**Philippine Copyright 2022**

by the Authors and the

Parañaque City Campus

Polytechnic University of the Philippines

All rights reserved. Portions of this manuscript may be reproduced with proper referencing and due acknowledgment of the author.

# CLASSROOM SCHEDULING SYSTEM FOR PUP PARAÑAQUE CITY CAMPUS

A Thesis

Presented to the Faculty of Parañaque City Campus Polytechnic University of the Philippines

Parañaque City, Metro Manila

In Partial Fulfillment of the Requirements for the Degree Bachelor of Science in Information Technology

by

**DY, JOHN A.**

**MARTINEZ, SHERWIN IVAN C.**

**PAIRA, SOFIA G.**

**RAMOS, JOSEPH LAWRENCE R.**



### CERTIFICATION

This thesis, ***DEVELOPING A CLASSROOM SCHEDULING SYSTEM FOR PUP***

***PARANAQUE CAMPUS*,** prepared and submitted by **AARON JOSH F. MENDIETA, JOHN JUBIEN SANTOS, and JANINE VILLAR** in partial fulfillment of the requirements for the degree **BACHELOR OF SCIENCE IN INFORMATION TECHNOLOGY** has been examined and recommended for **Oral Examination**.

#### Evaluation Committee

ALDRIN P. OBSANGA

Adviser

MARVIN C. DE PEDRO CATHERINE R. LLAVE, MLing

Member Member

### APPROVAL

Approved by the **Panel on Oral Examination** on **April \_, 2022** with the grade of .

RIBERT R. ENIERGA, MIT

Chair

MARVIN C. DE PEDRO CATHERINE R. LLAVE, MLing

Member Member

Accepted in partial fulfilment of the requirements for the degree Bachelor of Science in Information Technology.

CLEOTILDE L. CRESCINI, DBA

Campus Director

ii

## ACKNOWLEDGMENT

First and foremost, praises and thanks to God the Almighty, for His showers of blessings throughout this research work to complete the research successfully. The researchers would like to express his deep and sincere gratitude to the people who have supported to complete the research work directly and indirectly. The researchers would like to thank the Polytechnic University of the Philippines – Parañaque Campus, especially to Prof. Catherine R. Llave for her kindness and agreeing to take part in the survey and for sharing their experience while using their existing manual system. Her cooperation made the objectives of this research realized.

iii

**CERTIFICATION OF ORIGINALITY**

This is to certify that the research work presented in this thesis entitled *Developing a Classroom Scheduling System for Polytechnic University of the Philippines Parañaque Campus,* 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 by the undersigned. This thesis does not contain words or ideas taken from published sources or written works that have been accepted as basis for the award of a degree from any higher education institution, except where proper referencing and acknowledgement were made.

**AARON JOSH F. MENDIETA**

**JANINE D. VILLAR**

**JOHN JUBIEN A. SANTOS**

Researchers February 5, 2022

iv

## ABSTRACT

|  |  |  |
| --- | --- | --- |
| Title | : | *Developing a Classroom Scheduling System* |
|  |  | For PUP Parañaque Campus |
| Researchers | : | Mendieta, Aaron Josh Santos, John Jubien Villar, Janine |
| Degree | : | Bachelor of Science in Information Technology |
| Institution | : | Polytechnic University of the Philippines |
| Year | : | 2022 |
| Adviser | : | Prof. Aldrin Obsanga |

This project was designed to help improve and change the manual process of creating schedules of professors at the Polytechnic University of the Philippines – Paranaque Campus, which lead to the creation of the classroom scheduling system.

The key features of the systems are to lessen the workload in a way that the system will suggest a schedule to the administrator, to put constraints, and the admin can print the schedule from the system into Excel or PDF file. These features will help to achieve the objectives of the system. The ISO Standard 25010 was used in this project to evaluate the performance of the developed software. A survey questionnaire was utilized in gathering data from the client and ten (10) IT professionals to evaluate the system.

v

The process used in developing the software was a rapid application development model that consists of a succession of short, iterative development cycles that result in completing the software through the phases of the following requirements: planning, user design, testing, and (d) implementation. The project progress is less monitored in planning and has more priority to the development tasks.

The overall data that have been gathered based on the survey result in an overall total mean of all parts of this survey, so the total mean of all these tables is “3.56” which falls on the strongly agreed interpretation of the four-point Likert scale that shows that the IT professionals and the client strongly agree that the system created was on par with the ISO Standard 25010.

Keywords: Polytechnic University of the Philippines, Classroom Scheduling System, Information System, Scheduling, Technology

vi

## TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
|  |  | Page |
| **Title Page** |  | i |
| **Certification and Approval Sheet** | | ii |
| **Acknowledgment** | | iii |
| **Certification of Originality** | | iv |
| **Abstract** |  | v |
| **Table of Contents** | | vii |
| **List of Tables** |  | ix |
| **List of Figures** | | x |
| **Chapter 1** | **The Problem and Its Setting** |  |
|  | Introduction | 1 |
|  | Purpose and Description | 3 |
|  | Objectives of the Study | 4 |
|  | Scope and Limitations of the Study | 5 |
|  | Definition of Terms | 6 |
| **Chapter 2** | **Review of Related Literature and Studies** |  |
|  | Technical Background | 8 |
|  | Difficulties of Manual Classroom Scheduling | 10 |
|  | Automated Classroom Scheduling | 12 |
|  | Various Algorithm used to create Scheduling Systems | 13 |
|  | Implementation of Scheduling System | 14 |
|  | Efficiency of Class Scheduling | 16 |
|  | Synthesis of the Reviewed Literature and Studies | 19 |

vii

|  |  |  |
| --- | --- | --- |
| **Chapter 3** | **Methodology** |  |
|  | Requirement Analysis | 20 |
|  | Requirements Documentation | 22 |
|  | Software Development Life Cycle | 23 |
|  | Design of Software, Systems, Product and Processes | 26 |
| **Chapter 4** | **Presentation, Analysis and Interpretation of Data** |  |
|  | User Interface | 29 |
|  | Statistical Treatment of Data | 37 |
| **Chapter 5** | **Summary of Findings, Conclusions and Recommendations** |  |
|  | Summary of Findings | 56 |
|  | Conclusion | 61 |
|  | Recommendations | 63 |
| **References** |  | 64 |
| **Appendices** |  | 72 |
|  | Appendix 1 Permission Letter | 73 |
|  | Appendix 2 Project Charter | 74 |
|  | Appendix 3 Cost Estimate | 75 |
|  | Appendix 4 Questionnaire | 76 |
|  | Appendix 5 Picture with the Client | 81 |
|  | Appendix 6 Transcript of the Interview | 82 |
|  | Appendix 7 Source Code | 88 |
|  | Appendix 8 Certificate of Grammarian | 171 |
|  | Appendix 9 Plagiarism Check Result | 172 |
|  | Appendix 10 Biographical Statement | 174 |

viii

## LIST OF TABLES

|  |  |  |
| --- | --- | --- |
| Number | Title | Page |
| 1 | **Respondents Perception to Functional Suitability** | 38 |
| 2 | **Respondents Perception to Performance Efficiency** | 49 |
| 3 | **Respondents Perception to Compatibility** | 40 |
| 4 | **Respondents Perception to Usability** | 42 |
| 5 | **Respondents Perception to Reliability** | 44 |
| 6 | **Respondents Perception to Security** | 46 |
| 7 | **Respondents Perception to Maintainability** | 48 |
| 8 | **Respondents Perception to Portability** | 52 |
| 9 | **Summary of the overall data** | 54 |

ix

## LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| Number | Title | Page |
| 1 | **Context Diagram** | 20 |
| 2 | **Level 0 – Diagram** | 21 |
| 2.1 | **Level 1 - Diagram (3.0 Manage Schedule)** | 21 |
| 3 | **ER Diagram** | 22 |
| 4 | **RAD Model of Classroom Scheduling System** | 23 |
| 5 | **IPO** | 25 |
| 6 | **System Architecture** | 26 |
| 7 | **Login Interface** | 29 |
| 7.1 | **Dashboard Interface** | 30 |
| 7.2 | **System Settings Interface** | 31 |
| 7.3 | **Course Window** | 31 |
| 7.4 | **Room Window** | 32 |
| 7.4.1 | **Room Window** | 33 |
| 7.5 | **Professor Window** | 33 |
| 7.5.1 | **Professor Window** | 34 |
| 7.5.1.1 | **Add New Schedule Interface** | 35 |
| 7.6 | **Subject Window** | 36 |
| 7.6.1 | **Subject Window** | 36 |

**Chapter 1**

**THE PROBLEM AND ITS SETTING**

Every year, despite the rising trend in technology, many institutions and colleges in the Philippines still conduct everyday operations using a manual procedure. One of these is a scheduling conflict that the institution prepares each semester. It is crucial, especially for students and teachers, to have an accurate and precise timetable so they can plan ahead. The majority of public universities and colleges manually alter and create schedules for different bachelor's degrees and year levels. (Kurniawan, 2017)

At all educational levels, students must have a set of class times. These schedules are provided to teachers and students at the beginning of each semester or school year. The course name, building name, room number, course number, instructor's name, and the day and hour of the class are often included in a schedule. Additionally, it will include information about any labs or lectures that are required for the course. The most common methods for creating the class schedule are manual methods or Microsoft Excel. (Aziz, 2018)

Creating the class schedules, allocating professor workloads, and planning room usage for the faculty and students for the semester before classes ever begin. The manual system is more time-consuming and labor-intensive, which can occasionally result in conflicts in the designated rooms for the various courses and year levels and a need to revise class schedules (Anuta, 2019).

The creation of class schedules for the various courses in the school might be made easier by switching from the manual scheduling procedure to a system-based one. Scheduling mistakes and conflicts can be reduced, and time and resources can be saved.

Timetables provide effective scheduling across numerous courses and year levels. Students and lecturers are less likely to become confused by a well-organized class schedule. Simple mistakes can delay planning and need additional time to correct (Sayson, 2018). One of the primary benefits of using a classroom scheduling system in schools is that it reduces workload, requires less planning time, and results in less mistakes being made (Starben, 2021).

Every semester in colleges and universities, course times, locations, and instructors should be organized by teaching and curriculum framework. Scheduling follows a set of criteria that consider aspects such as the time and location of teaching, as well as the teachers. It says that Class scheduling in most colleges and universities is manually implemented by teaching staff, who encounter numerous difficulties because of the large number of courses and teachers, required instruction places, and multiple constraints. (Wen-jing, 2018).

However, some schools in Metro Manila created their class schedules manually. This approach necessitates additional time to act, arrange, and amend the department head-provided class schedules, space utilization, and instructor workload. The university officials in charge of planning the class schedule and allocating students to these classes must take their effects into account. An efficient class timetable combines a straightforward structure with enough latitude for the teacher to adjust to unanticipated events during school hours.

Unquestionably, one of the many tasks that each division head must complete before classes begin is setting up class schedules, teacher workloads, and room usage for the students and faculty in each department. However, the school created the class schedule manually. The manual system requires more time and labor to plan, arrange, and amend the department heads' submitted class schedules, room usage, and instructor load. In response to these issues, the researchers got the idea of developing a class scheduling system using JavaScript and MongoDB. Through this development, operational errors have been reduced and resources like time and labor have been preserved. (Gamale, 2020)

When Polytechnic University of the Philippines - Paranaque was founded in 2011, the campus' construction initially took place in a warehouse. To create a schedule for both students and professors, the campus administrator uses manual scheduling methods that include a combination of paper-based requests, Excel, and printing enormous Excel spreadsheets. During the years, while conducting a scheduling table can take hours or even days, manual scheduling helps the campus manage the time allotted for instructors and students.

Currently, the issue in PUP Paranaque, according to the Academic Head, is that students are having issues with their designated rooms because some sections already have an assigned room but there are other students who are loitering or occupying it, which results in the classroom being unavailable due to incorrect scheduling of students' classes for each course.

**Purpose and Description**

The suggested system is a desktop program that can generate a timetable based on either the professor's or the room's schedule. The scheduling would be managed by the software that was developed. A login form, calendar search, and amending subject, room, or professor information are just a few of the software's vital functions.

The main functions of the systems are to reduce burden by suggesting an administrator's timetable, imposing restrictions, and allowing the administrator to print into an Excel or PDF file the schedule taken from the system. These attributes and some others that the researchers will add to the system will aid in achieving the system's goals:

**• The system will categorize how much time is allocated for part-time, regular, instructors with TS loads, etc.**

**• In the event that a professor has already gone over the permitted 40 hours, the system will alert the administrator.**

**• The system will generate or supply the professor's schedule.**

**• The system will let the administrator know whether or not the selected professor has a full schedule.**

**• Only the administrator is able to access the system.**

**Objectives of the Study**

**General Objective**

This system's primary goal is to develop the existing classroom scheduling system for the Polytechnic University of the Philippines Paranaque Institution, which will aid the campus in achieving the following specific goals:

**Specific Objectives**

1. To develop the Classroom Scheduling System,

2. To evaluate the effectiveness of the created PUP Paranaque Classroom Scheduling System

3. To evaluate the performance of the developed software using ISO Standard 25010

**Scope and Limitation**

The scope of the project is to grant administrators access to the system's features. The university will be able to establish and look for a certain professor or class schedule thanks to this suggested initiative. The data can be exported as an Excel file after being displayed in table format. The academic head's computer has access to the established Classroom Scheduling System. Without the administrator's account, which grants access to the system, it cannot be accessed.

The fact that this study is not linked to the university's information system and that only the academic head can use it and input data into the system limits its applicability.

**Definition of Terms**

1. **ElectronJS -** employing HTML5, CSS, and JavaScript, is a runtime framework for developing desktop-suite applications.
2. **End-user -** The person who will use a computer system when it has been properly built and configured.
3. **Excel / Spreadsheets -** a spreadsheet that has rows and columns of cells in it that may be used to calculate, sort, and arrange data. May also be used to assign schedules and manage timetabling.
4. **ExpressJS -** is a Node.js web application framework. It offers a lot of features that speed up and simplify the creation of web applications.
5. **FET -** is a free tool that automatically creates a schedule for a university, high school, or school. It makes use of an effective and speedy timetabling mechanism.
6. **Genetic algorithms -** a computer and artificial intelligence (AI) heuristic search technique.
7. **Local Access Network (LAN) –** Access points, cables, routers, and switches make up a computer network, usually referred to as a local area network, which enables devices to connect to web servers and internal servers located within a single building or campus.
8. **NodeJS -** a JavaScript runtime environment and framework that is open-source and cross-platform for running web applications outside of the client's browser.
9. **ReactJS -** a JavaScript package that combines the efficiency of JavaScript with a creative rendering technique to create highly dynamic websites that respond quickly to user input.

**Chapter 2**

### REVIEW OF LITERATURE AND STUDIES

This chapter presents the related studies about the Classroom Scheduling System reviewed by the researchers that further enhanced the study’s conceptualization and helped conceptualize the study and explain the cognitive design that influenced the researchers’ actions.

#### Technical Background

The school's current approach for scheduling classes is manual. The school lacks the current technology required and vital for a quick and simple method of handling the scheduling of the classrooms as well as the record searching process and production into an excel file.

The system was initially made by the former group and somehow, the app was helpful for the school and the client requested to add more features that will be useful for the future.

For the implementation of the proposed system, the school needs to provide a computer unit. Using the computerized classroom scheduling system and taking advantage of the technology in processing data, record keeping, searching, and their scheduling system will provide an effective means of processing information for the schools’ class schedules.

The researchers will use JavaScript as the main programming language, MongoDB to be used as a local database server on the computer, and Microsoft Excel to extract the data inside the database into a timetable.

JavaScript is a text-based programming language designed for creating a variety of applications that runs on any framework (Unity, NodeJS). The researchers

decided to use JavaScript because it is best for beginners, and it is mostly used to create a wide range of applications (Mukherjee 2021).

MongoDB is the most used database that includes all the tools needed to manage and navigate databases. The researchers decided to use MongoDB because it can be used easily as a free database admin tool and MongoDB uses collections and documents instead of tables and rows, as in traditional relational databases (Taylor, 2022).

The researchers used ReactJS for the front-end of the program, it is also the number choice because it is mostly used on applications these days, which run smoothly (Moore 2022).

ElectronJS is a framework that can make a website into a desktop application and the researchers used ElectronJS because it is well-known and it can increase the development of an application, also it could make application development significantly easier (Dryka, M. 2020).

ExpressJS is a framework to set-up a RESTful API and for back-end development. ExpressJS is used because it has a small learning curve and can be easily deployed and helps to build a server with RESTful API (White, L. 2021).

NodeJS is a framework for running JavaScript applications. The researchers used NodeJS because it is the heart of the system and without it, the system wouldn’t even run. It is fast to build real-time applications and increases the efficiency of development and it also executes properly (Gawron, K. 2018).

Classroom slots must be distributed to instructors and their courses from various departments along the timeline of each, according to the institution's specified rules. Some universities employed a manual way to organize the class schedule and load the professors. Class schedules, teacher load, and space in each department for students and staff are just a few of the many tasks that each department chair must complete before classes begin. Some schools, on the other hand, adopted a manual method of the class schedule. Because of the vast number of classes, class scheduling at a university is a large, difficult, and time-consuming undertaking. (Corchado, J. et al., 2020).

