DEVELOPMENT OF THE PUP TAGUIG FACULTY LOADING AND SCHEDULING SYSTEM

A Capstone Project

Presented to

Polytechnic University of the Philippines – Taguig Branch

In Partial Fulfillment of the

Systems Analysis & Design

Second Semester A.Y. 2023-24

Bachelor of Science in Information Technology

By

Martinez, Emmanuel Q.

Malaluan, Kyla Rica C.

Naoe, Adrian B.

Rasquero, Via Clariz A.

JULY 2024

CERTIFICATION OF ORIGINALITY

This is to certify that the research work presented in this capstone project, **DEVELOPMENT OF THE PUP TAGUIG FACULTY LOADING AND SCHEDULING SYSTEM** for the degree Bachelor of Science in Information Technology at the Polytechnic University of the Philippines embodies the result of original and scholarly work carried out by the undersigned. This capstone project does not contain words or ideas taken from published sources or written works that have been accepted as the basis for the award of a degree from any other higher education institution, except where proper referencing and acknowledgement were made.

EMMANUEL Q. MARTINEZ
Researcher

KYLA RICA C. MALALUAN
Researcher

ADRIAN B. NAOE
Researcher

VIA CLARIZ A. RASQUERO
Researcher

_____
Date Signed

CHAPTER 1

THE PROBLEM AND ITS BACKGROUND

1. Introduction

In the ever-evolving landscape of academic institutions, the efficient management of faculty resources and course scheduling stands as a cornerstone for fostering a dynamic and effective learning environment. The Polytechnic University of the Philippines (PUP) Taguig, with its rich history and commitment to excellence, has continually sought to optimize its administrative processes to meet the growing demands of a diverse student body and changing educational paradigms.

The PUP Taguig Faculty Loading and Scheduling System is a vital component in the university's administrative machinery which traces its roots back to an era when manual processes dominated academic resource allocation. The system was conceived as a response to the burgeoning need for a more systematic approach to faculty loading, a pivotal task in ensuring that the right educators are assigned to the right courses.

Over the years, the system has been a witness to the ebb and flow of academic transformations. From the integration of new programs to shifts in pedagogical methodologies, the demands on the faculty loading and scheduling process have grown in complexity and diversity. What once sufficed as an adequate solution has found itself challenged by the accelerating pace of change in the educational landscape.

As academic programs diversified, accreditation standards evolved, and technology became an integral part of education, the limitations of the existing system became increasingly apparent. Manual entry processes, lack of adaptability to new curricula, and a user interface that struggled to meet modern usability standards have cumulatively contributed to inefficiencies in faculty resource allocation and scheduling.

The challenges faced by the PUP Faculty Loading and Scheduling System are not unique to this institution alone; they mirror a broader trend in educational institutions globally. The limitations of traditional systems in adapting to the demands of contemporary education have become a common narrative, underscoring the need for innovative solutions that blend technological advancements with the intricacies of academic administration.

This study seeks to bridge the historical foundations of the PUP Taguig Faculty Loading and Scheduling System with the current imperatives of educational administration. By understanding where the system has come from, we gain a nuanced perspective on the intricacies and nuances that have shaped its current state. The challenges posed by the past illuminate a path forward—a path of enhancement which integrates the ever-evolving curricula, new technological advancements, and user-centric design principles.

As we delve into the existing PUP Taguig Faculty Loading and Scheduling System, we do so with the intent to not just address existing deficiencies but to contribute to the enhancement and progress that propels PUP's academic administration into a future characterized by efficiency, adaptability, and a commitment to providing an optimal learning experience for both faculty and students.

1.1 Overview of the Current System Process

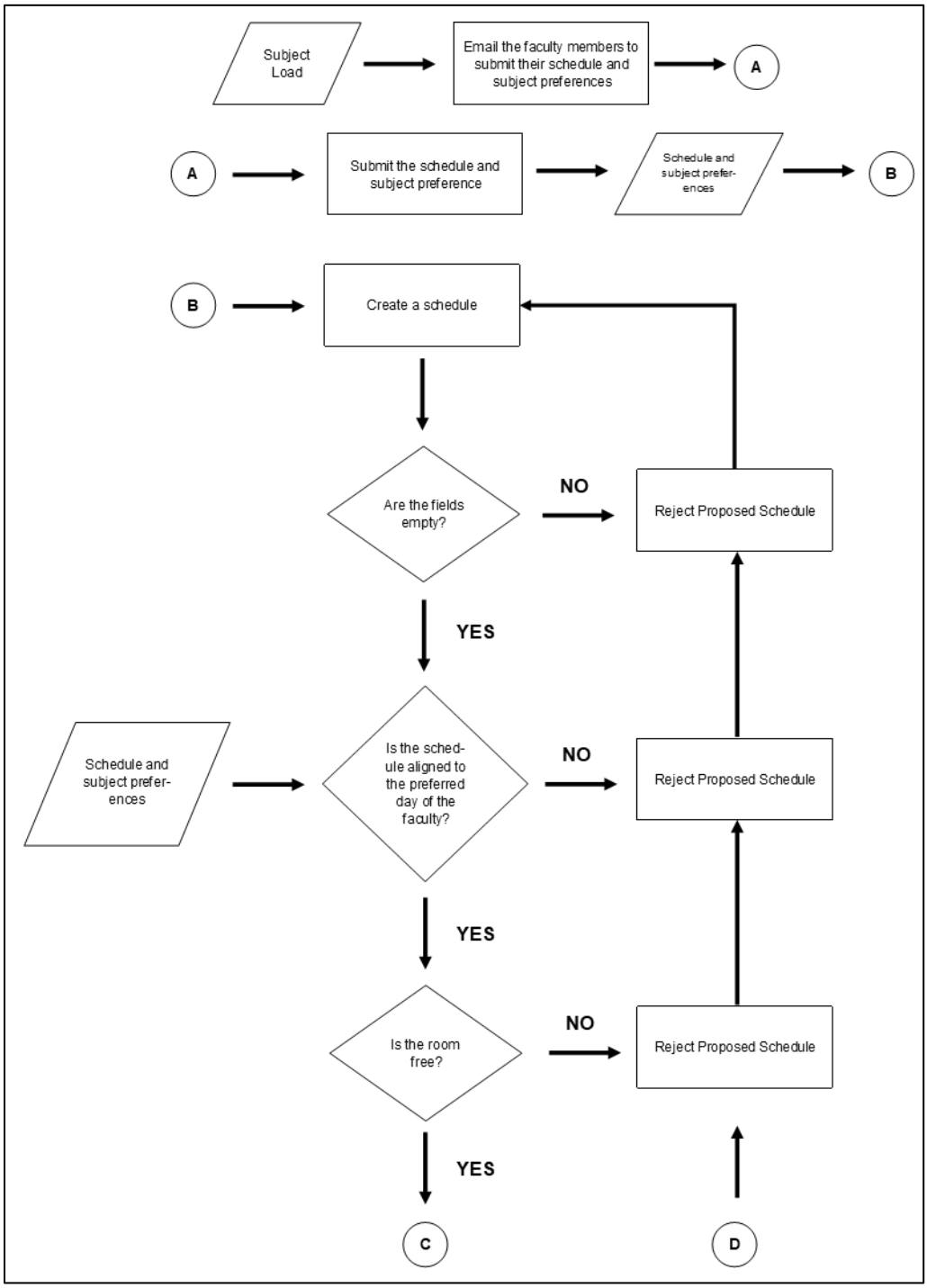


Figure 1a. The current process flow of the PUPT-FLS System

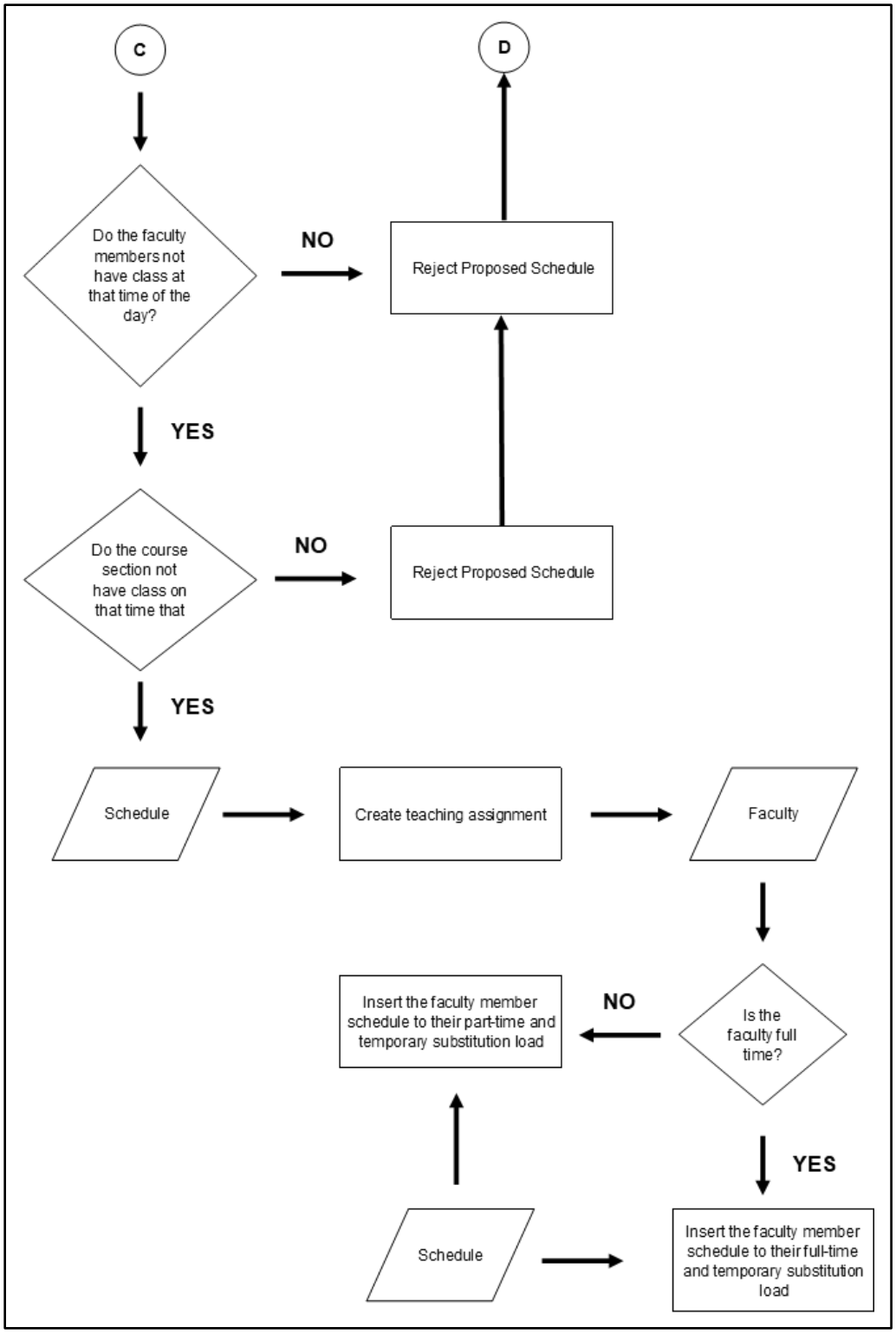


Figure 1b. The current process flow of the PUPT-FLS System

1.2 Background of the Organization

1.2.1. Organizational Hierarchy

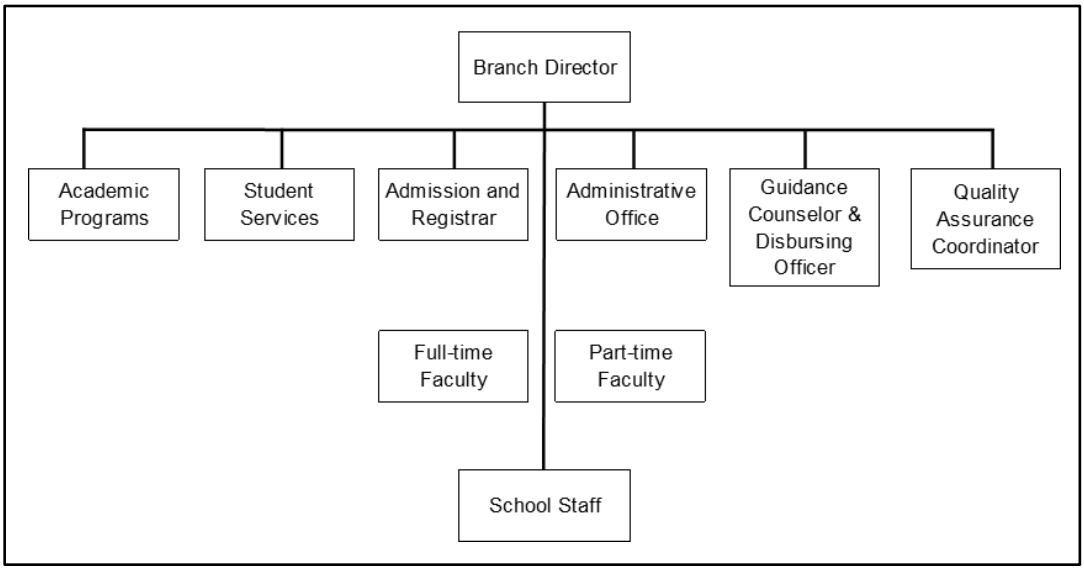


Figure 2. Organizational Hierarchy Overview at PUP Taguig

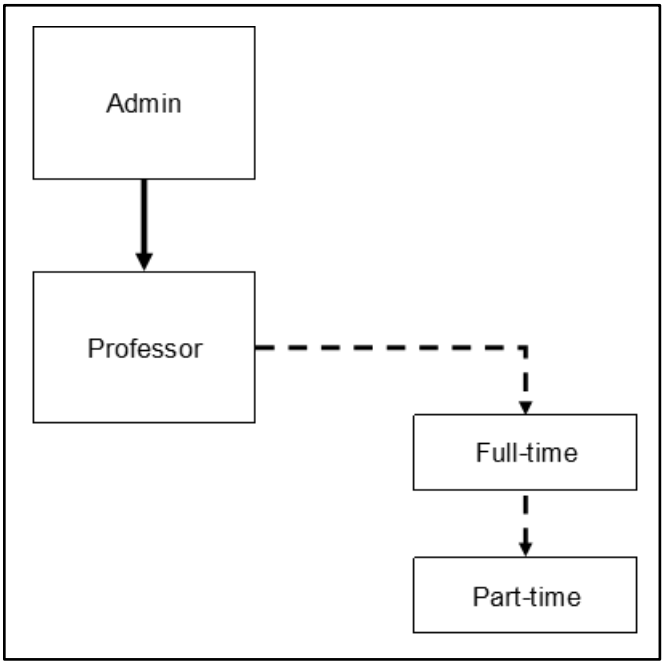


Figure 3. The Flow of Assigning Faculty Roles in PUP Taguig FLS System

1.2.2. Policies and Procedures

The following are the policies and procedures of PUP Taguig in terms of faculty loading:

- **Classification of Faculty Members**
  - Full-Time Faculty Member
  - Designee Faculty Member
  - Part-time Faculty Member
  - Temporary Faculty Member

Classification	Teaching Load (maximum units)
Full-Time	15
Designee	12
Part-Time	12
Temporary	12

The 2024 Faculty Manual outlines the teaching load and units allowed for different types of faculty members at the Polytechnic University of the Philippines. Below is an analysis of the different types of faculty and the units allowed for each:

1. **Full-time Faculty**

- **Teaching Load:** Full-time faculty members are assigned a standard teaching load of **15 units per semester**. This includes 15 hours of teaching per week along with research, extension, and consultation duties.
- **Total Weekly Hours:** The workweek for full-time faculty is **40 hours**, with **15 hours dedicated to teaching**, **10 hours to quasi-teaching activities** (research and extension), and other duties such as consultation.

2. **Faculty with Administrative Designations:**

   - **Deans and Directors:** Faculty designated as deans or directors have a reduced teaching load of **3 units**.
   - **Associate Deans, Chairpersons, and Faculty Assistants:**

     These positions have a reduced load of **6 units**.

3. **Part-time Faculty:**

   - **Teaching Load:** Part-time faculty members can teach a maximum of **12 units per semester**, with any teaching load beyond this considered as temporary substitution (TS) load.
   - **Appointment:** Part-time faculty members are appointed on a semestral basis, depending on available teaching load.

4. **Special Designations:**

   - Faculty involved in **extension services** are eligible for a **6-unit reduction** from their regular load but may still handle part-time teaching loads, capped at **12 units**.

Factors in selecting teaching load and assigning schedules

Faculty availability:

- This takes into account the preferred schedules submitted by the faculty members and their preferred subjects

Faculty academic qualifications:

- the faculty meets the minimum requirements of educational level for which he/she is being hired.

Faculty Evaluation:

- refers to the result of the semestral evaluation from the students. If a faculty member has a three (3) consecutive evaluation grade below 2.0 (“fair”), their maximum load will be reduced

Room availability

- Refers to the available regular and laboratory rooms

Course availability

- Refers to the availability of offered courses under a certain program

Program availability

- Refers to the availability of offered programs in a school year

Administrative Roles

- If a faculty member is a branch official or designee, their teaching load is reduced compared to the full-time/regular faculty

1.3 Statement of the Problem

The current PUP Faculty Loading and Scheduling System face inefficiencies in adapting to the evolving educational landscape. Manual processes and limitations in system capabilities hinder the optimal allocation of faculty resources and the seamless integration of new curricula. These challenges contribute to scheduling conflicts, reduced system performance, and a less-than-optimal user experience. Addressing these issues is critical to fostering an effective and adaptive academic environment.

The specific challenges in the Faculty Loading and Scheduling project are as follows:

- *Inefficient and Time-Consuming Manual Processes:* The current process requires excessive paperwork and repeated data entry as each assignment must be recorded multiple times across different piles of information (faculty, rooms, courses, and programs) which leads to substantial administrative workload burden.

- *High Error Rate and Inconsistencies:* The manual handling of scheduling tasks often results high error rates in assigning faculty, rooms, and time slots, leading to scheduling conflicts, missing schedules, and inconsistencies across curricula impacting both faculty and students.

- *Failure of Existing Online Solutions:* Previous attempts to digitize the scheduling process have failed due to outdated UI/UX and critical functional flaws such as inability to handle multiple curricula, resulting in lack of adoption and continued dependence on inefficient manual methods.

- *Lack of Integrated Data Management:* The absence of a centralized platform for managing faculty preferences, curriculum requirements, program details, and room allocations leads to fragmented data handling, making it difficult to generate coherent and optimized schedules.

1.4 Theoretical Framework

The development of the enhanced PUP Taguig Faculty Loading and Scheduling System (FLSS) is grounded in theoretical concept of the study conducted by Raymundo et al. (2016) titled *Development of an Automated Faculty Loading, Room Utilization, Subject and Student Scheduling System (AFLRUS4) for Bulacan Polytechnic College* which offers insights and evidence that reinforce the key theoretical concepts underpinning the FLSS, with a particular focus on the aspects of automation, efficiency, and data management.

The findings of Raymundo et al. provide tangible evidence of the positive impact of automation on the scheduling processes within an academic institution. The AFLRUS4 system demonstrated the capability to streamline intricate scheduling tasks, showcasing a reduction in manual effort. This aligns seamlessly with the theoretical concept of efficiency embedded in the FLSS framework. The study illustrates how the implementation of an automated system not only expedites scheduling procedures but also minimizes the likelihood of errors, contributing to a more streamlined and error-free allocation of faculty resources. Furthermore, the study by Raymundo et al. highlights the practical importance of system flexibility within an FLSS. Academic environments are dynamic, subject to changes in faculty availability, student enrollment, and course requirements. The AFLRUS4 system's adaptability to such changes underscores the necessity for a flexible FLSS that can accommodate the evolving needs of an academic institution. This flexibility aligns closely with the theoretical framework's emphasis on optimizing faculty resources by ensuring that the FLSS can adjust to the dynamic nature of academic operations.

1.5 Conceptual Framework

The *Optimizing Faculty Workload Management: A Revamped Approach for PUP Taguig's Faculty Loading and Scheduling System* aims to improve the overall performance of this system for all users. This study is inspired by the Input-Process-Output (IPO) model for enhancing the Scheduling and Loading system. It enables users to add new curricula, access availability schedules, store data in the database, and experience a seamlessly user-friendly interface while using this system.

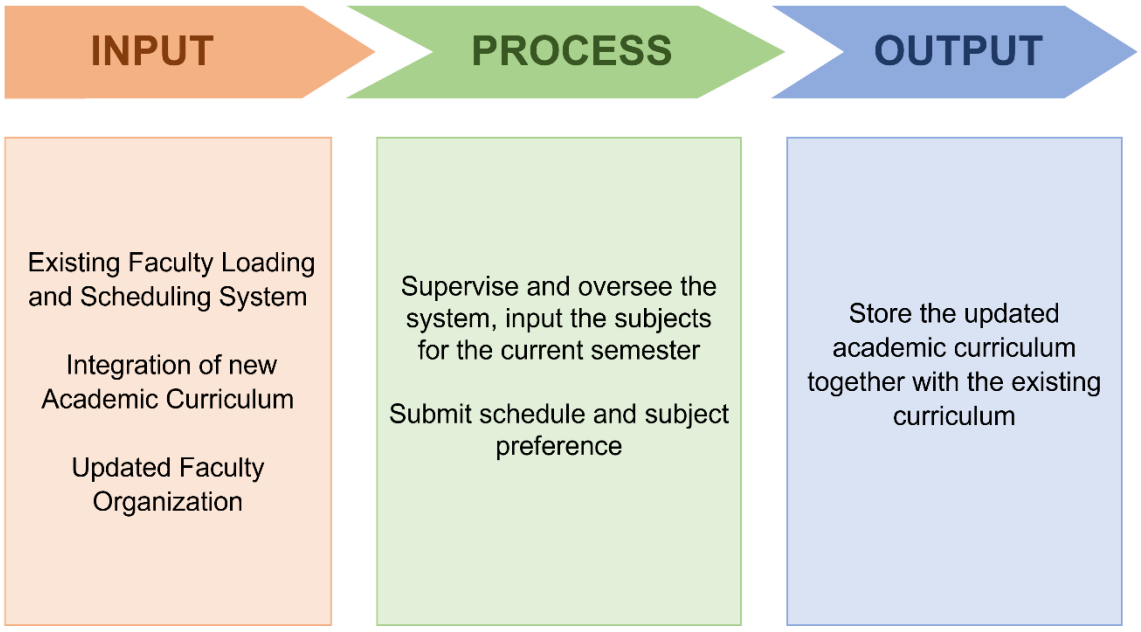


Figure 4. Input-Process-Output Model of the PUPT Faculty Loading and Scheduling System

1.6 Objectives

General Objectives

The overarching goal of this project is to enhance the efficiency and effectiveness of faculty workload management and class scheduling at PUP Taguig.

