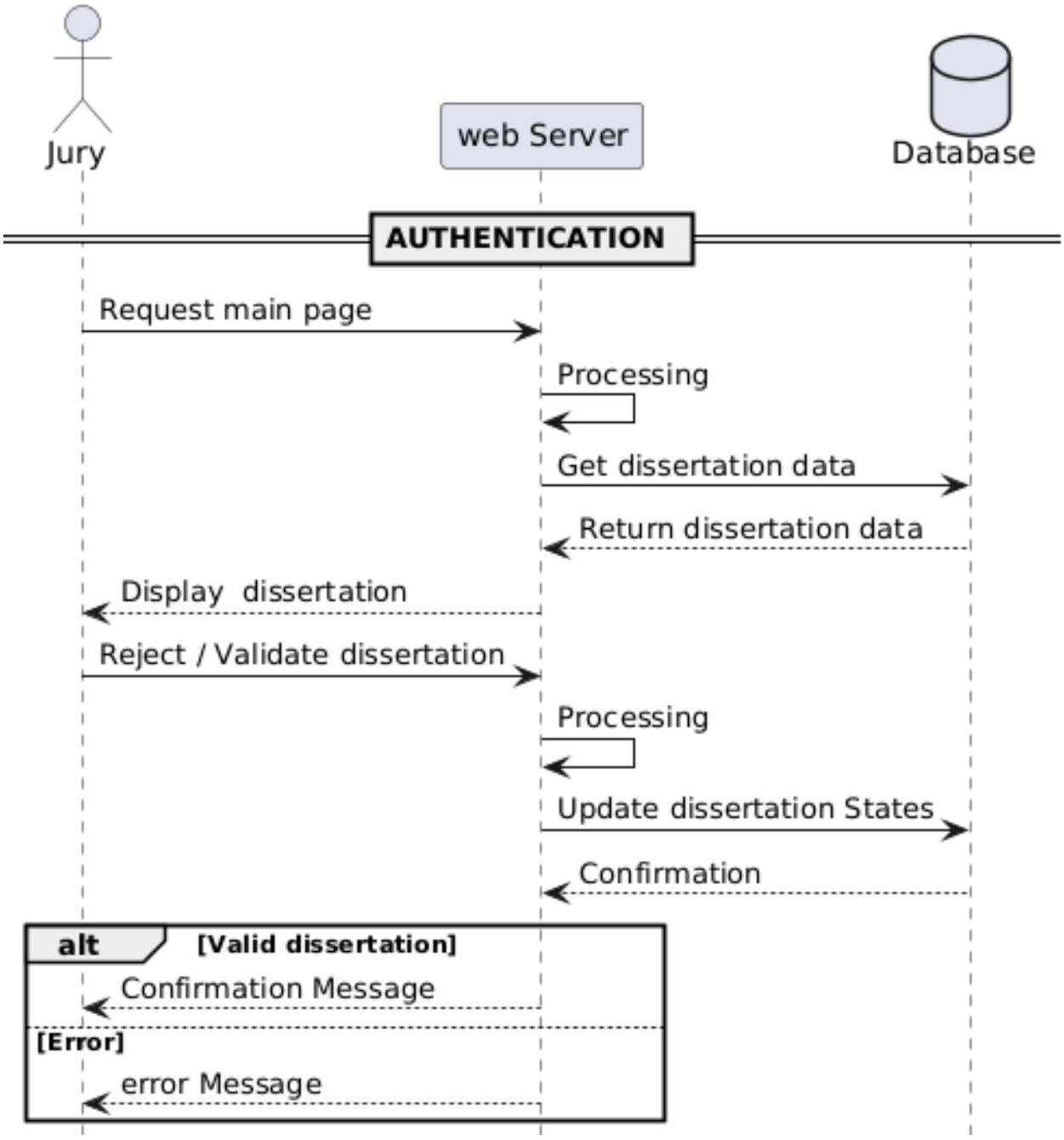


Figure 2.17: Sequence Diagram: Review Dissertation Projects (Jury)

Figure 2.17 illustrates how a Jury Member accesses and evaluates assigned dissertation projects. After login, the jury member requests the list of dissertations and student groups. Upon making a validation or rejection decision, the system updates the project status and returns appropriate confirmation or error messages.

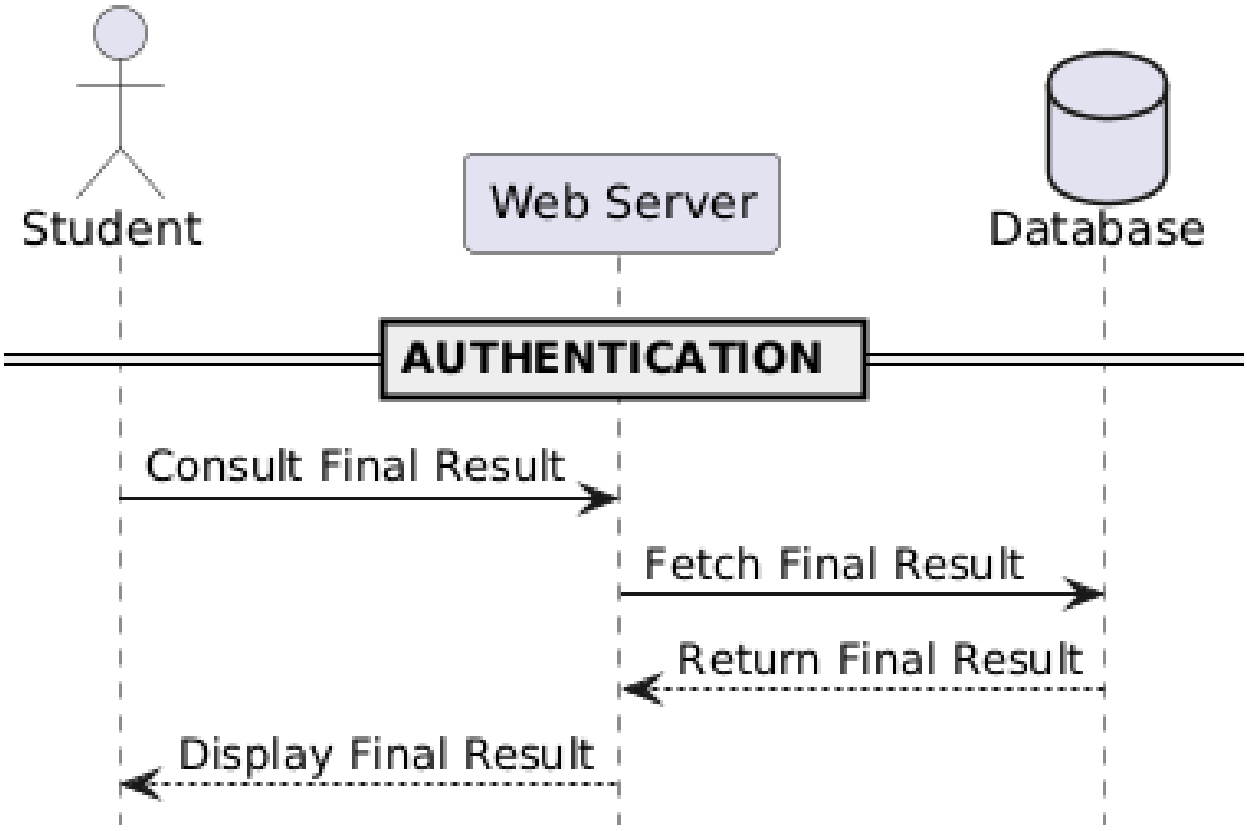


Figure 2.18: Sequence Diagram: View Final Dissertation Grade (Student)

Figure 2.18 describes the process by which a Student retrieves their final dissertation grade. Once authenticated, the system queries the database and displays the result. If no grade is available, a message is shown. The interface is dynamically updated to reflect any changes or errors encountered.

2.3.1.5 Class Diagram

The class diagram models the core structure of the final-year dissertation management system in accordance with the Odoo development paradigm. It leverages several of Odoo's built-in models and modules to facilitate essential functionalities such as authentication, event scheduling, departmental structure and portal access.

Specifically, the system uses Odoo's res.users and res.groups models to implement user authentication and role-based access control. Users can include Students, Teachers, Jury Members, Scientific Committee Members and the Department Head, each belonging to designated groups that define their access rights and functional roles. The res.partner model is utilized to store shared contact and login information for users with portal access, enabling frontend interaction through student and teacher portals.

To model administrative and organizational information, we extend Odoo's hr.employee and hr.department classes. These allow the system to associate academic actors with departments, facilitate internal hierarchy representation and manage roles within the academic institution. Event-based operations, such as announcements, session scheduling and dissertation evaluations, are handled using the event.event model. This approach ensures seamless integration with the website module and the publication of activities on the frontend.

Each class in the diagram corresponds to a database table, with fields, relationships and business logic implemented through Odoo's ORM (Object-Relational Mapping). This structure allows for efficient data manipulation, inheritance and modular extension, following best practices in enterprise software development.

Figure 2.19 illustrates the overall static class diagram of the custom dissertation management module. Core models such as Dissertation, StudentGroup, Teacher, Jury, Review and Evaluation are defined, with relationships reflecting real-world interactions among stakeholders.

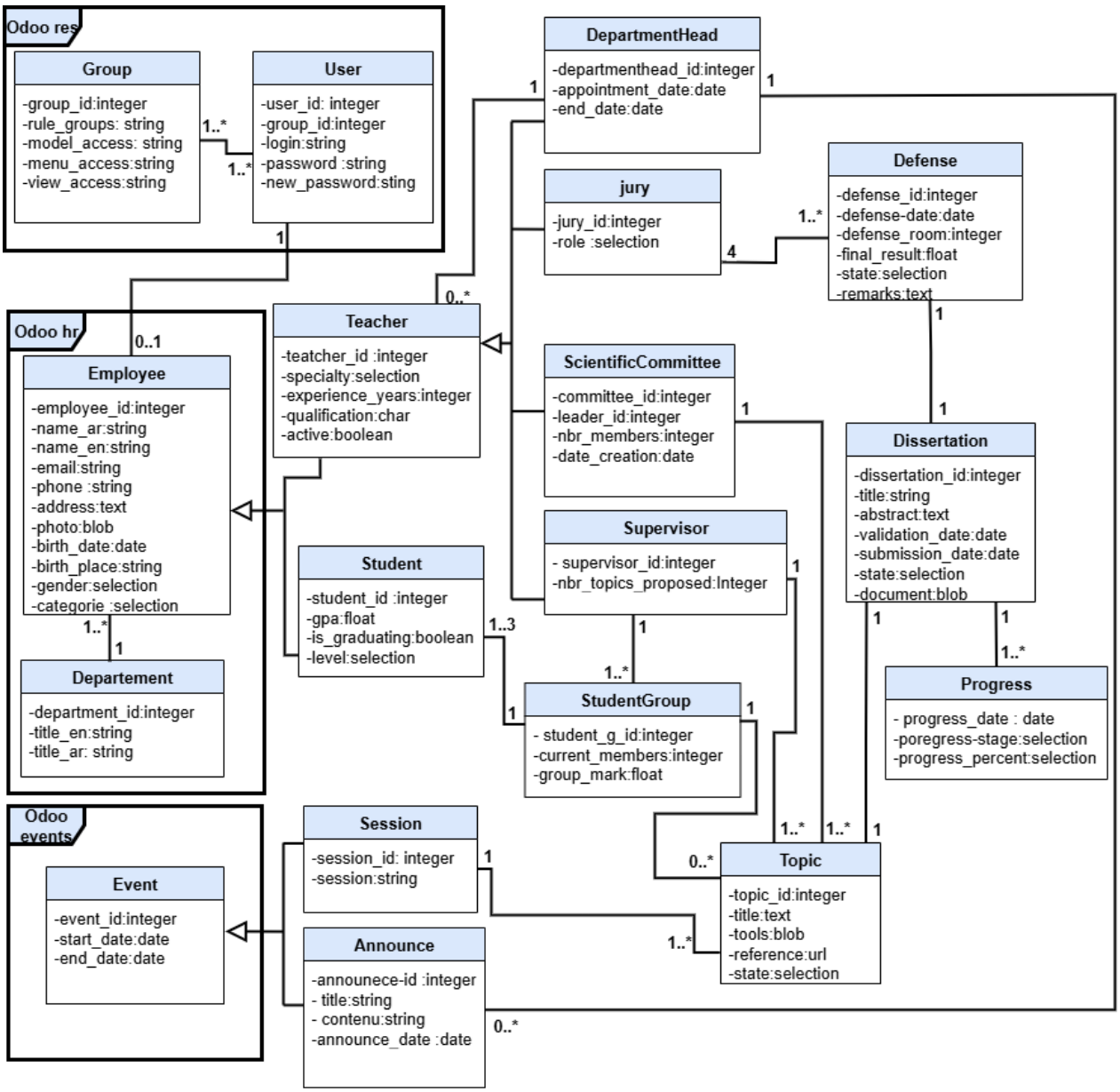


Figure 2.19: Class Diagram.