Manual course scheduling can be very complicated and time-consuming especially when using Excel Sheets, often even in violation of rigid and soft constraints. Soft restrictions typically apply to teacher expectations and student preferences as part of the schedule. Some of the schools will undoubtedly arrange the lesson schedule at the beginning of each semester. The process of setting up a school schedule can be a complicated and time-consuming problem. The process of preparing the schedule also requires high precision in order to avoid clashes between the resources involved in it. On scheduling in general, several resources need to be organized, including subjects, students, teachers, and study rooms (Ciptayani, P. et al. 2018).

Spreadsheets like Excel sheets are helpful in scheduling but can also be time-consuming when mistakes are made and if workers need a schedule on time, it will be frustrating to make a timetable. It is very prone to errors and a person will use a new spreadsheet just to redo the mistakes since using spreadsheets needs to be managed manually and carefully. (Gallegos, J., 2019)

#### *Automated Classroom Scheduling*

The advantages of the existing classroom scheduling system will help the students and lecturers to communicate easier than manual scheduling of appointments. Also it will allow the students to know the availability of their lecturers. Instead of looking for their lecturers manually, the proposed system will allow the student to view whether their lecturers are available or not and they can set their appointments to their lecturers, also it will allow the professors to arrange their schedules and they have the authority to accept or decline the appointment set by their students through internet. (Corpuz, D. 2018)

Automated Class Scheduling System has been in human requirements since created for managing time effectively. It is common teaching in schools, colleges, and other areas. Timetable preparation was performed manually in the early days with a single person or community participating in planning it with their hands, which requires a lot of effort and time. Scheduling even the smallest constraints can take much time, and the case is even worse when the number of constraints or the amount of data to deal with

increases. In such cases, a perfectly designed timetable is reused for the whole generation without any changes, proving dull in such situations (Bautista, J. et al. 2020).

Appointment Scheduling System designed for universities so that students can schedule appointments with their university’s faculty and staff regarding academic, employment, immigration, or personal issues. The scheduling system has the ability of being embedded to the Computer Science Capstone project iAcademic, which I took part in as a front-end developer. (Noori, 2021)

Each patient is assigned a specific path over a subset of the considered resources and each step needs to be scheduled. The main aim of these problems is to let each patient visit the resources in his or her subset within the allotted time to receive timely care. This is important because a delayed diagnosis or treatment may result in adverse health effects. Additionally, with multi-appointment scheduling, hospitals have the opportunity to augment patient satisfaction, allowing the patient to visit the hospital less frequently. To structure the growing body of literature in this field and aid researchers in the field, a classification scheme is proposed and used to classify the scientific work on multi-appointment scheduling in hospitals published before the end of 2017. The results show that multi-appointment scheduling problems are becoming increasingly popular. In fact, multi-appointment scheduling problems in hospitals are currently gaining progressively more momentum in the academic literature. (Marynissen, J. 2019)

#### *Various Algorithm used to create Scheduling Systems*

Activities such as rearranging and reshuffling classrooms could take more time and complexity. Scheduling the timetable is an important action that may be done at any academic institution to ensure that the lesson's syllabus is completed on time or that the scheduling project or assignment is completed within a certain time frame, and it helps easily to do the activities without a waste of time and most importantly, to finish tasks on time. A new algorithm was designed to solve the university's time problem and it has a heuristic approach to timing optimization to boost the efficiency in preparing the classroom. (Izah R. Ahmad et al. 2018).

Traditionally, medical appointments have been made with schedulers over the telephone or in person. These methods are based on verbal communications with real people and allow for maximum flexibility in complicated situations However, because these traditional methods require the intervention of schedulers, the ability to get a timely appointment is not only limited by the availability of appointment slots, but also by the schedulers and phone lines. Patients’ satisfaction with appointment booking is influenced by their ability to book at the right time with the right health service providers. (Zhao, P., 2017)

Initially the idea of the Brilliant Scheduler project was derived as a result of a critical challenge faced by the zAdministration of Educational Media at KAU. The idea is now capable of being used as an automated Web-based scheduling system. The main focus of the Brilliant Scheduler system is to work with the common standards for the scheduling system with decreased time and

effort. This system is segmented by allotting each registered course with a classroom and a supervisor. Moreover, it was concluded that the establishment of Brilliant Scheduler system was not an easy task. The current un-automated scheduling process of video-conferencing is complicated, which requires time and effort. This study proposed such techniques that would allow better understanding of the scheduling process. Therefore, it introduced a developed and comprehensive solution, which has focused on the automated course scheduling process. A unified e-portal was presented by the Brilliant Scheduler that facilitates the users to deliver the requests for the reservation of classrooms required for the video-conferencing course. This system has the ability to provide instant feedback about the reservation by checking the availability automatically. Furthermore, the system can also appoint a supervisor for each reserved course. (Meccawy, M. 2018)

The server is the central component of the system. Its main function is to mediate communication between the Conecte SUS Cidadão app and PEC e-SUS APS. The server is also responsible for storing appointment scheduling data and service configurations, ensuring that messages are only exchanged between the Conecte SUS Cidadão app and enabled PEC e-SUS APS installs, thus guaranteeing secure data transmission and storage. (Celuppi 2021)

#### *Implementation of Scheduling System*

Class scheduling is a significant difficulty for classrooms, students, and staff who are responsible for planning the school's class schedules. With so many resources to manage, such as teachers, students, subjects, courses, and schedules, the challenge is becoming increasingly complex and to solve this, a classroom scheduling software can help reduce the problem (Tuaycharoen, N. et al. 2018)

#### *Efficiency of Class Scheduling*

Class schedulers significantly assist instructors in decreasing their workloads so that they can get to know their students and classes on managing their own schedules. The administrator can utilize the class scheduling system in the most basic method to determine campus data, and the user can establish and manage relational tables such as a class table, instructor table, and class table. (Abdullah 2019)

Scheduling classes is one of the reasons that the school really needs planning before enrollment plays a crucial role in providing students with an understanding of how to handle their time. As time goes by, numerous problems occur to schools and colleges’ scheduling processes, such as time spent finding solutions to time conflicts. Some of the assignments for teachers, students, and

room schedules take several days or even weeks to prepare, tendency, the planned schedules have not been followed. Because of the faculty's time frame in

delivering the contents, this kind of issue has a substantial impact on the students' behavior to attend courses. (Labuanan, F. et al. 2019).

Activities such as rearranging and reshuffling classrooms could take more time and complexity. Scheduling the timetable is an important action that may be done at any academic institution to ensure that the lesson's syllabus is completed on time or that the scheduling project or assignment is completed within a certain time frame and it helps easily to do the activities without a waste of time and most importantly, to finish tasks on

time. A new algorithm was designed to solve the university's time problem and it has a heuristic approach to timing optimization to boost the efficiency in preparing the classroom. (Izah R. Ahmad et al. 2018)

A scheduling system is a much more reliable software system to produce the schedule straightforwardly. But some schools are using manual ways to process the schedule it takes the workforce for the arrangement and revision provided by the department heads. And it needs unity and a workforce to generate a robust process and solution. (Gemino, A. 2017)

The purpose of this article is to estimate the effect of class start time on a student’s grade. If class start time has an effect on student performance, this issue needs to be studied to find ways to improve students’ grades. This paper contributes to the existing literature on time of day and student achievement. No other studies have focused exclusively on microeconomics classes. Furthermore, the current literature is dated, and this paper updates this information about time of day and student achievement. I hypothesize that morning classes will display higher grades for both male and female students, as compared to their peers registered in later class times. Differences in the performance of students with different class times may arise because of student characteristics, class characteristics, and differences in the selection mechanisms for morning and afternoon classes. I examine the effect of student characteristics and account for the possibility that higher achieving students may generally register in early morning classes. (Aldaghir, M. 2017)

The results were viewed through the theory of constructivism, as it is used to advocate for forms of block scheduling to promote increased instructional techniques and student academic achievement. Although the schedules taken in totality not show an improved student academic performance based on the schedule under which instruction occurred, the individual course analysis reflected statistically significant differences in the content area of math. The findings of this research promote positive social change by adding to the understanding of the effectiveness of different schedules on student academic achievement. (Childers, E. 2018)

#### *Synthesis of the Reviewed Literature and Studies*

There are many insights that are said above of what is the importance of scheduling and what can be the use of scheduling. The schedule is very important for managing the operation, particularly when the activity has been carried out in a large organization and has been maintained for a long time or routine. Scheduling the timetable is an important activity that can also be done in any academic institution or non-academic institution. This is to ensure that the syllabus of the lesson can be completed on time, or that the scheduling project or task can be completed within a given time span.

Scheduling plays the role of time managing to prevent issues of conflicting schedules for everyone. Automated Class Scheduling System has been in human requirements since they thought of creating managing time effectively. It is commonly used in teaching schools, colleges, and other areas. Manual scheduling strategies take an immense amount of time, and numerous courses and classroom disputes typically occur. Many types of scheduling can be used in workplaces and schools. The insights about the scheduling presented above are some ideas of what a schedule can do to manage the circumstances. Scheduling not just by time but also the plans that need to be done at its specific time and place.

**Chapter 3**

**METHODOLOGY**

This chapter deals with the researchers’ method and technique in conducting the study, system development, design, and developed systems.

#### *Requirement Analysis*

The gathered data will be used as the basis for the design of the system. The client required the system to be able to export the data into an excel file or PDF. The system must be able to show the schedule depending on what is searched, room or professor. A list of rooms is given to be included in the system database.

#### *Process Model using Use Case Diagram*

**Use case stories/use case description.**

· *"As an Academic Head, I want to create a schedule for students and professors easily through automation."*

*Acceptance Criteria: Must be able to create class schedules for students and professors at PUP Parañaque automatically.*

|  |  |
| --- | --- |
| **Use cases** | **Users/Actors** |
| Login to system | Academic Head |
| Add rooms | Academic Head |
| Create scheduled time and date | Academic Head |
| Add courses | Academic Head |
| Add subject | Academic Head |

****

**Figure 1. Use case diagram**

**Figure 2. Activity diagram**

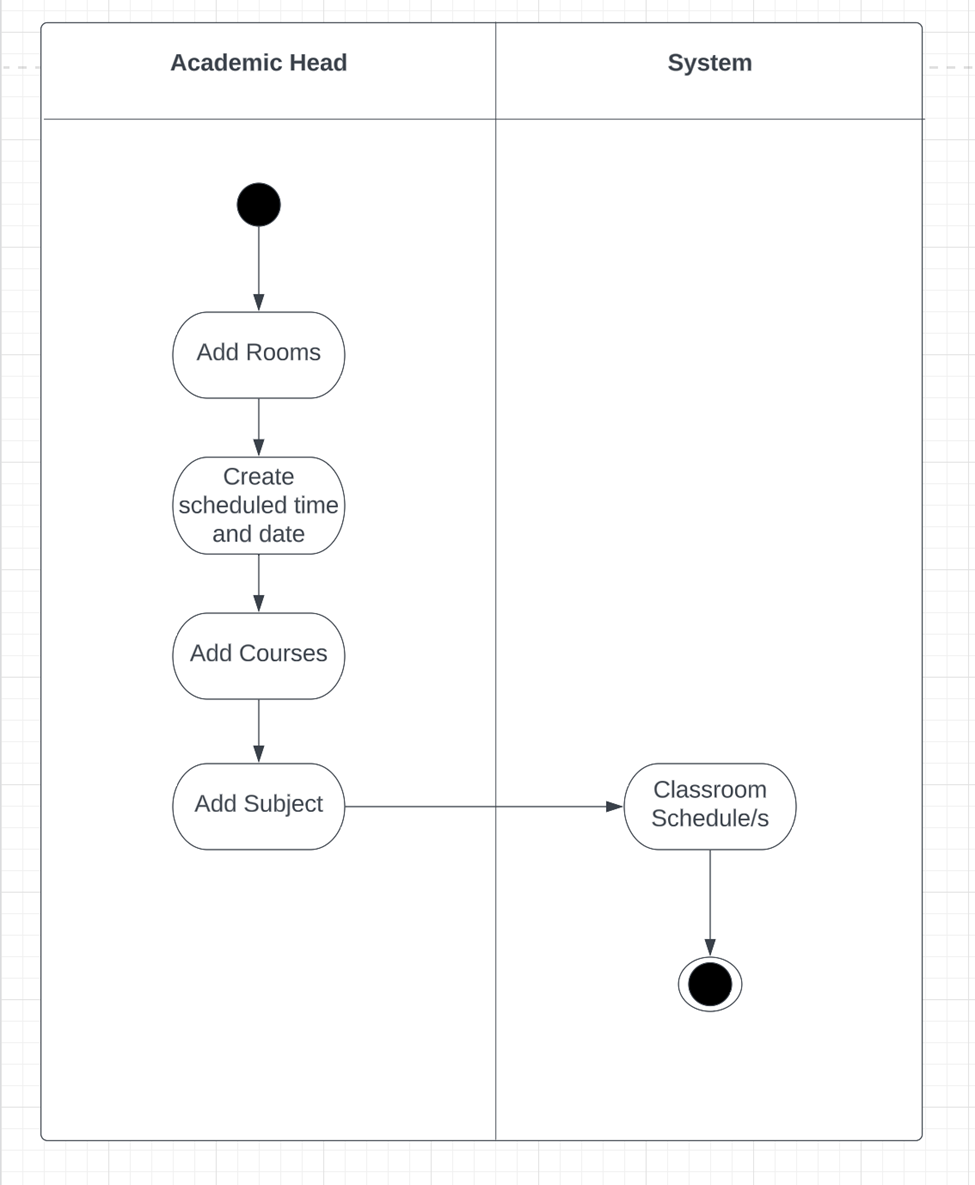
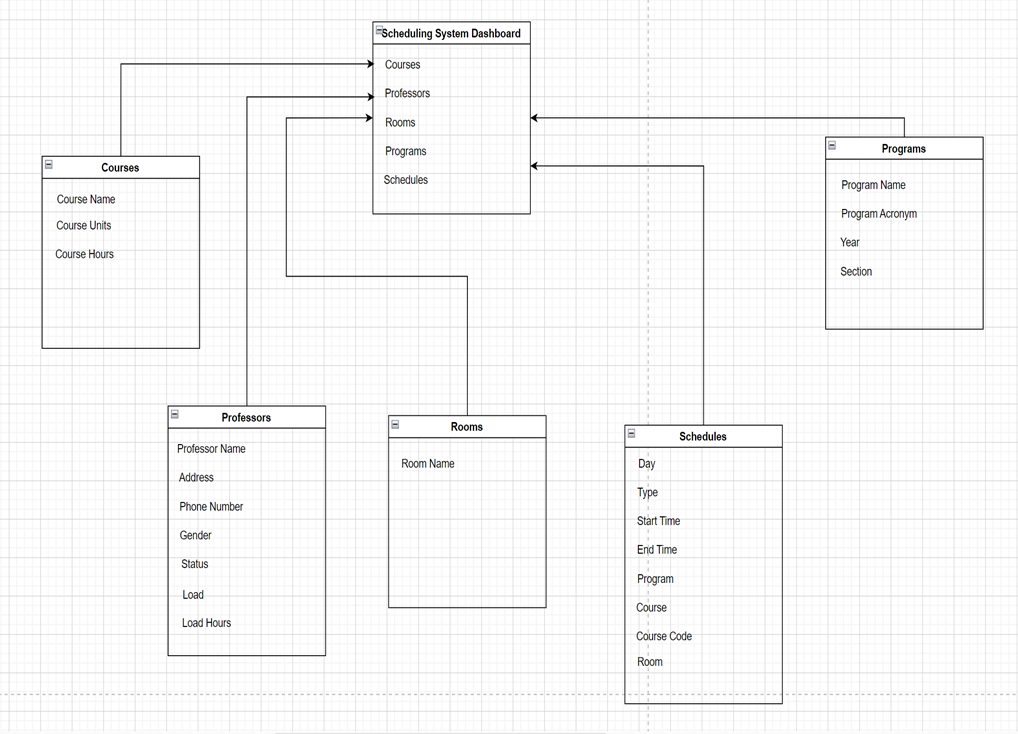
****

Figure 3. **Domain Model Class Diagram**



#### *Requirements Documentation*

**Account Module**

This module provides functionality for the academic head or administrator of the campus. The admin already has an existing account in the system it is not necessary to create another account, and the admin is the only user who will operate the system. However, the administrator can change the information of their account inside the system.

* Update password inside the system.
* Change role into user or administrator

#### *Information Module*

This module provides functionality for the administrator to fill up new information of professors, subjects and rooms inside the system It will allow the admin to manage the information of the database.

* Add/Update/Delete information of professors, rooms, subjects, academic year and courses to the scheduling system.

#### *Schedule Module*

This module provides functionality to the admin who will be able to operate the system to fill every field of the schedule table.

* Add/Update/Remove existing information on professors, rooms, subjects, academic year and courses from the database.
* Adjust time for assigning professor schedule in the schedule table.

#### *Software Development Life Cycle*

The researchers selected the Rapid Application Development Model for the development of the Classroom Scheduling System. It is a process model in the System Development Life Cycle (SDLC) that consists of a succession of short, iterative development cycles that result in completing the software. Its process is based on prototyping without any prior preparation, the project progress is less monitored in planning and has more priority to the development tasks.

The following are the phases involved in the development of the Classroom Scheduling System as shown in Figure 4.