By introducing a more comprehensive and innovative approach, the enhanced system aims to provide administrators with a powerful tool to navigate the intricacies of faculty workload and scheduling.

Specific Objectives

In pursuit of optimizing the Faculty Loading and Scheduling System, the project is set to achieve key milestones, as outlined in the following objectives:

- *Streamlined Scheduling Process:* To develop a web-based app that streamlines the faculty loading and scheduling process, eliminating excessive paperwork and repeated data entry across different categories, thereby decreasing the administrative workload and improving efficiency.

- *Error Reduction and Conflict Prevention:* To implement conflict detection features within the system to eliminate errors associated with manual scheduling to ensure that faculty assignments, room allocation, and schedules are accurate, reliable, and free of conflicts.

- *Modern, User-Friendly System Design:* To design a modern, user-friendly, and fully functional web-based scheduling system that addresses the limitations of previous systems, including the inability to handle multiple curricula, encouraging widespread adoption for faculty and administrators.

- *Integrated Data Management Platform:* To establish an integrated data management platform that centralizes faculty preferences, curriculum requirements, program details, and room allocations, facilitating seamless data handling and enabling the generation of optimized and coherent schedules.

1.7 Scope and Delimitation

This study focuses on implementing specific key enhancements to the existing PUP Taguig Faculty Loading and Scheduling System. These enhancements include the integration of the new curriculum, migration to the LARAVEL framework, and a redesign of the user interface for improved usability and accessibility. Academic stakeholders such as Academic Administrators, Curriculum Developers, and Faculty Members who interact with the system will be primary involved. Their experiences and feedback are crucial for evaluating the effectiveness of the enhancements. The scope also includes the measurement and analysis of system performance metrics, such as response time, resource utilization, and overall efficiency. These metrics will provide quantitative insights into the impact of the enhancements.

On the other hand, this study is limited to the Faculty Loading and Scheduling System within PUP Taguig and does not extend to other systems or departments within PUP Taguig nor any other PUP branches; integration with external systems beyond the scope of faculty loading and scheduling process will not be addressed in this research. While security features will be implemented and evaluated in the enhanced system, the study does not involve ethical hacking or extensive security testing beyond the scope of system evaluation.

The study does not extend to a long-term impact assessment of the enhanced system beyond the immediate post-implementation phase. The focus is on the initial evaluation and user feedback. Furthermore, this study also acknowledges that certain technical limitations may exist, such as hardware constraints or compatibility issues with the legacy system. These constraints may influence the implementation of certain enhancements.

1.8 Significance of the Study

A system capable of seamlessly and efficiently loading and scheduling is crucial for professors. Having such a system implemented in PUP-Taguig holds significant importance for the institution's performance and operation. Specifically, the following will benefit from the enhancement of the PUP Taguig Faculty Loading and Scheduling System:

Academic Administrators: Academic administrators will experience streamlined decision-making processes for faculty workload allocation, resulting in a balanced and optimized distribution of teaching assignments.

Curriculum Developers: Curriculum developers will benefit from an enhanced system capable of seamlessly integrating new curricula. This ensures that course offerings align with emerging academic standards, providing a more relevant and engaging learning experience for students.

Faculty Members: Faculty members will enjoy an improved user experience and increased efficiency in managing their teaching assignments. The redesigned user interface and enhanced functionalities contribute to higher satisfaction and a more user-friendly interaction with the system.

Students: Students will indirectly benefit from the optimized faculty loading and scheduling system. A well-organized and efficiently managed system contributes to a smoother academic environment, potentially reducing scheduling conflicts and ensuring a more consistent learning experience.

Accreditation Bodies: Accreditation bodies responsible for evaluating the academic quality of the institution may benefit indirectly from a system that ensures compliance with accreditation standards through efficient faculty assignment and curriculum integration.

Information Technology (IT) Personnel: IT personnel responsible for system maintenance will benefit from the migration to the LARAVEL framework, which provides a more modern, scalable, and maintainable technological infrastructure. This ensures easier maintenance and future adaptability.

Future Researchers and Developers: The findings and best practices established in this study serve as a valuable resource for future researchers and developers in the field of academic administration. The study contributes to the body of knowledge on the integration of research findings into the technological enhancement of educational systems.

1.9 Definition of Terms

Efficiency: In the context of this study, efficiency refers to the system's ability to streamline processes, reduce manual interventions, and optimize the allocation of faculty resources and course scheduling.

Adaptability: The system's capacity to easily incorporate changes, updates, and new requirements, allowing it to evolve and remain relevant in response to the dynamic nature of educational programs and administrative needs.

User Satisfaction: The overall contentment and positive experiences of Academic Administrators, Curriculum Developers, and Faculty Members interacting with the enhanced system, measured through surveys, feedback, and user testimonials.

System Performance Metrics: Quantitative measures used to assess the performance of the PUP Faculty Loading and Scheduling System, including but not limited to response time, resource utilization, and overall operational efficiency.

Modernization: The process of incorporating contemporary technologies, methodologies, and design principles into the system to enhance its functionality, security, and user experience.

Academic Environment: The encompassing atmosphere and conditions within the educational institution, shaped by administrative processes, resource allocation, and the interaction between faculty and students.

CHAPTER 2

REVIEW OF RELATED LITERATURE AND STUDIES

2.1 Introduction

This chapter aims to provide a comprehensive understanding of the existing relevant literature and studies related to the development and implementation of faculty loading and scheduling systems in the context of higher educational institutions. The following studies showcase the efficient distribution of faculty responsibilities, and the creation of optimal scheduling systems are essential components in fostering a conducive academic environment. This review will lay the groundwork for the rationale, theoretical framework, and methodologies employed for this study.

2.2 Faculty Workload Optimization and Scheduling Systems

The study of Abed (2021) titled *Development Of A Faculty Workload Management System Using Optimization Algorithms* addresses the prevalent concern of faculty workload imbalance in Jordan Universities by proposing a novel Faculty Workload Management System (FWMS) utilizing optimization algorithms. Recognizing the inherent efficiencies of manual scheduling, the FWMS leverages genetic algorithms to meticulously assign courses to faculty members, considering their expertise, preferences, and pre-defined workload limits. This innovative approach demonstrably reduced the number of overloaded faculty members,

contributing to improved faculty well-being and potentially enhanced student learning outcomes. Abed's study emphasizes the critical roleof accurate data and user acceptance in ensuring the successful implementation and sustained efficacy of such systems within academic institutions.

In their research work titled *ClassSchedule: A Web-Based Application for School Class Scheduling with Real-Time Lazy Loading*, Nuengwong et al. (2018) introduced an innovative method to address the complexities associated with class scheduling in educational institutions. The study delves into the intricate management of resources, encompassing teachers, students, subjects, classrooms, and periods, and highlights the inherent limitations of traditional manual scheduling methods employing pens and paper. The authors underscore the insufficiencies of existing software and algorithms in meeting the diverse demands imposed by distinct departments and overarching school policies. In response to these challenges, they developed *ClassSchedule*—a sophisticated web-based application utilizing technologies such as PHP, HTML, CSS, AJAX, JavaScript, and MySQL. This advanced tool represents a departure from conventional approaches, streamlining the scheduling process and providing educators with the capability to schedule classes and access real-time data seamlessly. Key features of *ClassSchedule* include the facilitation of collaborative scheduling among multiple department representatives, allowing for simultaneous editing of scheduling data and real-time assessment of resource collisions.

Meanwhile, the Development of Online-Class Scheduling Management System Conducted by the Case Study of Department of Social Science: Faculty of Humanities and Social Sciences Suan Sunandha Rajabhat University" by Wipada Chaiwchan and Patcharee Klinhom, explores the intricacies of implementing an online-class scheduling management system. The study places a primary emphasis on enhancing the efficiency and convenience of class scheduling procedures for educators and students alike. Executed as a web-based application, the system is

meticulously developed using PHP as the programming language and MySQL as the designated database management system. The research delves into the nuanced nature of class scheduling, taking into account factors such as course quantities, student volumes, and the specific

physical attributes of individual classrooms. The scholarly discourse unfolds by highlighting the transformative role of information technology in shaping educational paradigms and facilitating communication. The authors illustrate how technology, particularly online scheduling systems, streamlines class schedule management, benefiting both faculty and students. The study further investigates the application of genetic algorithms in solving complex scheduling challenges, showcasing their effectiveness in reducing complexity, saving labor time, and minimizing costs. Methodologically, the research employs a comprehensive evaluation strategy, incorporating Black-Box testing and obtaining feedback from both experts and general users. The positive outcomes of the evaluation affirm the system's high satisfaction levels among users, indicating its potential for effective implementation in authentic educational settings. In summary, the study underscores the profound significance of technologically-driven solutions, exemplified by the online-class scheduling management system, in improving educational processes and meeting user requirements.

Faculty members are at the forefront in every learning institution. They teach and nurture the minds of students. The tasks of faculty members are not only limited to teaching; they check the large volume of test papers, which add burden to their work. This is due to the reason that faculty members handle as much as fifteen teaching loads every semester. The more teaching loads, the higher the salary they received (Etcuban, 2013) This study emphasizes the pivotal role of faculty members within educational institutions. These professionals are responsible for educating

and guiding students, and their duties extend beyond teaching. They also have to manage a significant volume of test papers, which contributes to their workload. This added burden is notably high because faculty members typically handle up to fifteen teaching assignments per semester. Interestingly, their compensation often correlates with the number of teaching assignments they undertake, meaning that higher teaching loads lead to increased salary, as mentioned by Etcuban (2013).

2.3 Historical Perspectives on Manual Scheduling

Miranda (2010) explained that a university, historically, subjects were scheduled its courses and assigned classrooms manually by trial and error, a slow and tedious process with average execution times that had ballooned to four weeks. In the article entitled Solving the Problem in timetabling (2010), typically, a manual solution requires expert attention and can take many weeks for large educational establishments. Moreover, because of the problem complexity, planners are not always able to make the best decisions, building schedules that are inconsistent with teaching requirements and do not satisfy all teachers' needs. Miranda's (2010) study reveals the historical reliance of universities on manual methods for scheduling courses and assigning classrooms, a slow and laborious process that involved trial and error, taking an average of four weeks for execution. This manual approach, as detailed in

"Solving the Problem in Timetabling" (2010), demanded expert attention and often consumed numerous weeks, particularly in larger educational institutions. However, due to the intricate nature of the task, planners encountered difficulties in making optimal decisions. Consequently, the resultant schedules frequently fell short in meeting teaching requirements and satisfying the diverse needs of teachers, highlighting inconsistencies and inefficiencies within the scheduling system in educational settings.

2.4 University Scheduling Challenges: A Philippine Context

Baccay & Cabahug (2019) address the multifaceted challenges of university scheduling in the Philippines. Their Class Scheduling and Faculty Loading System (CSFLS) utilizes a modified particle swarm optimization algorithm to generate efficient schedules, considering course requirements, room availability, faculty preferences, and minimizing conflicts. The study extends its optimization scope beyond workload management to include classroom utilization, showcasing its potential for broader resource optimization within universities. While recommending further testing and user training for enhanced adoption and user satisfaction, Baccay & Cabahug's work demonstrates the effectiveness of optimization algorithms in addressing complex challenges in university scheduling.

In a study conducted by Botangen (2016), Central Luzon State University struggles with challenges in their manual scheduling system, encompassing the laborious task of organizing classes, recurrent delays in finalizing schedules, and errors leading to conflicts. Face-to-face discussions among registrars are essential for conflict resolution yet hindered by geographical distances and varying priorities among units. To tackle these hurdles, the university adopted a web-based Class-Scheduling system, integrating technologies like Apache, PHP/HTML/JavaScript, and MySQL, fostering collaboration among dispersed registrars. This innovative system operates through block, room, and faculty scheduling categories, streamlining outcomes to create a cohesive schedule. Despite challenges like delays and conflicts, the system's integration of compatible outcomes ensures a comprehensive and optimized scheduling solution.

According to Louie et al. (2018), in their paper titled "Predictive Analytics Implementing Genetic Algorithm on Class Scheduling System," concerns related to faculty course-loading and student accommodation at Trimex Colleges in Laguna,

Philippines are addressed. With a notable increase in student population, there is a need for an efficient class scheduling and faculty course-loading system. The primary objective of this study is to develop a system capable of swiftly generating class schedules and faculty course-loading while predicting the demand for course sections based on data from previous terms. Their study introduces an inclusive system that integrates predictive analytics and genetic algorithms to address challenges associated with class scheduling and faculty course loading at Trimex Colleges. The system's efficiency lies in its ability to swiftly generate class schedules and faculty course loading, resulting in substantial time and resource savings. Furthermore, the system exhibits predictive capabilities, anticipating the need for course sections by leveraging insights derived from historical data. The results of the software evaluation attest to the system's success in achieving the outlined objectives, positioning it as a valuable tool for Trimex Colleges.

In an article by Ortega et. al (2015) titled, *Online Class Scheduling and Faculty Loading System within a Decision Support Framework*, they discussed the challenges faced by academic department heads in creating course schedules to accommodate student enrollment needs. The study focuses on developing a class scheduling and loading system using a knowledge-based approach, heuristic functions, and rule sets within a decision support framework. The goal is to optimize available resources, such as classrooms and faculty, and enhance the efficiency of the scheduling process. The research identifies key processes in creating course schedules, including regular offerings, checking teacher availability, and allocating breaks.

Constraints, such as room availability, faculty preferences, and workload considerations, are highlighted. The study emphasizes the importance of aligning faculty members with their qualifications and preferences, contributing to the overall

effectiveness of the scheduling process. The article introduces an online system prototype that allows for data entry, room availability checks, and faculty assignment validation, aiming to streamline the scheduling and loading procedures. Overall, the research concludes that the developed online system is feasible and recommends continuous improvement and maintenance for enhanced functionality and adaptability to evolving constraints and requirements. The translated outcome of the study is an operational web system used by the University of Cebu-Banilad Campus for class scheduling and faculty loading, contributing to better resource management and overall efficiency.

In a study conducted by Jalandoon et. al (2023) titled *Development of Faculty Loading and Credential Management for STI College Alabang,* they tackled the challenges in manually creating schedules for each room and section, and manually assigning loads to faculty. They assessed the level of efficiency and effectiveness of a developed web-based online system in terms of functionality, efficiency, compatibility, usability, reliability, security, and maintainability. They also looked into the satisfaction of end users based off system performance in terms of speed, reliability, functionality, and usability. Their study found out that the developed system meets the mentioned standards and increased the efficiency in managing teaching load in the said university.

2.5 Synthesis and Relevance to the Current Study

The reviewed literature and studies present a comprehensive landscape of faculty workload optimization and scheduling systems in the higher education context, each offering unique insights and approaches. Drawing upon these studies is essential to establish a foundation for the rationale, theoretical framework, and methodologies applied in the current study titled "Optimizing Faculty Workload: Enhancing the PUP Taguig Faculty Loading and Scheduling System." The synthesis

of these works provides valuable perspectives on the challenges faced by academic institutions and the innovative solutions devised to address them.

In examining Abed's (2021) Faculty Workload Management System (FWMS) and Nuengwong et al.'s (2018) ClassSchedule, the integration of optimization algorithms emerges as a crucial component in achieving efficient faculty assignments and streamlined class scheduling. These studies collectively advocate for the adoption of genetic algorithms to meticulously match faculty expertise, preferences, and workload limits, resulting in improved well-being for faculty members and potentially enhanced student learning outcomes.

The exploration of technology-driven solutions by Chaiwchan and Klinhom (2014) and Ortega et al. (2015) adds another layer to the synthesis. The implementation of web-based scheduling systems showcases the transformative impact of technology on educational processes. Leveraging PHP, HTML, CSS, AJAX, JavaScript, and MySQL, these systems not only streamline class schedule management but also provide real-time data access and collaborative scheduling features. The technological emphasis resonates with the contemporary landscape of educational technologies, aligning with the overarching goal of our study to enhance the PUP Taguig Faculty Loading and Scheduling System.

Miranda's (2010) historical perspective on manual scheduling serves as a crucial reminder of the challenges faced by universities in the past. The slow and error-prone nature of manual processes, taking weeks for execution, emphasizes the need for modern, efficient solutions. Our study acknowledges this historical context and endeavors to propel the PUP Taguig Faculty Loading and Scheduling System into a new era of effectiveness.

Addressing challenges specific to the Philippine context, Baccay & Cabahug's (2019) CSFLS, Botangen's (2016) case study on Central Luzon State University, and Louie et al.'s (2018) predictive analytics approach for Trimex Colleges underscore the multifaceted issues faced by Philippine universities. These challenges range from laborious manual scheduling systems and recurrent delays to the need for predictive analytics in accommodating a growing student population. Our study, focusing on PUP Taguig, aligns with these contextual challenges and endeavors to provide tailored solutions that consider the unique dynamics of the Philippine higher education landscape.

The synthesis of these diverse studies reinforces the significance of our research objectives in optimizing the PUP Taguig Faculty Loading and Scheduling System. By integrating optimization algorithms, adopting technology-driven solutions, acknowledging historical challenges, and addressing specific Philippine contextual issues, our study aims to enhance the existing system. The infusion of a new curriculum, migration to a new framework, and revamping the user interface align with the evolving landscape of educational technologies and best practices identified in the literature. The insights gained from these studies will guide the implementation and evaluation phases of our proposed enhancements, ensuring a more robust and efficient faculty workload optimization and scheduling system at PUP Taguig.

CHAPTER 3

RESEARCH METHODOLOGY

This chapter outlines the systematic approach undertaken to conduct the research, providing a comprehensive overview of the research design, data sources, construction of research instruments, data generation procedures, ethical considerations, and the proposed structure of the system under investigation.

3.1 Research Design

This study adopts a quantitative research design, focusing on the systematic collection and analysis of numerical data to assess the impact of enhancements to the Polytechnic University of the Philippines Taguig Faculty Loading and Scheduling System. The quantitative research design emphasizes objective measurement, statistical analysis, and structured data collection methods to evaluate specific metrics related to system performance, efficiency, and user satisfaction.

3.1.1 Quantitative Measures:

The research design involves the quantification of key variables through structured surveys and questionnaires. Participants, including academic administrators and faculty members, will provide numerical responses using Likert scales and closed-ended questions. The survey instruments are designed to capture quantitative data on user satisfaction, perceived system efficiency, and overall feedback on the enhanced features.

3.1.2 System Performance Metrics:

Objective metrics related to system performance, response time, and resource utilization will be automatically collected by the enhanced system. These quantitative measures aim to assess the impact of the enhancements on the technical aspects of the Faculty Loading and Scheduling System. Data generated by the system, including log files and analytics, will serve as crucial sources for quantitative analysis.

3.2 Research Instrument

In this study, the research instrument is crafted in accordance with the ISO 25010 standard, a comprehensive framework for evaluating software product quality and system performance. ISO 25010 outlines a set of quality characteristics and sub-characteristics crucial for assessing the efficiency, usability, and reliability of software systems. Specifically, the research instrument employed in this study takes the form of structured surveys and questionnaires. These instruments are designed to align with internationally recognized criteria for software quality, as outlined in the ISO standard, ensuring a systematic and rigorous evaluation of the targeted aspects of software performance.

3.2.1 ISO 25010 Quality Characteristics:

The research instrument will focus on the following ISO 25010 quality characteristics, tailored to the specific context of the enhanced Polytechnic University of the Philippines (PUP) Faculty Loading and Scheduling System:

1. **Performance Efficiency:**

- *Aspect of Focus:* Evaluation of system performance metrics, including response time and resource utilization.

- *Instrumentation:* Automated tracking of response time, system-generated metrics related to efficiency, and participant feedback on perceived system speed and responsiveness.

2. **Usability:**

- *Aspect of Focus:* Assessment of user interface design, accessibility, and overall user experience.
- *Instrumentation:* Structured surveys and questionnaires based on ISO 25010 sub-characteristics such as learnability, operability, and user satisfaction. These instruments will gather quantitative data on user perceptions of the system's usability.

3. **Reliability:**

- *Aspect of Focus:* Evaluation of system stability, accuracy, and error handling.
- *Instrumentation:* Automated collection of system-generated data on error rates and reliability. Surveys will also include questions related to users' experiences with system reliability.

3.2.2 Data Collection Methods:

The research instrument employs a combination of automated system metrics and participant responses through surveys and questionnaires. The system will automatically generate data related to performance and reliability, while structured surveys will capture user perceptions aligned with the ISO 25010 usability characteristics.

3.2.3 Structured Surveys and Questionnaires:

Structured surveys and questionnaires will be designed based on ISO 25010 guidelines to gather quantitative data from participants. These instruments will include Likert scales and closed-ended questions, enabling participants to provide numerical feedback on specific aspects of system performance, usability, and reliability.