Additionally, Figure 2.20 [1] highlights Odoo's native access control architecture (res.users, res.groups, ir.model.access and ir.rule). This structure is reused and extended in our module to provide secure, role-based access to different system features. Each user can belong to one or more groups and each group is associated with a defined set of permissions. This modular security model ensures a flexible and scalable approach to managing diverse academic roles.

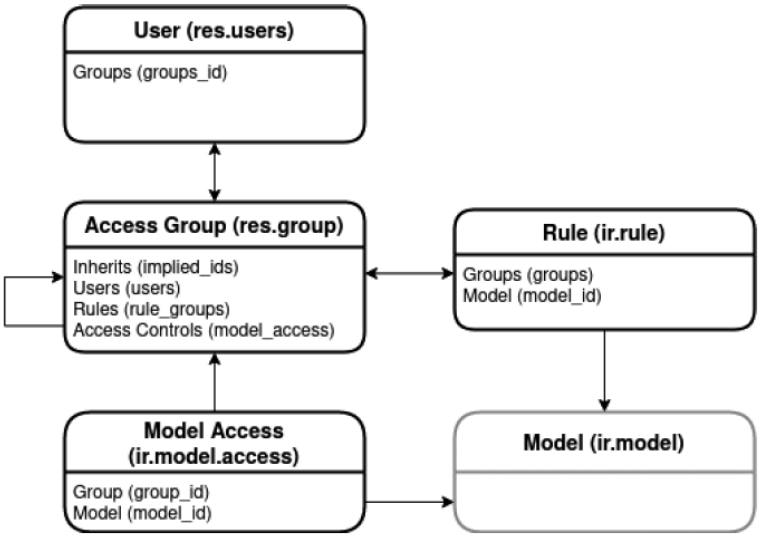


Figure 2.20: Security-Related Data Model[1].

This architecture promotes clean separation of concerns, modular reuse of Odoo's core features and extensibility of the system in future academic or administrative contexts.

2.3.1.6 Deployment Diagram

Figure 2.21 illustrates the deployment architecture of our web-based dissertation management platform, developed using Odoo 16 and extended with a custom module dedicated to the management of final-year projects.

The architecture follows a client-server model. Users interact with the system through a Web Client (browser), which serves as the frontend interface. These interactions are transmitted via HTTP or HTTPS protocols to the Odoo application server. The server implemented on Proxmox Linux-based (Ubuntu) virtual machine (VM) handles core business logic—such as topic proposals, group formation, dissertation submissions, evaluations and notifications—through the custom module integrated into the Odoo ecosystem.

The Odoo server communicates with a PostgreSQL database via its Object-Relational Mapping (ORM) layer, ensuring efficient and consistent storage, retrieval and manipulation of structured data. Additionally, asynchronous operations such as email notifications, approvals and alerts are managed by an external Mail Server using standard SMTP/IMAP protocols.

This modular and layered deployment ensures maintainability, scalability and robustness. It also supports secure, real-time communication among stakeholders (students, supervisors, jury members and department administrators), while preserving data integrity and enabling centralized application management.

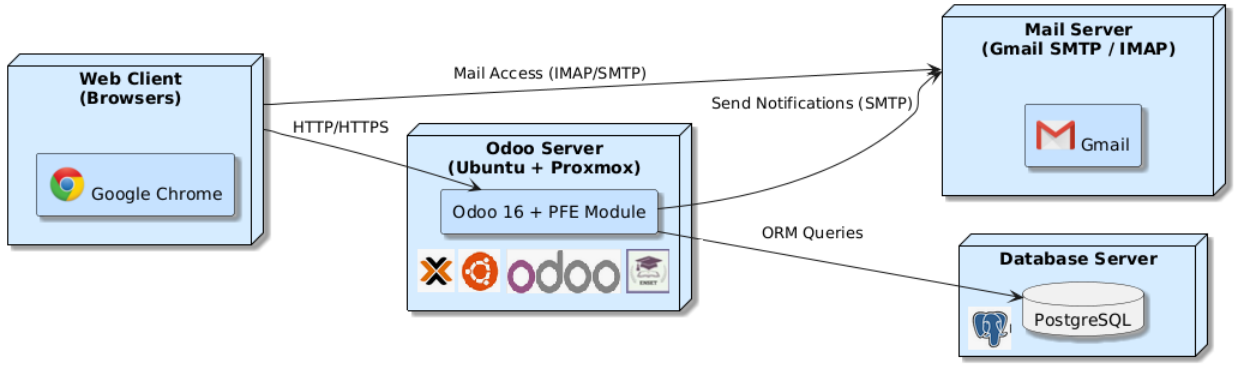


Figure 2.21: Deployment Diagram of the Web-Based Dissertation Management System.

2.4 Conclusion

In this chapter, we have outlined the key steps in specifying and designing the system, starting with an in-depth analysis of requirements, including stakeholder identification, feasibility study and workflow understanding. We then defined both functional and non-functional requirements to ensure that the solution meets user needs effectively. To structure and visualize the system, we leveraged various UML diagrams: use-case, activity, flow-chat, class and deployment diagrams. These tools allowed us to represent the system's structure, interactions and operational flow, laying the groundwork for a robust, modular and scalable architecture. Following this analysis and design work, the next chapter will focus on the implementation phase, where we will present the development of our application: an Odoo module for managing final-year dissertations.

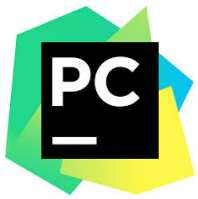
Implementation

CHAPTER 2
IMPLEMENTATION

2.1 Introduction

This chapter details the technical implementation of the custom Odoo module for managing Final Year Projects (PFE) at ENSET-Skikda. Building upon the comprehensive understanding of Odoo's fundamentals (Chapter 1) and the detailed requirements analysis and design specifications (Chapter 2), this chapter describes the practical steps taken to translate the proposed design into a functional application within the Odoo framework. It will cover the selection and utilization of technical tools, the architectural considerations specific to Odoo development, the module's internal structure, the implementation of data models, user interfaces for both the backend web client and frontend website/portals and the crucial aspects of security access control. Finally, the chapter will discuss the deployment process and summarize the key challenges encountered during the implementation phase.

2.2 Technical Tools and Frameworks



The implementation of the custom Odoo module for managing Final Year Projects (PFE) at ENSET-Skikda relied on several key technical tools and frameworks. The primary Integrated Development Environment used for writing and debugging the module's code was PyCharm [1]. PyCharm is an integrated development environment (IDE) for Python. It provides code analysis, a graphical debugger, integrated unit testing and version control support. PyCharm is developed by JetBrains and built on their IntelliJ platform.

The core backend business logic, models and controllers were developed using the Python 3.8 programming language, leveraging Odoo's extensive Python API. Python is dynamically type-checked and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library [2].

[1] https://en.wikipedia.org/wiki/PyCharm
[2] https://en.wikipedia.org/wiki/Python_(programming_language)

The Odoo Object-Relational Mapping (ORM) framework was essential. It is a concept or technique which acts as a bridge between the programming language (Python) and the database (Postgres). The ORM helps to execute SQL queries without writing them explicitly. Once the ORM is configured in an application, the user can use the Object Oriented Programming (OOP) concepts like classes and objects to interact with the database[3].

The module structure and user interface views definitions were handled using Extensible Markup Language (XML) [4]. XML is used extensively in Odoo for defining various components such as data models (implicitly via Python), database records (like menus, actions, security rules) and especially the structure and layout of user interface views (form, tree, kanban, etc.).

The Frontend Stack for the Web Client UI utilized Odoo's built-in JavaScript framework, relying on Odoo Web Language (OWL) [5] and QWeb templating engine[6] for dynamic rendering and interactive components. The Frontend Website/Portals for students and teachers incorporated standard web technologies: HTML for structure (defined in QWeb templates), styling primarily via CSS (Bootstrap [7]) adapted through Odoo's theming for responsiveness and client-side interactivity implemented with JavaScript[8].

For version control, GitHub was utilized as the primary platform to manage the source code throughout the development lifecycle. Its powerful versioning features enabled efficient collaboration, branch management and real-time tracking of code changes, which were essential in coordinating tasks across the development team. GitHub's integrated tools—such as commit history, pull requests and issue tracking—provided a robust environment for reviewing, merging and documenting code updates. The complete source code of the developed Odoo module is publicly available in the following repository:[9]

The data backend utilized PostgreSQL, Odoo's default database, which stores all application data, including module-specific records, configuration and user information. PostgreSQL [10] is an open-source, highly stable database system that supports various SQL features such as sequences, foreign keys, subqueries, triggers, as well as different user-defined types and functions.

[3] https://www.odoo.com/documentation/18.0/developer/reference/backend/orm.html
[4] https://en.wikipedia.org/wiki/XML
[5] https://www.odoo.com/documentation/18.0/fr/developer/reference/frontend/owl_components.html
[6] https://www.odoo.com/documentation/18.0/developer/reference/frontend/qweb.html
[7] https://www.odoo.com/documentation/18.0/developer/reference/frontend/qweb.html
[8] https://developer.mozilla.org/fr/docs/Web/JavaScript
[9] https://github.com/SamRepository/odoo_dev/tree/16.0
[10] https://kinsta.com/fr/base-de-connaissances/qu-est-postgresql/

2.3 Odoo System Architecture

Odoo's architecture is fundamentally a three-tier system comprising the database (PostgreSQL), the application server (Python/ORM) and the client interfaces (Web Client/Backend and Website or Portal/Frontend) (See Figure 2.1)[11].

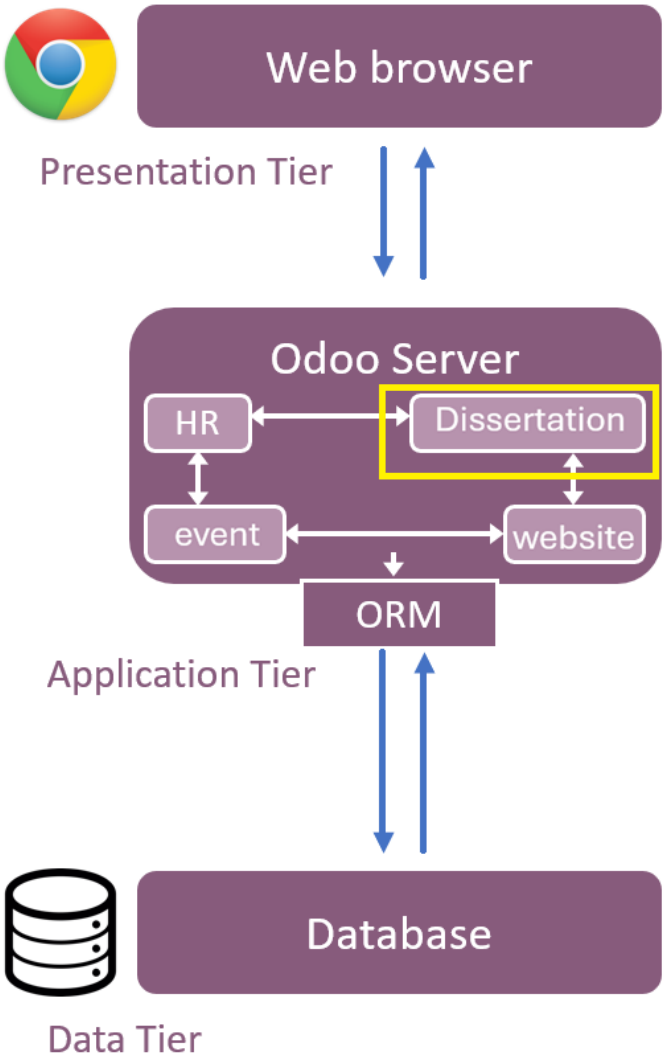


Figure 2.1: The Three-Tier Architecture of Odoo.

The application server, built on Python, processes business logic, manages data via the ORM and mediates between the database and the client layers. The database provides persistent storage for all application data. As shown in Figure 2.2, Odoo presents two main user interfaces: the Website/Frontend and the Backend Web Client[12].