Figure 4: RAD Model of Classroom Scheduling System

Diagram

Description automatically generated

#### *Requirements Planning*

In this phase, the researchers will interview the target beneficiary of the system. In conducting a research interview, then followed by gathering data to identify the desired and needed features in the proposed system. After gathering data, the researchers will work on the paper and design the proposed system

#### *User Design*

This phase defines that the requirements obtained and analyzed are ready to be designed in terms of what programming language to use, designing the system’s interface, and its flow of operation.

#### *Prototype*

The researchers will code a part of the system and it will proceed to the test phase.

#### *Test*

In this phase, after designing the prototype of the system, the researchers will test what the system's flow would be so that they can simply manage and understand how to utilize it. If there are any bugs and errors in this process, the researchers will quickly detect them.

#### *Refine*

In the refining process, researchers will fix the bug in the current state of the system and then it will be rebuilt and tested again.

#### *Testing*

After the cycle is complete in the user design phase, the system is proven effective and efficient, unlike the current system used. This is when the system is implemented and installed in the campus office.

#### *Implementation*

During the implementation, errors or defects may exist, which would require repairs during additional testing of the software. The system will be given to the client and the processes will be evaluated. In this phase the effort is also required to be implemented, resolving the identified problems, and also planning for sustaining the system. Monitoring the performance of the software is also included during this phase.

Figure 5: IPO

Diagram

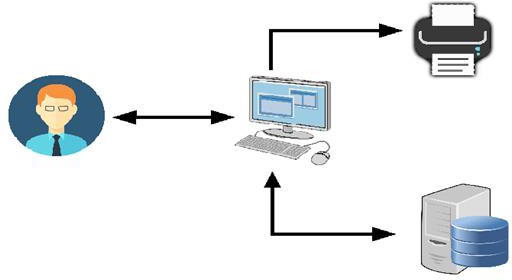
Description automatically generated

#### *Design of Software, Systems, Product and Processes*

The classroom scheduling system can be accessed or used by the account given to the admin. The user can access the system by using the desktop connected to the server by LAN to view and update the schedule information.

This figure shows the system layout of the proposed system. The one who can access this system is the administrator.

Figure 6: System Architecture



System

Administrator

Output Database

#### *Design of Software*

The user interface of the Classroom Scheduling shows the design of each window starting from the login interface where the admin can type the password, then the hamburger menu where the admin can navigate the various functions of the system. The other interfaces like professor, course etc. windows have the same designs, and it displays a table where they can manage the information of professors, subject, room, and course window.

#### *Development and Testing*

The development of the system involves a series of production activities. There is a chance of errors occurring at any stage. Because of the human inability to perform and communicate with perfection, testing the faulty modules system needs to be done. To test the system, the researchers are compiling the codes into a .exe file to run and check if the system is running as intended.

The researchers use ISO 25010 as the basis for their questionnaire for the evaluation of their system, they've also selected the following quality characteristics from the product quality model mentioned above: functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability and portability.

For testing the system, the researchers will follow the strategies given below.

#### *Unit Testing*

This will test the individual units or components of a software. The purpose is to validate that each unit of the software code performs as expected. This test isolates a section of code and verifies its correctness.

#### *Integration Testing*

Integration testing guarantees that the software and modules are working together.

This tests all of the module's interfaces to ensure that they work properly if integrated.

#### *System Testing*

This will test the system that validates the complete and fully integrated software product. It aims to satisfy the user when the system meets all requirements of the client’s specifications.

#### *Acceptance Testing*

The end-user or the client will perform this testing to verify/accept the software system before moving the software application to the production environment.

#### *Implementation Plan*

The developed system will be installed at the PUP Paranaque Campus after the researcher’s system is completed and tested with different strategies. After presenting and the client accepted the system, the researchers will provide continued support to the system. The researchers will train and teach the client how to use the system and its different functions.

**Chapter 4**

### PRESENTATION, ANALYSIS AND INTERPRETATION OF DATA

This chapter will discuss the analysis of the system, its interfaces and the evaluation of the respondents who participated to answer the survey for the developed system.

#### User Interface

Figure 7: Login Interface

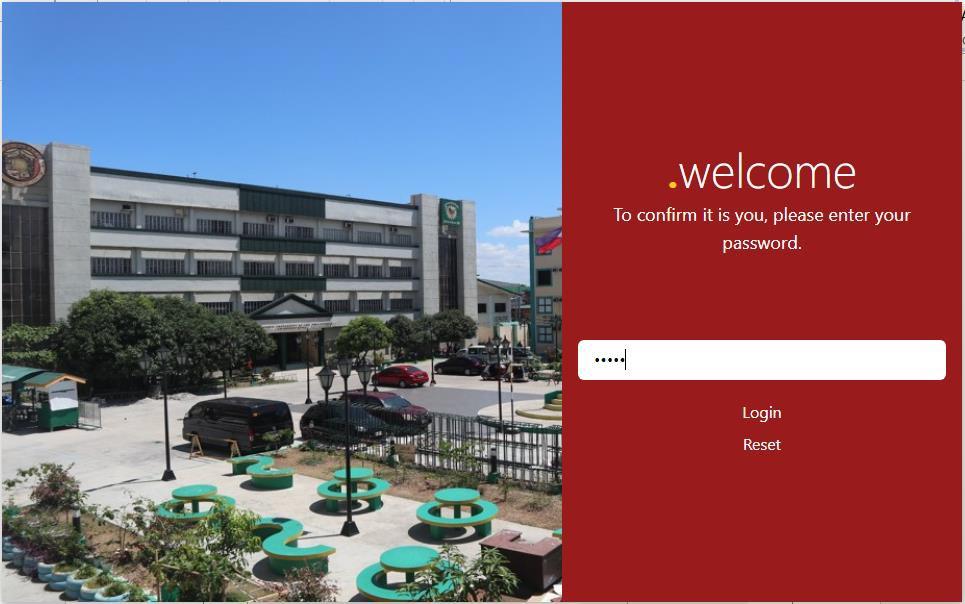
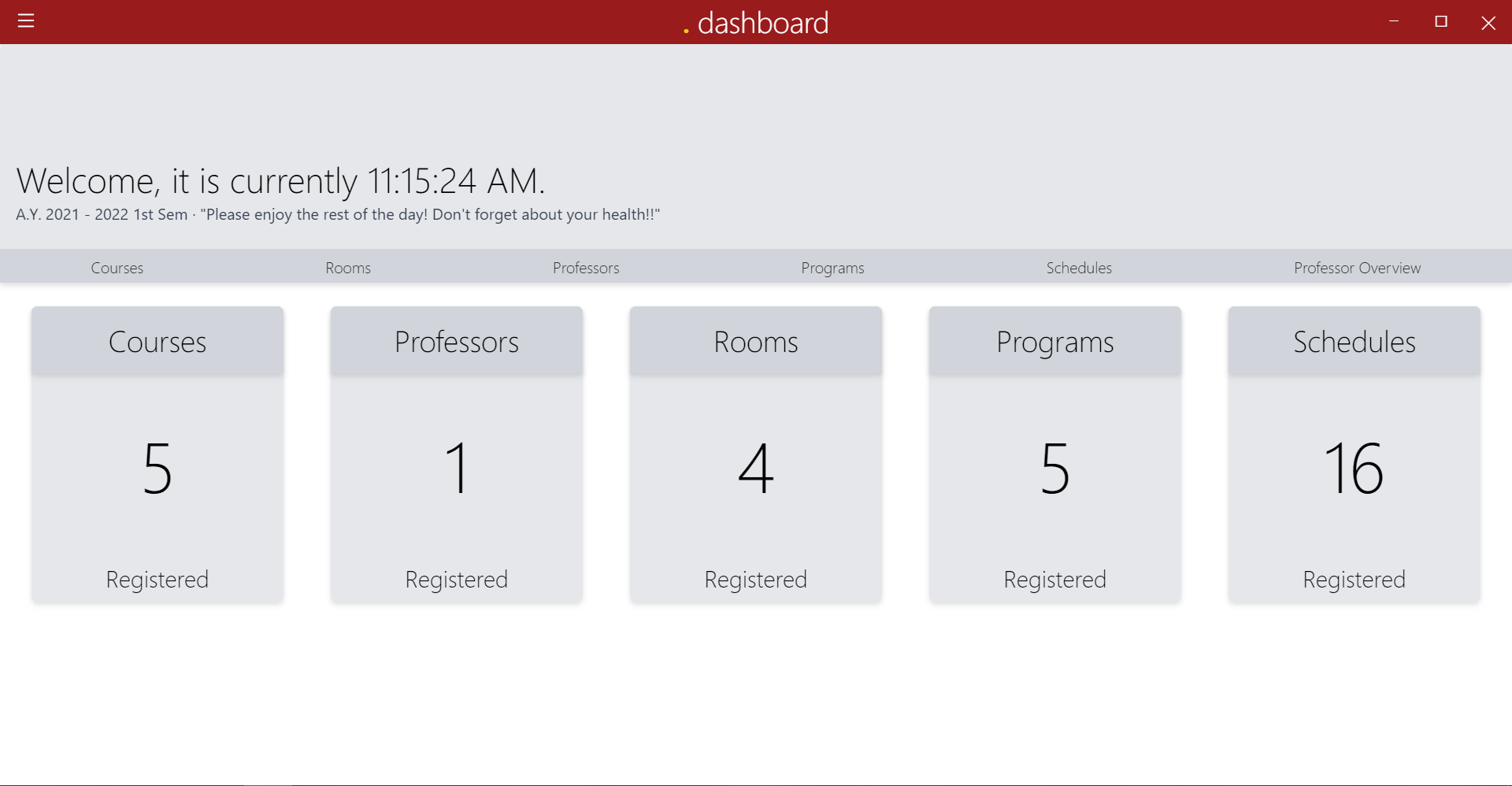


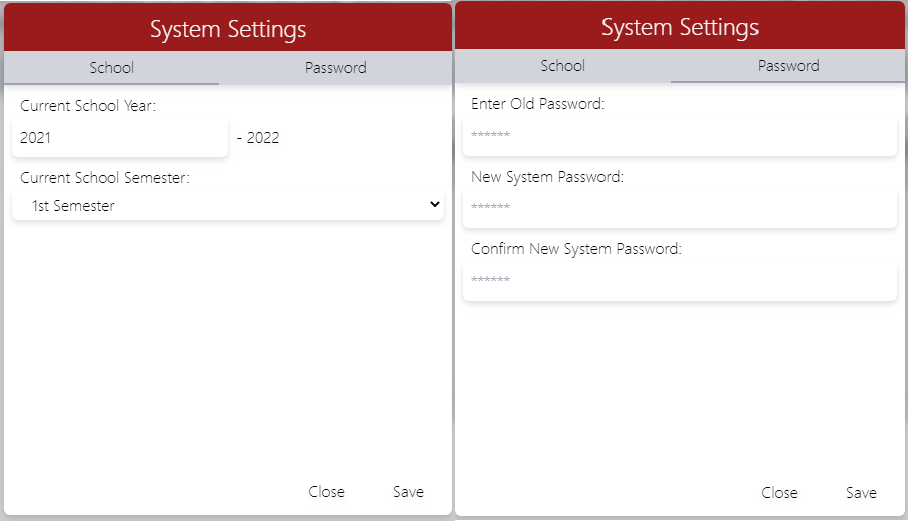
Figure 7, shows the login interface in which the user will need to input the default password in order for the user to access and use the system. The user can also reset the password if the user can’t remember its password.

Figure 7.1 – Dashboard Interface

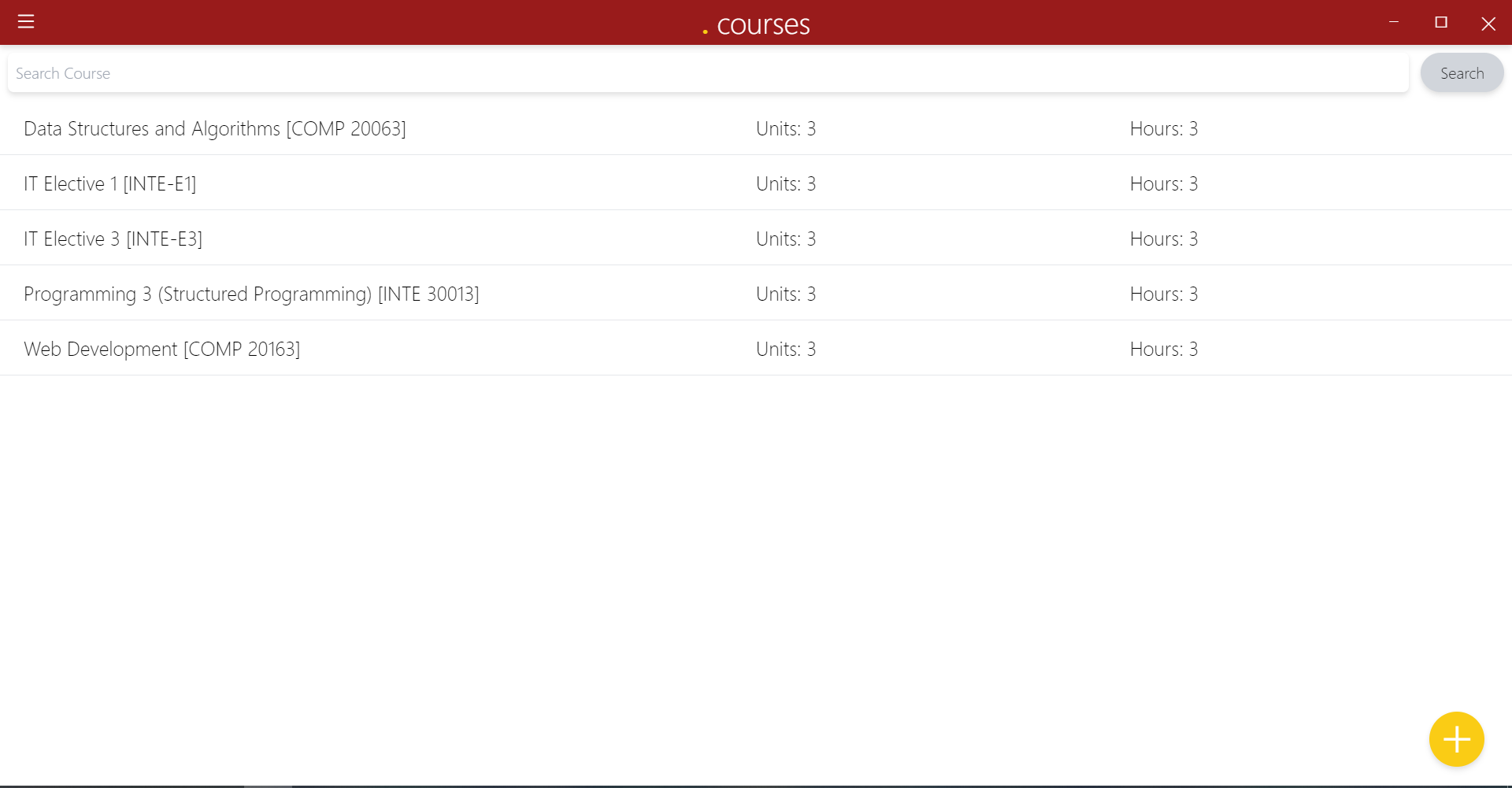


In Figure 7.1, the Dashboard Interface is the first window that will be shown to the user, and in this window, it will show the summary report of how many rooms, courses, professors and programs are inside the system. The user can also navigate fast to the room, courses, professor or programs window. It also shows the current semester and academic year and the current time of the day.

Figure 7.2 – System Settings Interface

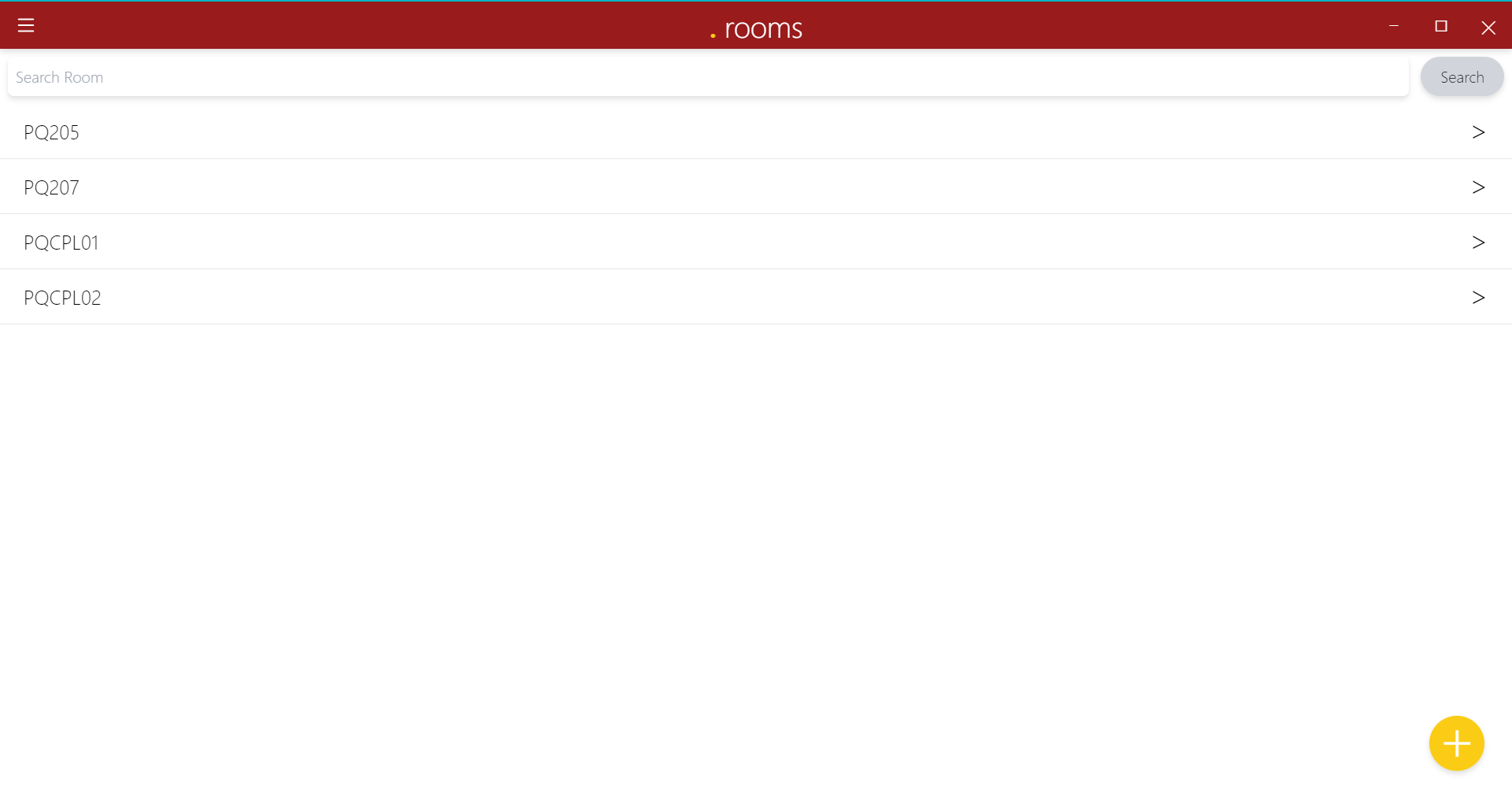


In Figure 7.2, this is the interface where the user can change the current semester and academic year, and the user can also change their password.