By adopting ISO 25010 as the foundation for the research instrument, this study ensures a systematic and internationally recognized approach to assessing the enhanced PUP Faculty Loading and Scheduling System. The chosen quality characteristics guide the instrument development, facilitating a comprehensive evaluation of the software product's performance and user satisfaction.

3.3 Data Gathering Procedure

The researchers employed a comprehensive data gathering approach, primarily utilizing structured interview with key stakeholders, specifically the school director. These interview, guided by a specific protocol, explored participant past experiences, challenges, and expectations regarding to the previous faculty loading and scheduling system. In-depth interviews with the school director, the primary end-user, and the system developer provided valuable insights into user perspectives and technical aspects. Additionally, purposive sampling was applied to carefully select faculty members involved with the system, ensuring diverse perspectives. This multifaceted data collection strategy, combining structured interviews with strategic ones and purposive sampling, allowed for a thorough understanding of the faculty loading and scheduling system at PUP Taguig, informing the proposed optimized approach.

3.3.1 Sampling Method

This study employs a purposive sampling method to select participants who possess specific expertise and roles within the academic community of Polytechnic University of the Philippines (PUP). Purposive sampling is a deliberate and non-random method chosen to target individuals with in-depth knowledge and experience relevant to the faculty loading and scheduling system. In this section, we detail the rationale behind choosing purposive sampling and provide insights into the selection criteria.

3.3.1.1 Rationale for Purposive Sampling

Expertise and Experience:

The primary objective of this research is to gain insights from individuals who hold key roles in the academic administration and faculty at PUP. Academic administrators, responsible for overseeing and managing faculty resources, and faculty members, who directly engage with the faculty loading and scheduling system, possess critical expertise and experience relevant to our study. Purposive sampling allows us to intentionally target individuals with specific knowledge about the existing system and, if applicable, those who will interact with the enhanced system.

Targeted Perspectives:

By selecting participants purposefully, we aim to capture a diverse range of perspectives within the academic community. This includes individuals from different departments, varying levels of experience, and distinct roles within academic administration and faculty. The targeted perspectives contribute to a more comprehensive understanding of the faculty loading and scheduling

system, ensuring that insights are reflective of the diverse nature of the academic community at PUP.

Overall, purposive sampling serves as a strategic approach to ensure that our study benefits from the rich insights of individuals with specific roles and experiences within the academic community at PUP. While acknowledging the intentional selection process, the study aims for diversity within the chosen participant group to enhance the robustness and generalizability of our findings.

3.4 Ethical Considerations

Prior to conducting interviews, the researchers obtained explicit consent from the school director, our primary client, seeking permission to engage in the study. The consent process included a detailed explanation of the study's purpose, procedures, and the confidential nature of the information to be gathered. An agreement was established with the school director, affirming that all details obtained during the research process would be treated with the utmost confidentiality and would solely remain within the research team. This commitment aligns with the principles of ethical conduct and the requirements of the Republic Act No. 10173, known as the Data Privacy Act. Furthermore, the researchers ensured that the personal information of all participants, particularly faculty members, was handled with strict confidentiality to protect their privacy and adhere to ethical standards.

3.5 System Architecture

The system architecture serves as a conceptual framework illustrating the operational dynamics of the system. It also serves as a representation of the system. It stands as essential instrument in offering a comprehensive perspective of the overall system structure.

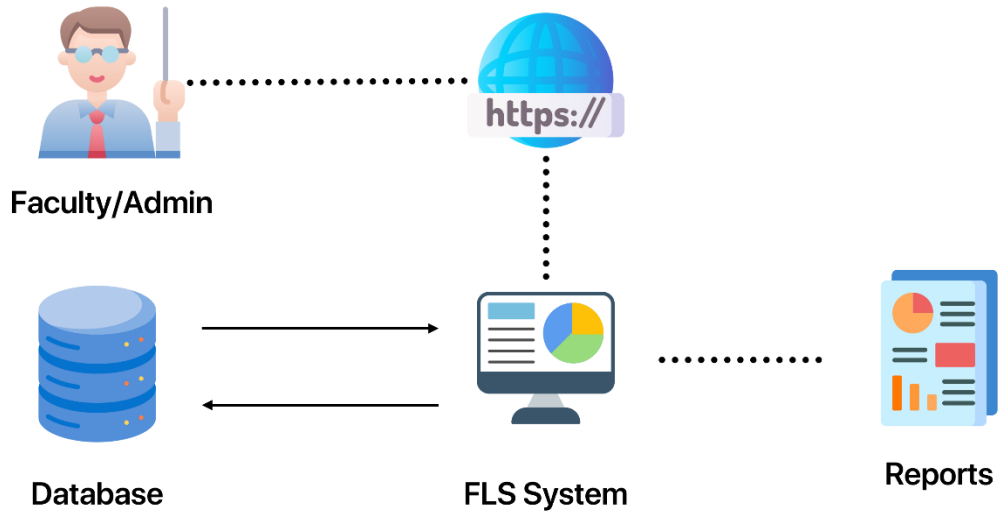


Figure 5. System Architecture

This illustration depicts the interaction between the faculty members, administrator and the Faculty Loading and Scheduling system. The administrator is responsible for overseeing the system using a computer. The system uses the raw data stored in the database server to make sure that the necessary information is properly registered. Faculty members have the ability to see important details and reports through the system.

3.5.1 System Flowchart

Figure (). User Login Module

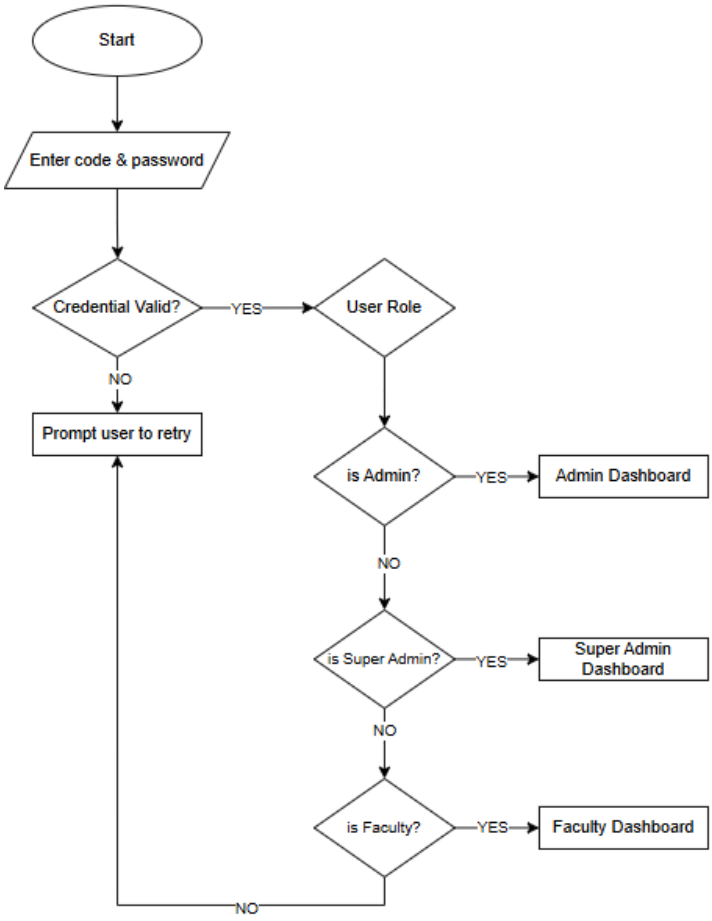


Figure () depicts the process for logging into the system. The login process begins when a user inputs their unique code and password. The system validates these credentials, ensuring they match the stored information. If the credentials are invalid, the system prompts the user to retry. Upon successful validation, the system checks the user's role. Based on the user's role, they are directed to the appropriate dashboard. An admin user is given access to the Admin Dashboard, while a super admin is redirected to the Super Admin Dashboard. Faculty members will be directed to faculty dashboard. If the user does not fit any of these categories, the login attempt is denied.

Figure (). Submit Preferences in the Faculty Module

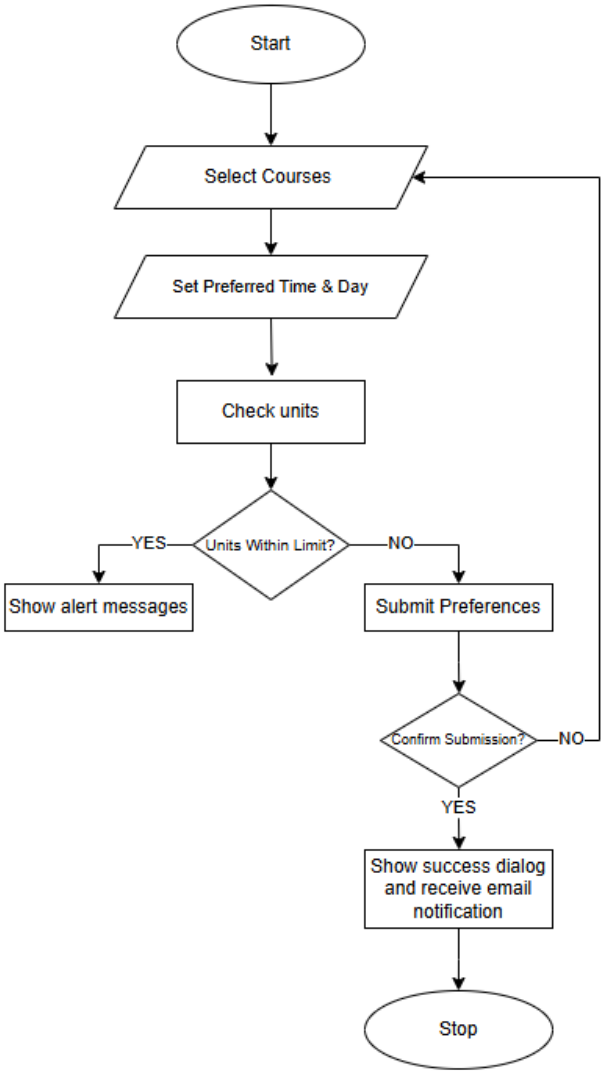


Figure () illustrates the process of selecting and submitting course preferences. The user starts by choosing courses and setting preferred times and days. The system checks if the total number of units is within the allowed limit. If the units exceed the limit, an alert is shown; otherwise, the user submits their preferences. After confirming submission, a success message is displayed, and an email notification is sent, completing the process.

Figure (). Scheduling and Loading in the Faculty Module

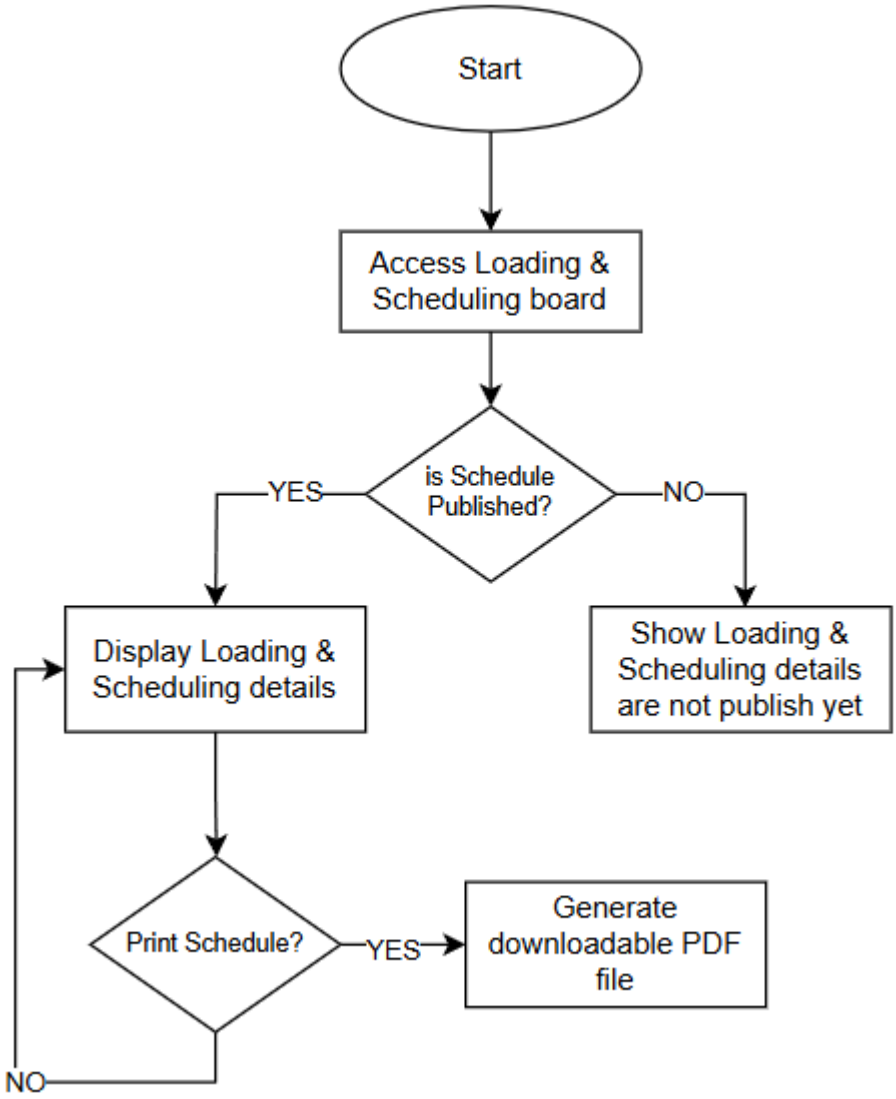


Figure () illustrates the process for accessing, viewing, and printing loading and scheduling details. The user begins by accessing the loading and scheduling board. The system checks if the schedule has been published. If it is published, the loading and scheduling details are displayed. If not, the system informs the user that the details have not yet been published. After displaying the details, the user is given the option to print the schedule. If the user chooses to print, the system generates a downloadable PDF file. If printing is not selected, the process ends.

Figure (). Viewing and Generating Faculty Information in the Super Admin Module

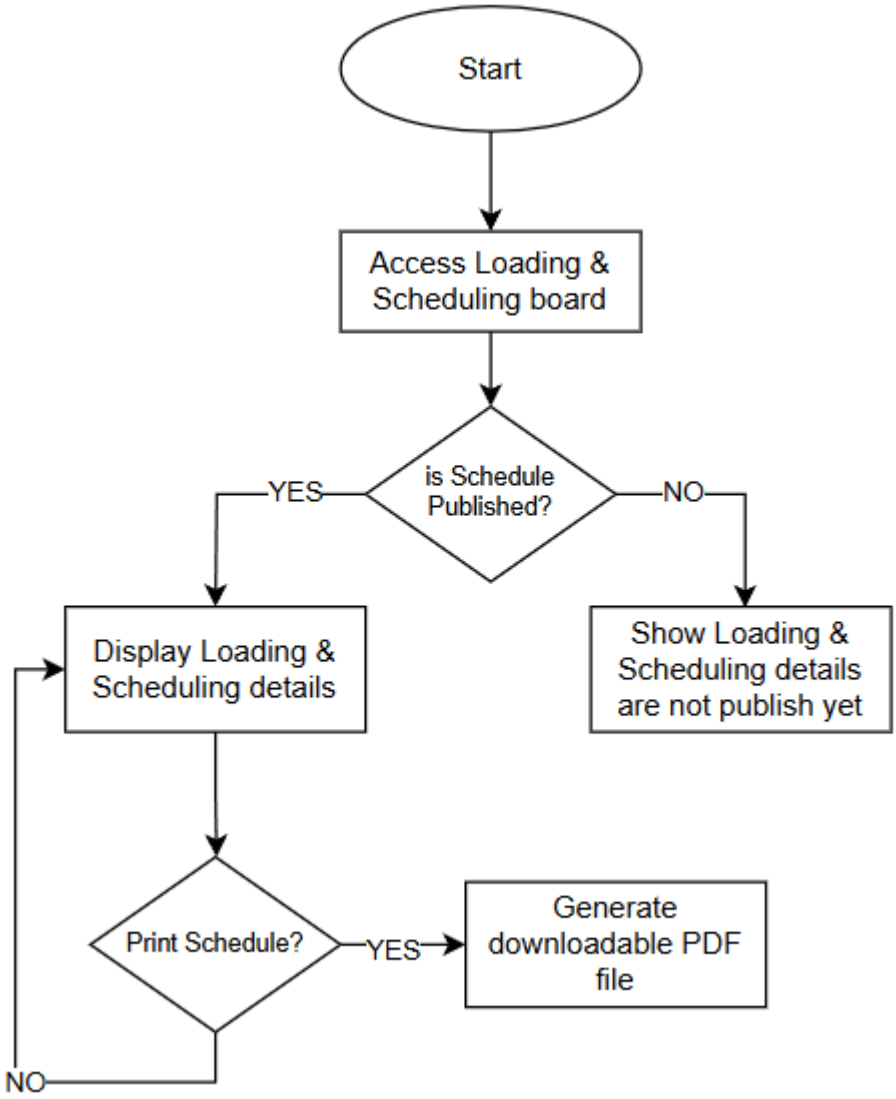


Figure () illustrates the process of viewing and generating a PDF for faculty details. The process starts when the user accesses the faculty management board. If the user chooses to generate a PDF, the system displays a dialog box containing the faculty details for review. If the user opts to print the PDF, the system generates a downloadable PDF file. If the user decides not to print, the process loops back to the review stage or stops without generating the file. The process concludes once the PDF is generated or the decision to not print is finalized.

Figure (). Adding Faculty Information in the Super Admin Module

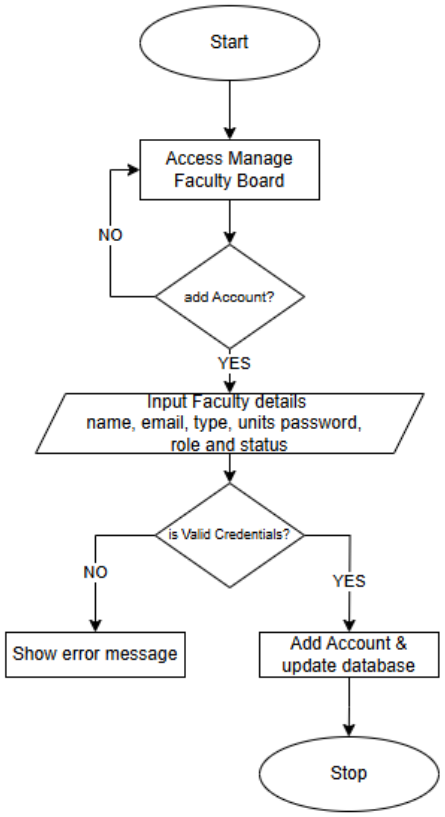


Figure () illustrates the process for adding a faculty account to the system. The user begins by accessing the faculty management board. If the user chooses to add an account, they input the faculty details, including name, email, type, units, password, role, and status. The system then checks the validity of the entered credentials. If the credentials are invalid, an error message is shown. If the credentials are valid, the account is added to the database, and the system is updated. The process then concludes.

Figure (). Editing Faculty Account in the Super Admin Module

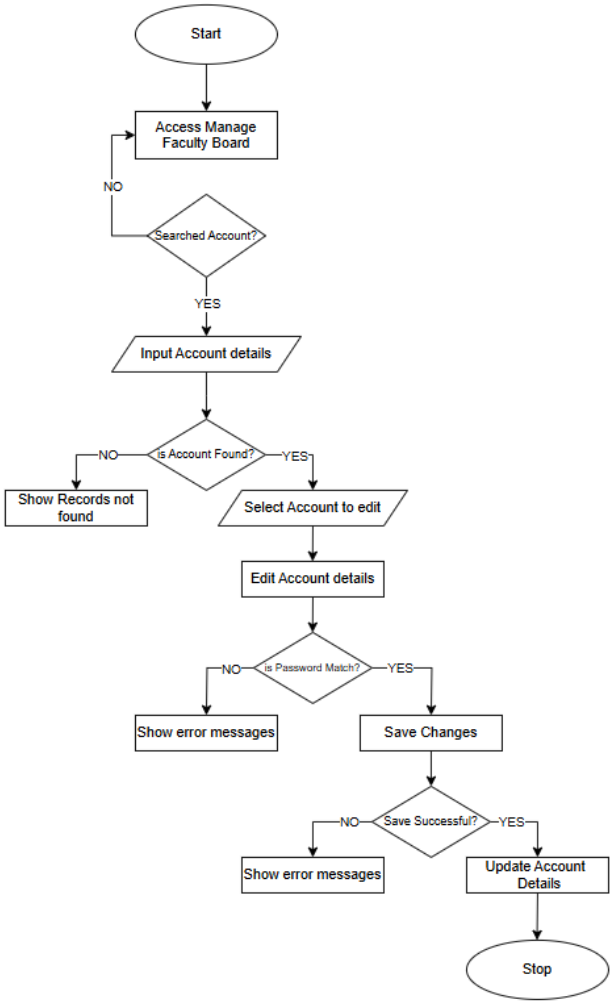


Figure () illustrates the process for managing faculty accounts. The user begins by accessing the faculty management board. If an account search is initiated, the user inputs the account details. If the account is found, the user selects it for editing. If the account is not found, the system shows an error message. Once the account is selected, the user proceeds to edit the account details. The system verifies if the password matches. If it does not, an error message is shown. If the password matches, changes are saved, and the system checks if the save was successful. If successful, the account details are updated, otherwise, an error message is displayed, and the process concludes.