[11] https://2021.desosa.nl/print/projects/odoo/posts/essay_2/
[12] https://ajscript.com/odoo/view-type/understanding-web-client-architecture/

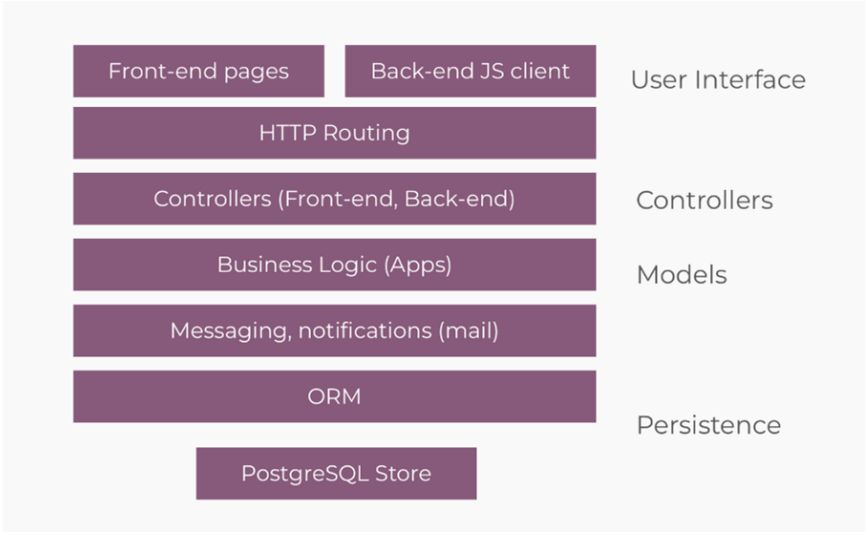


Figure 2.2: The Odoo Architecture Layers.

The Website/Frontend, typically used for public pages or dedicated user portals, follows a traditional Model-View-Controller (MVC) [13] pattern on the server side. User requests are routed to Python controllers, which interact with the models to fetch data. The controller then renders an HTML view using QWeb templates and sends it to the user's browser. This flow is illustrated in the upper section of Figure 2.3.

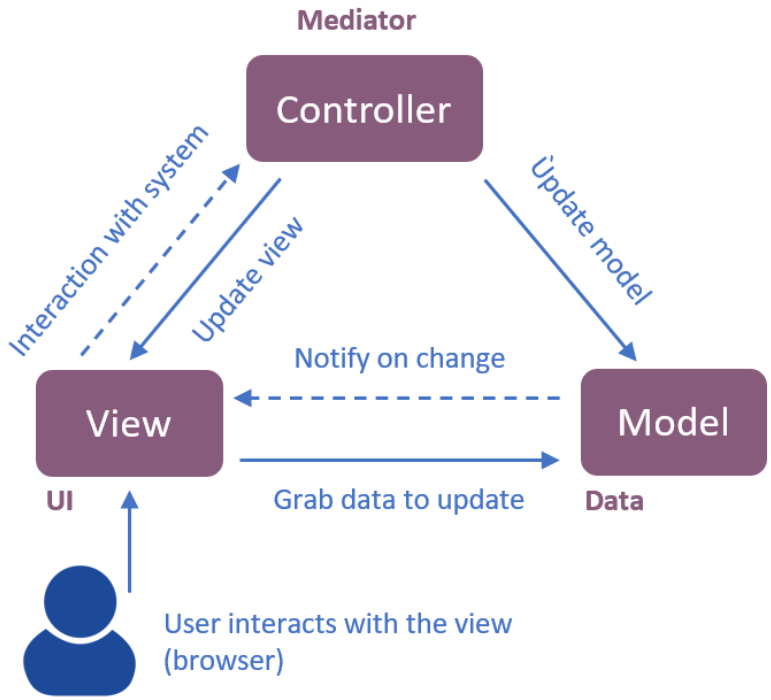


Figure 2.3: The MVC (Model-View-Controller) Pattern.

[13] https://en.wikipedia.org/wiki/Model

2.3.1 Understanding Odoo Web Client Architecture

The Backend Web Client operates as a Single Page Application (SPA). It is designed for internal users managing business processes and features a dynamic and interactive interface. Although an SPA, it also adheres to MVC principles on the client side (using JavaScript). Client-side components, including models, controllers and renderers, interact with the server-side via Odoo's Remote Procedure Calls (RPC) [14] mechanism (leveraging the ORM for data access). This allows for dynamic updates without full page reloads. The relationship between the server-side and client-side components in the Backend Web Client architecture is depicted in the lower section of Figure 2.4.

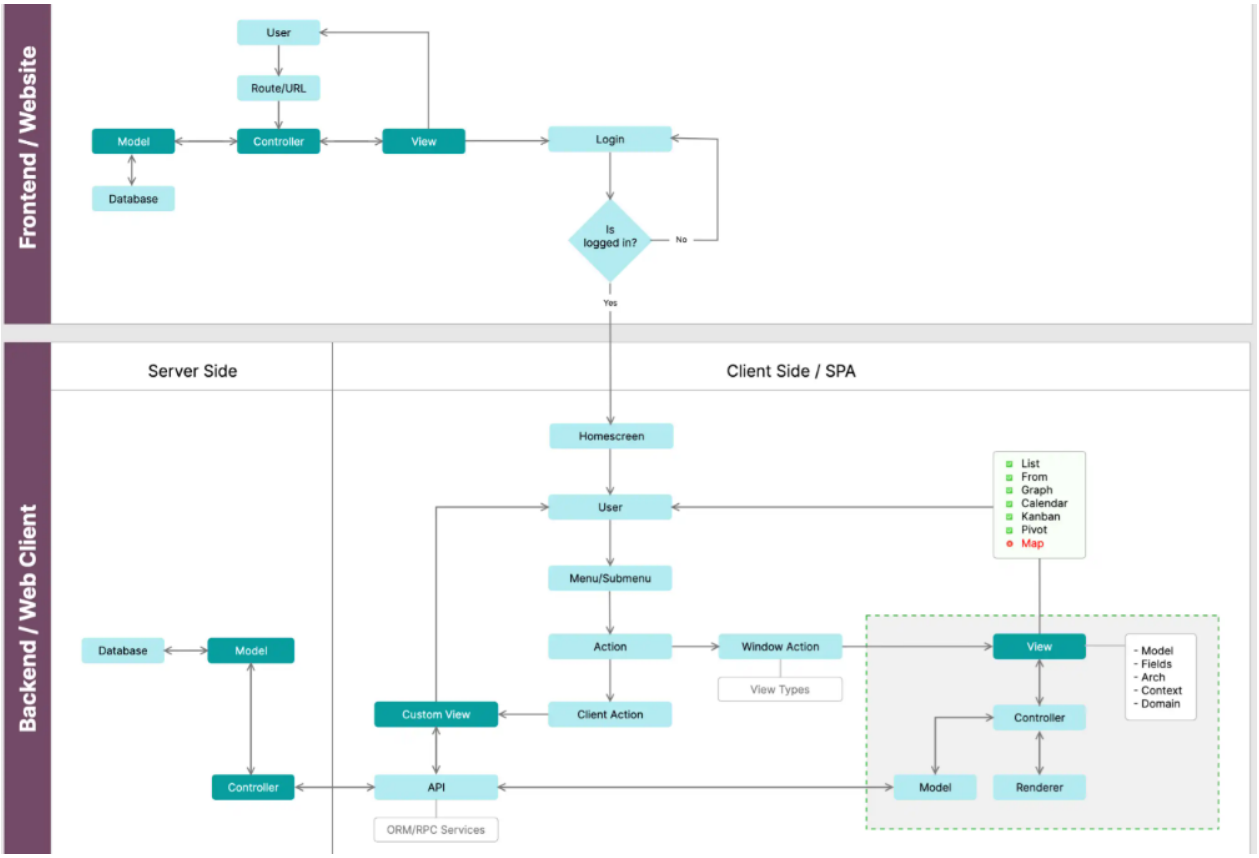


Figure 2.4: Odoo Front-end/Website Back-end/Web-client Interaction Flow. [15]

2.3.2 Setting Up Odoo 16 Development Environment

To establish a stable and functional development environment for Odoo 16 on a Windows operating system, a structured sequence of installation steps must be carefully executed. This process ensures that all dependencies and services required by the Odoo framework are correctly configured, thereby enabling smooth development and testing of the custom module. The installation procedure includes the setup of Python, PostgreSQL, relevant libraries and the Odoo source code itself. Table 2.1 summarizes the essential steps for installing and configuring Odoo 16 on a Windows platform.

[14] https://www.odoo.com/documentation/18.0/developer/reference/external_api.html
[15] https://ajscript.com/odoo/view-type/understanding-web-client-architecture/

Table 2.1: Installation Steps for Odoo 16 on Windows.

	Description	**Commands / Actions**	**Explanation**
1	Install required tools	Python 3.8, PostgreSQL, Git, Node.js, Visual Studio Build Tools, PyCharm, wkhtmltopdf	These tools are essential for running Odoo 16 and compiling required packages.
2	Create a folder for Odoo	`C:\odoo16`	Organizes your Odoo project files in one location.
3	Clone Odoo 16 from GitHub	`git clone --depth 1 --single-branch --branch 16.0 https://github.com/odoo/odoo.git`	Downloads the Odoo 16 source code from GitHub with minimal history.
4	Open the project in PyCharm	Use PyCharm as the development environment	PyCharm helps manage and edit Python code efficiently.
5	Create a virtual environment	Use Python 3.8 in a `venv`	Isolates Python packages for this project to avoid conflicts.
6	Install Python dependencies	`pip install setuptools wheel` `pip install -r requirements.txt`	Installs essential build tools and all Python libraries required by Odoo.
7	Install RTL CSS using npm	`npm install -g rtlcss`	Enables support for right-to-left languages like Arabic in the web interface.

8	Create the `odoo.conf` file	`[options]`
		`admin_passwd = admin`
		`db_host = localhost`
		`db_port = 5432`
		`db_user = odoo`
		`db_password = odoo`
		`addons_path = C:\odoo16\odoo\addons`
		`bin_path = C:/Program Files/wkhtmltopdf/bin`
			Configuration file used by Odoo to connect to the database and define paths.

9	Run Odoo from PyCharm	`python odoo-bin -c odoo.conf`	Starts the Odoo server using the settings defined in the config file.
10	Access Odoo in the browser	`http://localhost:8069`	Opens the Odoo web interface to create or manage your database.

2.3.3 Odoo Module Structure

Odoo modules are organized in a consistent and standardized directory layout, which facilitates modular development, maintainability and scalability. Each directory serves a specific role in the functionality of the module. Below is a description of the main components of an Odoo module:

- `__init__.py`: Initialization file that makes Python packages importable. It specifies which submodules (e.g., models, controllers) are loaded when the module starts.
- `__manifest__.py`: Contains the metadata and configuration of the module, such as its name, version, dependencies and data files.
- `models/`: Defines the business logic and data models using Python classes. It is the core layer that interacts with the Odoo ORM and the database.
- `views/`: XML files that define user interface elements (e.g., form views, tree views, kanban views) for the models.
- `controllers/`: Handles web and HTTP requests. It defines custom routes and logic for front-facing portals and web client communication.
- `data/`: Contains initial data to be loaded upon module installation, such as predefined records (e.g., groups, menus, or statuses) using XML or CSV.
- `demo/`: Stores demo data and files (e.g., Python, XML, JS, CSS) used when creating a test database with demo mode enabled.
- `doc/`: Holds documentation files about the module, typically in formats like PDF, DOC, or Markdown (md, rst).
- `i18n/`: Contains translation files (.pot, .po) for enabling multilingual support across the interface.
- `wizard/`: Implements transient models and modal dialogs for step-by-step operations or user inputs using XML and Python.
- `report/`: Used for defining reports, including templates and logic for generating printable or exportable reports (e.g., PDF).
- `security/`: Defines access control lists (ACLs) and record rules using XML or CSV to manage user roles and permissions.
- `static/`: Hosts front-end assets such as JavaScript, CSS and image files required for rendering dynamic and responsive UIs.
- `populate/`: (Optional) Python scripts that generate large test datasets for development and testing purposes.
- `tests/`: Contains unit test cases written in Python to validate module functionality.
- `tools/`: (Optional) Stores miscellaneous utility scripts and helper functions that do not logically belong to any other directory.

This structured layout enables modularity and clarity in development, allowing developers to focus on specific functional aspects without affecting others.

Figure 2.5 presents a graphical representation of the typical folder structure and purpose of each directory in an Odoo module[16].