Figure 7.3 – Course Window

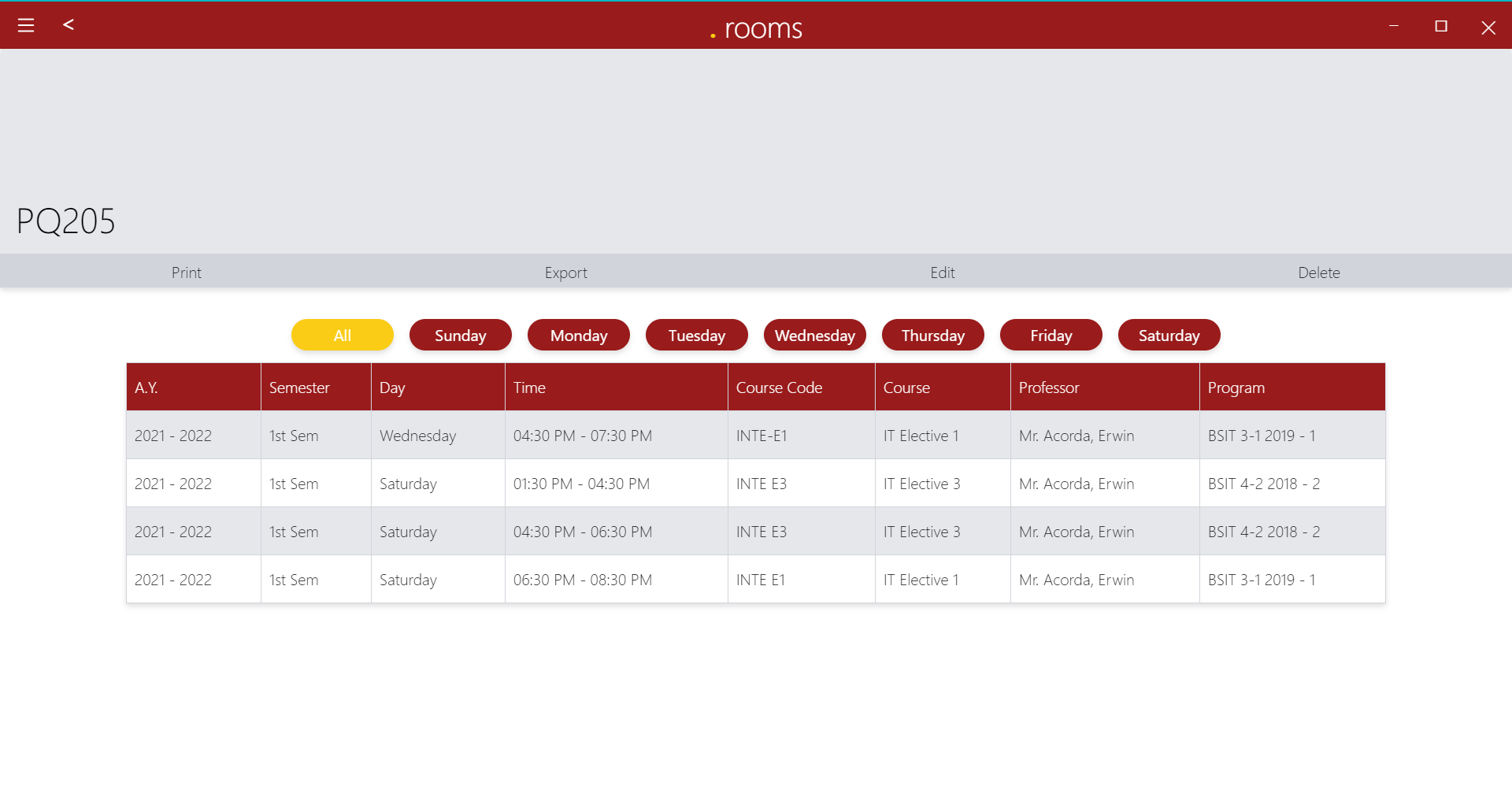
In Figure 7.3, the course window shows the name of the subject and the user can also add new information so that it can be saved in the database and used for making a schedule.

Figure 7.4 – Room Window

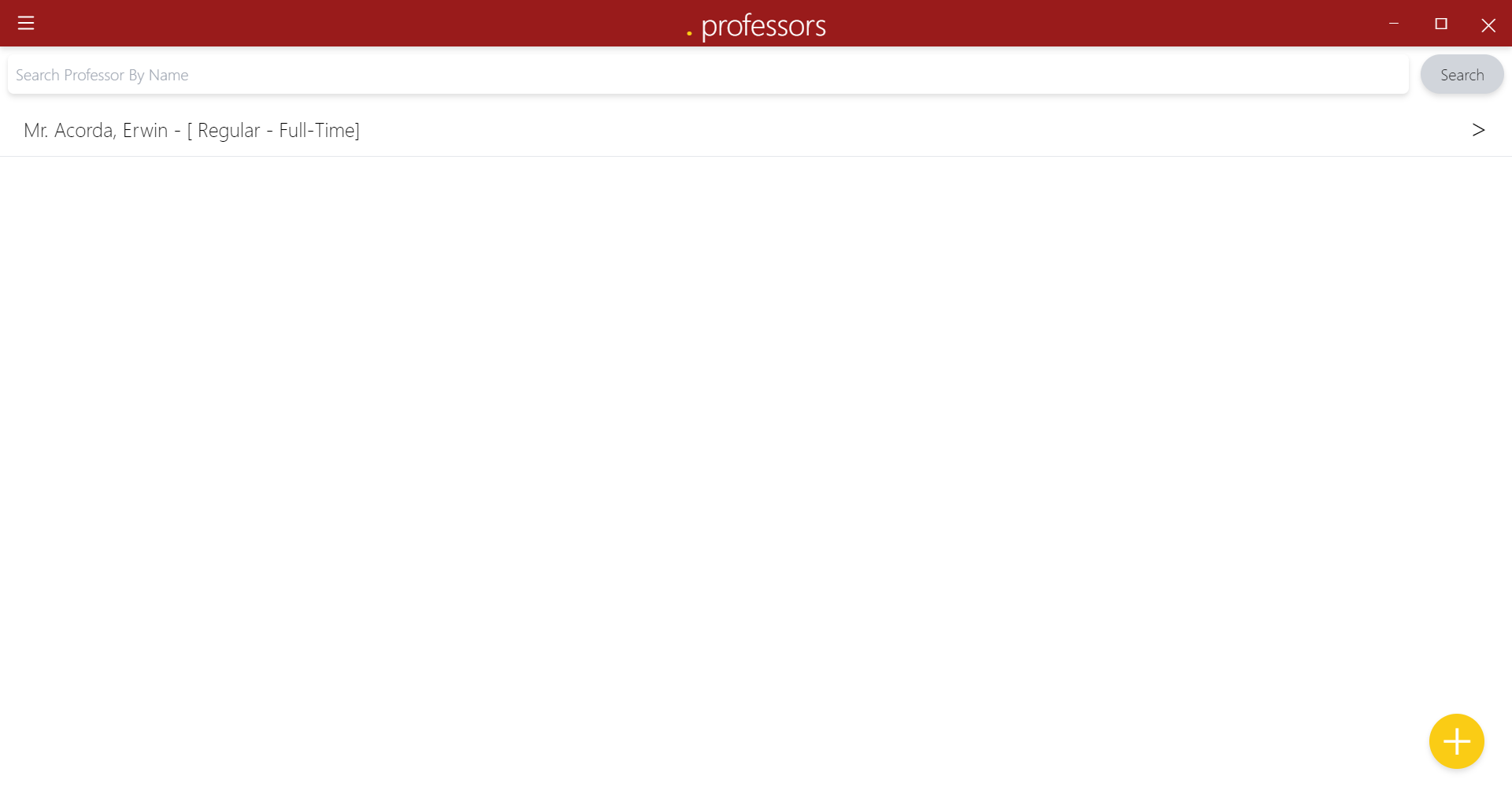


In Figure 7.4, the user can manage rooms so that the user can use it for assigning schedules.

Figure 7.4.1 – Room Window

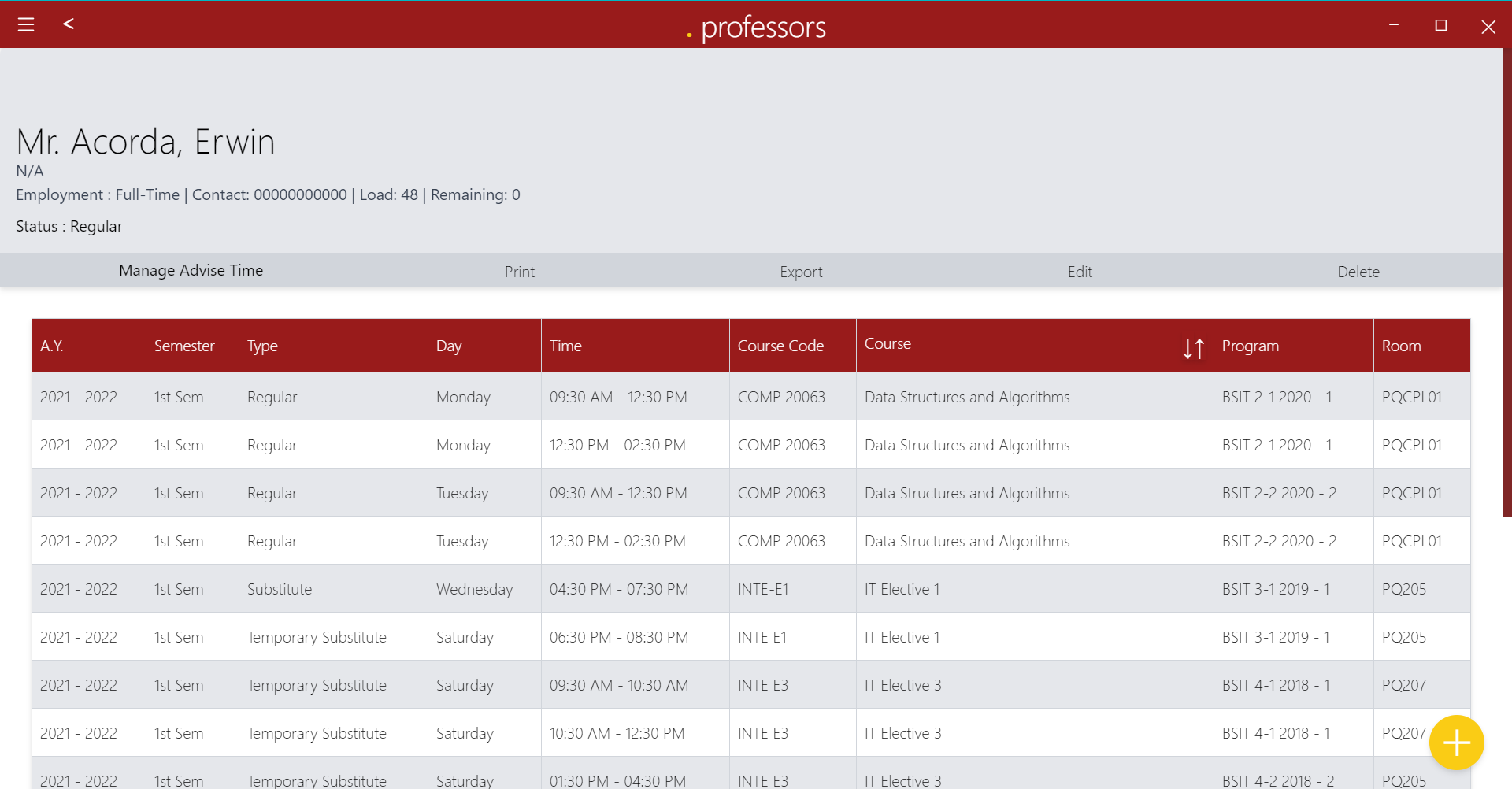


In Figure 7.4.1, the user will proceed to this if the user clicks a room in the room window, and it will show the data that are assigned from the Professor Window. The user can also pick a day and print it to PDF or export it into Excel and the user can also delete the data.

Figure 7.5 – Professor Window

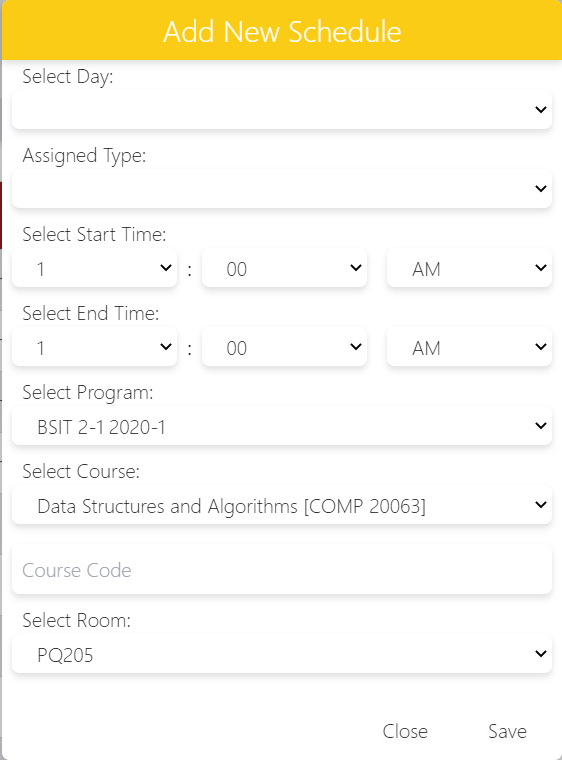
In Figure 7.5, the user can add information for the professor that he/she desires and the user needs to put the name of a professor, its address and his employment status. The user can also put the allotted time of the professor in order for the user to have limits on assigning a schedule to a professor.