Figure (). Deleting Faculty Account in the Super Admin Module

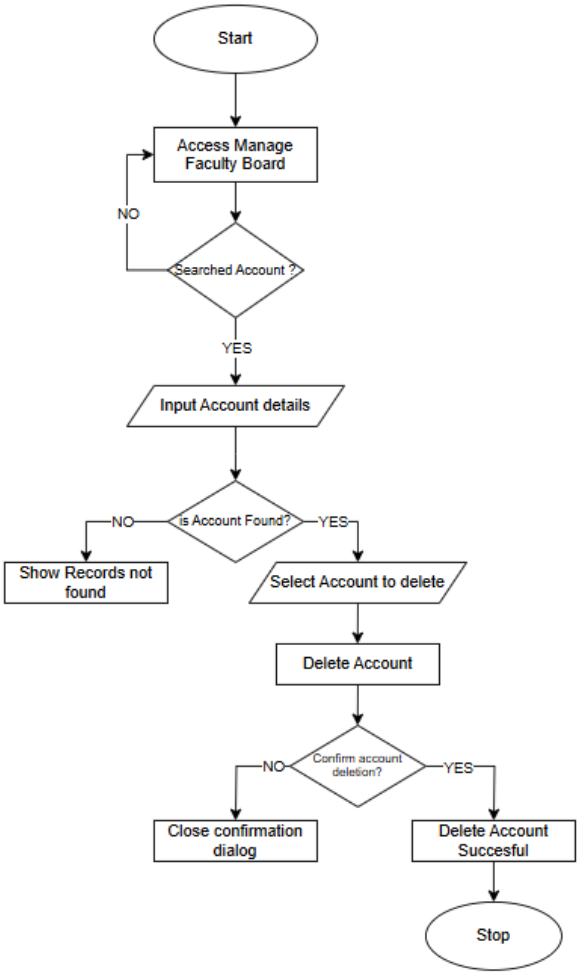


Figure () illustrates the process for deleting a faculty account. The user starts by accessing the faculty management board and searches for the account they want to delete. If no account is found, the system shows a "Records not found" message. If the account is found, the user inputs the account details and selects it for deletion. The system then checks if the faculty member is currently active in the current academic year (A.Y.), with any teaching load or schedule. If the faculty member is inactive, the account is deleted. However, if the faculty is active, the system displays a message indicating that the account cannot be deleted while active. The process then concludes.

Figure (). Viewing and Generating Room Report in the Super Admin Module

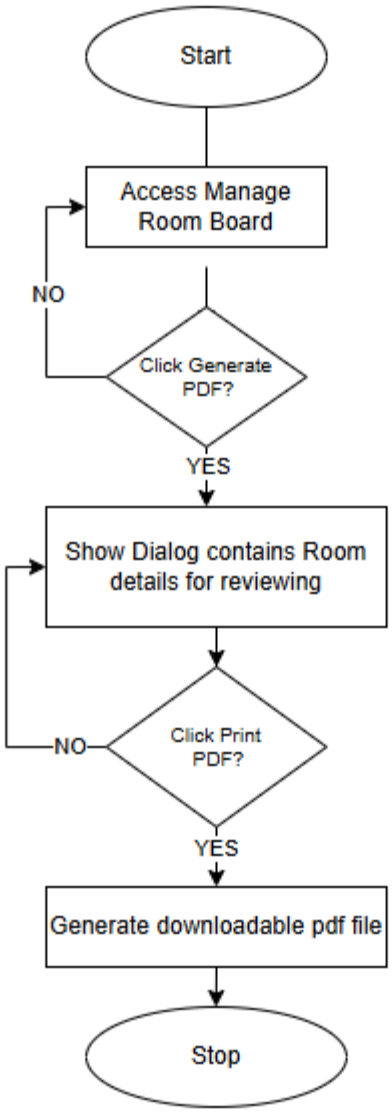


Figure () illustrates the process for generating and reviewing room details in PDF format. The user begins by accessing the room management board. If the user opts to generate a PDF, the system displays a dialog containing the room details for review. The user can then decide whether to print the PDF. If the print option is selected, the system generates a downloadable PDF file. If not, the process returns to the review stage. Once the PDF is generated, the process ends.

Figure (). Adding Room in the Super Admin Module

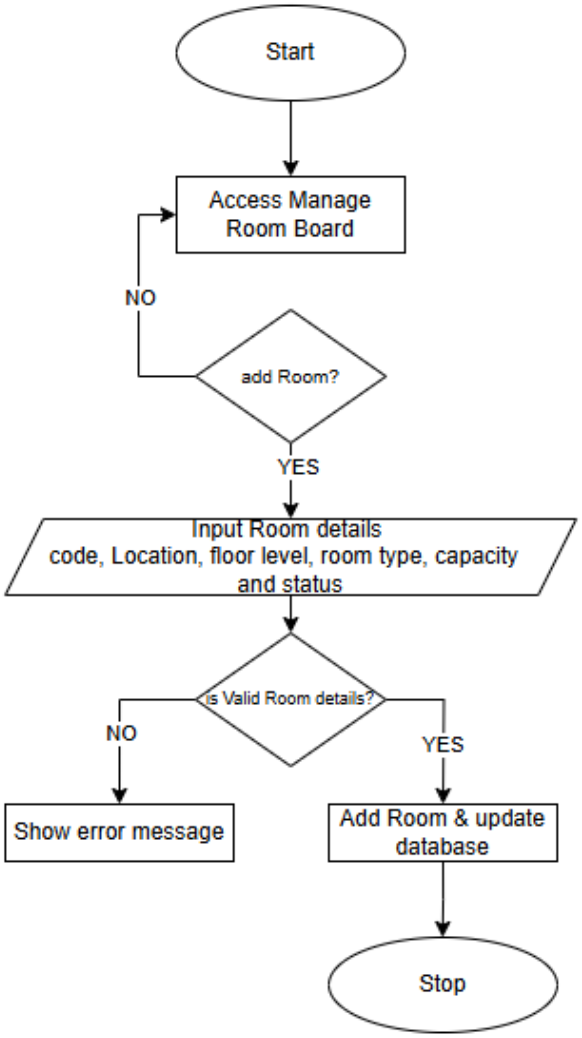


Figure () illustrates the process for adding a room to the system. The user begins by accessing the room management board. If the user decides to add a room, they input the room details, including the code, location, floor level, room type, capacity, and status. The system then checks if the room details are valid. If the details are invalid, an error message is shown. If the details are valid, the room is added to the system, and the database is updated. The process concludes once the room is successfully added or if an error is encountered.

Figure (). Editing Room details in the Super Admin Module

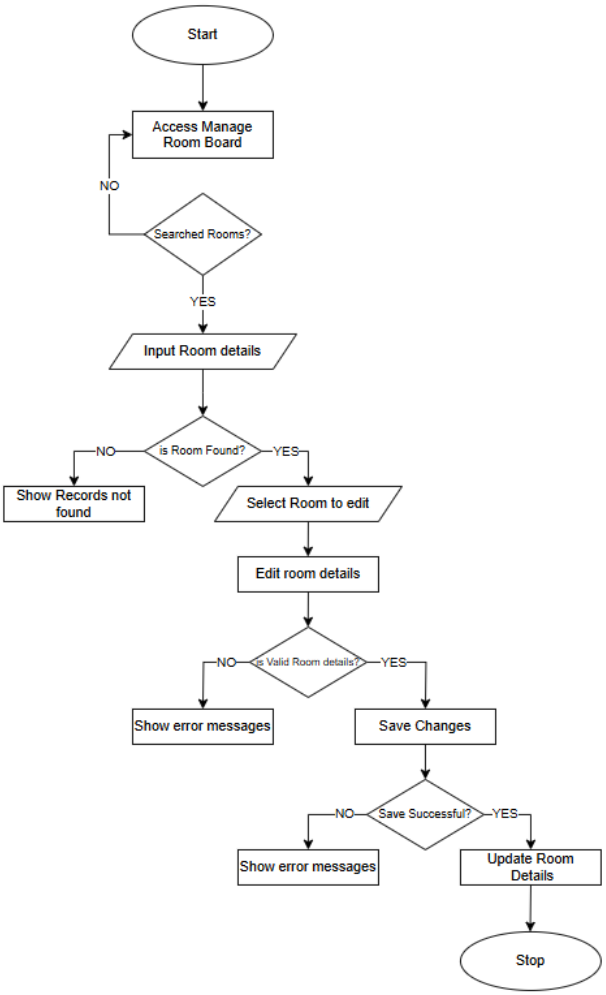


Figure () illustrates the process for searching and editing room details. The process starts when the user accesses the room management board. If the user searches for a room, they input the room details. If the room is found, the user selects the room for editing; otherwise, the system displays a "Records not found" message. Once the room is selected, the user can proceed to edit the room details. The system then validates the details entered. If the details are invalid, an error message is shown. If valid, the changes are saved, and the system checks if the save was successful. If the save is unsuccessful, another error message is shown. If successful, the room details are updated, and the process concludes.

Figure (). Deleting Room details in the Super Admin Module

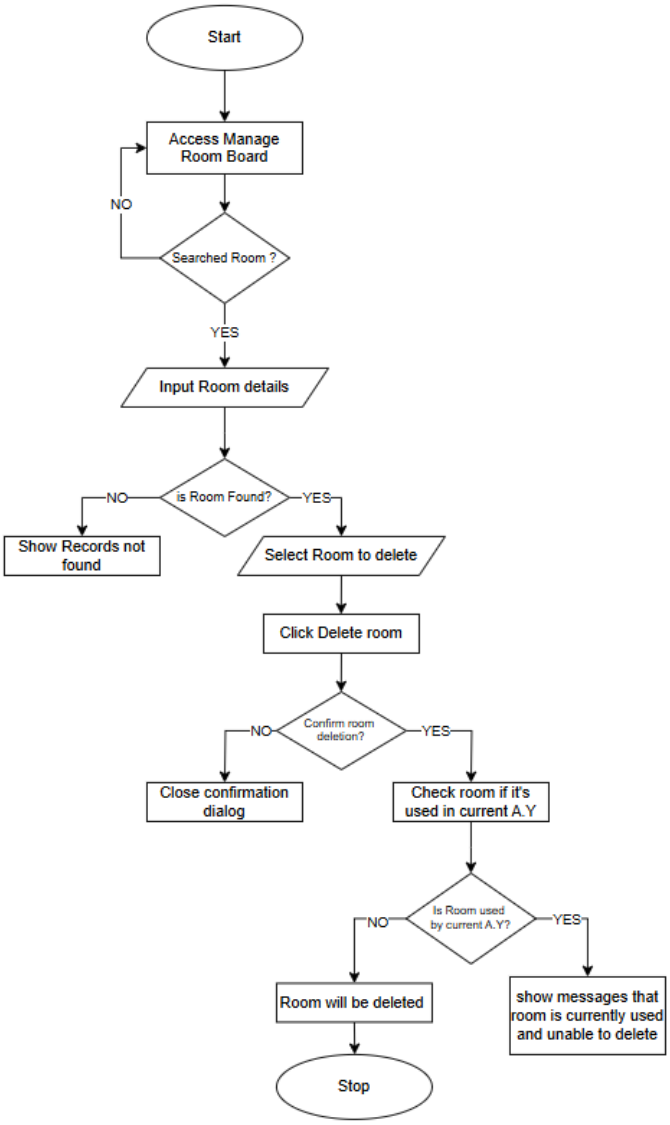


Figure () shows the process for deleting a room. The user searches for a room in the room management board. If the room is found, they select it and click delete. The system checks if the room is in use during the current academic year. If it's in use, a message appears saying the room cannot be deleted. If not, the room is deleted. If the room is not found or the deletion is canceled, the process stops.

Figure (). Viewing and Generating Program Report in the Super Admin Module

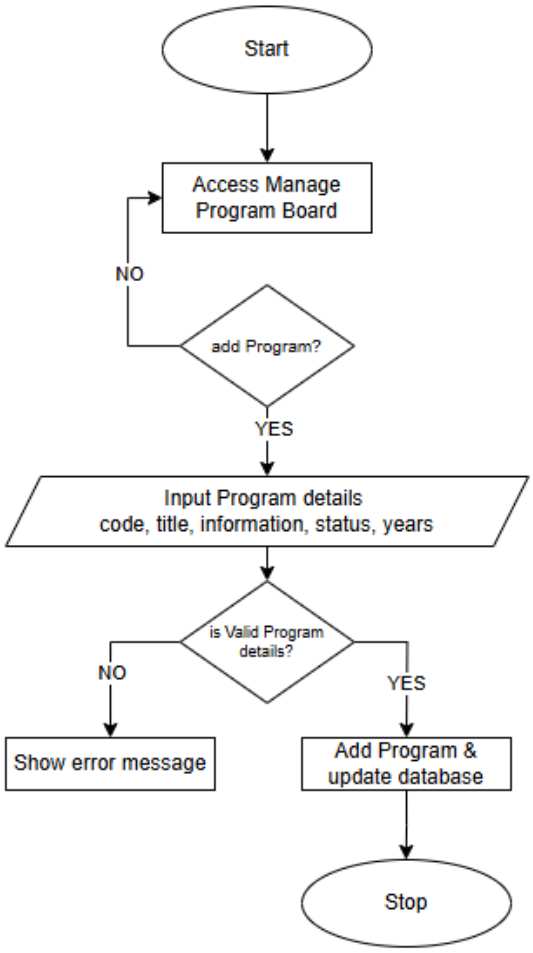


Figure () illustrates the process for adding a program to the system. The user begins by accessing the program management board. If they choose to add a program, they input the program details, including the code, title, information, status, and years. The system checks if the program details are valid. If the details are invalid, an error message is shown. If the details are valid, the program is added, and the database is updated. The process then concludes.

Figure (). Adding Program in the Super Admin Module

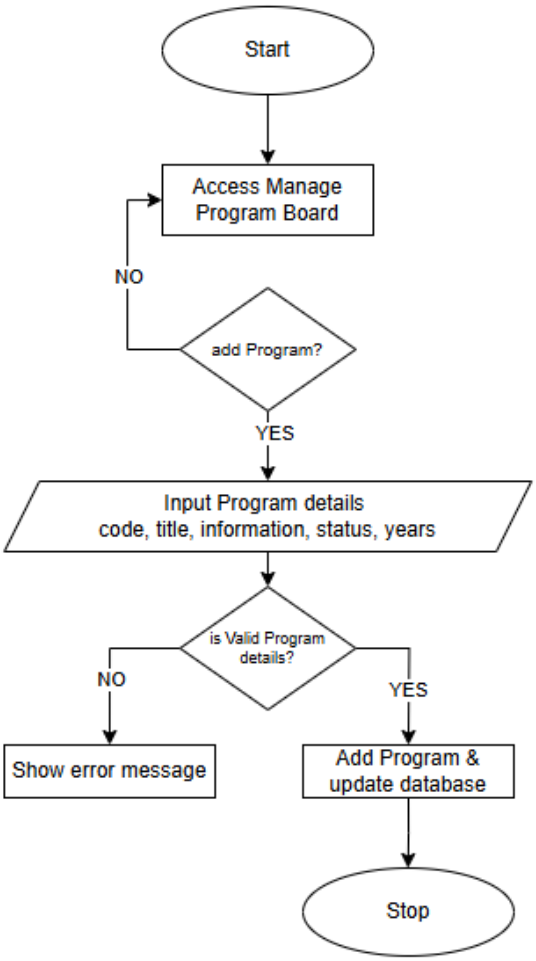


Figure () illustrates the process of adding a new program to the system. The user accesses the program management board and chooses whether to add a new program. If they proceed, they input the program details such as the code, title, information, status, and duration (years). The system then validates the details entered. If the details are invalid, an error message is displayed. If the details are valid, the program is added to the system, and the database is updated, concluding the process.

Figure (). Editing Program Details in the Super Admin Module

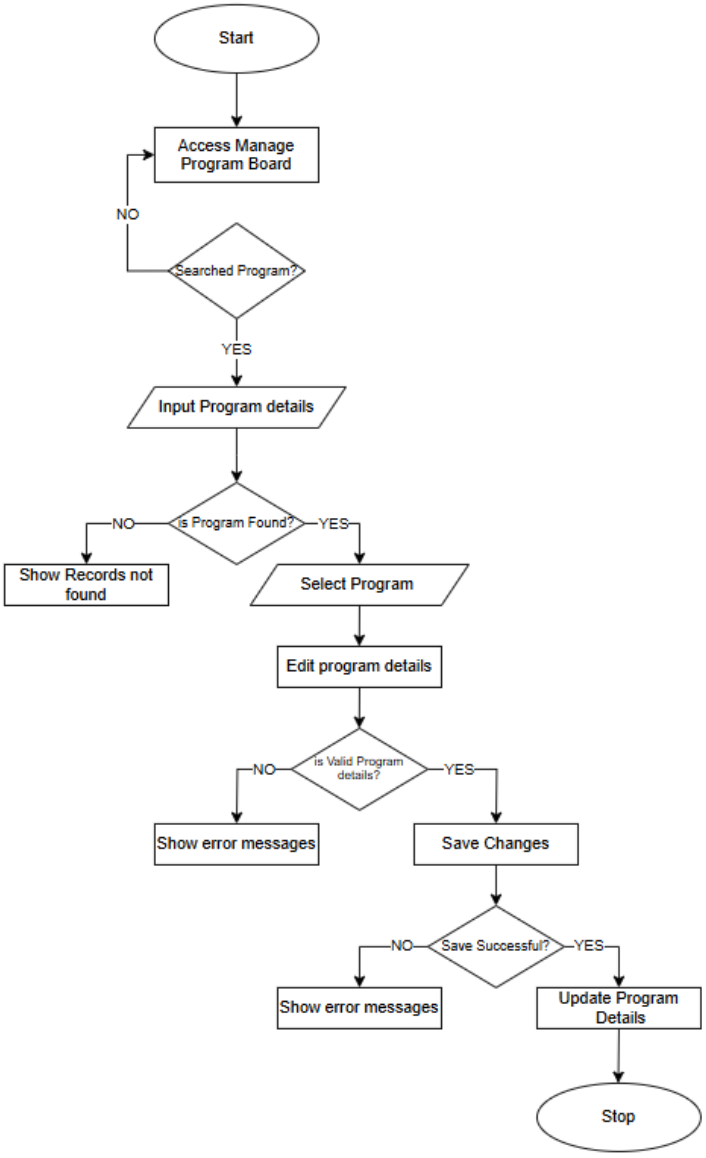


Figure () shows the process of searching and editing a program. The user accesses the program management board and searches for a program. If found, they select and edit it; if not, a "Records not found" message is shown. After editing, the system checks if the changes are valid. If valid, the changes are saved, and the program details are updated. If not, an error message is displayed. The process then ends.

Figure (). Deleting Program in the Super Admin Module

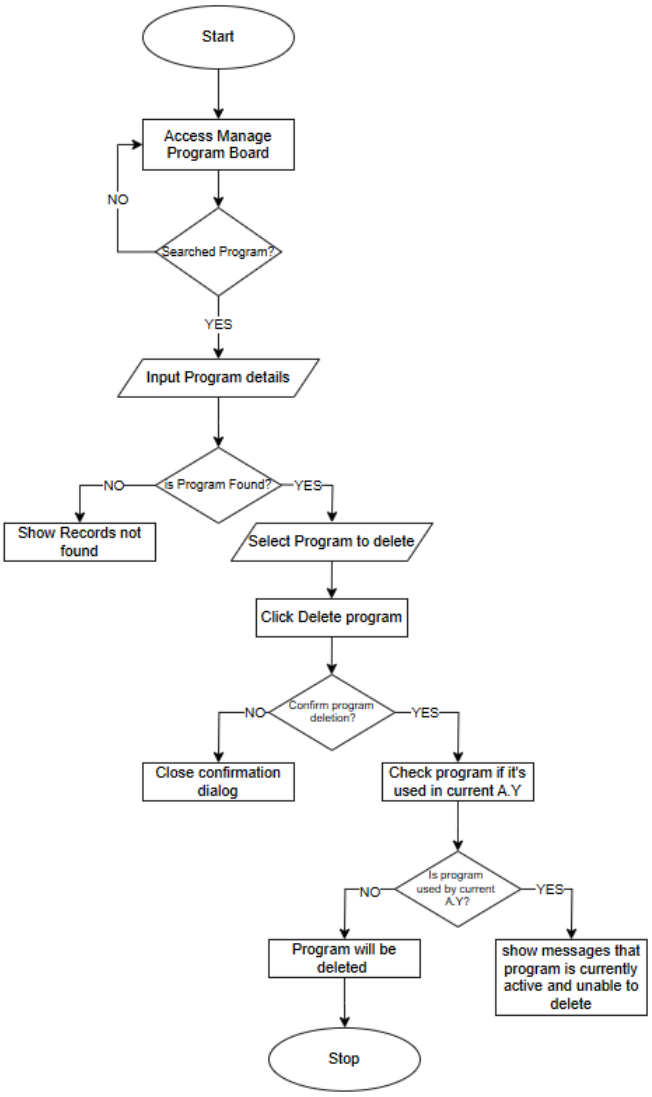


Figure () shows the process for deleting a program from the system. The user starts by accessing the program management board and searching for the program. If the program is found, it is selected for deletion; if not, a "Records not found" message appears. After selecting the program, the user confirms the deletion. The system then checks if the program is currently active in the current academic year. If the program is active, a message is shown indicating it cannot be deleted. If the program is inactive, it is deleted, and the process concludes.