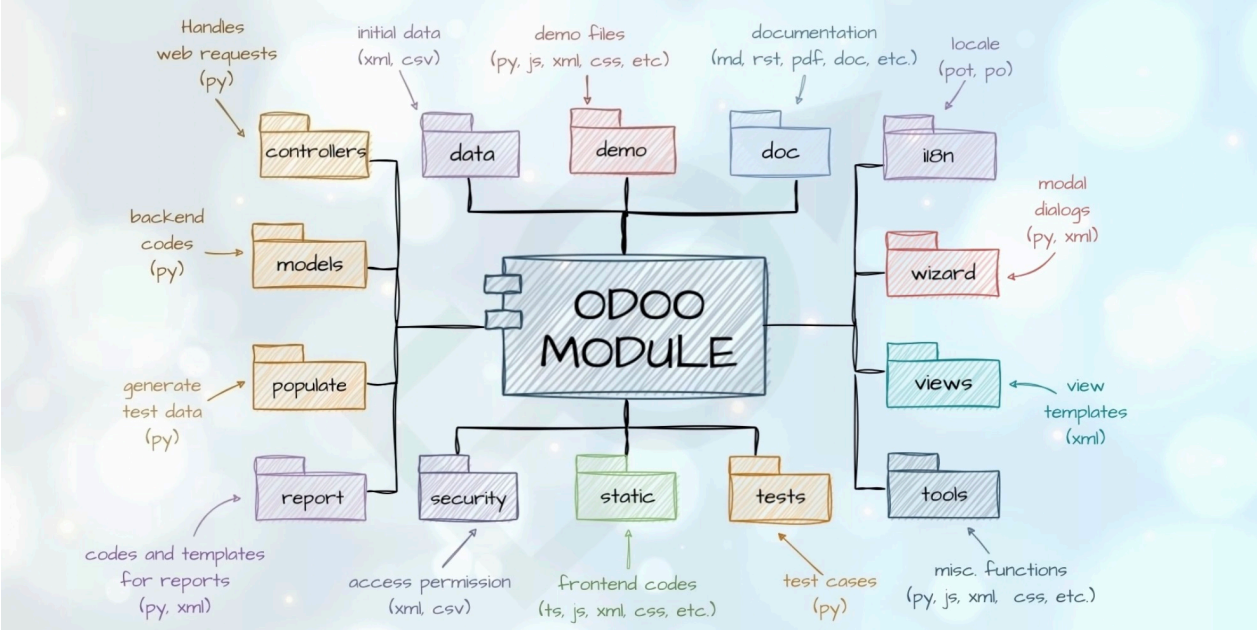


Figure 2.5: Standard Directory Layout of an Odoo Module.

This consistent structure helps developers navigate modules and integrate custom code within the Odoo ecosystem. Listing 2.1 shows the ___manfest___.py file of our module, which is central to defining the module and listing its data files.

As shown in Figure 2.6 , the "Dissertation Management" icon appears on the Odoo dashboard, confirming that the module was correctly installed and recognized by the system.

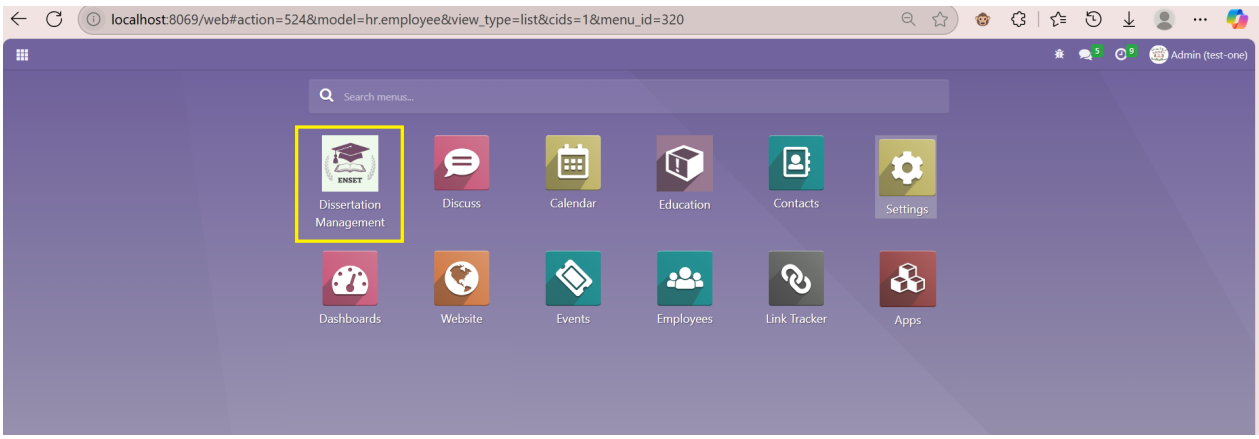


Figure 2.6: Successful installation of the PFE custom module.

[16] https://erpsolutions.oodles.io/blog/odoo-architecture-technical-deployment/

Listing 2.1: Sample___manifest___.py File Snippet.

```
{
    'name': "Final Year Dissertation Management",
    'summary': "Odoo module for managing final year dissertations at the Department of
        ↪ Mathematics and Computer Science at ENSET SKIKDA",
    'author': "Department of Mathematics and Computer Science",
    'version': '16.0.0.1',
    'category': 'Education',
    'sequence': 1,
    'application': True,
    'installable': True,

    'depends': ['base', 'web', 'web_responsive', 'website', 'portal', 'hr', 'contacts', '
        ↪ event', 'mail'],

    'data': [
        'security/groups.xml',
        'security/ir.model.access.csv',
        'security/ir_rule.xml',
        'security/portal.xml',
        'security/topic_security.xml',
        'data/teacher_categories.xml',
        'data/education_specializations.xml',
        'data/event.xml',
        'data/email.xml',
        'data/ir_sequence_data.xml',
        'views/education_specialization.xml',
        'views/student.xml',
        'views/sgroupe.xml',
        'views/topic.xml',
        'views/topic_categories.xml',
        'views/dissertation.xml',
        'views/defense.xml',
        'views/departement.xml',
        'views/menu.xml',
        'views/website_menu.xml',
        'views/choose_diss.xml',
        'views/website_topic_submit_templateA.xml',
        'views/template_defense_creat.xml',
        'views/portal_template.xml',
        'views/portal_dissertation_portal.xml',
        'views/Topic_details.xml',
        'views/list_topic.xml',
        'views/list_topic_public.xml',
        'views/login.xml',

        'reports/dissertation_re.xml',
    ],

    'images': ['static/description/icon.png'],
}
```

2.3.4 Security and Access Control

Implementing robust security and access control is paramount in a multi-user application like Odoo. Odoo provides a robust Multi-Level Access Control system to manage permissions based on user groups. This system involves several layers of checks, as conceptualized in Figure 2.7

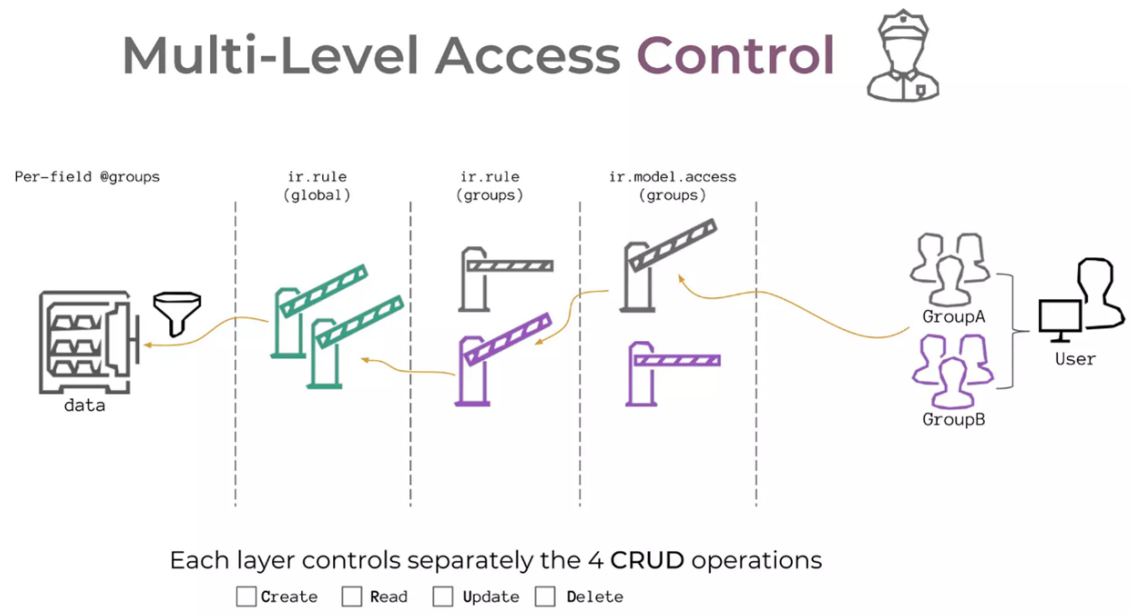


Figure 2.7: Multi-Level Access Control in Odoo (Conceptual).

The foundational layer is defined by Model Access Rights, configured primarily in the ir.model.access.csv file located within the module's security/ directory. Before assigning these rights, the corresponding user groups must first be defined. This is typically done in an XML file named ir_rule.xml, placed in the same security/ directory. Listing 2.2 presents an example (Teacher group) of how these user groups are defined. Once the groups are in place, the ir.model.access.csv file can be used to assign Create, Read, Write and Unlink (CRUD) permissions to each group for specific models. Listing 2.3 illustrates how this linkage is established between users group.xml and corresponded access rights in ir.model.access.cs.

Listing 2.2: Access Control Example highlighting Teacher Group and ir.rule.

```
 <!-- Teacher -->
<record id="group_teacher" model="res.groups">
    <field name="name">Teacher</field>
    <field name="category_id" ref="pfe.module_category_pfe"/>
    <field name="implied_ids" eval="[(4, ref('base.group_portal'))]"/>
</record>
<!-- TEACHER RULE -->

<!-- Teachers: Can see only their own proposed topics -->
<record id="rule_teacher_own_topics" model="ir.rule">
    <field name="name">Teacher: Own Topics</field>
    <field name="model_id" ref="model_pfe_topic"/>
    <field name="domain_force">[('teacher_id.user_id', '=', user.id)]</field>
    <field name="groups" eval="[(4, ref('pfe.group_teacher'))]"/>
    <field name="perm_read" eval="1"/>
    <field name="perm_write" eval="0"/>
    <field name="perm_create" eval="0"/>
    <field name="perm_unlink" eval="0"/>
</record>
```

Listing 2.3: Access Control Example highlighting topic ir.model.access.csv.

```
id,name,model_id:id,group_id:id,perm_read,perm_write,perm_create,perm_unlink
access_pfe_topic_admin,Access Topic for Admin,model_pfe_topic,base.group_system,1,1,1,1
access_pfe_topic_department_head,Access Topic for Department Head,model_pfe_topic,pfe.
    ↪ group_department_head,1,1,1,1
access_pfe_topic_teacher,Access Topic for Teacher,model_pfe_topic,pfe.group_teacher,1,0,1,0
access_pfe_topic_committee,Access Topic for Committee,model_pfe_topic,pfe.
    ↪ group_committee_scientifique,1,1,1,0
```

Building upon model access rights, Record Rules (defined using `ir.rule` records in XML files—typically under the `security/` directory) provide a more granular level of control. These rules apply domain filters to a model, restricting access to specific records based on conditions related to the user or the record data itself. Figure 2.8 depicts how `ir.rule` records are linked to user group.xml. For example: a record rule for the `pfe.project` model linked to the `group_pfe_student` group could apply the domain:

```
[('dissertation_id.group_id.student_ids.user_id', '=', user.id)]
```

ensuring Students only see projects associated with their own record. Similarly, a rule for Teachers could limit visibility to projects where they are the Supervisor or a Committee Member. These rules act as additional filters after the initial model access check, allowing for fine-grained control over data visibility.

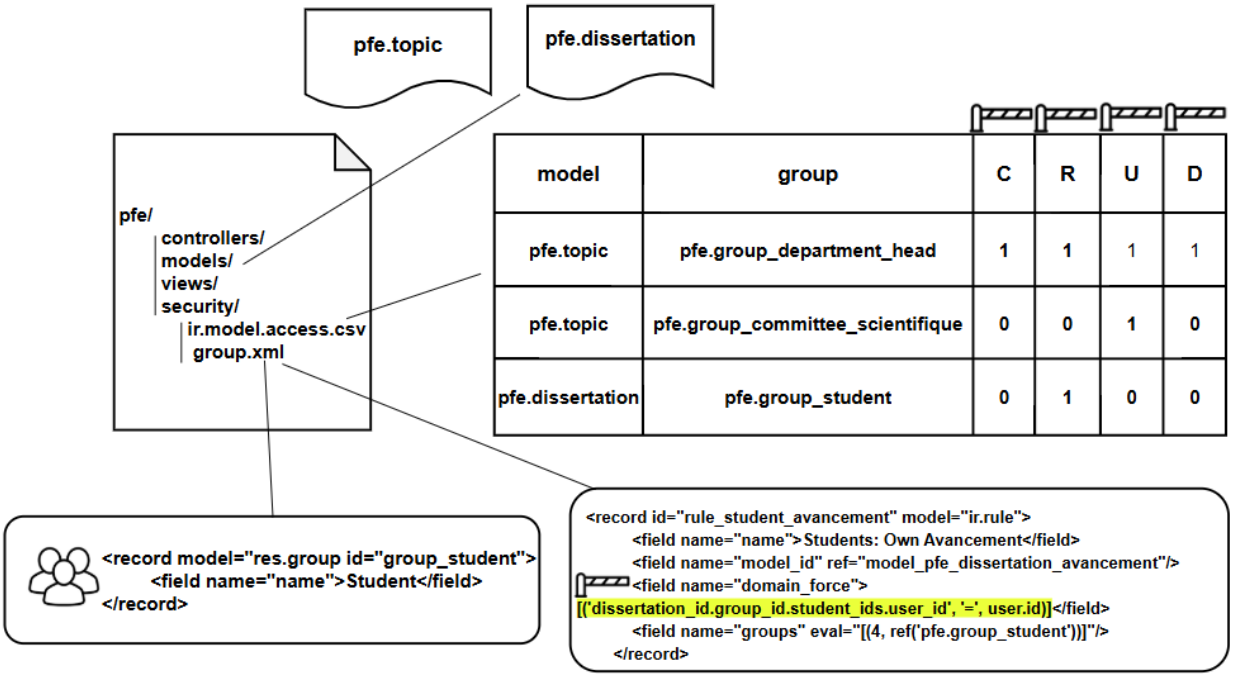


Figure 2.8: Implementation of Record Rules and Access Rights Using Security Groups.