Figure 7.5.1 – Professor Window



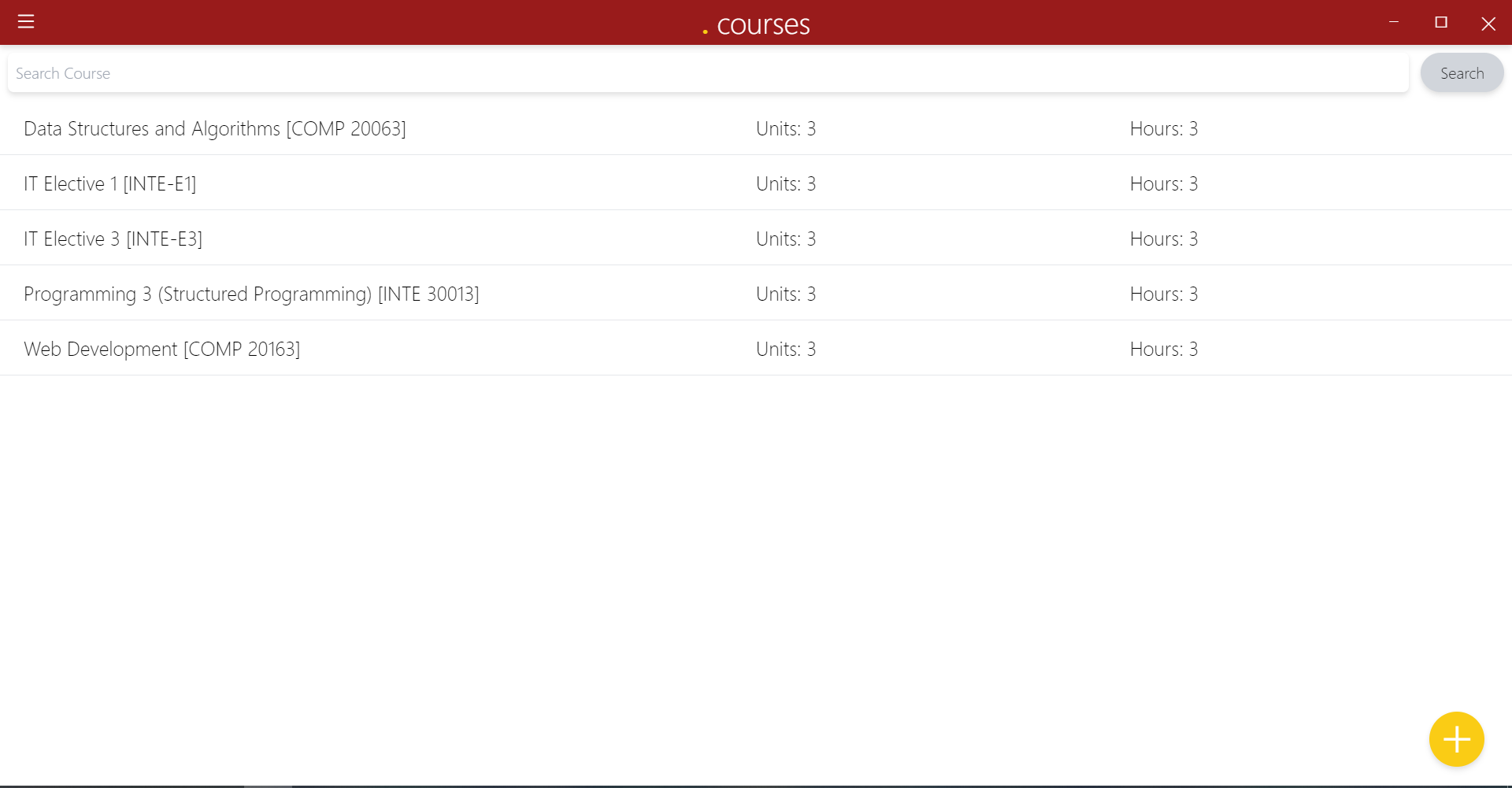
In Figure 7.5.1, the user will be directed to this window after clicking the name of the professor in the previous window. The user can add information by pressing the add button in the lower right part of the screen. It also shows the information of the professor that you recently added from the professor window.

Figure 7.5.1.1 – Add New Schedule Interface



In figure 7.5.1.1, the user will see this tiny window after pressing the add button. In this small window, the user will enter all the information that he/she added from the course, room and programs window and that information will automatically be added to the professor window in order for them to be used when the user is going to assign a schedule for the professor.

Figure 7.6 – Subject Window



In Figure 7.6, the user can manage a subject in this window by pressing the add button. The user can type the name of the program, its acronym and the year and section of the program.

Figure 7.6.1 – Subject Window



In Figure 7.6.1, in this window, the user will be directed in this window from the program window. The user can also print, export and edit the details of the assigned schedule that is set from the professor window.

#### Statistical Treatment of Data

Likert Scale was used to interpret items in the questionnaire. Likert Scale is a psychometric response scale primarily used in questionnaires to obtain participants’ preferences or degree of agreement with a statement or set of statements. Respondents are asked to indicate their level of agreement with a given statement by way of an ordinal scale. The range and interpretation of the four-point Likert Scale are shown in the table below:

3.26 – 4.00 - Strongly Agree

2.51 – 3.25 - Agree

1.76 – 2.50 - Disagree

1.00 – 1.75 - Strongly Disagree

Table 1

#### Respondents Perception to Functional Suitability

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **A. Functional Suitability** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Functional**  **completeness -** The system covers all the specified tasks and user objectives. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| **2. Functional correctness**  - The system provides the correct results with the needed degree of precision. | 8 | 3 | 0 | 0 | 11 | 3.73 | Strongly Agree |
| **3. Functional**  **appropriateness** - The system facilitates the accomplishment of specified tasks and objectives. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| Average Mean | | | | | | 3.61 | Strongly Agree |

In Functional Suitability table 4.1, the survey shows that in Functional Completeness 6 out of 11 respondents answered “Strongly Agree” while on the other hand 5 out of 11 respondents answered “Agree” with the mean of “3.55”. On the Functional Correctness, 8 out of 11 answered “Strongly Agree” and 3 out of 11 responded with

“Agree” with the result mean of “3.73”. While in the Functional Appropriateness, 6 of the respondents responded with “Strongly Agree” and 5 of the respondents answered with “Agree” and with the mean of “3.55”. The average mean on this table is “3.61”.

Table 2

#### Respondents Perception to Performance Efficiency

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **B. Performance Efficiency** | 4 | **3** | 2 | 1 | Total | Mean | Interpretation |
| **1. Time behavior** - The system’s response and processing times and throughput rates when performing its functions, meet requirements. | 5 | 4 | 2 | 0 | 11 | 3.27 | Strongly Agree |
| **2. Resource utilization** - The system’s amounts and types of resources used when performing its functions, meet requirements. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| **3. Capacity** - The system’s maximum limits of parameter meet requirements. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |

|  |  |  |
| --- | --- | --- |
| Average Mean | 3.48 | Strongly Agree |

In Performance Efficiency table 4shows that 11 respondents answer some of the questions under this table. In Time Behavior, 5 out of 11 respondents answered “Strongly Agree”, while 4 of it answers “Agree” and 2 of the responses answered “Disagree” and this has an average mean of “3.27”. On Resource Utilization, 6 respondents answered “Strongly Agree”. While 5 of the respondents answered “Agree” and with these it has a mean of “3.55”. While on the Capacity part 7 out of 11 respondents answered “Strongly Agree” and 4 of them answers “Agree”. The average mean of this table is “3.48”.

Table 3

#### Respondents Perception to Compatibility

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **C. Compatibility** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Co-existence** - The system can perform its required functions efficiently while sharing a common  environment and | 5 | 6 | 0 | 0 | 11 | 3.45 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| resources with other products, without detrimental impact on any other product |  |  |  |  |  |  |  |
| **2. Interoperability** - The system can exchange information and use the information that has been exchanged. | 6 | 4 | 1 | 0 | 11 | 3.45 | Strongly Agree |
| Average Mean | | | | | | 3.45 | Strongly Agree |

In Compatibility table 4.3, the survey shows that 11 answers the following question. In the Co-existence part, 5 of them gives an answer of “Strongly Agree” and 6 answered “Agree” with a mean of “3.45”. In the Interoperability, 4 of the respondents answered “Agree” while on the other hand, 1 respondent give the answer “Disagree” with a mean of “3.45”. All the collected data result in an average mean of “3.45”.

Table 4

#### Respondents Perception to Usability

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **D. Usability** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1.No**  **Appropriateness recognizability** - The system allows users to recognize if it is appropriate for their needs. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| **2. Learnability** - The system can be used by specified users to achieve specified goals of learning to use the application with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |
| **3. Operability** - The system has attributes that make it easy to operate and control. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| **4. User error protection** - The system protects  users against making errors. | 6 | 3 | 2 | 0 | 11 | 3.36 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **5. User interface aesthetics** - The system’s user interface enables pleasing and satisfying interaction for the user. | 5 | 6 | 0 | 0 | 11 | 3.45 | Strongly Agree |
| **6. Accessibility** - The system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of the use | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |
| Average Mean | | | | | | 3.53 | Strongly Agree |

The Usability table 4.4, it shows the data collection gathered from the respondents. In the first part of The No Appropriateness Recognizability 6 out of 11 respondents gave an answer of “Strongly Agree” while 5 of them answers “Agree” and with these, it has a mean of “3.55”. On the second part of this table which is Learnability, 7 of the respondents answered “Strongly Agree” and 4 out of 11 responses an answer of “Agree” with a mean of “3.64”. In the Operability part of this table, 6 out of 11 respondents answered “Strongly Agree and 5 out of it answered “Agree” with a mean of “3.55”. Next is User Error Protection, 6 out of the 11 respondents give an answer of “Strongly Agree” and 3 out of this answered “Agree” and 2 of the respondents answers “Disagree” with a mean of “3.36”. In User Interface Aesthetics, 5 respondents answers “Strongly Agree” and 6 out of the 11 respondents answer “Agree” with a mean of “3.45”. In Accessibility, 7 of the respondents

answers “Strongly Agree” and 4 out of 11 respondents answers “Agree” with the mean of “3.64” and with all the gathered data it gives an average mean of “3.53”.

Table 5

#### Respondents Perception to Reliability

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **E. Reliability** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Maturity** - Degree to which a system, product or component meets needs for reliability under normal operation. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |
| **2. Availability** - Degree to which a system, product or component is  operational and accessible when required for use. | 4 | 7 | 0 | 0 | 11 | 3.36 | Strongly Agree |
| **3. Fault**  **Tolerance** - Degree to which a system, product or component operates as intended despite the presence of hardware or software faults. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **4. Recoverability**  - Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system. | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |
| Average Mean | | | | | | 3.55 | Strongly Agree |

In Reliability table 4.5, 11 respondents answer the questions under this table. On the first part, which is Maturity, 7 out of 11 respondents answered “Strongly Agree,” and the remaining 4 responded with “Agree,” with a mean of “3.64”. In the Availability part, 4 out of 11 respondents answered “Strongly Agree,” and the other 7 respondents answered “Agree” with a mean of “3.36”. In Fault Tolerance, 7 out of 11 respondents answered “Strongly Agree” while the other respondents gave the answer of 4 “Agree,” with the result mean of “3.64”. On the Recoverability, 6 “Strongly Agree” answers are given by the respondents, and 5 out of them give an answer of “Agree” that results in a mean of “3.55”. And on this table gives an average mean of “3.55”.

Table 6

#### Respondents Perception to Security

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **F. Security** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Confidentiality**  - The system ensures that data are accessible only to those authorized to have access. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |
| **2. Integrity** - The system prevents  unauthorized access to, or modification of, computer programs or data. | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |
| **3. Non-**  **repudiation** - The | 7 | 4 | 0 | 0 | 11 | 3.64 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| system can be proven to have taken place, so that the events or actions cannot be repudiated later. |  |  |  |  |  |  |  |
| Average Mean | | | | | | 3.64 | Strongly Agree |

In Security table 4.6, it shows that there are 11 respondents are willingly answered different survey questions. In the Confidentiality part, there are 7 respondents who answered “Strongly Agree” while 4 of the respondents answered “Agree,” so it results in a mean of “3.64”. In the Integrity part of this table, 7 out of the 11 respondents answered “Strongly Agree,” and the 4 of them gave an answer of “Agree” with a calculated mean of “3.64”. While in the Non-Repudiation, 7 out of the 11 respondents gave an answer of “Strongly Agree” while the remaining 4 respondents answered “Agree” and with a result mean of “3.64”. In this part of the table, the calculated average mean for this is “3.64”.

Table 7

#### Respondents Perception to Maintainability

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **G.**  **Maintainability** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Modularity** - Degree to which a system or computer program is  composed of discrete components such that a change to one component has minimal impact on other components. | 8 | 3 | 0 | 0 | 11 | 3.73 | Strongly Agree |
| **2. Reusability** -  Degree to which an asset can be | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| used in more than one system, or in building other assets. |  |  |  |  |  |  |  |
| **3. Analyzability** - Degree of effectiveness and efficiency with which it is  possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a  product for  deficiencies or causes of failures, or to identify parts to be modified. | 8 | 3 | 0 | 0 | 11 | 3.73 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **4. Modifiability** - Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality. | 8 | 2 | 1 | 0 | 11 | 3.64 | Strongly Agree |
| **5. Testability** - Degree of effectiveness and efficiency with which test criteria can be  established for a system, product or component and tests can be performed to determine  whether those | 8 | 3 | 0 | 0 | 11 | 3.73 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| criteria have been met. |  |  |  |  |  |  |  |
| Average Mean | | | | | | 3.67 | Strongly Agree |

In the maintainability table 4.7, there are 11 respondents that answer a different questions on the survey. In the Modularity part of this table, 8 out of 11 respondents give the answer of “Strongly Agree” while 3 of them answer “Agree” and with a mean of “3.73”. The next one is Reusability, there are 6 respondents who answer “Strongly Agree” then there are 5 respondents who give an answer of “Agree” with a result mean of “3.55”. The next part under this Maintainability table is Analyzability, where there are 8 respondents who answer “Strongly Agree” and the remaining 3 respondents answered “Agree” and a result means of “3.73”. Next is Modifiability, wherein 8 of the respondents gives an answer of “Strongly Agree”, 2 out of the 11 answered “Agree” and 1 respondent with an answer of “Disagree” and with a mean of “3.64”. The last one is Testability, there are 8 out of 11 respondents who answered “Strongly Agree” and 3 out of the 11 answered “Agree” with a calculated mean of “3.73”. and for this table, there is an average mean of “3.67” which is the result of the given answer provided by the respondents.

Table 8

#### Respondents Perception to Portability

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **H. Portability** | 4 | 3 | 2 | 1 | Total | Mean | Interpretation |
| **1. Adaptability** - Degree to which a product or system can effectively and efficiently be adapted for  different or evolving hardware, software or other operational or usage environments. | 4 | 7 | 0 | 0 | 11 | 3.36 | Strongly Agree |
| **2. Installability** - Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or  uninstalled in a | 6 | 5 | 0 | 0 | 11 | 3.55 | Strongly Agree |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| specified environment. |  |  |  |  |  |  |  |
| **3. Replaceability**  - Degree to which a product can replace another specified software product for the same purpose in the same environment. | 7 | 2 | 2 | 0 | 11 | 3.45 | Strongly Agree |
| Average Mean | | | | | | 3.45 | Strongly Agree |

The last table is Portability, wherein in this part respondents are given some survey questions to collect data from them. On the Adaptability part, there are 4 respondents who answers “Strongly Agree” and 7 out of the 11 answered “Agree” and the result means is “3.36”. The next one is Installability, wherein 6 out of the 11 respondents answered “Strongly Agree” and 5 of the respondents give an answer of “Agree”’ with a result mean of “3.55”. Next is Replaceability, that there are 7 out of the 11 respondents answered “Strongly Agree” 2 of the respondents answered “Agree” and the other 2 of the respondents answered “Disagree” which results in a mean of “3.45”. and on this table, it has an average mean of “3.45” that is based on the collected data on this part.

Table 9

#### Summary of the overall data

|  |  |  |  |
| --- | --- | --- | --- |
| **A. Functional Suitability** | **Mean** | 3. Operability | 3.55 |
| 1. Functional completeness | 3.55 | 4. User error protection | 3.36 |
| 2. Functional correctness | 3.73 | 5. User interface aesthetics | 3.45 |
| 3.Functional appropriateness | 3.55 | 6. Accessibility | 3.64 |
| **B. Performance Efficiency** | **Mean** | **E. Reliability** | **Mean** |
| 1. Time behavior | 3.27 | 1. Maturity | 3.64 |
| 2. Resource utilization | 3.55 | 2. Availability | 3.36 |
| 3. Capacity | 3.64 | 3. Fault Tolerance | 3.64 |
| **C. Compatibility** | **Mean** | 4. Recoverability | 3.55 |
| 1. Co-existence | 3.45 | **F. Security** | **Mean** |
| 2. Interoperability | 3.45 | 1. Confidentiality | 3.64 |
| **D. Usability** | **Mean** | 2. Integrity | 3.64 |
| 1.No Appropriateness recognizability | 3.55 | 3. Non-repudiation | 3.64 |

|  |  |  |  |
| --- | --- | --- | --- |
| **G. Maintainability** | **Mean** | 2. Installability | 3.55 |
| 1. Modularity | 3.73 | 3. Replaceability | 3.45 |
| 2. Reusability | 3.55 | **Overall Mean =** | **3.56** |
| 3. Analyzability | 3.73 |  | |
| 4. Modifiability | 3.64 |
| 5. Testability | 3.73 |
| **H. Portability** | **Mean** |
| 1. Adaptability | 3.36 |

The overall data that have been gathered based on the survey, it results in an overall total mean of all parts of this survey so the total mean of all these tables is “3.56”.

**Chapter 5**

### SUMMARY OF FINDINGS, CONCLUSION, AND RECOMMENDATIONS

This chapter discusses the summary of findings, conclusion, and recommendation.

#### Summary of Findings

Upon conducting a survey and finding the results that are translated from the questionnaire that has been prepared by the researchers, the results stated:

Functionality:

1. 54.55% of the respondents answered “Strongly Agree” on the functional completeness of the system while on the other hand, 45.45% of the respondents answered, “Agree”.
2. 72.73% of the respondents answered “Strongly Agree” on the functional correctness of the system while on the other hand, 27.27% of the respondents answered, “Agree”.
3. 54.55% of the respondents answered “Strongly Agree” on the functional appropriateness of the system while on the other hand, 45.45% of the respondents answered, “Agree”.

Performance Efficiency:

1. 45.45% of the respondents answered “Strongly Agree” on the time behavior of the system and 36.36% of the respondents answered, “Agree”. 18.18% of the respondents disagreed with the system’s response and processing times.
2. 54.55% of the respondents answered “Strongly Agree” on the resource utilization of the system while on the other hand, 45.45% of the respondents answered, “Agree”.
3. 63.64% of the respondents answered “Strongly Agree” on the capacity of the system while on the other hand, 36.36% of the respondents answered, “Agree”.