Figure (). Viewing and Generating Curriculum Details in the Super Admin Module

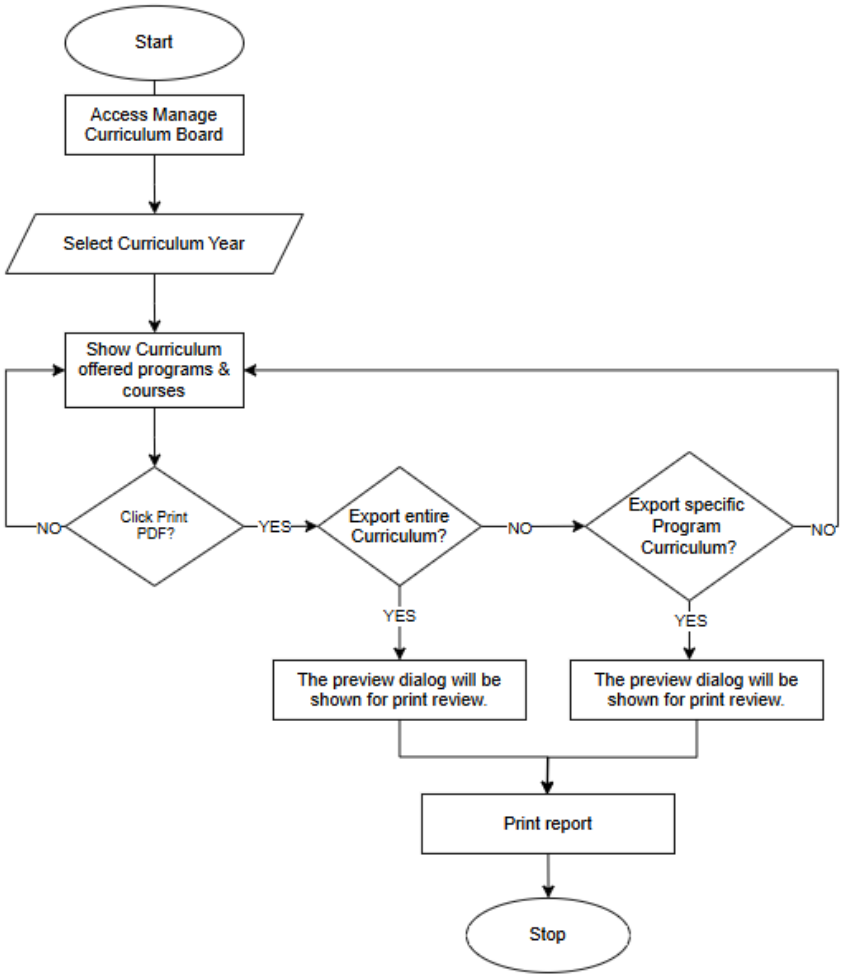


Figure () shows the process for exporting the curriculum program report. The process starts when the user accesses the "Manage Curriculum Board" and selects the desired curriculum year. Once the year is selected, the system displays the offered programs and courses. The user is then prompted with the option to print the curriculum as a PDF. If the user selects to print, the system will check whether to export the entire curriculum or just a specific program's curriculum. Depending on the selection, a preview dialog is shown for print review. After the review, the report is printed, and the process concludes. If the user chooses not to print, the process terminates without further action.

Figure (). Adding a Curriculum in the Super Admin Module

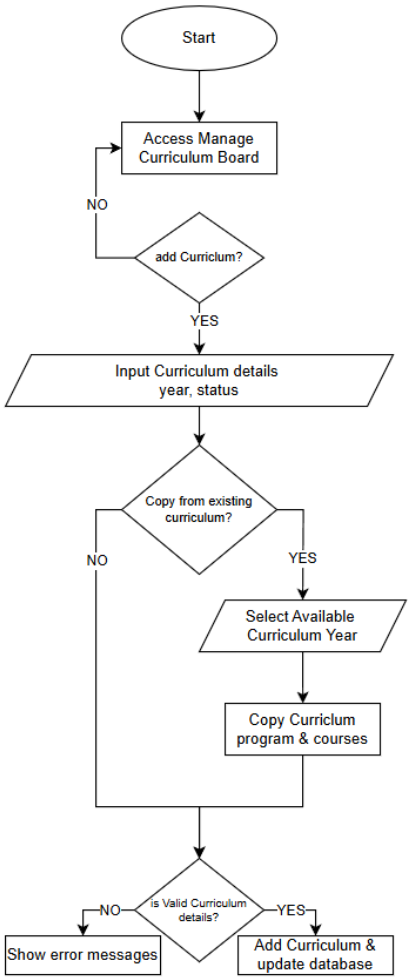


Figure () shows the process for adding a curriculum in the system. The process begins when the user accesses the "Manage Curriculum Board" and decides whether to add a curriculum. If the user chooses to proceed, they are required to input curriculum details, such as the year and status. The system then asks if the user wants to copy the curriculum from an existing one. If the user selects "Yes," they must choose the available curriculum year to copy the programs and courses. After copying, or if no existing curriculum is copied, the system checks if the curriculum details are valid. If the details are valid, the curriculum is added to the system, and the database is updated. If the details are invalid, the user is shown error messages, and the process concludes without adding the curriculum.

Figure (). Editing a Curriculum in the Super Admin Module

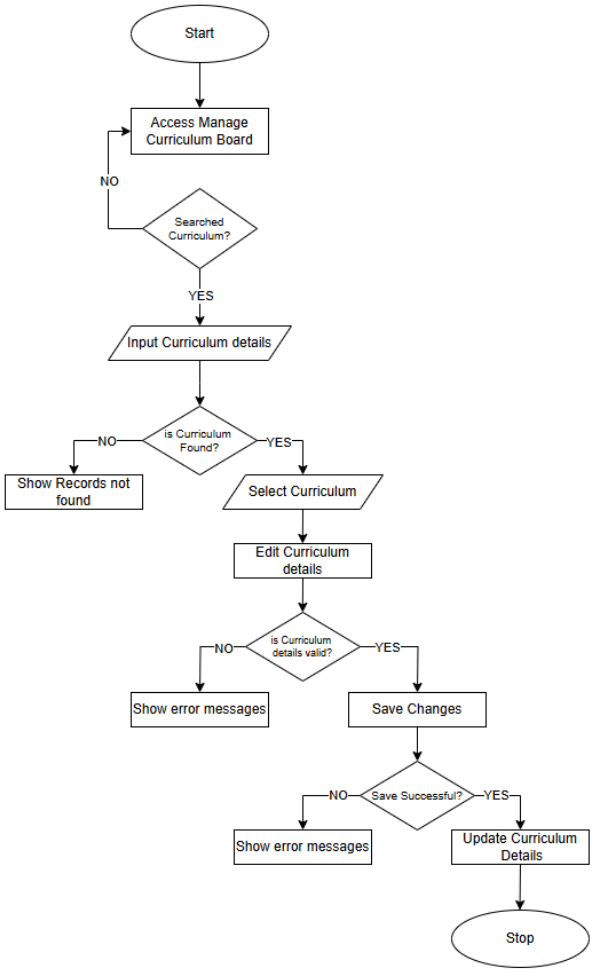


Figure () shows the process for editing a curriculum in the system. The process starts when the user accesses the "Manage Curriculum Board" and decides whether to search for a curriculum. If the user chooses to search, they are prompted to input the curriculum details. If the curriculum is found, the user selects it for editing; otherwise, a "Records not found" message is displayed. Once the curriculum is selected, the user can edit the curriculum details. After making changes, the system verifies if the entered details are valid. If the details are not valid, error messages are shown, and the process cannot proceed. If the details are valid, the user saves the changes. The system then checks if the save was successful. If not, an error message is displayed. If the save is successful, the curriculum details are updated, and the process concludes.

Figure (). Deleting a Curriculum in the Super Admin Module

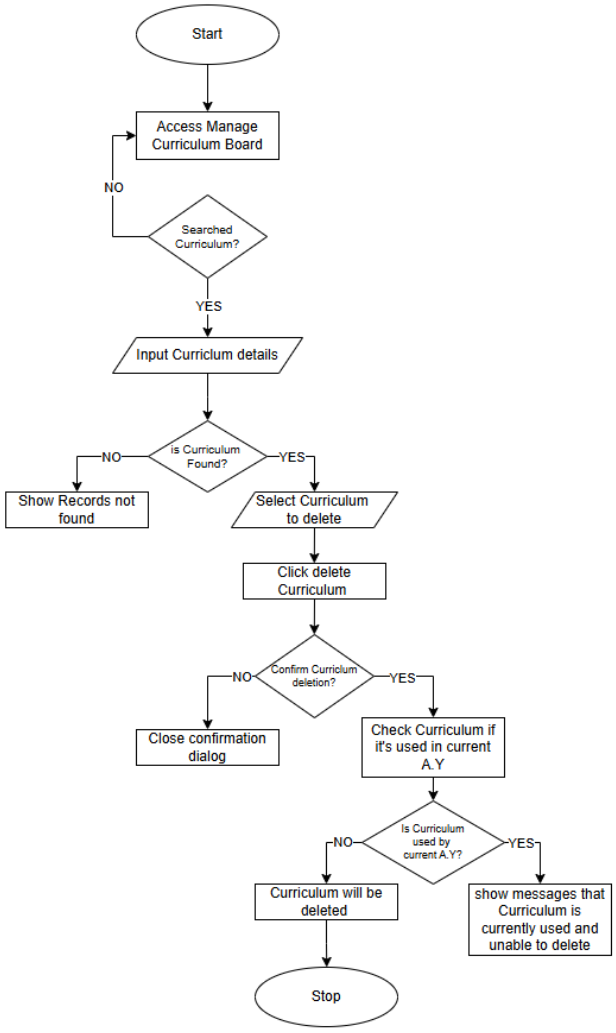


Figure () shows the process for deleting a curriculum from the system. The process starts with the user accessing the "Manage Curriculum Board" and deciding whether to search for a curriculum. If the user chooses to search, they are prompted to input the curriculum details. If the curriculum is found, the user selects it for deletion; otherwise, a "Records not found" message is displayed. After selecting the curriculum, the user clicks to delete it, and a confirmation dialog appears. If the user cancels the confirmation, the process stops. If the deletion is confirmed, the system checks if the curriculum is currently in use in the current academic year. If the curriculum is being used, a message is shown indicating that it cannot be deleted. If it is not in use, the curriculum is deleted, and the process ends.

Figure (). Assigning a Course to a Curriculum in the Super Admin Module

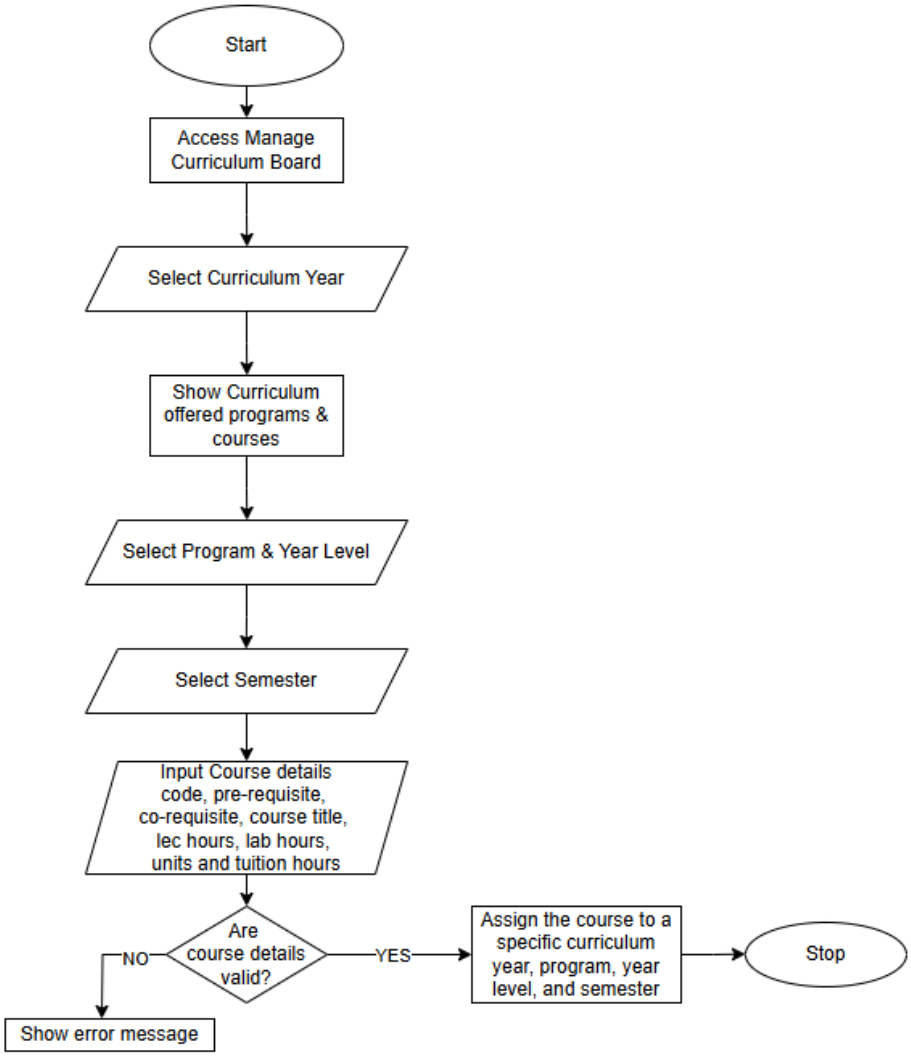


Figure () shows the process for assigning a course to a curriculum. The user starts by accessing the "Manage Curriculum Board" and selecting the curriculum year. After viewing the offered programs and courses, the user selects the program, year level, and semester. They then input course details, such as the course code, prerequisites, hours, and units. The system checks if the details are valid. If invalid, an error message is shown. If valid, the course is assigned to the selected curriculum, program, and semester, completing the process.

Figure (). Editing Course Details in the Super Admin Module

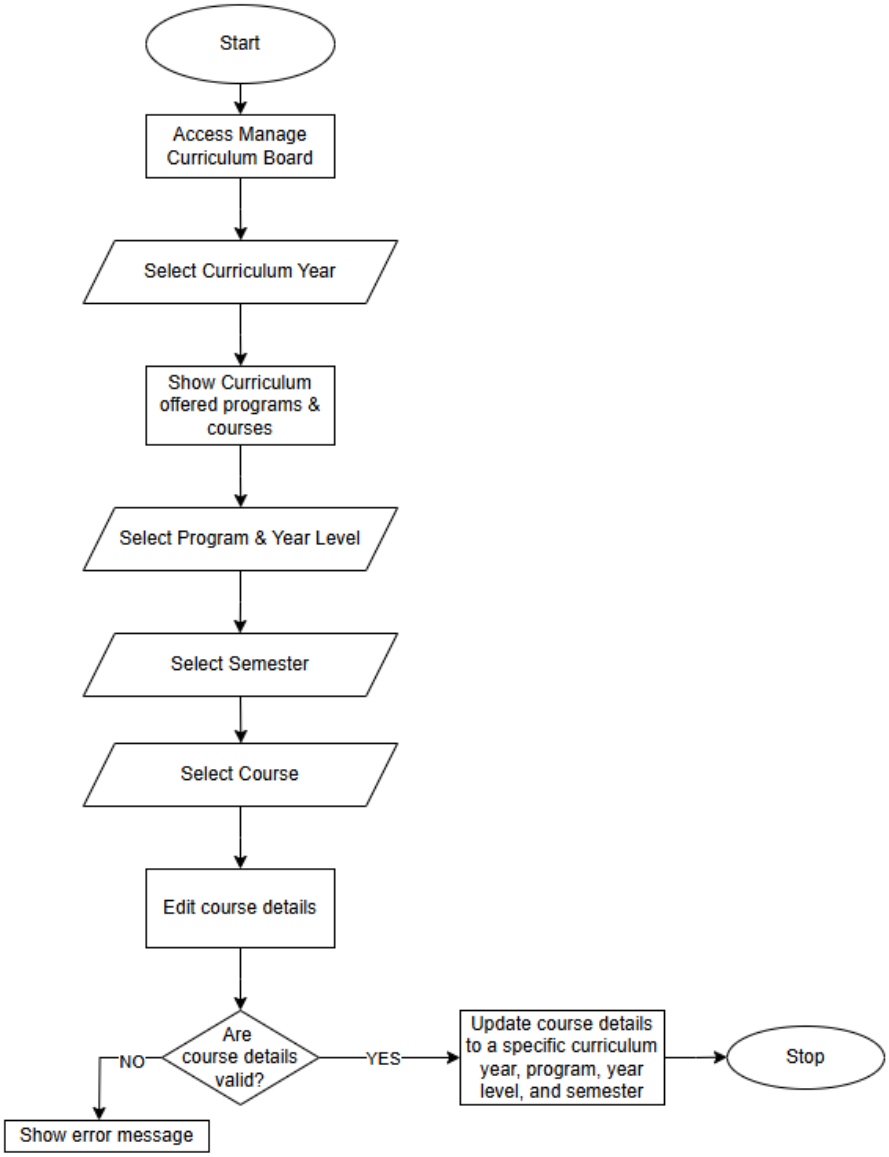


Figure () shows the process for editing course details in the system. The user begins by accessing the "Manage Curriculum Board," selecting the curriculum year, and viewing the offered programs and courses. They then choose the program, year level, semester, and specific course. After selecting the course, the user edits the course details. The system checks if the details are valid. If invalid, an error message is shown. If valid, the course details are updated, and the process ends.

Figure (). Deleting a Course in the Super Admin Module

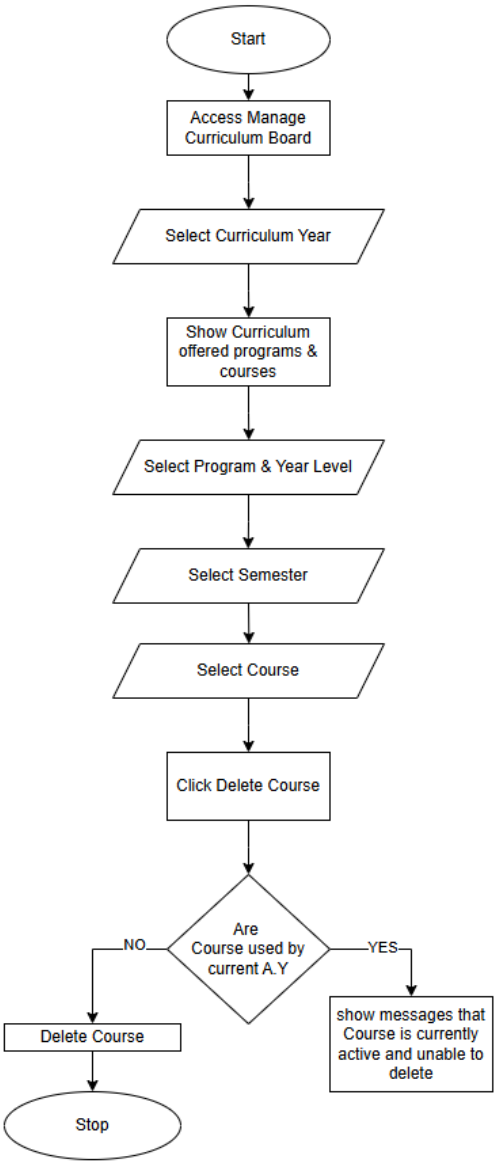


Figure () shows the process for deleting a course from the system. The user begins by accessing the "Manage Curriculum Board," selecting the curriculum year, and viewing the offered programs and courses. The user then selects the program, year level, semester, and the specific course. After selecting the course, the user clicks to delete it. The system checks if the course is currently being used in the current academic year. If the course is in use, a message is displayed indicating it cannot be deleted. If the course is not in use, it is deleted, and the process ends.

Figure (). Managing Programs in Specific Curriculum Year in the Super Admin Module

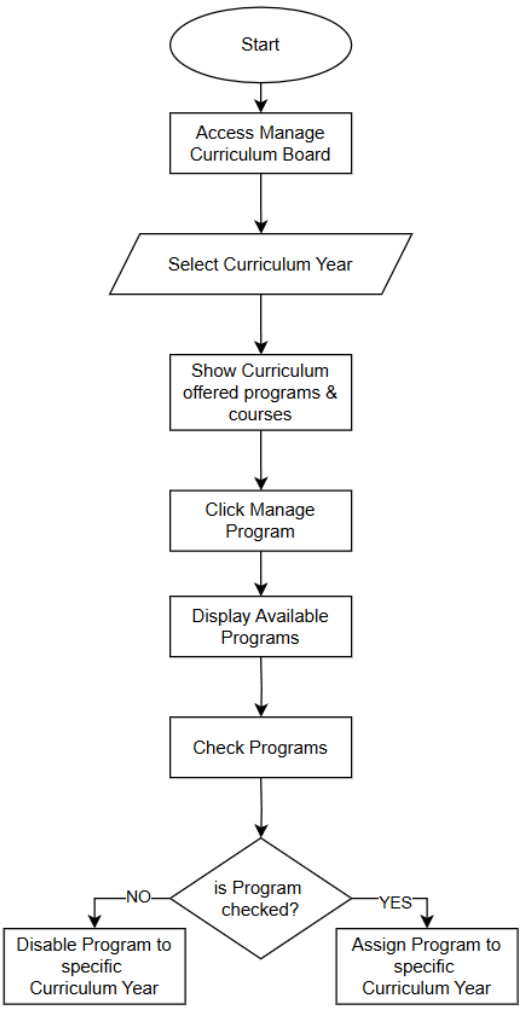


Figure () shows the process for managing programs in the system. The user begins by accessing the "Manage Curriculum Board" and selecting the curriculum year. The system then displays the offered programs and courses for that curriculum year. The user clicks on "Manage Program" to view the available programs. The programs are then checked to determine if they are selected. If a program is checked, it is assigned to the specific curriculum year. If it is not checked, the program is disabled for that curriculum year, and the process ends.