These layers, combined with user group assignments configured in the system, form the multi-level access control system that secures the PFE module's data and functionalities according to the defined user roles and their specific responsibilities within the PFE management process.

2.4 Workflow Implementation

The core functionalities and user interactions for managing the PFE lifecycle, as designed in Chapter 2, are implemented across the module's backend models, views and frontend components. This section details some of the realization of these workflow aspects in code and configuration.

2.4.1 Data Model Implementation

The data models, representing key entities in the PFE management system (e.g. student, teacher, topic, dissertation), were defined as Python classes inheriting from odoo.models.Model within the models/ directory. Each class defined the necessary fields (fields.Char, fields.Many2one, fields.Text, fields.Date, etc.) to store relevant data such as project: titles, descriptions, submission dates, student and supervisor links and project status. Computed fields and ORM methods were implemented within these model classes to encapsulate the business logic, such as automatically setting the project state based on actions (e.g., moving from 'Pending' to 'Validated' upon supervisor selection) or validating data integrity (e.g., ensuring a topic is unique upon submission). Listing 2.5 presents a code excerpt from the pfe.topic model. It highlights the structure of the model class in Python, including key fields such as the title, description, supervisor and the project status. This snippet exemplifies how Odoo's ORM is used to define data structure and enforce business rules directly within the model.

Listing 2.4: Definition of the pfe.topic Model in Python.

```
from odoo import models, fields, api, exceptions, _
from odoo.exceptions import UserError
from datetime import date

class Topic(models.Model):
    _name = 'pfe.topic'
    _inherit = ['portal.mixin', 'mail.thread', 'mail.activity.mixin', 'utm.mixin']
    _description = 'Dissertation Topic'
    _rec_name = 'name'
    name = fields.Char(
        string="SEQ",
        required=True,
        copy=False,
        readonly=True,
        default=lambda self: self.env['ir.sequence'].next_by_code('pfe.topic')
    )
    title= fields.Char(string="Topic Title", required=True)
    description = fields.Html(string="Description")
    tools = fields.Html(string="Tools")

    state = fields.Selection([
        ('pending', 'Pending'),
        ('validated', 'Validated'),
        ('rejected', 'Rejected'),
    ], string="State", default='pending', tracking=True)
    category_ids = fields.Many2many(
        'pfe.topic_category', 'topic_category_rel',
        'emp_id', 'category_id',
        string='Tags')
    supervisor_id = fields.Many2one('hr.employee', string="Supervisor", tracking=
        ↪ True)
    dissertation_id = fields.Many2one('pfe.dissertation', string="Dissertation",
        ↪ readonly=True)

    _sql_constraints = [
        ('unique_topic_name', 'UNIQUE(name)', 'Topic title must be unique!'),
    ]

    @api.model
    def create(self, vals):
        if vals.get('name', _('New')) == _('New'):
            vals['name'] = self.env['ir.sequence'].next_by_code('pfe.topic') or _
                ↪ ('New')

        if 'supervisor_id' in vals:
            supervisor = self.env['hr.employee'].browse(vals['supervisor_id'])
            category_teacher = self.env.ref('pfe.category_teacher')
            if category_teacher not in supervisor.category_ids:
                supervisor.category_ids = [(4, category_teacher.id)]

        return super(Topic, self).create(vals)
```

2.4.2 User Backend Interface Implementation (Views and Actions)

User interfaces in the Odoo Backend Web Client are defined through XML views stored in the `views/` directory. Different view types were implemented for the PFE module to provide various perspectives and interaction methods for the data. List/Tree/Form Views were created to display collections of records in a tabular format, suitable for presenting lists of all PFE projects or registered students with key information summarized in columns. The XML for these views defines the specific fields (`<field name="name"/>`, `<field name="state"/>`) to be shown in the table, allowing users to quickly browse and sort data. Form Views were designed to display and allow editing of individual records in detail, used for viewing or updating specific project details, entering student information, or submitting topic proposals. The XML for form views structures the fields using layout elements like `<sheet>`, `<group>` and `<field>` to create a user-friendly data entry and display interface. Other view types like Kanban Views were utilized to visualize records as interactive cards, aiding in tracking the workflow stages of PFE projects (e.g., `'Pending'`, `'Validated'`, `'Rejected'`) in a visual, drag-and-drop interface. Search Views were implemented to define searchable fields and filtering options, enabling users to efficiently find specific records within the various views based on criteria like project title, student name, or status.

These views are made accessible through Window Actions (`ir.actions.act_window`), which are also defined as records in XML. A window action serves as a bridge, specifying the target model (`res_model`) and the list of view types (`view_mode`) that should be available when this action is triggered (e.g., `tree`, `form`, `kanban`). These actions are then linked to Menu Items (`ir.ui.menu`), defined as records in XML, which organize the user interface navigation in the Odoo backend. Each menu item is typically linked to a specific action, directing the user to the corresponding view(s) for a particular model. Listing 2.5 provides a snippet illustrating the XML structure for defining simple List/Tree and Form views along with their corresponding action and menu items, demonstrating how these components are linked together to create navigable interfaces in the Odoo Web Client Backend.

Listing 2.5: Definition of the Topic Menus/ Actions /Views in XML.

```
<odoo>
    <data>

        <!-- Main Menu -->
        <menuitem id="pfe_main_menu" name="Dissertation Management" sequence="1"/>

        <!-- Submenu -->
        <menuitem id="menu_topic" name="Topic" parent="pfe_main_menu" action="
            ↪ action_pfe_topic"/>

        <!-- Action: Opens the list/form view for PFE Topics -->
        <record id="action_pfe_topic" model="ir.actions.act_window">
            <field name="name">Topics</field>
            <field name="res_model">pfe.topic</field>
            <field name="view_mode">kanban,tree,form</field>
        </record>

        <!-- Topic Tree View -->
        <record id="view_topic_tree" model="ir.ui.view">
            <field name="name">topic.tree</field>
            <field name="model">pfe.topic</field>
            <field name="arch" type="xml">
                <tree>
                    <field name="name"/>
                    <field name="title"/>
                    <field name="supervisor_id"/>
                    <field name="state"/>
                </tree>
            </field>
        </record>

        <!-- Topic Form View -->
        <record id="view_topic_form" model="ir.ui.view">
            <field name="name">topic.form</field>
            <field name="model">pfe.topic</field>
            <field name="arch" type="xml">
                <form string="Topic Information">
                    <sheet>
                        <header>
                            <button name="action_set_pending_button" type="object"
                                ↪  string="Set Pending"
                                    class="oe_highlight btn-warning"
                                    groups="pfe.group_committee_scientifique"
                                    attrs="{'invisible': [('state', '=', 'pending
                                        ↪ ')]}"/>
                        </header>
                        <div class="oe_title">
                            <h2 style="text-align:center; font-weight:bold; font-
                                ↪ family:'Times New Roman'; color:black;">
                                Topic Information
                            </h2>
                        </div>
                        <group string="General Information" colspan="2">
                            <field name="name"/>
                            <field name="title" required="1"/>
                            <field name="supervisor_id"
                                   domain="[('category_ids.name', '=', 'Teacher')
                                       ↪ ]"
                                   options="{'no_create': True}"/>
                        </group>
                        <group string="Details" colspan="2">
                            <field name="description" widget="html" placeholder="
                                ↪ Write a description here..."/>
                            <field name="tools" widget="html" placeholder="Write
                                ↪ tools here..."/>
                            <field name="category_ids" widget="many2many_tags"
                                ↪ options="{'color_field': 'color'}"/>
                            <field name="reference" required="1"/>
                        </group>
                    </sheet>
                </form>
            </field>
        </record>

    </data>
</odoo>
```

2.4.3 Web Portal Implementation

Custom web pages and functionalities for the Student and Teacher portals were implemented by extending Odoo's module. QWeb templates, typically located under `views/` or `static/src/xml/` within the module, were created to define the visual structure of the portal pages using standard HTML syntax augmented with Odoo-specific directives for dynamic content and control flow.

Dedicated Python controllers were developed in the `controllers/` directory to handle the server-side logic for these portal pages. These controllers receive user requests for specific portal URLs, interact with the PFE models (using the ORM) to fetch or manipulate data (e.g., processing a teacher's topic proposal submission, saving a teacher's inputs, or processing a student's topic selection) and then render the appropriate QWeb templates with the relevant information to be displayed in the user's browser.

This approach provided custom, user-friendly interfaces for teachers to propose topics and record evaluations and for students to choose from validated topics and track their project status, accessible outside the main Odoo Web Client Backend environment.

2.4.4 System Demonstration

The implemented module offers distinct interfaces adapted to the roles and responsibilities of its users within the PFE management system. As designed in Chapter 2, the system distinguishes between backend users—such as Department Heads and Scientific Committee members—and portal users like Students and Teachers.

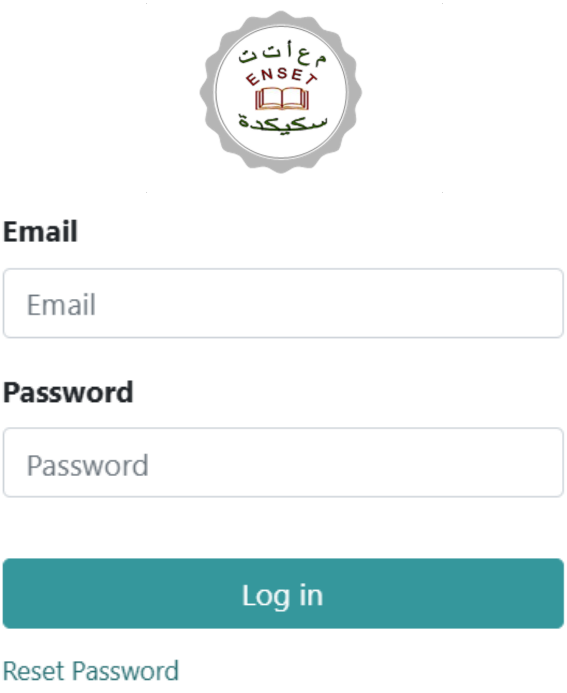


Figure 2.9: Screenshot: Login Page.

Figure 2.9 presents the authentication screen through which users securely access their respective user interface. Once authenticated, backend users are provided with a rich admin-

istrative interface via Odoo's Web Client. This interface facilitates centralized management of dissertation processes, including topic validation, group supervision, and jury coordination.

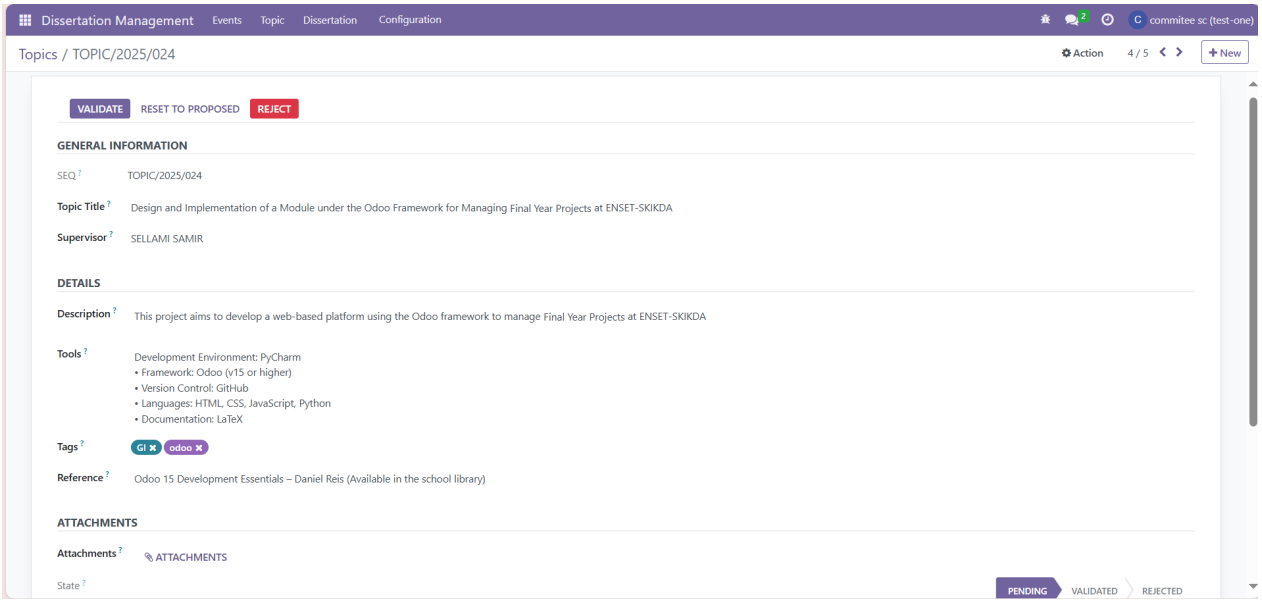


Figure 2.10: Screenshot: Scientific Committee Validates a Topic.