Compatibility:

1. 45.45% of the respondents answered “Strongly Agree” on the c-existence of the system while on the other hand, 54.55% of the respondents answered, “Agree”.
2. 54.55% of the respondents answered “Strongly Agree” on the interoperability of the system while on the other hand, 36.36% of the

respondents answered, “Agree”. 9.09% of the respondents disagreed with the interoperability process of the system.

Usability:

1. 54.55% of the respondents answered “Strongly Agree” on the system that allows users to recognize if it is appropriate for their needs and 45.45% of the respondents answered, “Agree”.
2. 63.64% of the respondents answered “Strongly Agree” on the learnability of the system while on the other hand, 36.36% of the respondents answered, “Agree”.
3. 54.55% of the respondents answered “Strongly Agree” on the operability of the system while on the other hand, 45.45% of the respondents answered, “Agree”.
4. 54.55% of the respondents answered “Strongly Agree” on the user error protection of the system while the 27.27% of the respondents answered, “Agree”. 18.18% of the respondents disagreed with the system’s ability to protect users against making errors.
5. 45.45% of the respondents answered “Strongly Agree” on the user interface aesthetics of the system while the 54.55% of the respondents answered, “Agree”.
6. 63.64% of the respondents answered “Strongly Agree” on the accessibility of the system while on the other hand, 36.36% of the respondents answered, “Agree”.

Reliability:

1. 63.64% of the respondents answered “Strongly Agree” on the maturity of the system while on the other hand, 36.36% of the respondents answered, “Agree”.
2. 36.36% of the respondents answered “Strongly Agree” on the availability of the system while on the other hand, 63.64% of the respondents answered, “Agree”.
3. 63.64% of the respondents answered “Strongly Agree” on the fault tolerance of the system while 36.36% of the respondents answered, “Agree”.
4. 54.55% of the respondents answered “Strongly Agree” on the recoverability of the system while the 45.45% of the respondents answered, “Agree”.

Security:

1. 63.64% of the respondents answered “Strongly Agree” on the confidentiality of the system while 36.36% of the respondents answered, “Agree”.
2. 63.64% of the respondents answered “Strongly Agree” on the integrity of the system while on the other hand, 36.36% of the respondents answered, “Agree”.
3. 63.64% of the respondents answered “Strongly Agree” on the non- repudiation of the system while on the other hand, 36.36% of the respondents answered, “Agree”.

Maintainability:

1. 72.73% of the respondents answered “Strongly Agree” on the modularity of the system while on the other hand, 27.27% of the respondents answered, “Agree”.
2. 54.55% of the respondents answered “Strongly Agree” on the reusability of the system while 45.45% of the respondents answered, “Agree”.
3. 72.73% of the respondents answered “Strongly Agree” on the analyzability of the system while 18.18% of the respondents answered, “Agree”.
4. 72.73% of the respondents answered “Strongly Agree” on the modifiability of the system while 27.27% of the respondents answered, “Agree”. 9.09% of the respondents disagreed with the system’s ability to be effectively and efficiently modified without introducing defects.
5. 72.73% of the respondents answered “Strongly Agree” on the testability of the system while on the other hand 27.27% of the respondents answered “Agree”

Portability:

1. 36.36% of the respondents answered “Strongly Agree” on the adaptability of the system while on the other hand, 72.73% of the respondents answered, “Agree”.
2. 54.55% of the respondents answered “Strongly Agree” on the installability of the system while on the other hand, 45.45% of the respondents answered, “Agree”.
3. 63.64% of the respondents answered “Strongly Agree” on the replaceability of the system while 18.18% of the respondents answered, “Agree”. 18.18% of the respondents disagreed with the system’s ability to which a product can replace another specified software product for the same purpose in the same environment.

#### Conclusion

Based on the findings of the study, the following conclusions were drawn:

The Academic Head encountered different problems in doing the manual way of creating class schedules. The problems that the university has encountered were as follows: (a) Students are having problems with their designated rooms, (b) Unavailability

of the classroom due to error scheduling of classes of students per course, (c) Students who are loitering in a vacant room and ended up breaking the chairs or table due to misuse, (d) Students are occupying wrong rooms or just loitering.

The researchers used the descriptive method for researching and determining the problems of the existing system. The data instrument used was an interview, the researchers used a direct verbal interaction with the respondent (Academic Head) to determine the main problems of the manual procedure of the scheduling system inside the campus.

To integrate the innovations of the developed system, these are the modules: (a) Login, (b) Professor, (c) Room, (d) Subject, (e) Course, and (f) Settings. The said modules were converted into a hamburger menu and buttons that were integrated into the system. Each button in the system has different functions according to the intended purpose.

The process used in developing the software was a rapid application development model that consists of a succession of short, iterative development cycles that result in completing the software through the phases of the following: (a) requirements planning,

(b) user design, (c) testing, and (d) implementation. The project progress is less monitored in planning and has more priority to the development tasks.

The evaluation of the developed system was evaluated by the respondents (End User and IT experts) in terms of the ISO Standard 25010 as follows: functionality, efficiency, compatibility, usability, reliability, security, maintainability, and portability. The overall summary assessment of the respondents to the developed system is evaluated as very acceptable with a weighted average of 3.56.

#### Recommendation/s

Based on the findings and conclusions of the study the following are hereby recommended for the improvement of the project:

1. To create accounts for professors to look into the created schedule in the system.
2. To provide additional features and modules in the scheduling system.
3. To add a search function for ease of determining vacant rooms or schedules.
4. To provide a “pick-up where you left off" function in case of a system crash or unexpected events during creating a schedule.
5. To improve the system’s responsiveness.
6. Notify the admin when there’s a conflict in schedule via

### REFERENCES

Abante, M. V., Abcede, K. J., Gonzaga, R. A., Ontoca, R. J. & Zapanta, J. N.. (2013). Class Scheduling System for URS – Binangonan Campus. Fatima University Research Journal, 5(1). Retrieved from https://[www.semanticscholar.org/paper/Class-Scheduling-](http://www.semanticscholar.org/paper/Class-Scheduling-) System-for-URS-%E2%80%93-Binangonan-Campus-Abante- Abcede/53098f002a4d7b1a20be5b7b497763a8d511c171#related-papers

Ahmad, I. R., Sufahani, S., Ali, M., & Razali, S. N. A. M. (2018, April 1). A Heuristics Approach for Classroom Scheduling Using Genetic Algorithm Technique. Journal of Physics: Conference Series. https://iopscience.iop.org/article/10.1088/1742- 6596/995/1/012050

Angela, Patricia & Gagno, M & Ii, Linda & Sarmiento, M & Kristel, Sheryl & Tesoro, Melchor & Batista, Theresa & Naval, Prospero. (2010). CSS: MULTI-OBJECTIVE COURSE SCHEDULING SYSTEM. Retrieved from https://[www.researchgate.net/publication/268056698\_CSS\_MULTI-](http://www.researchgate.net/publication/268056698_CSS_MULTI-) OBJECTIVE\_COURSE\_SCHEDULING\_SYSTEM

Bautista, J. R., Capistrano, Z. R., Francisco, A. A., & Reyes, A. C. (2020, March 20). Automated Class Scheduling System.docx - Automated Class Scheduling System A Design Project Presented to the Faculty of Institute of Computer Studies: Course Hero. Automated Class Scheduling System.docx - Automated Class Scheduling System A

Design Project Presented to the Faculty of Institute of Computer Studies | Course Hero. https://[www.coursehero.com/file/57620837/Automated-Class-Scheduling-Systemdocx/](http://www.coursehero.com/file/57620837/Automated-Class-Scheduling-Systemdocx/) Benjamin (2020, January 28). 7 Amazing Benefits Of School Management and Scheduling Software. Picktime. https://blog.picktime.com/benefits-of-school-management-software

Botangen, K. A. (2014). Web-Based Class Scheduling for a Collaborative Preparation of Block-Based Schedules. Research Gate. https://[www.researchgate.net/publication/306893805\_Web-](http://www.researchgate.net/publication/306893805_Web-) Based\_Class\_Scheduling\_for\_a\_Collaborative\_Preparation\_of\_Block- Based\_Schedules. DOI: DOI:10.7763/IPEDR.2014.V81.25

Botangen, Khavee Agustus & Khan, Concepcion. (2014). Class-Scheduling System for the Central Luzon State University(Abridged). https://[www.researchgate.net/publication/306894171\_Class-](http://www.researchgate.net/publication/306894171_Class-) Scheduling\_System\_for\_the\_Central\_Luzon\_State\_UniversityAbridged. DOI : 10.13140/RG.2.2.12729.57445

Branson, T. (2017, April 14). The 5 Best Reasons to Choose MySQL – and its 5 Biggest Challenges. Dataconomy. https://dataconomy.com/2017/04/5-reasons-challenges-mysql/

Brigoli, D. I., Ortega, E. P. & Ortega, M. D.. (2015). Online Class Scheduling and Faculty Loading System within a Decision Support Framework. JPAIR Institutional Research Journal, 6(1). Retrieved from https://ejournals.ph/article.php?id=12053. <http://dx.doi.org/10.7719/irj.v6il.363>

Budhi, G. S., Gunadi, K., & Wibowo, D. A. (2011). Genetic Algorithm for Scheduling Courses. Springer-Verlag Berlin Heidelberg. https://core.ac.uk/download/pdf/32452919.pdf

Capstoneguide. (2020, August 6). Automated Class Scheduling System Capstone Project Document. https://capstoneguide.com/automated-class-scheduling-system-capstone- project-document/

Corchado, J., & et al. (2020). WEB - BASED ACADEMIC CLASS SCHEDULING

SYSTEM. Course Hero. https://[www.coursehero.com/file/77255241/ACFrOgA0QX-](http://www.coursehero.com/file/77255241/ACFrOgA0QX-) KPDaiTiNdrSrRLEXhK0E5kSoNGEFQrkGlzmOZTBYsid60G- IlxrriYkytovazYej4PdWxmV3zr3qbcqFGy3GITBuQ/

De Francia, J. P. & Calimot, D. S.. (2013). Class Scheduling System for UIC-ITE using Genetic Algorithm. CLOUD, 1(1). Retrieved from https://ejournals.ph/issue.php?id=4

Donkor, G. (2014, April 1). A process for automated class scheduling at Ashesi. Ashesi Institutional Repository. https://air.ashesi.edu.gh/handle/20.500.11988/227

Enokela, J., & Mom, J. M. . (2012, May). Implementation of a time table generator using visual Retrieved

from:https://[www.researchgate.net/publication/274896750\_Implementation\_of\_a\_time\_t](http://www.researchgate.net/publication/274896750_Implementation_of_a_time_t) able\_generator\_using\_visual\_BasicNet

Evale, D. S. (2015). Integrated Class Scheduling System for Selected State Universities and Colleges with Satellite Campuses in the Philippines. International Journal of Engineering Research and General Science, 3(2). https://[www.semanticscholar.org/paper/Integrated-Class-Scheduling-System-for-](http://www.semanticscholar.org/paper/Integrated-Class-Scheduling-System-for-)

Gallegos, J. (2019, October 3). 5 Reasons Why Scheduling With Spreadsheets Just Doesn’t Work. Online Employee Schedule Software | Workforce Management – Humanity.Com. https://[www.humanity.com/blog/5-reasons-why-scheduling-with-](http://www.humanity.com/blog/5-reasons-why-scheduling-with-) spreadsheets-just-doesnt-work.html

Gamale, J., Anuta, E., & Sayson, Z. (2012). Automated class scheduling system. JPAIR Multidisciplinary Research, 9(1). https://[www.academia.edu/29553791/Automated\_Class\_Scheduling\_System.](http://www.academia.edu/29553791/Automated_Class_Scheduling_System) <http://dx.doi.org/10.7719/jpair.v9i1.11>

Gemino, A. (2017, March 22). A COMPUTERIZED CLASS SCHEDULING SYSTEM FOR CHRIST THE KING COLLEGE OF SCIENCE AND TECHNOLOGY.docx - CHRIST THE KING COLLEGE OF SCIENCE TECHNOLOGY A: Course Hero. CHRIST THE KING COLLEGE OF SCIENCE TECHNOLOGY A | Course Hero.

https://[www.coursehero.com/file/57728180/A-COMPUTERIZED-CLASS-SCHEDULING-](http://www.coursehero.com/file/57728180/A-COMPUTERIZED-CLASS-SCHEDULING-) SYSTEM-FOR-CHRIST-THE-KING-COLLEGE-OF-SCIENCE-AND-

TECHNOLOGYdocx/

J. B. Matias, A. C. Fajardo and R. P. Medina, “A Hybrid Genetic Algorithm for Course Scheduling and Teaching Workload Management,” 2018 IEEE 10th International

Conference on Humanoid, Nanotechnology, Information Technology,Communication and Control, Environment and Management (HNICEM), Baguio City, Philippines, 2018, pp. 1- 6, https://ieeexplore.ieee.org/document/8666332. DOI : 10.1109/HNICEM.2018.8666332

Labuanan F. et al (2019) Application of Representation and Fitness Method of Genetic Algorithm for Class Scheduling System. (2019). International Journal of Recent Technology and Engineering, 8(2), 1816–1821. https://[www.ijrte.org/wp-](http://www.ijrte.org/wp-) content/uploads/papers/v8i2/B1026078219.pdf. DOI: 10.35940/ijrte.B1026.078219

Legaspi, J. B., De Angel, R. M., Lagman, A., & Ortega, J. H. (2019). Web Based Course Scheduling System Using Greedy Algorithm. International Journal of Simulation: Systems, Science & Technology. https://ijssst.info/Vol-20/No-S2/paper14.pdf. DOI : 10.5013/IJSSST.a.20.S2.14

Miranda, J., Rey, P. A., & Robles, J. M. (2012). A Web architecture based decision support system for course and classroom scheduling. Decision Support Systems, 52(2), 505–513. [http://repositorio.uchile.cl/handle/2250/128364.](http://repositorio.uchile.cl/handle/2250/128364) DOI: doi:10.1016/j.dss.2011.10.011

Mondejar, J. P. (2013). SPAEve Version 2: An Improved Course and Classroom Scheduling Adapted for Evening Classes. JPAIR Institutional Research, 1(1). https://[www.researchgate.net/publication/284500935\_SPAEve\_Version\_2\_An\_Improved](http://www.researchgate.net/publication/284500935_SPAEve_Version_2_An_Improved)

\_Course\_and\_Classroom\_Scheduling\_Adapted\_for\_Evening\_Classes. DOI:10.7719/irj.v1i1.203

Muhamad, W. Z. A. W., Adnan, F. A., & Yahya, Z. R. (2018). PROCEEDING OF THE INTERNATIONAL CONFERENCE ON MATHEMATICS, ENGINEERING AND

INDUSTRIAL APPLICATIONS 2018. In Research Gate. Kuala Lumpur. https://[www.researchgate.net/publication/328044145\_Solving\_university\_course\_timeta](http://www.researchgate.net/publication/328044145_Solving_university_course_timeta) bling\_problems\_using\_FET\_software. DOI:10.1063/1.5054251

Mukherjee, N., Mukherjee, N., Kasundra, P., & S. (2021, November). 4 reasons why JavaScript is so popular. Opensource.Com. https://opensource.com/article/20/11/javascript-popular

Narra, N. (2021, August 17). 7 Benefits of School Time Table Management Systems. GoSchooler. https://goschooler.com/benefits-of-school-timetable-management-systems/

Pokorny, Kian & Vincent, Ryan. (2013). Multiple constraint satisfaction problems using the A-star (A\*) search algorithm: classroom scheduling with preferences. Journal of Computing Sciences in Colleges. https://[www.researchgate.net/publication/262277120\_Multiple\_constraint\_satisfaction\_pr](http://www.researchgate.net/publication/262277120_Multiple_constraint_satisfaction_pr) oblems\_using\_the\_A- star\_A\_search\_algorithm\_classroom\_scheduling\_with\_preferences

Rettig, Michael D. (2019). The Effects of Block Scheduling. AASA. https://[www.aasa.org/schooladministratorarticle.aspx?id=14852#:~:text=The%20study%](http://www.aasa.org/schooladministratorarticle.aspx?id=14852&%3A~%3Atext=The%20study%25) 20reported%3A%20%22Overall%2C,or%20at%20most%20seven%20credits

Saptarini, N. G., Suasnawa, I. W., & Ciptayani, P. I. (2018). Senior high school course scheduling using genetic algorithm. Journal of Physics: Conference Series, 953, 012067. https://[www.researchgate.net/publication/322942231\_Senior\_high\_school\_course\_sche](http://www.researchgate.net/publication/322942231_Senior_high_school_course_sche) duling\_using\_genetic\_algorithm. DOI:10.1088/1742-6596/953/1/012067

Sasser, D. N. (2017, November 21). Block Vs. Traditional Scheduling Student Perspective. Education. https://education.seattlepi.com/block-vs-traditional-scheduling- student-perspective-1577.html

Sayson, Z. D. (2012). Automated Class Scheduling System. Ejournals. Retriebed from https://ejournals.ph/article.php?id=7525. DOI:<http://dx.doi.org/10.7719/pair.v9i1.11>

Tan, M. J. M., Jarilla, A. G., Joy, J., & Neria, C. B. A Comparative Study of Particle Swarm Optimization and Genetic Algorithm in a Course Scheduling System for a State University. Research Gate.

https://[www.researchgate.net/publication/345307026\_A\_Comparative\_Study\_of\_Particle](http://www.researchgate.net/publication/345307026_A_Comparative_Study_of_Particle)

\_Swarm\_Optimization\_and\_Genetic\_Algorithm\_in\_a\_Course\_Scheduling\_System\_for\_a

\_State\_University. DOI:10.13140/RG.2.2.18714.06081

Taylor, D. (2022, March 5). What is MongoDB? Introduction, Architecture, Features & Example. Guru99. https://[www.guru99.com/what-is-mongodb.html](http://www.guru99.com/what-is-mongodb.html)

Tuaycharoen, N., Prodpran V. and Srithong, B., “ClassSchedule: A web-based application for school class scheduling with real-time lazy loading,” 2018 5th International Conference on Business and Industrial Research (ICBIR), Bangkok, Thailand, 2018, pp. 210-214, https://[www.researchgate.net/publication/325910956\_ClassSchedule\_A\_web-](http://www.researchgate.net/publication/325910956_ClassSchedule_A_web-) based\_application\_for\_school\_class\_scheduling\_with\_real-time\_lazy\_loading.