Figure (). Adding Academic Years in the Admin Module

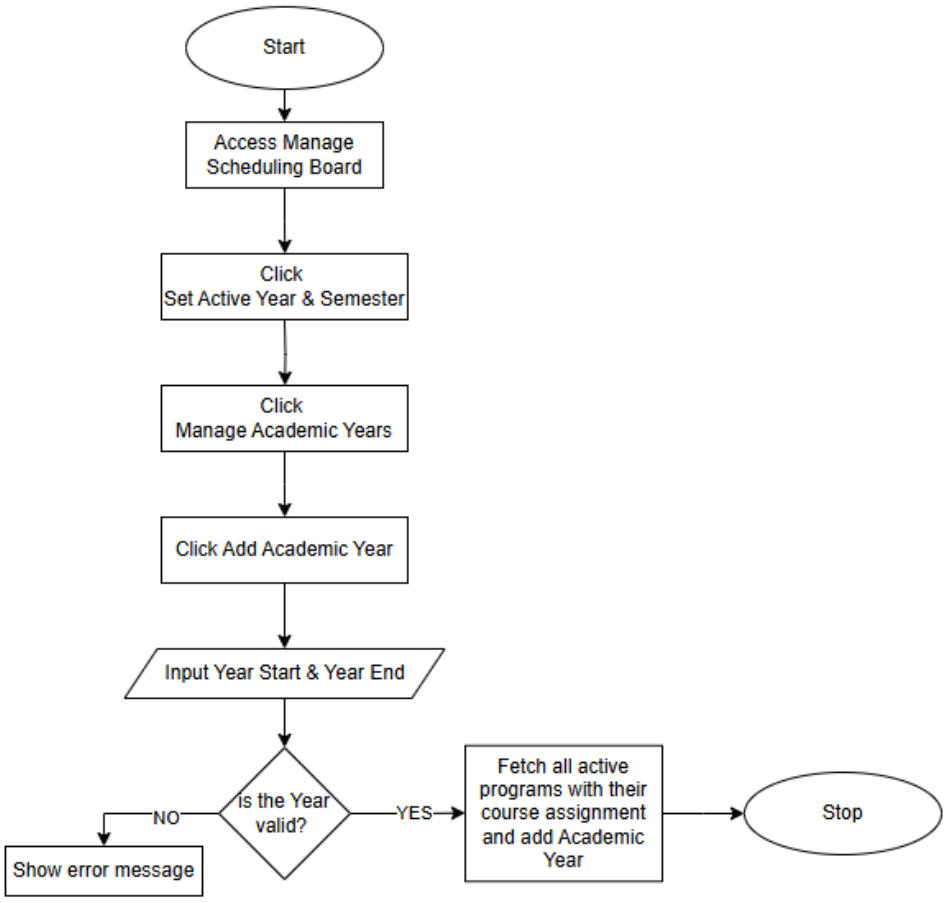


Figure () shows the process for managing academic years in the scheduling board. The user starts by accessing the "Manage Scheduling Board" and clicking on "Set Active Year & Semester." Then, the user navigates to "Manage Academic Years" and selects "Add Academic Year." After that, the user inputs the start and end years for the new academic year. The system checks if the year is valid. If not, an error message is displayed. If the year is valid, the system fetches all active programs with their course assignments and adds the new academic year, completing the process.

Figure (). Managing Year Levels Program Curriculum in the Admin Academic Year Board

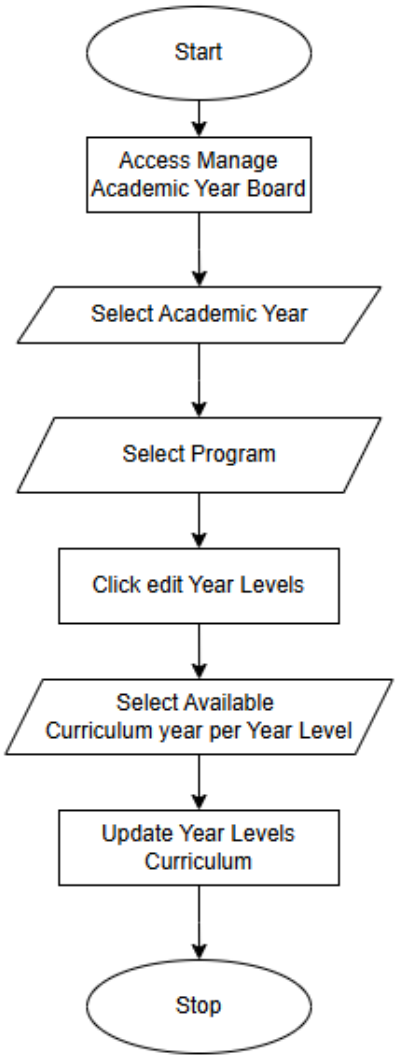


Figure () shows the process for updating year levels in the system. The user begins by accessing the "Manage Academic Year Board" and selecting the relevant academic year. Next, the user selects the program and clicks to edit the year levels. After that, the user selects the available curriculum year for each year level. Finally, the curriculum for the year levels is updated, and the process is complete.

Figure (). Managing Year Levels Program Sections in the Admin Academic Year Board

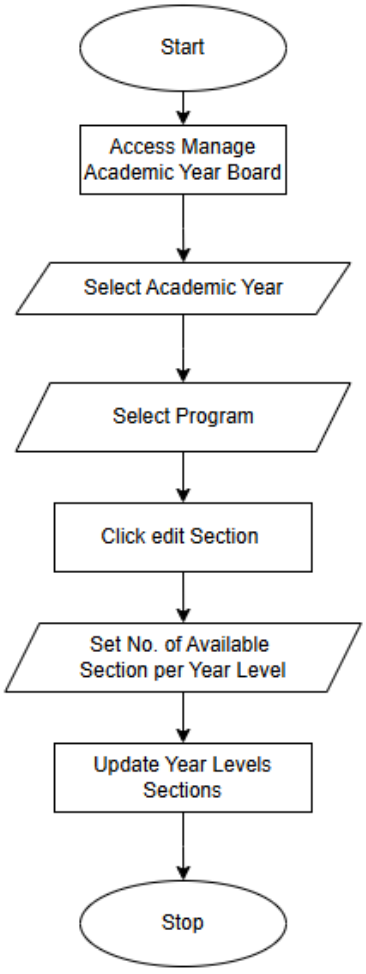


Figure () shows the process for updating sections in the system. The user begins by accessing the "Manage Academic Year Board" and selecting the academic year. After selecting the program, the user clicks to edit the sections. The next step is to set the number of available sections for each year level. Finally, the sections for the year levels are updated, and the process is complete.

Figure (). Deleting a Specific Program in the Selected Academic Year in Admin Academic Year Board

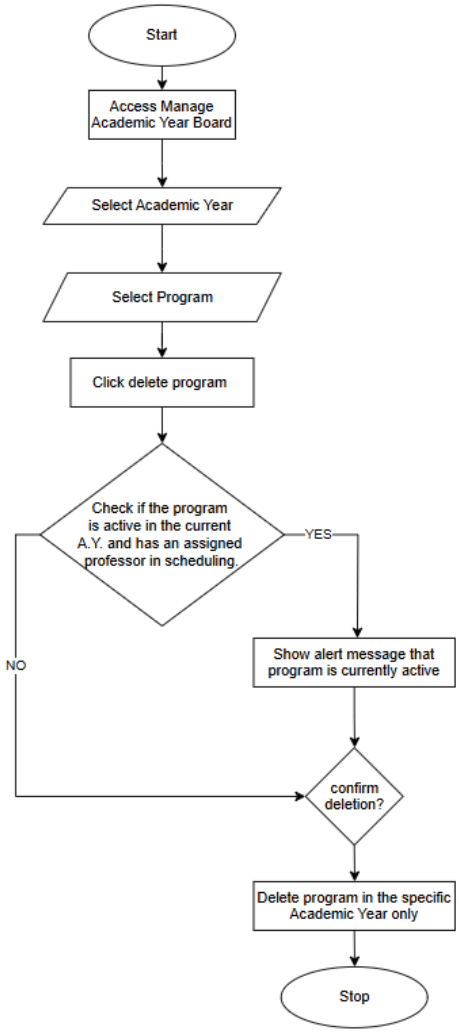


Figure () shows the process for deleting a program in the system. The user starts by accessing the "Manage Academic Year Board" and selecting the academic year and program. After selecting the program, the user clicks on "Delete Program." The system then checks if the program is active in the current academic year and has an assigned professor in scheduling. If the program is active, an alert message is displayed, indicating that the program cannot be deleted. If not, the system asks for confirmation to proceed with deletion. Upon confirmation, the program is deleted for the specific academic year, and the process concludes.

Figure (). Deleting a Academic Year in the in Admin Academic Year Board

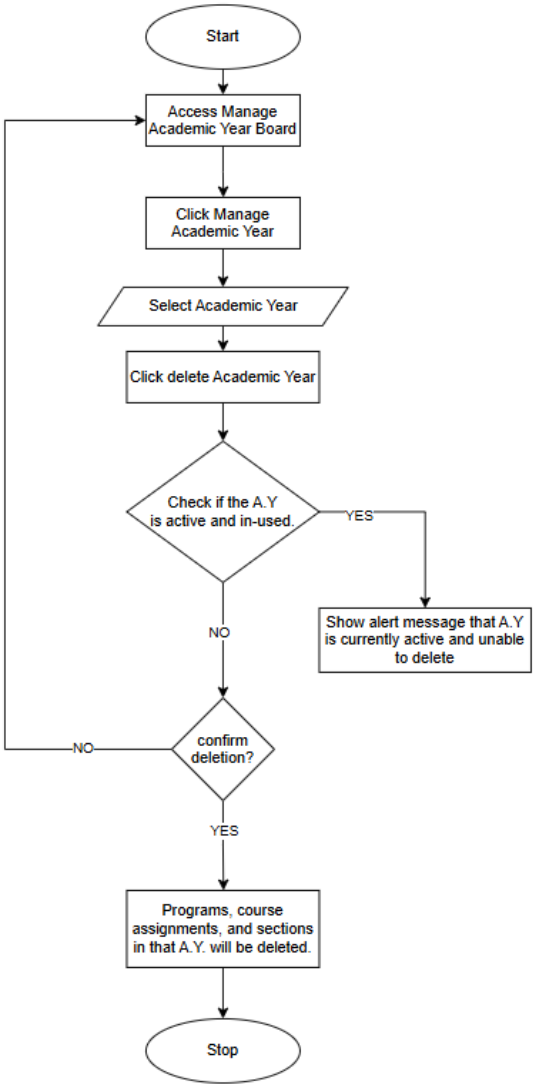


Figure () shows the process for deleting an academic year in the system. The user starts by accessing the "Manage Academic Year Board" and selecting "Manage Academic Year." After selecting the desired academic year, the user clicks "Delete Academic Year." The system then checks if the academic year is currently active and in use. If it is active, an alert message is shown, indicating that the academic year cannot be deleted. If the academic year is not in use, the system asks for confirmation to proceed with deletion. Upon confirmation, all programs, course assignments, and sections associated with that academic year will be deleted, and the process concludes.

Figure (). Set Active Semester in the Admin Scheduling Module

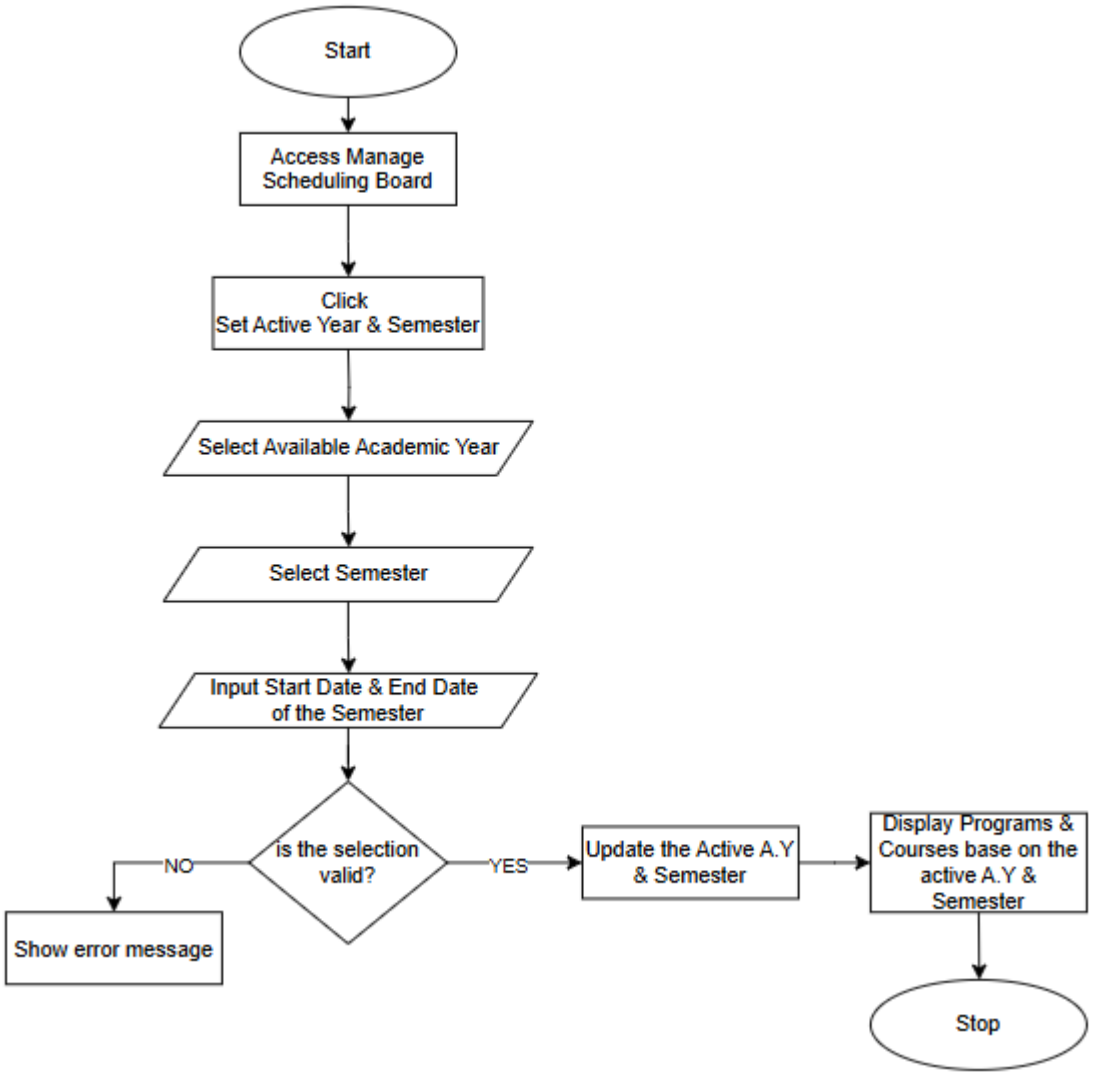


Figure () shows the process for setting the active academic year and semester in the system. The user begins by accessing the "Manage Scheduling Board" and selecting "Set Active Year & Semester." The next step is to choose an available academic year and semester. After selecting the semester, the user inputs the start and end dates for the semester. The system then checks if the selection is valid. If the selection is invalid, an error message is displayed. If valid, the active academic year and semester are updated, and the system displays programs and courses based on the newly set active year and semester, completing the process.

Figure (). Viewing and Generating Faculty Preferences in the Admin Module

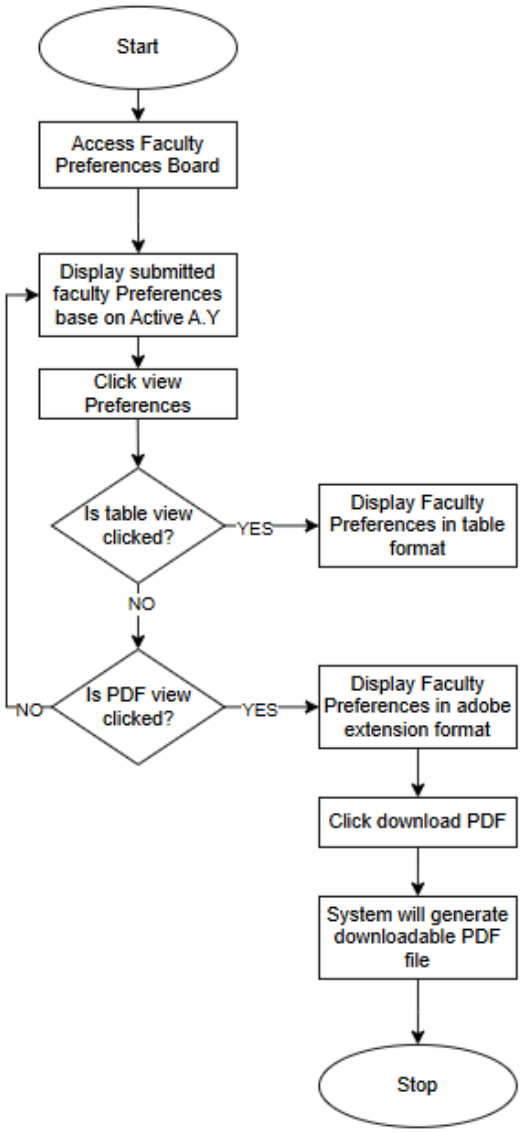


Figure () illustrates the process for viewing and generating faculty preferences in the system. The user starts by accessing the "Faculty Preferences Board" and displaying the submitted preferences based on the active academic year. The user can then click "View Preferences" to proceed. If the user selects the table view, the faculty preferences are displayed in a table format. If the PDF view is selected, the preferences are displayed in Adobe format. The user can then click "Download PDF," and the system will generate a downloadable PDF file, concluding the process.

Figure (). Exporting All Submitted Faculty Preferences

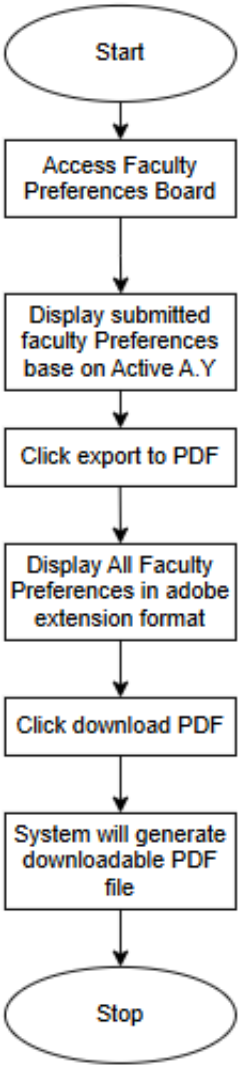
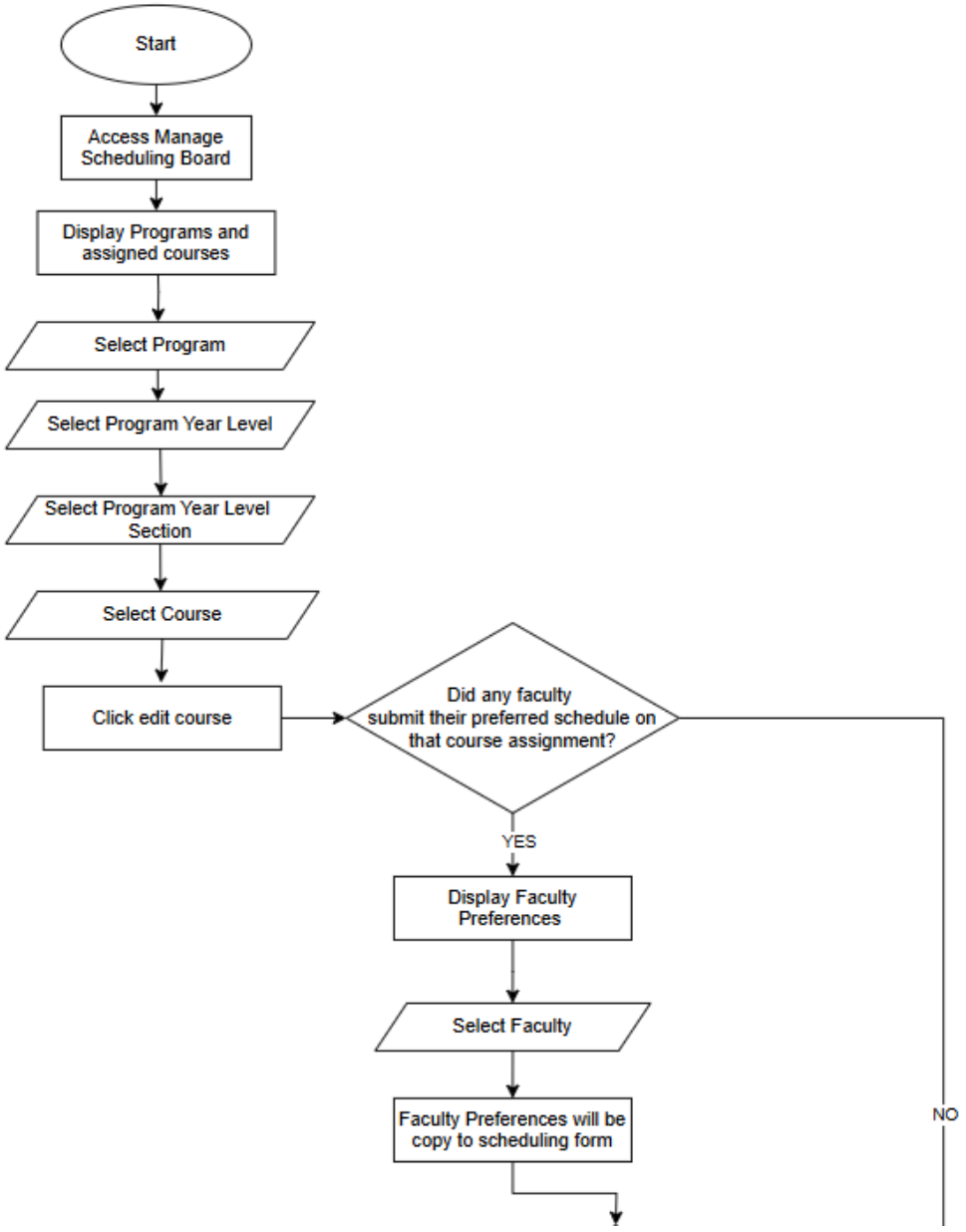


Figure () illustrates the process for exporting all submitted faculty preferences. The user begins by accessing the "Faculty Preferences Board" and displaying the submitted preferences based on the active academic year. The user then clicks "Export to PDF," and the system displays all faculty preferences in Adobe format. Afterward, the user clicks "Download PDF," and the system generates a downloadable PDF file, concluding the process.