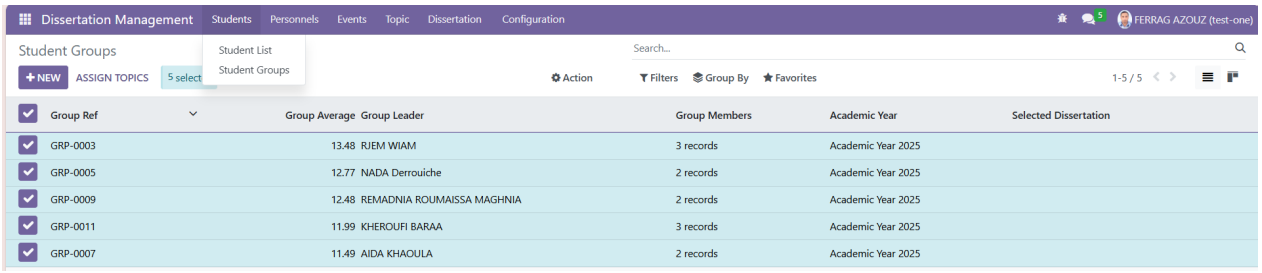


Figure 2.11: Screenshot: Department Head Assigns Dissertation Topics.

Figures 2.10 and 2.11 illustrate key administrative operations: topic validation by the Scientific Committee and the topic assignment process by the Department Head. The Odoo backend provides structured List and Form views that enable efficient navigation, real-time feedback, and editing functionalities.

In parallel, dedicated web portals were developed for Students and Teachers, extending usability beyond the backend and improving accessibility. These portals are accessible via the main website landing page (see Figure 2.12).

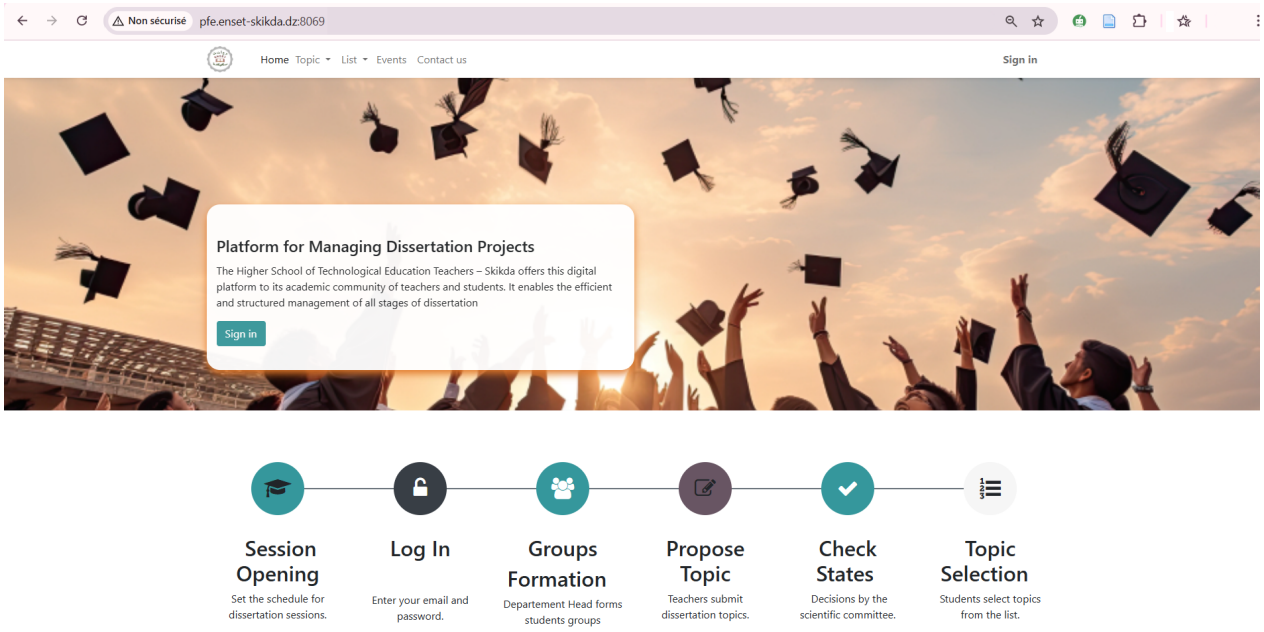


Figure 2.12: Screenshot: Website Main Landing Page.

For public users and Students access, the portal presents lists of validated dissertation topics along with their details (Figure 2.13), allowing students to explore available topics and select their preferences through a guided interface.

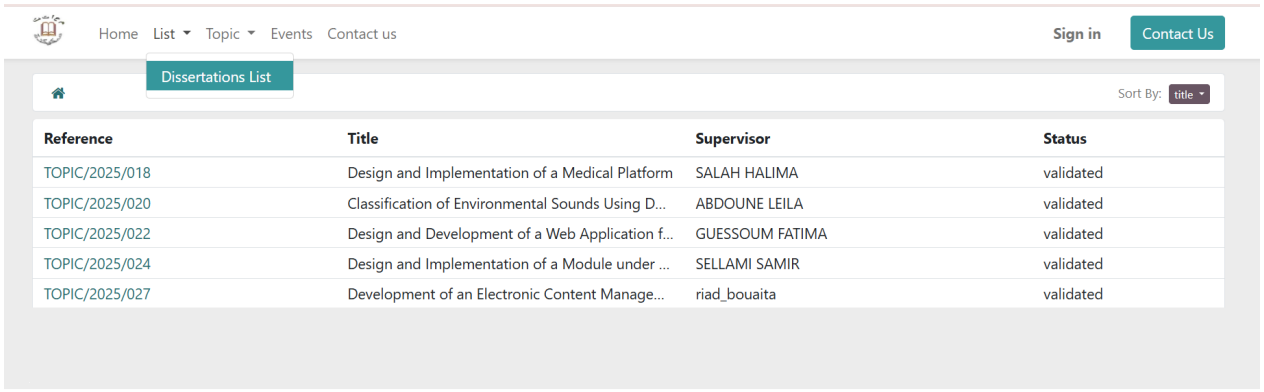


Figure 2.13: Screenshot: Topic List in Frontend Website.

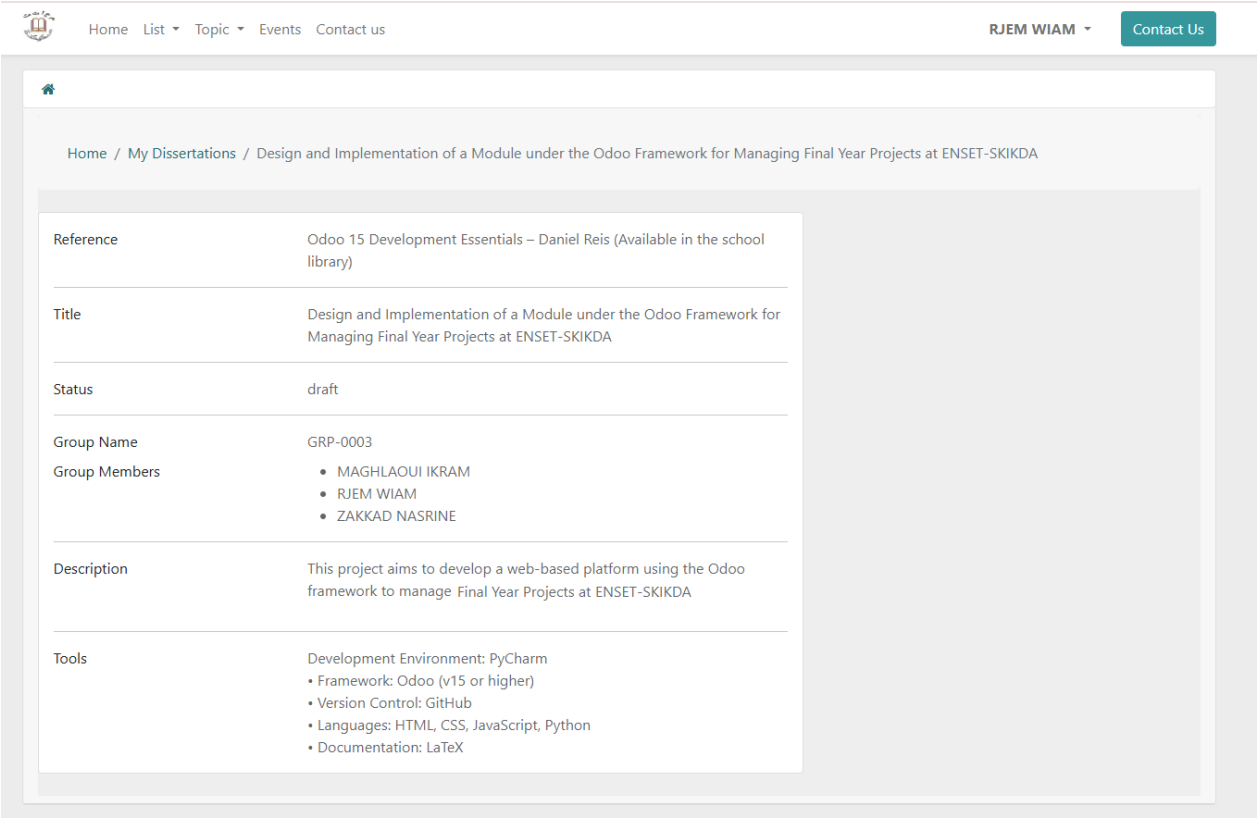


Figure 2.14: Screenshot: Portal Interface for Student.

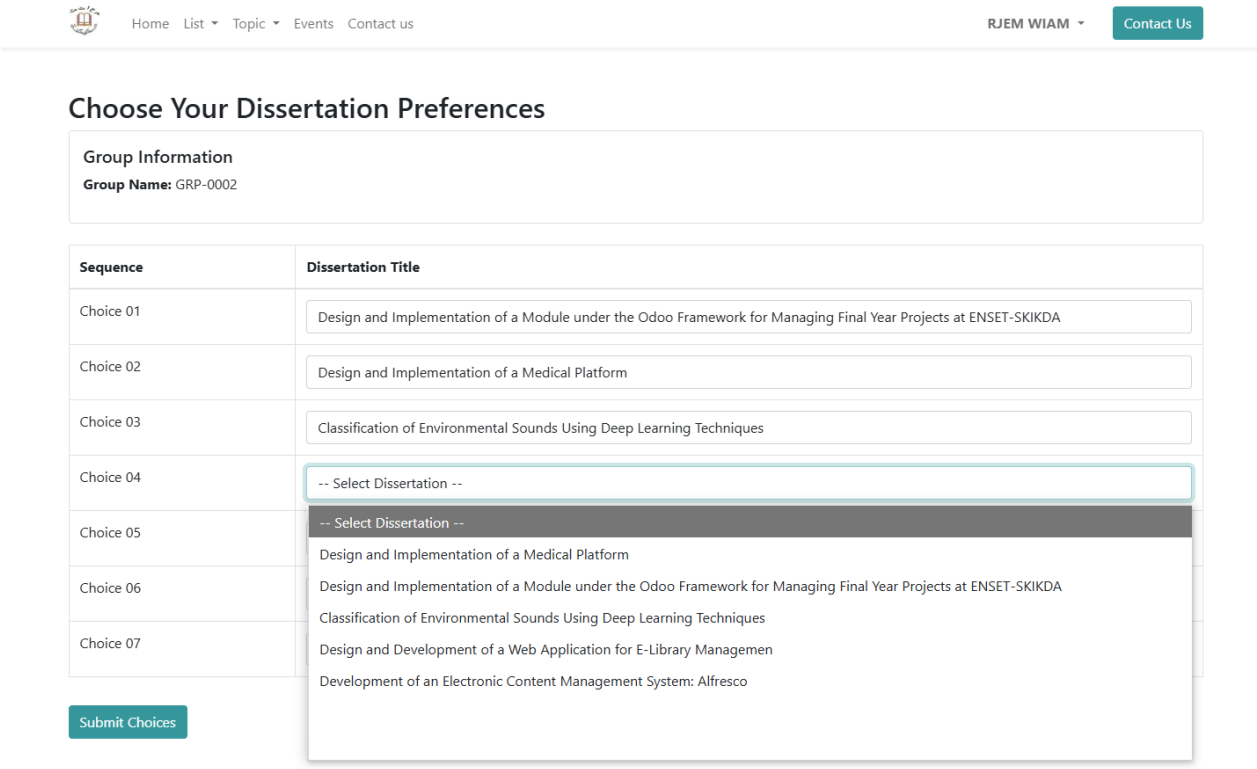


Figure 2.15: Screenshot: Student Topic Choices.