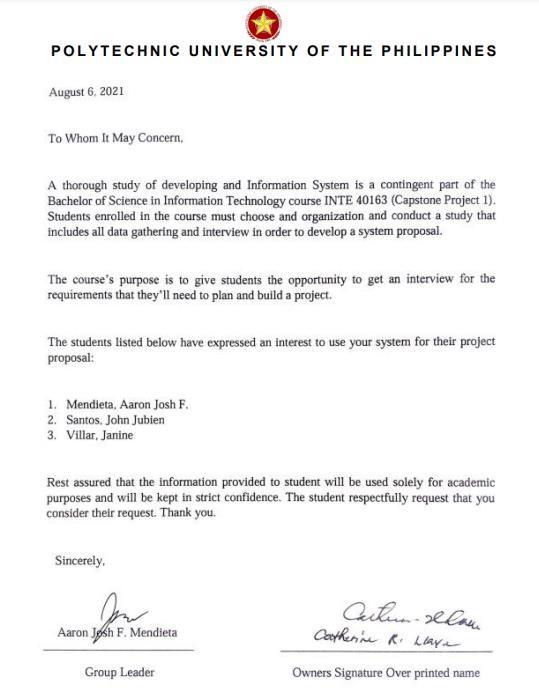
DOI:10.1109/ICBIR.2018.8391194

Uifalean, A. (2020, June 20). 6 Reasons Why Your School Needs Automated Scheduling. Teachngo. https://[www.teachngo.com/blog/6-reasons-automated-scheduling.php](http://www.teachngo.com/blog/6-reasons-automated-scheduling.php)

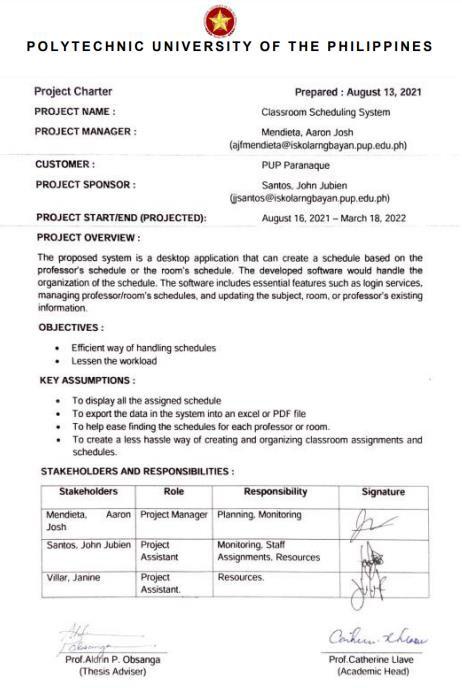
Wen-jing, W. (2018). Improved Adaptive Genetic Algorithm for Course Scheduling in Colleges and Universities. International Journal of Emerging Technologies in Learning (IJET), 13(06), 29. https://doi.org/10.3991/ijet.v13i06.8442

APPENDICES

**Appendix 1 PERMISSION LETTER**



**Appendix 2 PROJECT CHARTER**



**Appendix 3 COST ESTIMATE**

|  |  |
| --- | --- |
| **Items** | **Cost** |
| **Laptop** | **15,000php** |
| **Bond papers** | **600php** |
| **Printer Ink** | **1675php** |
| **Hardbound** | **2000php** |
| **Copyright** | **300php** |
| **Grammarian Check** | **500php** |
| **Plagiarism Check** | **100php** |
| **Notary** | **200php** |
| **Total** | **20,275php** |

**Appendix 4 QUESTIONNAIRE**

# QUESTIONNAIRE FOR ISO 25010: SOFTWARE QUALITY STANDARDS FOR CLASSROOM SCHEDULING SYSTEM

## PERSONAL INFORMATION

* 1. **Name :**
  2. **Type of Respondent :**
  3. **Address :**
  4. **Sex :**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **4** | **3** | **2** | **1** |
| **A. Functional Suitability** | | | | |
| 1. **Functional completeness** - The system covers all the specified tasks and user objectives. |  |  |  |  |
| 2. **Functional correctness** - The system provides the correct results with the needed degree of precision. |  |  |  |  |
| 3. **Functional appropriateness** - The system facilitates the accomplishment of specified tasks and objectives. |  |  |  |  |
| **B. Performance Efficiency** | | | | |
| 1. **Time behavior** - The system’s response and processing times and throughput rates when performing its functions, meet requirements. |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 2. **Resource utilization** - The system’s amounts and types of resources used when performing its functions, meet requirements. |  |  |  |  |
| 3. **Capacity** - The system’s maximum limits of parameter meet requirements. |  |  |  |  |
| **C. Compatibility** | | | | |
| 1. **Co-existence** - The system can perform its required functions efficiently while sharing a common environment and resources with other products, without detrimental impact on any other product |  |  |  |  |
| 2. **Interoperability** - The system can exchange information and use the information that has been exchanged. |  |  |  |  |
| **D. Usability** | | | | |
| 1. **No Appropriateness recognizability** - The system allows users to recognize if it is appropriate for their needs. |  |  |  |  |
| 2. **Learnability** - The system can be used by specified users to achieve specified goals of learning to use the application with effectiveness, efficiency, freedom from risk and satisfaction in a specified context of use. |  |  |  |  |
| 3. **Operability** - The system has attributes that make it easy to operate and control. |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 4. **User error protection** - The system protects users against making errors. |  |  |  |  |
| 5. **User interface aesthetics** - The system’s user interface enables pleasing and satisfying interaction for the user. |  |  |  |  |
| 6. **Accessibility** - The system can be used by people with the widest range of characteristics and capabilities to achieve a specified goal in a specified context of use. |  |  |  |  |
| **E. Reliability** |  |  |  |  |
| 1. **Maturity** - Degree to which a system, product or component meets needs for reliability under normal operation. |  |  |  |  |
| 2. **Availability** - Degree to which a system, product or component is operational and accessible when required for use. |  |  |  |  |
| 3. **Fault Tolerance** - Degree to which a system, product or component operates as intended despite the presence of hardware or software faults. |  |  |  |  |
| 4. **Recoverability** - Degree to which, in the event of an interruption or a failure, a product or system can recover the data directly affected and re-establish the desired state of the system. |  |  |  |  |
| **F. Security** |  |  |  |  |
| 1. **Confidentiality** - The system  ensures that data are |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| accessible only to those authorized to have access. |  |  |  |  |
| 2. **Integrity** - The system prevents unauthorized access to, or modification of, computer programs or data. |  |  |  |  |
| 3. **Non-repudiation** - The system can be proven to have taken place, so that the events or actions cannot be repudiated later. |  |  |  |  |
| **G. Maintainability** |  |  |  |  |
| 1. **Modularity** - Degree to which a system or computer program is composed of discrete components such that a change to one component has minimal impact on other components. |  |  |  |  |
| 2. **Reusability** - Degree to which an asset can be used in more than one system, or in building other assets. |  |  |  |  |
| 3. **Analysability** - Degree of effectiveness and efficiency with which it is possible to assess the impact on a product or system of an intended change to one or more of its parts, or to diagnose a product for deficiencies or causes of failures, or to identify parts to be modified. |  |  |  |  |
| 4. **Modifiability** - Degree to which a product or system can be effectively and efficiently modified without introducing defects or degrading existing product quality. |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 5. **Testability** - Degree of effectiveness and efficiency with which test criteria can be established for a system, product or component and tests can be performed to determine whether those criteria have been met. |  |  |  |  |
| **H. Portability** |  |  |  |  |
| 1. **Adaptability** - Degree to which a product or system can effectively and efficiently be adapted for different or evolving hardware, software or other operational or usage environments. |  |  |  |  |
| 2. **Installability** - Degree of effectiveness and efficiency with which a product or system can be successfully installed and/or uninstalled in a specified environment. |  |  |  |  |
| 3. **Replaceability** - Degree to which a product can replace another specified software product for the same purpose in the same environment. |  |  |  |  |

**Appendix 5 Picture with the Client**



**Appendix 6 Transcript of the Interview**

**Researchers:** Goodmorning po maam, we would like to interview po about po sa system na Classroom Scheduling System for PUP Parañaque

**Client:** Ah scheduling ng mga prof

**Researchers:** Opo, interview ka po naming maam kung ano po gusto niyong mangyari sa system and pano po yung mga magiging functions.

**Client:** Okay so ganito, about sa system, kasi ang mangyayari nyan kailangan ma- identify nyo kung may conflict sa room, sa schedule ng mga bata at sa schedule ni prof. So halimbawa mararanasan nyo naman yun minsan nung nasa face to face pa tayo nandun na kayo sa room nyo tapos may mag sasabi sainyo na estudyante eto nakalagay sa schedule namin,

**Researchers:** Ay opo, yes po

**Client:** Aan ang scheduling system. So kasi ang mangyayari kasi nyan bawat sem naka paskil na kung sino lang dapat yung mga estudyante nandun sa loob na room na yun. Kasi iniiwasan din dyan yun mga nasisirang upuan kasi may tumatambay na estudyante.

**Researchers:** Opo

**Client:** Diba alam nyo naman yung nangyari lastime? Yung nag dra-drawing ng ari ng lalaki

**Researchers:** Ay opo

**Client:** Kaya hanggang ngayon di pa sya nahuhuli, kung sino man yon. Yan yung mga vandalism na ganyan. Tsaka yung sa mga tumatambay na estudyante kaya may nasisira naupuan, kasi umuupo sila dun sa sa may mismong pinagsusulatan **Researchers:** Opo sa arm chair

**Client:** Yan ang scheduling nayan. So ang mayayari dyan eto pag inencode ko si prof halimbawa 7:30 to 10:30 jenifer aguas for posit communication room 301 **Researchers:** Opo

**Client:** So mabloblock na yun sa system ninyo.

**Researchers**: Ah okay po

**Client:** 7:30 to 10:30 monday halimbawa naka block na yun sa 301, kunwari nakalimutan ko si Maam Aguas tapos nandun pala siya sa 301 nilagay ko si Sir Acorda ng 7:30 to 10:30 sa 301, inonotify ako ni system na sasabihin “you cannot use the room anymore, you cannot use the time anymore. Because it is already in conflict.”

**Researchers:** Sige po maam Ahmm ayun po ba maam? So bali po maam ang

gusto po rin namin malaman is kung mga pwede pong features ng classroom scheduling po na gusto nyo po makita dun sa system.

**Client:** Yung features ng inyong magiging system?

**Researchers:** Opo.

**Client:** Kailangan meron syang leway dapat yan naka connect sya sa class record ni prof.

**Researchers:** Okay po.

**Client:** Pero di ko na pinapagawa sainyo dapat meron kayong way para kung sino pinapagawa ko ng class record may connect yan sainyo

**Researchers:** Okay po.

**Client:** So yun basta inonotify ako ni system na hindi na ako pwede dito si aguas si prof hanggang 40hrs lang pag nag exceed ako ng oras inonotify din ako na hindi na sya pwede mag exceed kasi professor has already reach 40hrs of schedule.

**Researchers:** Okay po.

**Client:** Tapos iba pa to ah kasi iba yung sa mga prof na regular iba din samin na may mga office halimbwa ako allowed lang ako ng 24hrs iba yung schedule namin ng regular sa mga prof ninyo.

**Researchers:** Okay po.

**Client:** Tapos kailangan yung system mag print ng excel and pdf tapos halimbawa meron din syang leway para sa changes. Halimbawa si prof, dapat limited lang para sa regular.

**Researchers:** Okay po.

**Client:** So halimbawa ako, kasi ang mga regular faculty kasi nahahati yung oras nila ng 1.5 to 1.5.

**Researchers:** Okay po. Maam. So halimbawa kung meron changes sa schedule nakapag suggest Si Sir ribert gusto nya gawing 3hrs. Ayun lang po maam?

**Client:** Oo ayun lang

**Researchers:** Ah sige po. Ano din po maam anong model po pwede namin gamitin kung waterfall or rapid? Yun po yung ano namin na naging pinaparevised ng depend po kami ng type chapter 1 to 3 sainyo po and then yung ano po yung model na gagamitin namin kung okay po ba rapid or waterfall, yung sa SDLC po. **Client:** Tingin ko rapid

**Researchers:** Ah sige po maam, okay po maam so okay na po yung features na note na natin po and then rapid gusto nyo din ayun lang po ba maam? May sasabihin pa po ba kayo?

**Client:** Wala naman na, sigurado ba naiintinidhan nyo? **Researchers:** Opo maam nag take notes para po ano namin. **Client:** Ah nag take down notes kayo?

**Researchers:** Opo **Client:** Saglit lang ah.. **Researchers:** Opo

**Client:** Ganito magiging itsura nya ah, nakikita nyo yung excel?

**Researchers:** Opo

**Client:** Halimbawa itong media ni mayas devera nakalagay na dyan yung kanyang oras so halimbawa itong mga colon ang nakalagay dyan mga part time. **Researchers:** Opo

**Client:** Kailangan 12hrs lang sya dapat may mag cocompute din na itong monday nato hanggang 12hrs lang basta everyday max nila 12hrs. So halimbawa meron akong coding ang color nato ay for part time load yung mga walang color ts load

kapag tinotall yan lessthan 40hrs, 36 may nagcocompute nyan. Tapos halimbawa itong si Sir Acorda sya ay regular faculty itong part time nya tapos itong color nato ito yung regular load nya so iba yung sa regular iba rin yung sa part time so ito yung mga ts load nya meron kasi kaming tatlong load na tinawag basta hindi sila mag eexceed sa 40 hrs.

**Researchers:** Ah okay po.

**Client:** So ganyan magiging input nya nandyaan yung course code nandyan yung subject code, section tsaka dapat nandyan din yung room, ayun nga yung problema ko ngayon hindi ko mailagay dyan kasi hindi ko namomonitor yung room. **Researchers:** Okay po

**Client:** So ayan ang output nyan si prof so dapat meron pang isang output yan ang nakalagay naman room 201 7:30 to 9:30 subject code, subject description tapos si prof.