Figure (). Scheduling Faculty to specific Course Assignment in the Admin Scheduling Module



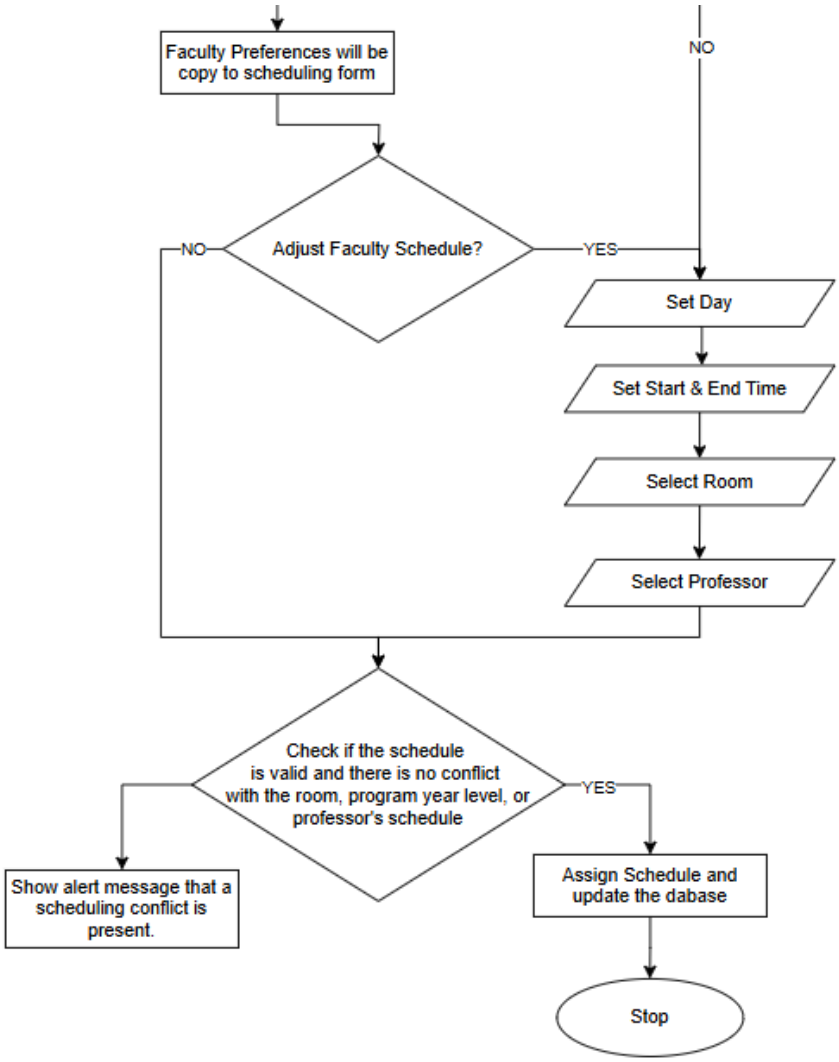


Figure () shows the process for scheduling of course assignment to specific program with specific year levels and sections. The user begins by accessing the "Manage Scheduling Board" and selecting a program, year level, section, and course. If a faculty member has submitted a preferred schedule for the course, the system displays those preferences. The user then selects the faculty and copies the preferences to the scheduling form. If needed, the user can adjust the schedule by setting the day, start and end times, room, and professor. The system then checks if the schedule is valid and free of conflicts with the room, program year level, or professor's schedule. If a conflict exists, an alert is shown. If the schedule is valid, it is assigned, and the system updates the database, completing the process.

Figure (). Exporting Faculty Assigned Load and Schedule in Admin Module

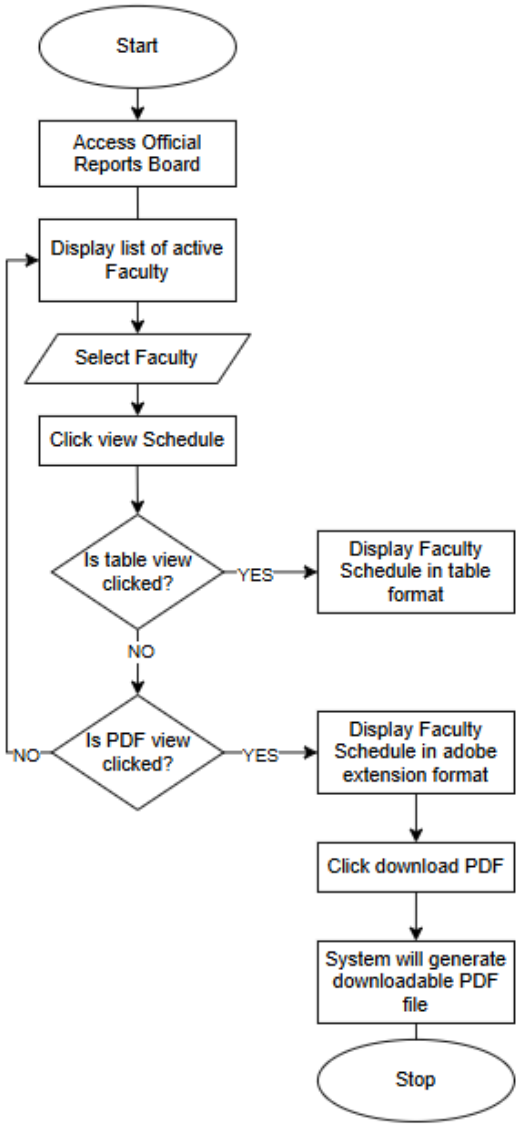


Figure () illustrates the process for exporting a faculty member's assigned load and schedule. The user starts by accessing the "Official Reports Board" and displaying the list of active faculty. After selecting a faculty member, the user clicks "View Schedule." If the table view is selected, the faculty schedule is displayed in table format. If the PDF view is selected, the schedule is displayed in Adobe format. The user can then click "Download PDF," and the system generates a downloadable PDF file, concluding the process.

Figure (). Exporting Program Assigned Load and Schedule in Admin Module

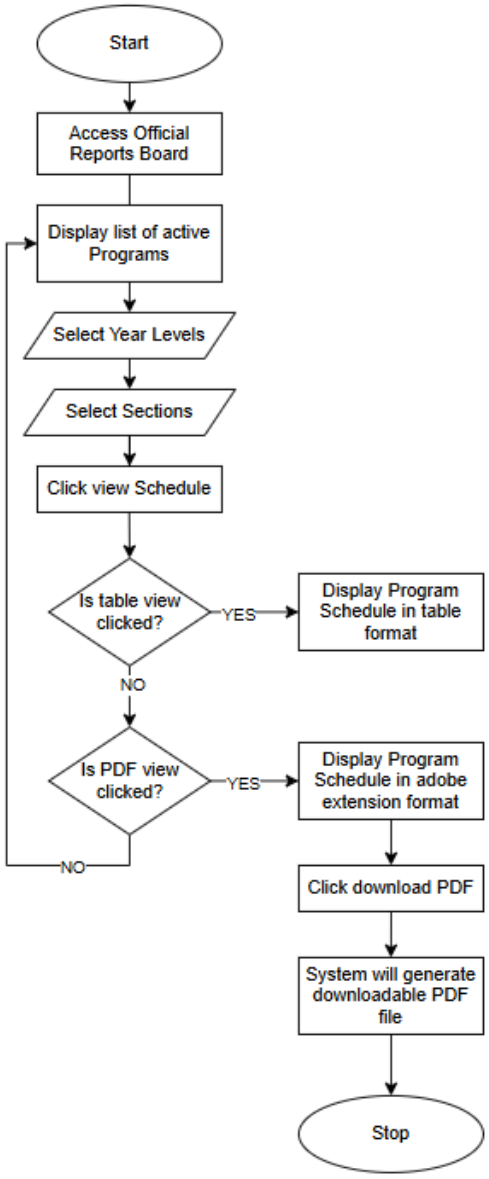


Figure () illustrates the process for exporting the schedule of a program. The user starts by accessing the "Official Reports Board" and displaying a list of active programs. The user then selects the year levels and sections before clicking "View Schedule." If the table view is selected, the program schedule is displayed in table format. If the PDF view is selected, the schedule is displayed in Adobe format. The user can then click "Download PDF," and the system generates a downloadable PDF file, concluding the process.

Figure (). Exporting Room Assigned Load and Schedule in Admin Module

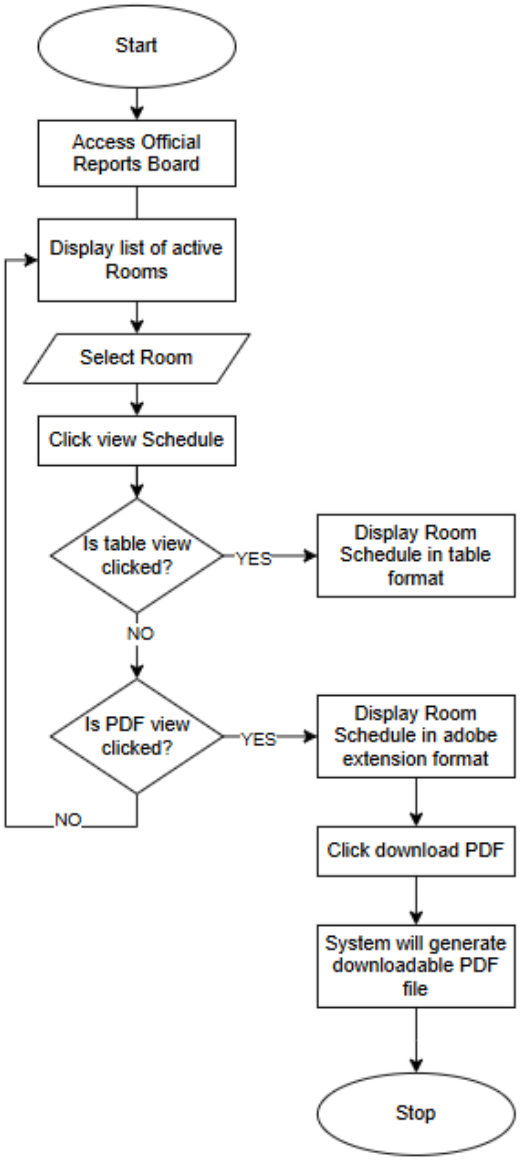


Figure () illustrates the process for exporting the schedule of a room. The user begins by accessing the "Official Reports Board" and displaying a list of active rooms. The user selects a room and clicks "View Schedule." If the table view is selected, the room schedule is displayed in table format. If the PDF view is selected, the schedule is displayed in Adobe format. The user can then click "Download PDF," and the system generates a downloadable PDF file, completing the process.

Figure (). Publishing Assigned Load and Schedule of Faculty in Admin Reports Module

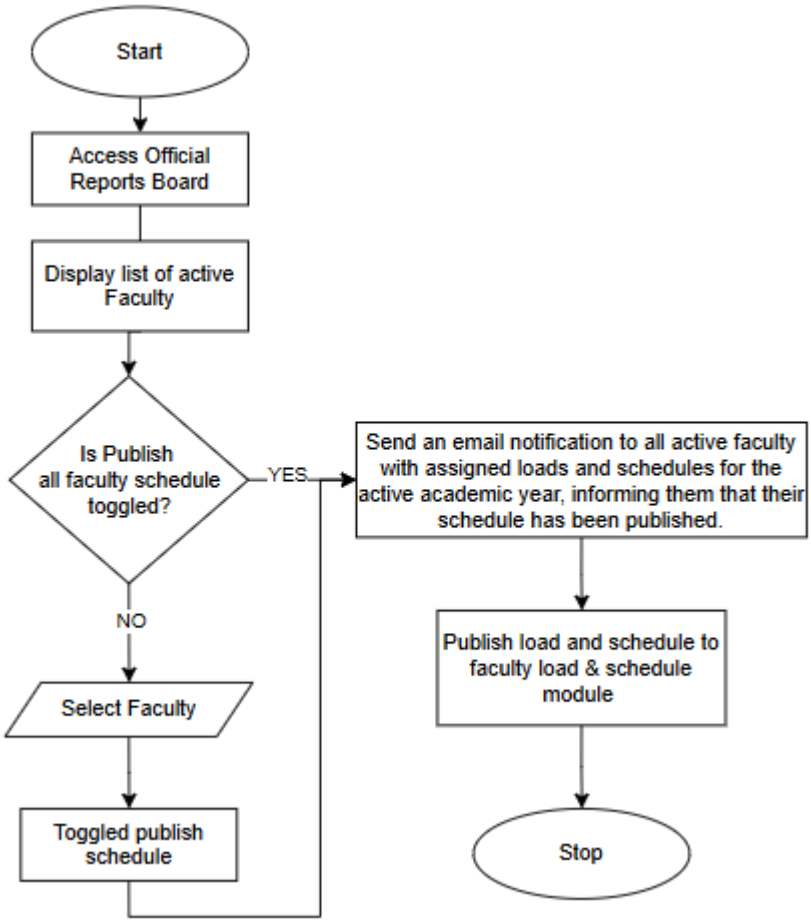


Figure () illustrates the process for publishing the load and schedule of faculty. The user begins by accessing the "Official Reports Board" and displaying a list of active faculty. If the option to "Publish All Faculty Schedules" is toggled, an email notification is sent to all active faculty with assigned loads and schedules for the active academic year, informing them that their schedule has been published. The system then publishes the load and schedule to the "Faculty Load & Schedule Module." If the "Publish All" option is not toggled, the user selects the individual faculty member, toggles the "Publish Schedule" option, and the schedule is published for that specific faculty member.

3.5.2 Database Design

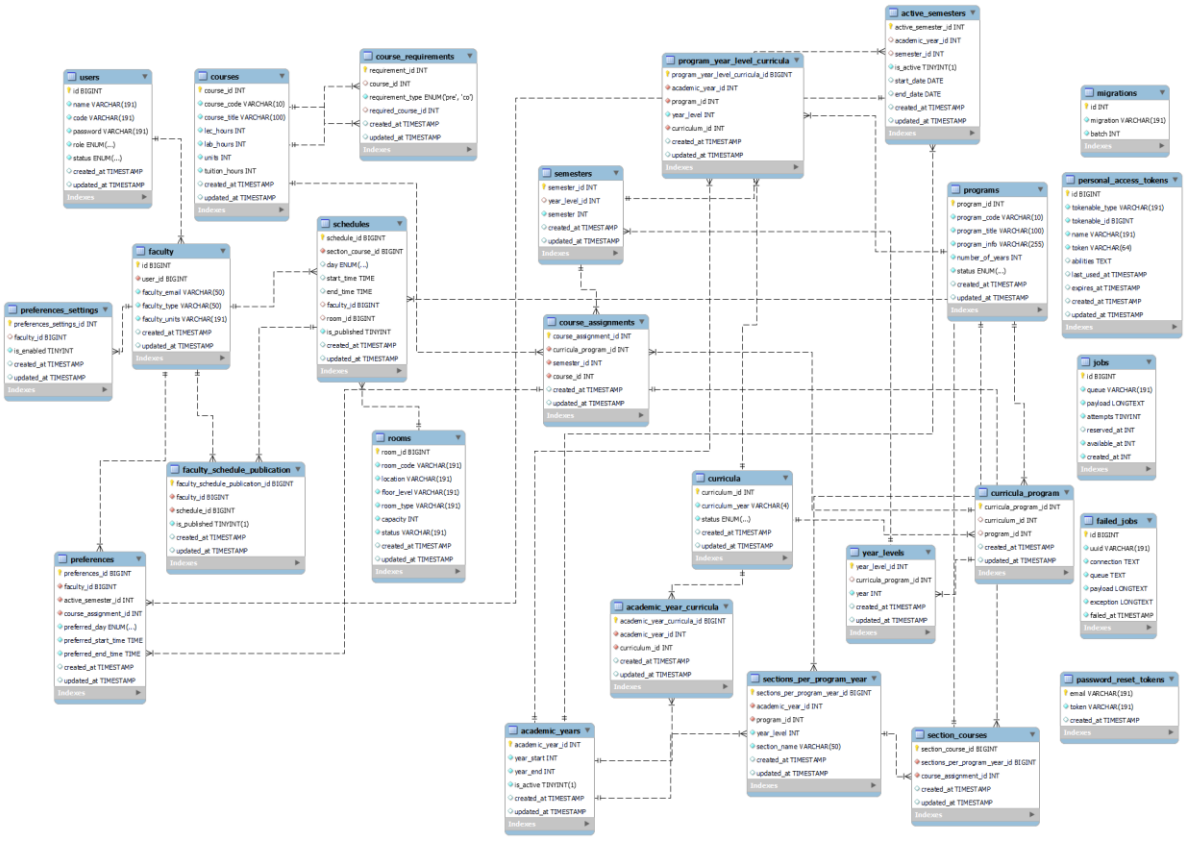


Figure 19: The new database design

3.5.3 Database Dictionary

users			
Field Name	**Data Type**	**Constraint**	**Description**
id	bigint UNSIGNED	PRIMARY KEY	Unique identifier for each flss user.
name	varchar(191)	NOT NULL	Full name of the user.

code	varchar(191)	NOT NULL, UNIQUE	Name or Label for the Type of KRA
password	varchar(191)	NOT NULL	Hashed password for user authentication.
role	Enum ('faculty','admin', 'superadmin')	NOT NULL	Role of the user within the system.
status	num ('Active','Inactive')	NOT NULL	Current status of the user account.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the user was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the user information was last updated

Table (). Data Dictionary – users

programs			
Field Name	**Data Type**	**Constraint**	**Description**
program_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each program.
program_code	varchar(10)	NOT NULL,	Unique code representing the program (e.g., BSIT).

		UNIQUE	
program_title	varchar(100)	NOT NULL	Official title of the program.
program_info	varchar(255)	NOT NULL	Brief description or information about the program.
number_of_years	int	NOT NULL	Duration of the program in years.
status	enum ('Active','Inactive')	NOT NULL	Current status of the program.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the program was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the program information was updated.

Table ().. Data Dictionary – programs

curricula			
Field Name	**Data Type**	**Constraint**	**Description**
curriculum_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each curriculum version.
curriculum_year	varchar(4)	NOT NULL,	Year associated with the curriculum (e.g., "2024").

		UNIQUE	
status	enum ('Active','Inactive')	NOT NULL	Current status of the curriculum.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the curriculum was created
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the curriculum was last updated.

Table ().. Data Dictionary – curricula

academic_years			
Field Name	**Data Type**	**Constraint**	**Description**
academic_year_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each program.
year	varchar(9)	NOT NULL	Academic year in a specific format (e.g., "2024-2025").
status	enum ('Active','Inactive')	NOT NULL	Current status of the academic year.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the academic year was created.

updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the academic year was updated last.

Table ().. Data Dictionary – academic_years

courses			
Field Name	**Data Type**	**Constraint**	**Description**
course_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each curriculum version.
course_code	varchar(10)	NOT NULL, UNIQUE	Year associated with the curriculum (e.g., "2024").
course_title	varchar(100)	NOT NULL	Current status of the curriculum.
units	int	NOT NULL	Timestamp when the curriculum was created
status	enum ('Active','Inactive')	NOT NULL	Timestamp when the curriculum was last updated.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the course was created

updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the course information was updated

Table (). Data Dictionary – courses

rooms			
Field Name	**Data Type**	**Constraint**	**Description**
room_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each room..
room_code	varchar(191)	NOT NULL	Code assigned to the room (e.g., "A201").
location	varchar(191)	NOT NULL	Location details of the room (e.g., "Building A").
floor_level	varchar(191)	NOT NULL	Floor level where the room is situated (e.g., "2nd").
room_type	varchar(191)	NOT NULL	Type of room (e.g., "Lecture", "Lab").
capacity	int	NOT NULL	Maximum number of occupants the room can accommodate.

status	varchar(191)	NULL DEFAULT NULL	Current status of the room (e.g., "Available").
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the room was added to the system.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the room information was updated

Table (). Data Dictionary – rooms

curricula_program			
Field Name	**Data Type**	**Constraint**	**Description**
curricula_ program_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each curriculum-program link.
curriculum_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated curriculum.

program_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated program.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the link was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the link was updated.

Table (). Data Dictionary – curricula_programs

year_levels			
Field Name	**Data Type**	**Constraint**	**Description**
year_level_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each curriculum-program link.
curricula_ program_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated curriculum-program link.

year	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Numerical representation of the year level (e.g., 1, 2, 3, 4).
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the year level was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the year level was updated.

Table (). Data Dictionary – year_levels

semesters			
Field Name	**Data Type**	**Constraint**	**Description**
semester_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each curriculum-program link.
year_ level_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated curriculum-program link.

semester	int	DEFAULT NULL, FOREIGN KEY	Numerical representation of the year level (e.g., 1 = 1st sem, 2 = 2nd sem, 3rd summer,).
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the semester was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the semester was updated.

Table (). Data Dictionary – semesters

course_assignments			
Field Name	**Data Type**	**Constraint**	**Description**
course_ assignment_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each course assignment.
curricula_ program_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated curriculum-program link.

semester_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated semester.
course_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the assigned course.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the course assignment was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the course assignment was updated.