Figures 2.14 and 2.15 highlight the Student portal, through which students can view

their group's dissertation, consult status, and choose validated topics. The portal provides a transparent and intuitive experience aligned with student needs.

On the other hand, Teachers interact with a dedicated portal (Figure 2.16) where they can view their proposed topics and associated dissertations. They can also submit new topic proposals through an intuitive submission form (Figure 2.17).

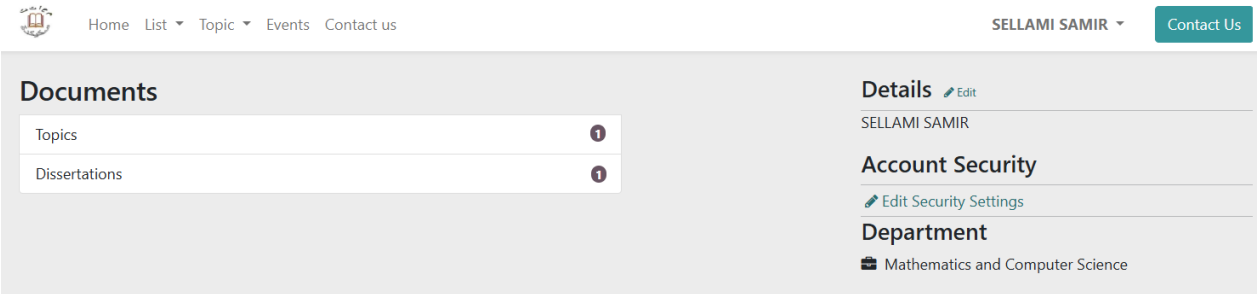


Figure 2.16: Screenshot: Portal Interface for Teacher.

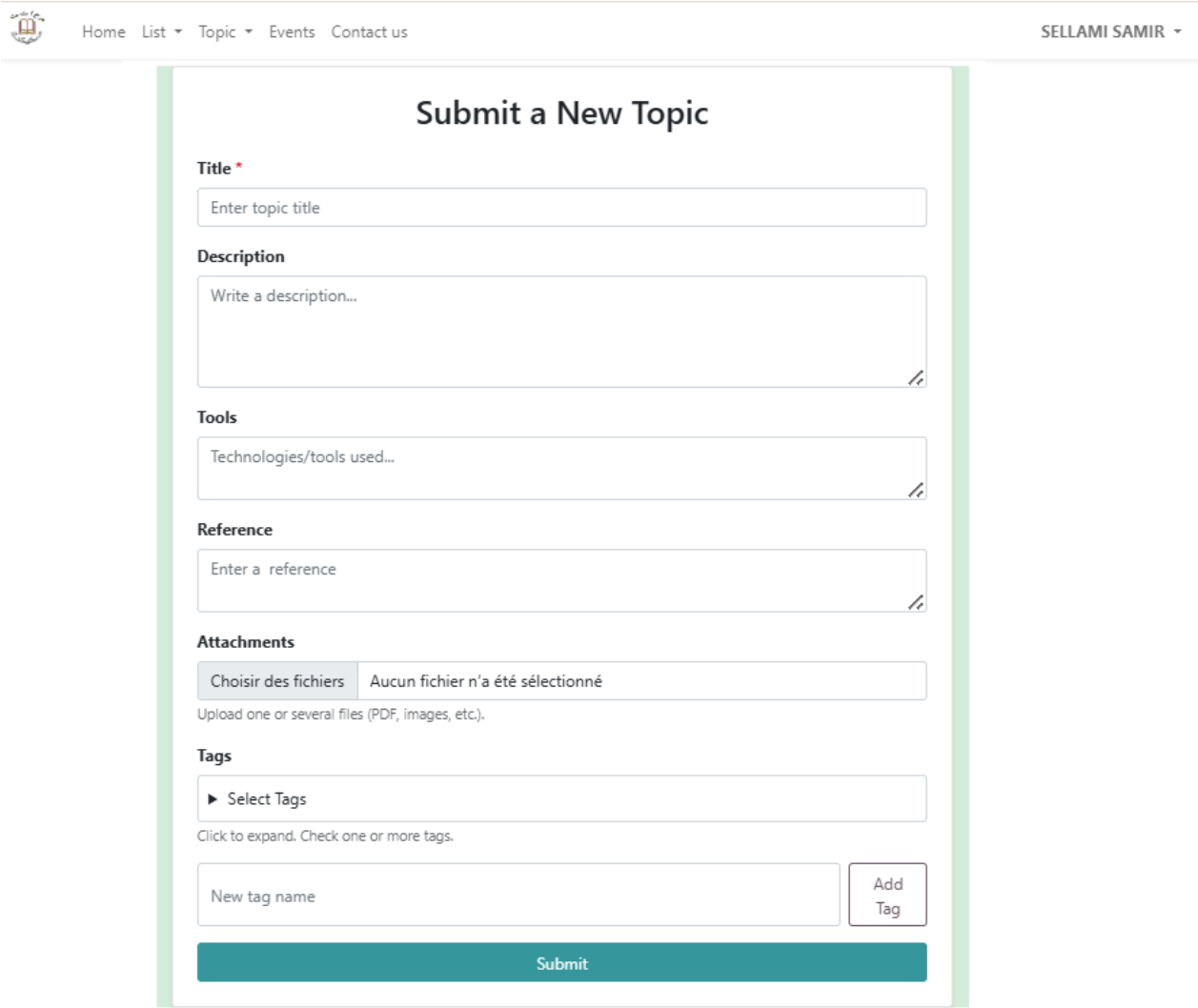


Figure 2.17: Screenshot: Teacher Topic Submission Form.

Overall, this demonstration confirms that the system provides well-differentiated, role-specific interfaces. By combining Odoo's backend flexibility with user-friendly frontend website and portals, the platform ensures both administrative efficiency and accessibility for students and faculty alike. It is important to note that the screenshots presented here represent only a subset of the system's functionalities. The full application includes additional views, forms, and interactions that could not be fully illustrated within the scope of this section.

2.5 Source Code and Repository

The complete source code for the *"PFE Management ENSET-Skikda"* Odoo module—encompassing all implemented models, views, security definitions, data files, controllers, and web-portal components—is publicly available for review and further development on GitHub. The repository can be accessed at (See Figure 2.18).:

`https://github.com/SamRepository/odoo_dev/tree/16.0`

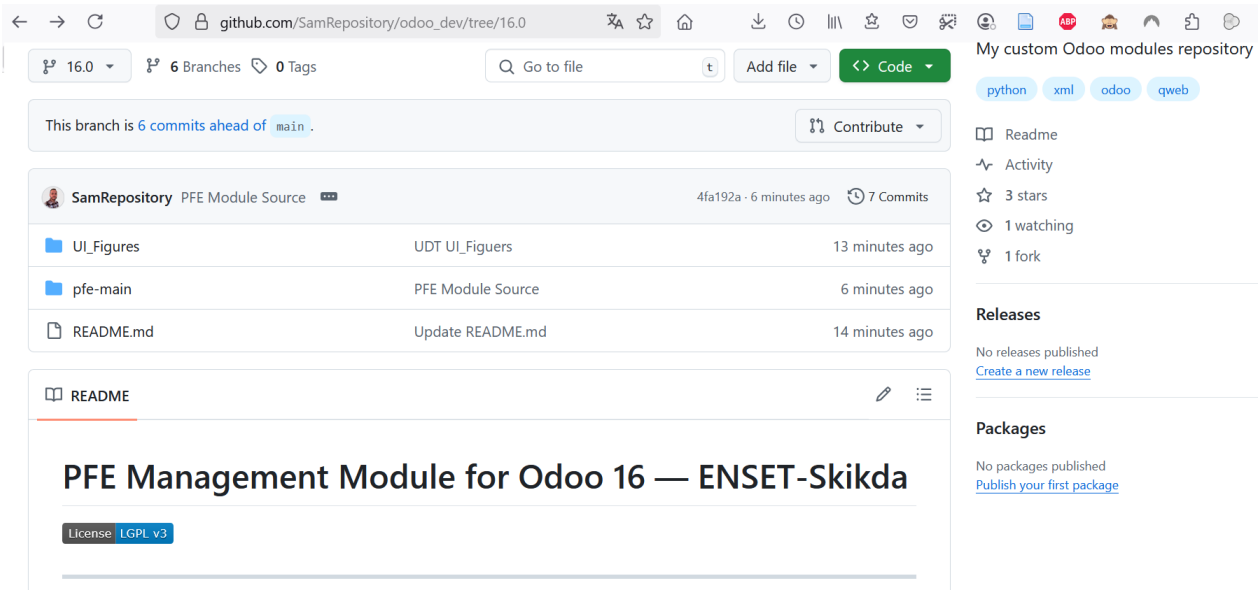


Figure 2.18: GitHub repository page of the *PFE Management ENSET-Skikda* module.

This repository provides full transparency into the implementation details and serves as a foundation for potential contributions, extensions, or deployment in other academic settings.

2.6 Deployment

The developed Odoo PFE module, along with the required Odoo server instance and its dependencies, was deployed onto a virtual server environment hosted at the ENSET-Skikda IT service and machines room. The deployment infrastructure utilized a Proxmox Virtual Server running the Linux operating system (Ubuntu 24.04), which provides a stable and flexible platform for hosting. The deployment process involved several key steps:

1. Provisioning and configuring a suitable VM within the Proxmox environment with adequate computational resources (CPU, RAM) and storage to host the Odoo instance and its database (See Figure 2.19).

2. Installing the chosen Linux distribution (Ubuntu 24.04 Server) and necessary system packages required by Odoo and PostgreSQL.

3. Installing and configuring the PostgreSQL database server, creating a dedicated database for the Odoo instance and configuring appropriate user permissions.

4. Setting up the Odoo 16 (Community Edition) server instance, including installing the required Python 3.10 version and libraries, typically within a dedicated virtual environment (VENV) to manage dependencies.

5. Cloning the source code of the Odoo community edition and the custom PFE module from the GitHub repository into the designated Odoo addons path.

6. Configuring the Odoo server's configuration file (`odoo.conf`) to connect to the PostgreSQL database, specify the addons path including the custom module and set other relevant parameters.

7. Starting the Odoo server and installing the *"PFE Management ENSET-Skikda"* module via the Odoo Apps interface after updating the list of available applications, which loads the XML and CSV data files into the database.

8. Setting up a production-ready web server (Nginx Web Server) to act as a reverse proxy for the Odoo instance. This server handles incoming HTTP requests from users and forwards them to the Odoo server, while also managing static files and potentially handling SSL termination for secure access (HTTPS).

9. Configuring DNS records to map a specific domain name (`pfe.enset-skikda.dz:8069`[17]) to the server's public IP address, making the deployed application accessible to users via a standard web URL.

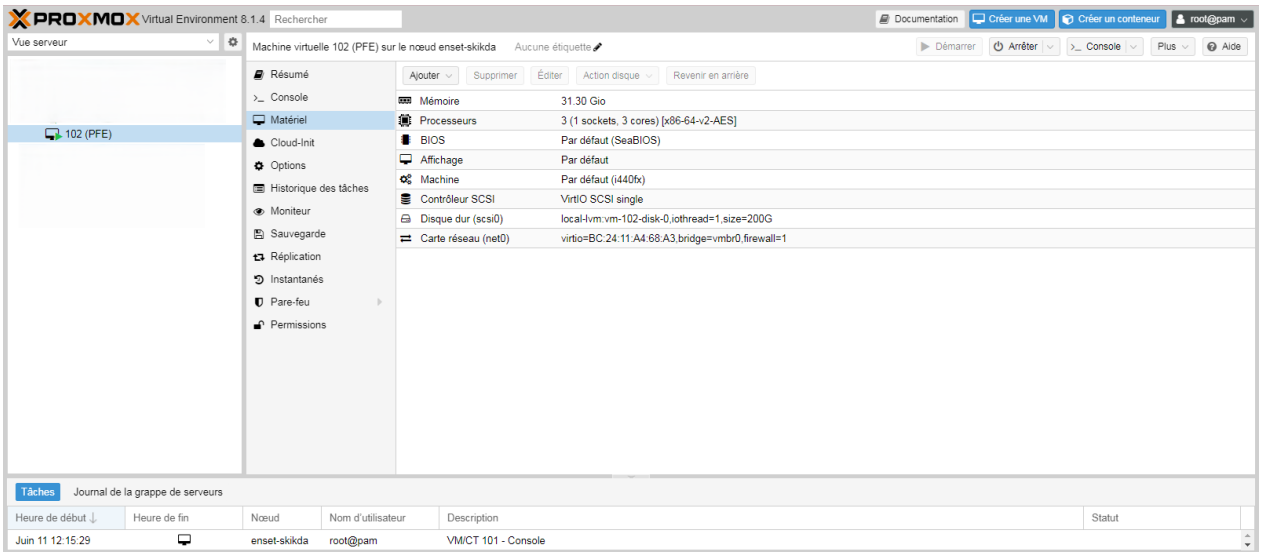


Figure 2.19: Screenshot: Proxmox VM (Ubuntu) Setup for Odoo.

[17] `http://pfe.enset-skikda.dz:8069/`

2.7 Challenges Faced

During the implementation, several challenges were encountered, primarily related to translating the detailed design specifications into Odoo's specific development patterns and managing the project scope within the available time.

- **Time Constraints:** The strict timeline allocated for the dissertation project was a primary constraint. This necessitated prioritizing core functionalities and implementing them efficiently, leading to the omission or simplification of some desirable features and preventing extensive fine-tuning and optimization of user interfaces and backend processes.
- **Incomplete Workflow Coverage:** Due to the time limitations, certain sub-processes or specific edge cases within the comprehensive PFE management workflow, such dissertation evaluation and defenses schudelling, fully automated notification systems for all status changes, detailed analytical reporting and dashboards, advanced features like plagiarism checking integration, or robust handling of complex scenarios like project cancellations or changes in committee members, could not be completely implemented or thoroughly tested.

These challenges highlight potential areas for future development and iteration to enhance the module's completeness, robustness and usability, moving towards a fully comprehensive PFE management solution.

2.8 Conclusion

This chapter presented the technical implementation details of the PFE management module for ENSET-Skikda, detailing the module structure, the utilization of Odoo's system architecture and key development tools and the specific implementation of data models, user interfaces (both backend and frontend portals) and the security access control system. The implementation successfully leveraged the Odoo framework to create a functional application addressing the core requirements of the PFE management process for different user roles. While constraints related to project scope and time limitations were encountered, the completed functional module provides a robust foundation for future expansion and refinement. The availability of the full source code on GitHub facilitates transparency and potential future development by the institution or the wider Odoo community.

GENERAL CONCLUSION AND PERSPECTIVES

Summary of the Project

The present dissertation focused on addressing the challenges associated with the manual management of final-year dissertation projects at the Higher School of Technological Education in Skikda (ENSET-Skikda). The traditional process, characterized by paper-based submissions, delayed communication and inefficient topic assignment, was shown to be time-consuming, error-prone and lacking transparency.

To overcome these limitations, we proposed a workflow design and implemented a custom web module using the Odoo framework. The objective was to streamline the workflow for all key stakeholders, including department head, students, teachers and the scientific committee. The project followed a structured methodology that included requirement analysis, system modeling using UML diagrams, iterative development with Agile principles and partial deployment on Proxmox Ubuntu virtual server environment.

The final system supports core functionalities such as students group formation, topic announcement and validation, submission of dissertation files, assignment of reviewers and result publication. The modular and scalable nature of the Odoo framework enabled us to design a reusable and extensible solution that meets the specific needs of the academic context.

Challenges Faced

Throughout the development process, several challenges were encountered. A major difficulty was the steep learning curve of the Odoo framework, particularly understanding its object-relational mapping (ORM), module inheritance mechanisms and QWeb-based templates definitions. In addition, technical constraints related to environment setup, version compatibility and module integration posed obstacles during implementation.

Time management was also a concern, as the academic calendar imposed strict deadlines, leaving limited time for user testing and refinement. Nevertheless, most of these challenges were addressed through documentation review, consultation with supervisors and continuous testing.

Limitations of the Work

Despite the significant progress achieved, several limitations remain. Due to time constraints, not all phases of the dissertation workflow were fully implemented—particularly the dissertation evaluation procedures and the generation of detailed reporting dashboards.

Integration with external institutional platforms, such as the national student record system Progres, was also outside the scope of this project and remains a planned future enhancement.

Another limitation lies in the localization of the user interface. Currently, the system's front-end and backend views are available only in English. Arabic translation of labels and content has not yet been implemented. However, the Odoo framework provides built-in internationalization (i18n) support and facilitates translation workflows, which will be leveraged in future iterations to provide full bilingual support.

Additionally, user testing was limited to a small internal group and performance benchmarking under real academic workloads was not conducted. Broader usability testing and deployment in a live academic environment will be necessary to validate robustness, scalability and user experience.

Future Enhancements

Future work can address the current limitations and extend the platform with new features. Recommendations include:

- Implementation of multilingual support by leveraging Odoo's built-in internationalization (i18n) framework to enable interface translation and localization for Arabic and other languages along with detailed documentation user guides.
- Integration with external platforms such as Progres for importing student administrative data, GitHub for source code tracking and Turnitin for plagiarism detection.
- Enhancement of reporting tools for supervisors and administrators, including dashboards and performance metrics.
- Full implementation of dedicated reviewers and jury members with responsive design and improved user experience.
- Deployment of load-balancing techniques to ensure availability and scalability during peak usage.
- Development of AI-powered features such as automatic project-topic suggestions and matching based on students' academic records and preferences.

Final Reflection

This project served as a valuable learning experience, combining theoretical knowledge with practical application. It provided deep insight into Odoo development, collaborative software engineering and academic workflow digitization. The resulting application marks an important step toward modernizing dissertation management at ENSET-Skikda, with promising potential for institutional adoption and future expansion.

LIST OF ACRONYMS

- **API:** Application Programming Interface.
- **CPU:** Central Processing Unit.
- **CRUD:** Create, Read, Update, Delete.
- **CSS:** Cascading Style Sheets.
- **DNS:** Domain Name System.
- **ENSET:** École Normale Supérieure de l'Enseignement Technologique.
- **ERP:** Enterprise Resource Planning.
- **FTP:** File Transfer Protocol.
- **HTML:** HyperText Markup Language.
- **HTTP:** HyperText Transfer Protocol.
- **HTTPS:** HyperText Transfer Protocol Secure.
- **IDE:** Integrated Development Environment.
- **IMAP:** Internet Message Access Protocol.
- **IP:** Internet Protocol.
- **JSON:** JavaScript Object Notation.
- **JS:** JavaScript.
- **MVC:** Model-View-Controller.
- **NFRs:** Non-Functional Requirements.
- **ORM:** Object-Relational Mapping.
- **OWL:** Odoo Web Library.

- **PDF:** Portable Document Format.
- **PEM:** Professeur de l'Enseignement Moyen.
- **PES:** Professeur de l'Enseignement Secondaire.
- **PFE:** Projet de Fin d'Études.
- **QWeb:** Odoo's Templating Engine.
- **RAM:** Random Access Memory.
- **RBAC:** Role-Based Access Control.
- **RPC:** Remote Procedure Call.
- **SMTP:** Simple Mail Transfer Protocol.
- **SPA:** Single Page Application.
- **SQL:** Structured Query Language.
- **SSL:** Secure Sockets Layer.
- **UI:** User Interface.
- **UML:** Unified Modeling Language.
- **URL:** Uniform Resource Locator.
- **UX:** User Experience.
- **VM:** Virtual Machine.
- **XML:** Extensible Markup Language.

Bibliography

BIBLIOGRAPHY

[1] D. Reis, *Odoo 15 Development Essentials — Fifth Edition: Enhance Your Odoo Development Skills to Create Powerful Business Applications*, 5th ed. Birmingham, United Kingdom: Packt Publishing, Feb. 2022, online: https://www.packtpub.com/product/odoo-15-development-essentials-fifth-edition/9781801813204.

[2] General Secretariat of the Government, "Official journal of the people's democratic republic of algeria, no. 46," 2009. [Online]. Available: https://www.joradp.dz/FTP/jo-francais/2009/F2009046.pdf

[3] ENSET Skikda, "Internal regulations of the national higher school of technology," 2024, accessed 11 Jan 2025. [Online]. Available: http://www.enset-skikda.dz/index.php/enset/2021-11-08-10-14-19.html

[4] People's Democratic Republic of Algeria, "Executive decree no. 05-500 of 29 December 2005 – missions and special rules for non-university higher-education institutions (art. 7)," 2005, government decree.

[5] ENSET-Skikda, "Introductory brochure of the higher school of teachers in skikda," 2024. [Online]. Available: https://eddirasa.com/etudiant-ens-skikda/

[6] ENSET Skikda, "Organizational structure of the national higher school of technology in skikda," 2024. [Online]. Available: http://www.enset-skikda.dz/index.php/enset/2021-11-08-10-16-28.html

[7] Department of Mathematics and Computer Science, ENSET Skikda, "Overview of academic programs and student statistics," Dec. 2024. [Online]. Available: http://www.enset-skikda.dz/index.php

[8] F. Azouz, "Interview with department head: Overview of the workflow of final-year projects in ENSET-Skikda," Dec. 2024, personal communication, December 2024.

[9] H. Khouchmen, "Scientific committee report of the mathematics and computer science department," Dec. 2024, internal report.

[10] H. Khouchmen, "Interview with the leader of the scientific committee on equal averages among student groups," Dec. 2024, personal communication.

[11] L. Souici, "Conception et réalisation d'une application pour la gestion des soutenances," Master's thesis, Université Mouloud Mammeri, Tizi-Ouzou, 2012.

[12] A. Bensadi, "Conception et réalisation d'une application web pour la gestion des projets de fin d'Études," Master's thesis, Université M'hamed Bougara Boumerdès, Boumerdès, 2018.

[13] M. M. Meddah, "Conception et réalisation d'une application web pour la gestion de projets de fin d'Études," Master's thesis, Université Abdelhamid Ibn Badis, Mostaganem, 2021, accepted 21 Mar 2022. [Online]. Available: http://e-biblio.univ-mosta.dz/handle/123456789/20473

[14] V. Semushin, "Improving software development life cycle for Odoo ERP system," Bachelor's thesis, Metropolia University of Applied Sciences, 2023. [Online]. Available: https://www.theseus.fi/handle/10024/801059

[15] R. O. A. E. Abdelmuniem, "Parallel processing for data retrieval in Odoo enterprise resource planning reporting system," Master's thesis, Universiti Malaya, Kuala Lumpur, Apr. 2021. [Online]. Available: http://studentsrepo.um.edu.my/14140/

[16] I. A. Masroor, "The power of Odoo: The all-in-one business software," Sep. 2024. [Online]. Available: https://www.manystrategy.com/what-is-odoo-guide/

[17] R. Srivastava and K. Team, "Advantages and disadvantages of Odoo ERP," Oct. 2024. [Online]. Available: https://www.ksolves.com/blog/odoo/advantages-and-disadvantages-of-odoo-erp

[18] V. Agrawal, "What is Agile methodology: Concepts, process, & benefits," 2025. [Online]. Available: https://www.simplilearn.com/tutorials/agile-scrum-tutorial/what-is-agile

[19] J. A. Highsmith, *Agile Software Development Ecosystems*. Addison–Wesley Professional, 2002. [Online]. Available: https://books.google.com/books/about/Agile_Software_Development_Ecosystems.html?id=uE4FGFOHs2EC

[20] M. Keeling, "Activity 22 – context diagram," in *Design It!* The Pragmatic Bookshelf, 2017. [Online]. Available: https://www.oreilly.com/library/view/design-it/9781680502923/f_0139.xhtml

[21] S. Bennett, K. Lunn, and J. Skelton, *Schaum's Outline of UML*, 2nd ed. McGraw-Hill, 2005.

[22] EdrawSoft, "How to create different UML diagrams?" 2025. [Online]. Available: https://www.edrawsoft.com/fr/uml-introduction.html

[23] S. Taibi, "Une Étude orientée objectifs pour la réalisation d'un module sous Odoo de gestion des stages à l'ENSET-Skikda," Master's thesis, Université du 20 Août 1955, Skikda, 2021. [Online]. Available: https://www.academia.edu/51362167/des_Stages_%C3%A0_lENSET_SKIKDA_R%C3%A9alis%C3%A9_par_Encadr%C3%A9_par

[24] GeeksforGeeks, "Class diagram: (UML)," 2025. [Online]. Available: https://www.geeksforgeeks.org/unified-modeling-language-uml-class-diagrams/

[25] K. Fakhroutdinov, "Uml deployment diagrams," 2009, page updated 27 May 2025. [Online]. Available: https://www.uml-diagrams.org/deployment-diagrams.html