Table (). Data Dictionary – course_assignment

course_requirements			
Field Name	**Data Type**	**Constraint**	**Description**

requirement_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each course requirement.
course_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the course that has the requirement.
requirement_type	enum ('pre','co')	NOT NULL	Type of requirement: 'pre' for prerequisite, 'co' for corequisite.
required_course_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the course that is required.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the requirement was created.

updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the requirement was updated.

Table (). Data Dictionary – course_requirements

sections_per_program_year			
Field Name	**Data Type**	**Constraint**	**Description**
section_per_ _program_ year_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each section.
academic _year_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated academic year.
program_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated program.
year_level	int	NOT NULL	Numerical representation of the year level (e.g., 1, 2, 3, 4).

section_name	varchar(50)	NOT NULL	identifier for the section (e.g., "1", "2").
created_at	timestamp	NULL DEFAULT NULL	Timestamp when section per program was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when section per program was updated.

Table (). Data Dictionary – sections_per_program_year

section_courses			
Field Name	**Data Type**	**Constraint**	**Description**
section_course_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each section-course association.

section_per_ _program_ year_id	bigint UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated section.
course_ assignment_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated course assignment.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the association was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the association was last updated.

Table (). Data Dictionary – section_courses

faculty			
Field Name	**Data Type**	**Constraint**	**Description**
id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each faculty member.
user_id	varchar(50)	NOT NULL, FOREIGN KEY	Reference to the associated user in the users table
faculty_email	varchar(50)	NOT NULL, UNIQUE	Official email address of the faculty member.
faculty_type	int	NOT NULL	Type of faculty (e.g., "Full-time", "Part-time")
faculty_units	varchar(50)	NOT NULL	Number of teaching units assigned to the faculty member.

created_at	timestamp	NULL DEFAULT NULL	Timestamp when faculty was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when faculty was updated.

Table (). Data Dictionary – faculty

schedules			
Field Name	**Data Type**	**Constraint**	**Description**
schedule_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each schedule entry.
section_ course_id	varchar(50)	NOT NULL, FOREIGN KEY	Reference to the associated section-course.

day	enum ('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday')	NOT NULL, UNIQUE	Day of the week when the course is scheduled.
start_time	time	DEFAULT NULL	Start time of the scheduled course
end_time	time	DEFAULT NULL	End time of the scheduled course.
faculty_id	bigint UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the faculty assigned to the course
room_id	bigint UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the room where the course is held

is_published	tinyint	NOT NULL, DEFAULT '0'	Indicates whether the schedule is published (1) or not (0)
created_at	timestamp	NULL DEFAULT NULL	Timestamp when schedule was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when schedule was updated.

Table (). Data Dictionary – schedules

active_semesters			
Field Name	**Data Type**	**Constraint**	**Description**
active_ semester_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each active semester entry.

academic_year_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated academic year.
semester_id	int UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated semester.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the active semester was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the active semester was last updated.

Table (). Data Dictionary – active_semester

preferences			
Field Name	**Data Type**	**Constraint**	**Description**

preferences_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each preference record.
faculty_id	bigint UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated faculty member.
active_semester_id	int UNSIGNED	NOT NULL, UNIQUE	Reference to the active semester context.
course_assignment_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the course assignment related to the preference.
preferred_day	enum ('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday',	NOT NULL	Preferred day of the week for teaching.

	'Saturday', 'Sunday')		
preferred_start_time	time	NOT NULL	Preferred start time for teaching.
preferred_end_time	time	NOT NULL	Preferred end time for teaching.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when preference was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when preference was updated.

Table (). Data Dictionary – preferences

preferences_setting			
Field Name	**Data Type**	**Constraint**	**Description**

preferences_settings_id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each preferences setting.
faculty_id	bigint UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated faculty member.
has_request	tinyint		1 for request made for preference, 0 for no request.
is_enabled	tinyint	NOT NULL, DEFAULT '1'	Indicates if preferences are enabled (1) or disabled (0).
global_deadline	date	NULL	Global deadline of preference for all users, if applicable.
individual_deadline	date	NULL	Specific submission preference deadline for individual users.

created_at	timestamp	NULL DEFAULT NULL	Timestamp when the setting was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the setting was updated.

Table (). Data Dictionary – preferences_setting

faculty_schedule_publication			
Field Name	**Data Type**	**Constraint**	**Description**
faculty _schedule_publication_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each publication record.
faculty_id	bigint UNSIGNED	DEFAULT NULL, FOREIGN KEY	Reference to the associated faculty member.
schedule_id	bigint UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated schedule.

is_published	tinyint(1)	NOT NULL, DEFAULT '0'	Indicates if the schedule is published (1) or not (0).
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the publication record was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the publication record was last updated.

Table (). Data Dictionary – faculty_schedule_publication

academic_year_curricula			
Field Name	**Data Type**	**Constraint**	**Description**
academic_ year_curricula _id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each academic year-curriculum link.

academic_ year_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated academic year.
curriculum_id	int UNSIGNED	NOT NULL, FOREIG N KEY	Reference to the associated curriculum.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the link record was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the link record was last updated.

Table (). Data Dictionary – academic_year_curricula

program_year_level_curricula			
Field Name	**Data Type**	**Constraint**	**Description**
program_ year_level_ curricula_id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each program-year-level-curriculum link.
academic_ year_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated academic year.
program_id	int UNSIGNED	NOT NULL, FOREIG N KEY	Reference to the associated program.
year_level	int	NOT NULL	Numerical representation of the year level (e.g., 1, 2, 3, 4).

curriculum_id	int UNSIGNED	NOT NULL, FOREIGN KEY	Reference to the associated curriculum.
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the association record was created.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the association was last updated.

Table (). Data Dictionary – program_year_level_curricula

migrations			
Field Name	**Data Type**	**Constraint**	**Description**
id	int UNSIGNED	PRIMARY KEY,	Unique identifier for each migration record.

migration	varchar(191)	NOT NULL	Name of the migration file (e.g., "create_users_table").
batch	int	NOT NULL	Batch number indicating the order of migrations.

Table (). Data Dictionary – migrations

personal_access_tokens			
Field Name	**Data Type**	**Constraint**	**Description**
id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each personal access token.

tokenable_type	varchar(191)	NOT NULL	Type of the model that owns the token (e.g., "App\Models\User")
tokenable_id	bigint UNSIGNED	NOT NULL, FOREIGN KEY	ID of the model that owns the token.
name	varchar(191)	NOT NULL, UNIQUE	Indicates if the schedule is published (1) or not (0).
abilities	text	NULL	JSON array specifying the abilities granted by the token.
last_used_at	timestamp	NULL DEFAULT NULL	Timestamp when the token was last used..
expires_at	timestamp	NULL DEFAULT NULL	Timestamp when the token expires
created_at	timestamp	NULL DEFAULT NULL	Timestamp when the token was created

updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the token was last updated

Table (). Data Dictionary – faculty_schedule_publication

jobs			
Field Name	**Data Type**	**Constraint**	**Description**
id	bigint UNSIGNED	PRIMARY KEY,	Unique identifier for each job entry.
queue	varchar(191)	NOT NULL	Name of the queue where the job will be processed
payload	longtext	NOT NULL	Serialized data containing job details and information.
attempts	Tinyint UNSIGNED	NOTNULL	Number of attempts made to process this job.

reserved_at	int UNSIGNED	NULL	Timestamp when the job was reserved for processing.
available_at	int UNSIGNED	NOT NULL	Timestamp when the job was reserved for processing..
created_at	timestamp	NULL DEFAULT NULL	Timestamp indicating when the job becomes available for processing.
updated_at	timestamp	NULL DEFAULT NULL	Timestamp when the job was created..

Table (). Data Dictionary – jobs

3.6 Functional Specifications

This section includes the system boundaries of the proposed system which are shown through a use case diagram.

3.6.1 System Boundaries

The PUP-Taguig will employ the Faculty Loading and Scheduling system, utilizing data and processes compiled by the group. This system is designed to handle administrative tasks and manage details and transactions for both administrators and professors. Use-case diagram was a vital tool in helping developers define the requirements for the system. It provides insight into how each subsystem communicates and outlines the necessary flow of transactions.

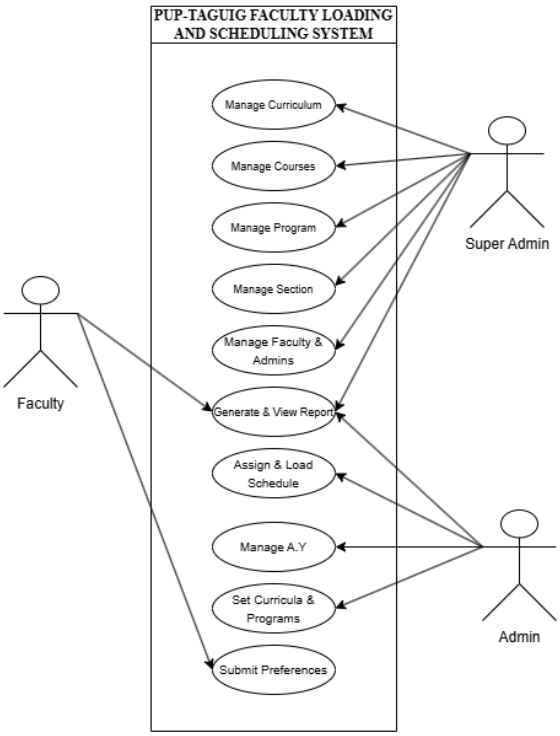


Figure 20: Use Case Diagram for Administrator and Faculty users of the enhanced PUP Taguig Faculty Loading and Scheduling System

3.7 Development Methodology

3.7.1 Process Model

With this project, researchers are using the Agile Methodology to develop the system's needs. The Agile Methodology assists the team in handling changing requirements and delivering high-quality products that meet or exceed our clients' expectations. The first step is the planning phase, during which researchers gather all the needed data related to the project, requirements from the client, and all the data and other elements necessary to start the project. The second phase involves designing, wherein researchers create plans based on the first phase. After designing, the third phase involves development based on the planning and designing phases from both previous stages. Testing occurs after the development phase and includes functional testing, user acceptance testing, and end-to-end testing. Releasing takes place after all the testing succeeds without errors or problems. During this phase, researchers implement the system and accept feedback from the client and target users, marking the beginning of the feedback phase. Using this model in developing a system helps researchers create a more effective and efficient solution for the client's main problems.

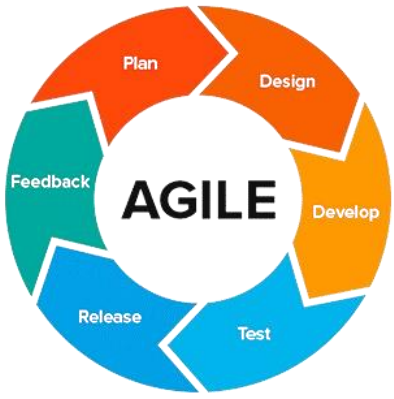


Figure 21. Process Model – Agile Development

3.7.2 Quality Plan

The objective of the quality management plan is to outline the procedures and activities required to ensure the system's quality is met and achieved. In the context of the PUPT Faculty Loading and Scheduling System project, the following steps will be taken to fulfill the client's needs and adhere to system requirements:

- *Review of Documents:* The aim is to verify if the document aligns with the system's functions and requirements. This involves reading the document, checking sources, and ensuring the accuracy of all involved data.

- *Code Review:* This step is intended to identify and rectify oversights made during the initial development phase. The purpose is to prevent the presence of unstable code that could negatively impact the system's processes, fostering continuous improvement.

- *System Testing:* Ensuring that the system meets the specified requirements outlined in the document is the focus of system testing. This step also aims to confirm that the system operates smoothly as intended. Detailed test cases will be written and executed to assess each feature of the system.

For an accurate evaluation of the system to be developed, researchers will use a survey based on ISO 25010, the International Standard for Software and Data Quality. This standard serves as a guide for creating a high-quality software product.

Below is a list of characteristics to consider when developing quality software:

- Functionality
- Reliability
- Usability
- Efficiency
- Compatibility
- Security
- Maintainability

3.8 Validity and Reliability

The success and credibility of any research study hinge on the robustness of its data, requiring a meticulous examination of both validity and reliability. In this section, we delve into the strategies and measures implemented to ensure the trustworthiness of the study outcomes.

3.8.1 Validity

Internal Validity: To ensure the internal validity of this study, rigorous measures have been implemented to accurately measure the impact of enhancements to the Polytechnic University of the Philippines (PUP) Faculty Loading and Scheduling System. The following steps have been taken:

1. **Alignment with Research Objectives:**

- All data collection instruments, including surveys and automated system metrics, have been meticulously designed to align with the specific research objectives. This ensures that the gathered data directly contributes to addressing the research questions and testing the study hypotheses.

2. **ISO 25010 Framework:**

- The research instrument, particularly the surveys, is structured based on the ISO 25010 standard to evaluate software product quality. By aligning with internationally recognized criteria for software quality, the study enhances the internal validity of the collected data.

3. **Pilot Testing:**

- Prior to full-scale implementation, the research instruments underwent pilot testing with a small group of participants. Feedback from the pilot study

facilitated refinement and ensured that the instruments effectively measure the intended constructs.

3.8.2 Reliability

Consistency and Stability: Reliability measures have been implemented to ensure the consistency and stability of the research findings. Key steps taken include:

1. **Structured Surveys:**
   - The structured surveys employ established scales and standardized questions, contributing to the reliability of responses. Likert scales, for instance, offer a consistent format for participants to express their opinions.

2. **Automated System Metrics:**
   - The use of automated system metrics, such as response time and error rates, ensures objective and consistent measurements of system performance. These metrics contribute to the overall reliability of the technical assessments.

3. **Pilot Testing and Calibration:**
   - Pilot testing not only served as a validity check but also allowed for the calibration of the instruments to enhance their reliability. Adjustments were made based on the pilot study outcomes to ensure that the instruments consistently measure the intended variables.

3.8.3 Addressing Potential Bias

Mitigation Strategies: Recognizing the potential for bias in the research process, the study incorporates several strategies to minimize and address bias:

1. **Transparent Communication:**

- Participants are provided with clear and transparent communication regarding the purpose of the study, the voluntary nature of their participation, and the unbiased intent of the research.

2. **Diverse Participant Selection:**

- Purposive sampling is employed to intentionally select participants based on specific roles and expertise. However, efforts have been made to ensure diversity within the chosen group, minimizing the risk of bias in participant perspectives.

3. **Anonymous Data Collection:**

- Responses collected through surveys are anonymized to encourage honest and unbiased feedback. Participants are assured that their responses will be kept confidential, fostering an environment conducive to open and unbiased input.

4. **Continuous Monitoring:**

- Throughout the data collection process, the research team actively monitors for potential sources of bias. Any identified issues are promptly addressed to maintain the integrity of the study.

By addressing potential sources of bias and implementing measures to enhance both validity and reliability, this study aims to produce robust and trustworthy findings that contribute meaningfully to the understanding of the enhanced PUP Faculty Loading and Scheduling System.

3.9 Development Tools and Technologies

In the pursuit of creating an effective and robust system, careful consideration was given to the selection of development tools and technologies. This section provides a general overview of the overarching technological framework employed in the creation of the PUP Taguig Faculty Loading and Scheduling System. The choices made in this realm were driven by a combination of factors, including project requirements, scalability, maintainability, and alignment with industry best practices.

3.9.1 Software Development Framework

The enhancement of the Polytechnic University of the Philippines (PUP) Faculty Loading and Scheduling System involves the adoption of both the Laravel and Angular frameworks. Laravel, a robust and widely-used PHP framework, is chosen for its versatility, scalability, and adherence to modern software development practices, providing a strong backend foundation. Meanwhile, Angular, a popular front-end framework developed by Google, is selected for its ability to build dynamic, responsive, and interactive user interfaces. The combination of Laravel for the back-end and Angular for the front-end ensures a seamless integration of the new curriculum, offering a more efficient and user-friendly experience.

3.9.2 Database Management System

The study employs MySQL as the primary database management system for storing and retrieving data related to faculty loading, scheduling, and curriculum information. MySQL is recognized for its reliability, performance, and ease of integration with Laravel, making it a suitable choice for managing the system's database.

3.9.3 Front-End Development Technologies

The user interface (UI) redesign component of the study leverages cutting-edge front-end development technologies to enhance both visual aesthetics and user experience. These technologies are carefully selected to create a responsive, engaging, and intuitive interface. The key front-end development technologies employed in this study include:

1. **HTML5/SCSS:**
    - HTML5 provides the core structure of the user interface, while SCSS (Sassy CSS) is utilized for styling with enhanced capabilities over traditional CSS, offering better organization, reusability, and maintainability. This combination ensures flexibility and responsiveness for a seamless experience across devices.
2. **Angular 18 with Angular Material:**
    - Angular 18 is used as the primary framework for building dynamic and robust front-end applications. It allows the development of responsive, scalable, and high-performance web applications. Angular Material, a UI component library for Angular, is integrated to create visually appealing and consistent user interfaces, adhering to modern design standards and enhancing user experience with pre-built, responsive UI components.
3. **JavaScript (TypeScript):**
    - JavaScript, specifically TypeScript (the recommended language for Angular), is employed to implement dynamic and interactive elements within the user interface. This ensures a modern, efficient, and user-friendly design.
4. **NPM (Node Package Manager):**

- NPM, a modern library and package manager for JavaScript, is used to manage dependencies and streamline the development process. It provides access to a wide range of libraries and tools that help in optimizing performance and adding advanced functionality to the system.

3.9.4 Version Control and Collaboration

Git is employed as the version control system to manage the source code and facilitate collaborative development. The use of Git allows for efficient tracking of changes, collaboration among team members, and the ability to roll back to previous versions if needed.

3.9.5 Project Management

For effective project management and collaboration, the study utilizes Notion. Notion provides a comprehensive and flexible platform for project planning, task management, and collaboration, enabling the development team to organize work seamlessly and enhance communication.

3.9.6 Hosting and Deployment

Hostinger is selected as the hosting and deployment platform for the enhanced system. Hostinger offers reliable and cost-effective hosting solutions, ensuring the availability and accessibility of the system for end-users. The deployment process is streamlined through Hostinger's infrastructure, contributing to a smooth and efficient release of updates.

3.9.7 Security Measures

Security is a paramount consideration in the development process. The study incorporates industry-standard security practices and tools to ensure the protection of sensitive data and system integrity. Key security measures include:

1. **SSL/TLS Encryption:**
   - Secure Sockets Layer (SSL) and Transport Layer Security (TLS) encryption are implemented to secure data transmission between the user interface and the server, ensuring that all communication remains private and tamper-proof.

2. **Testing of Hosting Environment:**
   - The hosting environment undergoes rigorous security testing using OWASP ZAP (Open Web Application Security Project Zed Attack Proxy) to identify vulnerabilities, such as cross-site scripting (XSS) and SQL injection, ensuring the system is protected from common web application attacks.
   - Qualys, a leading vulnerability management tool, is used to conduct in-depth scans of the server and network environment to detect potential weaknesses and ensure compliance with security standards.

3.9.9 Testing Tools

A variety of testing tools are employed to ensure the system's quality and reliability. PHPUnit is used for unit testing in Laravel, while Laravel Dusk handles browser testing to simulate user interactions. Postman is utilized for API testing to verify secure and accurate data transmission. For front-end testing in Angular, Jasmine and Karma are used together, with Jasmine providing a behavior-driven framework and Karma automating test execution across browsers. These tools work together to ensure comprehensive testing of both the back-end and front-end components.

REFERENCES

Abed, M. (2021). Development of a faculty workload management system using optimization algorithms. *International Journal of Scientific Research in Computer Science and Engineering,* 10*(*2), 327-333.

Baccay, J., & Cabahug, N. (2019). Class scheduling and faculty loading system using optimization algorithm. *International Journal of Advanced Trends in Computer Science and Engineering,* 9*(*4), 4243-4248.

Nuengwong, Tuaycharoen., Veerawat, Prodpran., Boonsong, Srithong. (2018). ClassSchedule: A web-based application for school class scheduling with real-time lazy loading. *IEEE International Conference on Business and Information Research (ICBIR)*, 210-214. doi: 10.1109/ICBIR.2018.8391194

Louie, F., Agustin, J. K. A., Losito, A. R. Mendoza, A. T., Factoriza, A. (2018). Predictive Analytics Implementing Genetic Algorithm on Class Scheduling System. *International Journal of Advanced Research in Computer Science,* 9(6), 18-22. doi: 10.26483/IJARCS.V9I6.6333

Botangen, K. A. (2016). Class-Scheduling System for the Central Luzon State University [Abridged]. Retrieved from ResearchGate website: (PDF) Class-Scheduling System for the Central Luzon State University

Miranda, J. (2010). eClasSkeduler: *A Course Scheduling System For The Executive Education Unit At The Universidad De Chile*. Retrieved on January 13, 2015 from http://goo.gl/6ljMn

Etcuban, J. O. (2013). Automated test generator for the faculty of the University of Cebu, Philippines. *IAMURE International Journal of Mathematics, Engineering & Technology*, 6, 30. Retrieved on March 2015 from http:// goo.gl/aLp1Ry

Ortega, E. P., & Brigoli, D. I. (2015). *Online Class Scheduling and Faculty Loading System within a Decision Support Framework*. JPAIR Institutional Research, 6(1), 61-77.

Chaiwchan, W., & Klinhom, P. (2014). *The Development of Online-Class Scheduling Management System Conducted by the Case Study of Department of Social Science: Faculty of Humanities and Social Sciences Suan Sunandha Rajabhat University.* International Journal of Computer and Information Engineering, 8(8), 2740-2744.