**Researchers**: Okay po.

**Client:** Ito yung ipapaskil ko sa room so kailangan nakalagay ang room number

**Researchers:** Sige po maam

**Client:** Ano ibigay ko pa ba ito sainyo para meron kayong pag basehan?

**Researchers:** Pwede naman po para may pag babasehan kami

**Client:** Sige, send ko nalang sa email niyo

**Researchers:** Ayun sige po maam, wala na po ba kayo madadagdag?

**Client:** Ayun lang, Yun lang naman.

**Researchers:** Ah sige po maam salamat po sa interview, na take notes po namin kaya nakukuha namin yung sinasabi nyo po so kami na po bahala dun sa system maam. Maraming salamat rin po nag reply kayo sa email namin

**Client:** Okay, sige. Thank you rin.

**Researchers:** Salamat po

**Appendix 7 Source Code**

const mongoose = require("mongoose"); const { ipcMain, dialog } = require("electron");

const path = require("path"); const xl = require("excel4node");

module.exports = function (win) {

const CourseModel = require(`${ dirname}/models/courseMo del.js`);

const ProfessorModel = require(`${ dirname}/models/professor Model.js`);

const ProgramModel = require(`${ dirname}/models/program Model.js`);

const RoomModel = require(`${ dirname}/models/roomMod el.js`);

const ScheduleModel = require(`${ dirname}/models/schedule Model.js`);

const SettingsModel = require(`${ dirname}/models/settingsM odel.js`);

let connectedStatus = false; const uri =

"mongodb://localhost:27017/dotsystems ";

mongoose.connect( uri,

{ useNewUrlParser: true, useUnifiedTopology: true },

async (err) => {

if (err != null) {

connectedStatus = false;

} else {

connectedStatus = true;

let initialSettingsCheck = await SettingsModel.find({

uid:

"time is gold",

});

if (initialSettingsCheck.length == 0) {

try

{

const newSettingsData = new SettingsModel({

uid: "time is gold",

schoolYear: "2018",

schoolSemester: "0",

systemPassword:

"admin",

});

await newSettingsData.save();

}

catch (err) {

console.log(err);

}

}

}

}

);

// When the window refreshes, the renderer will request for a connection again.

ipcMain.on("loading- screen", () =>

win.webContents.send("loa ding-screen", connectedStatus)

);

// Dashboard ipc handler. ipcMain.on("retrieve-

dashboard-course", () => {

try { ProfessorModel.count((err,

number) => {

CourseModel.count((err, if (err) {

number) => {

if (err) {

win.webContents.send("ret rieve-dashboard-course", "err");

throw err;

} else

win.webContents.send("retrieve- dashboard-course", number);

});

} catch (err) {

win.webContents.send("ret rieve-dashboard-course", err);

}

});

ipcMain.on("retrieve- dashboard-professor", () => {

try {

win.webContents.send("ret rieve-dashboard-professor", "err");

throw err;

} else

win.webContents.send("retrieve- dashboard-professor", number);

});

} catch (err) {

win.webContents.send("ret rieve-dashboard-professor", err);

}

});

ipcMain.on("retrieve- dashboard-room", () => {

try {

RoomModel.count((err,

number) => {

if (err) {

win.webContents.send("ret rieve-dashboard-room", "err");

throw err;

} else

win.webContents.send("retrieve- dashboard-room", number);

});

} catch (err) {

win.webContents.send("ret rieve-dashboard-room", err);

}

});

ipcMain.on("retrieve- dashboard-program", () => {

try {

ProgramModel.count((err,

number) => {

if (err) {

win.webContents.send("ret rieve-dashboard-program", "err");

throw err;

} else

win.webContents.send("retrieve- dashboard-program", number);

});

} catch (err) {

win.webContents.send("ret rieve-dashboard-program", err);

}

});

// Retrieve data for pages. ipcMain.on("retrieve-

course-data", async () => {

try {

let courseData = await CourseModel.find().sort({

name: 1,

code: 1,

units: 1,

hours: 1, "retrieve-room-

});

win.webContents.send(

data",

JSON.parse(JSON.stringif

"retrieve- y(roomData))

course-data",

JSON.parse(JSON.stringif y(courseData))

);

} catch (err) {

win.webContents.send("ret rieve-course-data", err);

}

});

ipcMain.on("retrieve-room- data", async () => {

try {

let roomData = await RoomModel.find().sort({

name: 1,

});

win.webContents.send(

);

} catch (err) {

win.webContents.send("ret rieve-room-data", err);

}

});

ipcMain.on("retrieve- professor-data", async () => {

try {

let ProfessorData

= await ProfessorModel.find().sort({

"name.first": 1,

"name.last": 1,

});

win.webContents.send(

"retrieve-professor-data", "retrieve-program-

data",

JSON.parse(JSON.stringif

y(ProfessorData))

);

} catch (err) {

win.webContents.send("ret rieve-professor-data", err);

}

});

ipcMain.on("retrieve- program-data", async () => {

try {

let ProgramData = await ProgramModel.find().sort({

name: 1, acronym:

1,

year: 1,

section: 1,

});

win.webContents.send(

JSON.parse(JSON.stringif y(ProgramData))

);

} catch (err) {

win.webContents.send("ret rieve-program-data", err);

}

});

ipcMain.on("retrieve- settings-data", async () => {

try {

let SettingsData = await SettingsModel.find();

win.webContents.send(

"retrieve-

settings-data",

JSON.parse(JSON.stringif y(SettingsData))

);

} catch (err) {

win.webContents.send("ret rieve-settings-data", err);

}

});

ipcMain.on("retrieve- schedule-professor-data", async (event, professorId) => {

console.log("Entered"); console.log(professorId); try {

let ScheduleData

= await ScheduleModel.find({

"professorData.\_id":

professorId,

}).sort({

day: 1,

"time.start.hours": 1,

"time.start.minutes": 1,

});

console.log("passed here");

console.log(JSON.parse(J SON.stringify(ScheduleData)));

win.webContents.send(

"retrieve-

schedule-professor-data",

JSON.parse(JSON.stringif y(ScheduleData))

);

} catch (err) {

console.log(err);

win.webContents.send("ret rieve-schedule-professor-data", err);

}

});

ipcMain.on("retrieve- schedule-program-data", async (event, programId) => {

console.log("Entered"); console.log(programId); try {

let ScheduleData = await ScheduleModel.find({

"programData.\_id":

programId,

}).sort({

day: 1,

"time.start.hours": 1,

"time.start.minutes": 1,

});

console.log("passed here");

console.log(JSON.parse(J SON.stringify(ScheduleData)));

win.webContents.send(

"retrieve-

schedule-program-data",

JSON.parse(JSON.stringif y(ScheduleData))

);

} catch (err) {

win.webContents.send("ret rieve-schedule-program-data", err);

}

});

ipcMain.on("retrieve- schedule-room-data", async (event, roomId) => {

console.log("Entered"); console.log(roomId); try {

let ScheduleData

= await ScheduleModel.find({

"roomData.\_id": roomId,

}).sort({

day: 1,

"time.start.hours": 1,

"time.start.minutes": 1,

});

console.log("passed here");

console.log(JSON.parse(J SON.stringify(ScheduleData)));

});

if

(courseNameExist.length >= 1) throw

win.webContents.send(

"retrieve-

schedule-room-data",

JSON.parse(JSON.stringif y(ScheduleData))

);

} catch (err) {

win.webContents.send("ret rieve-schedule-room-data", err);

}

});

ipcMain.on("save-data- course", async (event, courseData) => {

try {

let

courseNameExist = await CourseModel.find({

name:

courseData.name,

"Course name already exist.";

let

courseCodeExist = await CourseModel.find({

code:

courseData.code,

});

if (courseCodeExist.length >= 1) throw "Course code already exist.";

const newCourseData = new CourseModel({

name:

courseData.name,

code:

courseData.code,

units:

courseData.units,

hours:

courseData.hours,

});

await

let

roomNameExist = await

newCourseData.save(); RoomModel.find({

name:

win.webContents.send("sa ve-data-course", {

success:

roomData.name,

});

if

true,

message:

(roomNameExist.length >= 1) throw "Room already exist.";

"Added successfully.",

});

} catch (err) {

win.webContents.send("sa ve-data-course", {

success:

false,

message:

err,

});

}

});

ipcMain.on("save-data- room", async (event, roomData) => {

try {

const newRoomData = new RoomModel({

name:

roomData.name,

});

await newRoomData.save();

win.webContents.send("sa ve-data-room", {

success:

true,

message:

"Added successfully.",

});

} catch (err) { },

address:

win.webContents.send("sa ve-data-room", {

success:

false,

message:

err,

});

}

});

ipcMain.on("save-data- professor", async (event, professorData)

=> {

try {

const

newProfessorData = new ProfessorModel({

name: {

first: professorData.name.first,

last: professorData.name.last,

professorData.address,

employment: {

status: professorData.employment.status,

hours: professorData.employment.hours,

},

gender:

professorData.gender,

contact:

professorData.contact,

});

await newProfessorData.save();

win.webContents.send("sa ve-data-professor", {

success:

true,

message: "Added successfully.",

});

} catch (err) {

programData.section,

section:

});

if

(programExist.length >= 1) throw

win.webContents.send("sa ve-data-professor", {

success:

false,

message:

err,

});

}

});

ipcMain.on("save-data- program", async (event, programData)

=> {

try {

let programExist = await ProgramModel.find({

name:

programData.name,

year:

programData.year,

"Program already exist.";

const

newProgramData = new ProgramModel({

name:

programData.name,

acronym:

programData.acronym,

year:

programData.year,

section:

programData.section,

});

await newProgramData.save();

win.webContents.send("sa ve-data-program", {

success: true,

"Added successfully",

});

} catch (err) {

message:

`${scheduleData.time.start. hours}${scheduleData.time.start.minute s}`

) < 730

)

throw

win.webContents.send("sa ve-data-program", {

success:

false,

message:

err,

});

}

});

ipcMain.on("save-data- schedule", async (event, scheduleData)

=> {

// console.log(scheduleData);

try {

if (

parseInt(

"Time must start at 7:30 AM.";

if (

parseInt(

`${scheduleData.time.start. hours}${scheduleData.time.start.minute s}`

) > 2100

)

throw "Time must start at 7:30 AM.";

if (

parseInt(

`${scheduleData.time.end. hours}${scheduleData.time.end.minutes

}`

) < 730

)

});

console.log("Line:

throw 284");

"Time must end at 9:00 PM.";

console.log(scheduleData

if ( Gathered);

parseInt( console.log("Line:

286");

`${scheduleData.time.end.

hours}${scheduleData.time.end.minutes

}`

) > 2100

)

throw

"Time must end at 9:00 PM.";

console.log(scheduleData. professorData.\_id);

let scheduleDataGathered = await ScheduleModel.find({

"professorData.\_id": scheduleData.professorData.\_id,

let

scheduleDataHours = scheduleData.courseData.hours;

scheduleDataGathered.for Each((schedule) => {

scheduleDataHours += parseInt(schedule.courseData.hours);

});

console.log(

"Declared

Hours: ",

scheduleData.professorDat a.employment.hours

); day:

scheduleData.day,

console.log("Hours Consumed: ", scheduleDataHours);

if (scheduleData.professorData.employme nt.hours < scheduleDataHours)

throw "Professor has no remaining hours.";

console.log(

"Sanity

Check: ",

scheduleData.professorDat a.employment.hours < scheduleDataHours

);

scheduleDataGathered = await ScheduleModel.find({

"roomData.\_id": scheduleData.roomData.\_id,

});

let scheduleDataGatheredRoomSanityArra y = [];

scheduleDataGathered.for Each((schedule) => {

let s1 =

parseInt(

`${scheduleData.time.start. hours}${scheduleData.time.start.minute s}`

);

let e1 =

parseInt(

`${scheduleData.time.end. hours}${scheduleData.time.end.minutes

}`

);

let s2 =

parseInt(

`${schedule.time.start.hour s}${schedule.time.start.minutes}`

);

let e2 =

e2);

console.log("e1 < e2:", e1 <

console.log("e1 <= s2 &&

parseInt( e1 < e2", e1 <= s2 && e1 < e2);

`${schedule.time.end.hours

}${schedule.time.end.minutes}`

);

console.log("-------------------

- ");

|  |  |  |
| --- | --- | --- |
|  | console.log("Starting Room | console.log( |
| Check..."); |  |  |
|  |  | "(s1 >= e2 && s1 > s2) || (e1 |
|  | console.log("s1 >= e2: ", s1 | <= s2 && e1 < e2):", |
| >= e2); |  | (s1 |
|  |  | >= e2 && s1 > s2) || (e1 <= s2 && e1 < |
|  | console.log("s1 > s2: ", s1 > | e2) |
| s2); |  | ); |

console.log("s1 >= e2 &&

s1 > s2: ", s1 >= e2 && s1 > s2); console.log("-------------------

");

console.log("e1 <= s2:", e1

<= s2);

if ((s1 >= e2 && s1 > s2) || (e1 <= s2 && e1 < e2)) {

scheduleDataGatheredRoo mSanityArray.push(true);

} else

scheduleDataGatheredRoomSanityArra y.push(false);

});

let roomSanity =

true;

scheduleDataGatheredRoo mSanityArray.forEach((sanity, i) => {

console.log(`${i}:

${sanity}`);

if (sanity

== false) {

roomSanity = false;

}

});

if (!roomSanity) throw "Selected room is used around that time.";

console.log("Continuing to programData...");

scheduleDataGathered = await ScheduleModel.find({

"programData.\_id": scheduleData.programData.\_id,

day:

scheduleData.day,

});

let scheduleDataGatheredProgramSanityAr ray = [];

scheduleDataGathered.for Each((schedule) => {

let s1 =

parseInt(

`${scheduleData.time.start. hours}${scheduleData.time.start.minute s}`

);

let e1 =

parseInt(

Check...");

>= e2);

console.log("Starting Room

console.log("s1 >= e2: ", s1

console.log("s1 > s2: ", s1 >

`${scheduleData.time.end. hours}${scheduleData.time.end.minutes

}`

s2);

console.log("s1 >= e2 &&

parseInt(

);

let s2 =

s1 > s2: ", s1 >= e2 && s1 > s2);

console.log("e1 <= s2:", e1

<= s2);

`${schedule.time.start.hour

s}${schedule.time.start.minutes}`

);

let e2 =

parseInt(

e2);

console.log("e1 < e2:", e1 <

console.log("e1 <= s2 &&

e1 < e2", e1 <= s2 && e1 < e2);

`${schedule.time.end.hours

}${schedule.time.end.minutes}`

); console.log("-------------------

");

console.log(

"(s1 >= e2 && s1 > s2) || (e1

<= s2 && e1 < e2):",

(s1

>= e2 && s1 > s2) || (e1 <= s2 && e1 < e2)

);

console.log("-------------------

- ");

if ((s1 >= e2 && s1 > s2) || (e1 <= s2 && e1 < e2))

{

scheduleDataGatheredPro gramSanityArray.push(true);

} else

scheduleDataGatheredProgramSanityAr ray.push(false);

});

let programSanity

= true;

scheduleDataGatheredPro gramSanityArray.forEach((sanity, i) => {

console.log(`${i}:

${sanity}`);

if (sanity

== false) {

programSanity = false;

}

});

if (!programSanity) throw "Selected program is busy around that time.";

console.log("Continuing to professorData...");

scheduleDataGathered = await ScheduleModel.find({

"professorData.\_id": scheduleData.professorData.\_id,

day: scheduleData.day,

});

let scheduleDataGatheredProfessorSanity Array = [];

scheduleDataGathered.for Each((schedule) => {

let s1 =

parseInt(

`${scheduleData.time.start. hours}${scheduleData.time.start.minute s}`

);

let e1 =

parseInt(

`${scheduleData.time.end. hours}${scheduleData.time.end.minutes

}`

);

let s2 =

parseInt(

`${schedule.time.start.hour s}${schedule.time.start.minutes}`

);

let e2 =

parseInt(

`${schedule.time.end.hours

}${schedule.time.end.minutes}`

);

console.log("Starting Room

Check...");

console.log("s1 >= e2: ", s1

>= e2);

console.log("s1 > s2: ", s1 >

s2);

console.log("s1 >= e2 && s1 > s2: ", s1 >= e2 && s1 > s2);

console.log("e1 <= s2:", e1

<= s2);

e2);

console.log("e1 < e2:", e1 <

if ((s1 >= e2 && s1 > s2) || (e1 <= s2 && e1 < e2))

{

console.log("e1 <= s2 &&

e1 < e2", e1 <= s2 && e1 < e2);

console.log("-------------------

- ");

console.log(

"(s1 >= e2 && s1 > s2) || (e1

<= s2 && e1 < e2):",

(s1

>= e2 && s1 > s2) || (e1 <= s2 && e1 < e2)

);

console.log("-------------------

- ");

scheduleDataGatheredProf essorSanityArray.push(true);

} else

scheduleDataGatheredProfessorSanity Array.push(false);

});

let professorSanity = true;

scheduleDataGatheredProf essorSanityArray.forEach((sanity, i) => {

console.log(`${i}:

${sanity}`);

if (sanity

== false) {

professorSanity = false;

}

});

if (!professorSanity) throw "Professor is busy around that time.";

minutes: scheduleData.time.end.minutes,

},

const

newScheduleData = new ScheduleModel({

day:

scheduleData.day,

time: {

year: scheduleData.school.year,

semester:

},

school: {

start: {

hours: scheduleData.time.start.hours,

scheduleData.school.semester,

},

courseData: {

\_id:

minutes: scheduleData.time.start.minutes,

},

scheduleData.courseData.\_id,

name: scheduleData.courseData.name,

end: {

code:

hours: scheduleData.time.end.hours,

scheduleData.courseData.code,

units: scheduleData.courseData.units,

hours: scheduleData.courseData.hours,

},

professorData: {

scheduleData.professorData.\_id,

name: {

first:

\_id:

hours: scheduleData.professorData.employme nt.hours,

},

gender: scheduleData.professorData.gender,

contact: scheduleData.professorData.contact,

},

scheduleData.professorData.name.first,

programData: {

last: scheduleData.professorData.name.last,

},

address: scheduleData.professorData.address,

scheduleData.programData.\_id,

name: scheduleData.programData.name,

acronym:

\_id:

employment: {

status: scheduleData.professorData.employme nt.status,

scheduleData.programData.acronym,

year: scheduleData.programData.year,

section: scheduleData.programData.section,

},

roomData:

{

\_id:

scheduleData.roomData.\_id,

name: scheduleData.roomData.name,

},

});

await newScheduleData.save();

win.webContents.send("sa ve-data-schedule", {

success:

true,

message:

"Added successfully.",

});

} catch (err) {

console.log(err);

win.webContents.send("sa ve-data-schedule", {

success:

false,

message:

err,

});

}

});

ipcMain.on("edit-data- course", async (event, courseData, selectedData) => {

console.log(courseData,

selectedData);

try {

let

courseNameExist = await CourseModel.find({

name:

courseData.name,

});

if (courseNameExist.length >= 1) {

if (courseData.name != selectedData.name) {

await CourseModel.updateOne(

{ \_id:

throw "Course name already exist.";

}

}

let

courseCodeExist = await CourseModel.find({

code:

courseData.code,

});

if (courseCodeExist.length >= 1) {

if (courseData.code != selectedData.code)

{

throw "Course code already

exist.";

}

}

courseData.\_id },

{

$set: {

name: courseData.name,

code: courseData.code,

units: courseData.units,

hours: courseData.hours,

},

}

);

await ScheduleModel.updateMany(

{

"courseData.\_id": courseData.\_id },

{

$set: {

"courseData.name": courseData.name,

"courseData.code": courseData.code,

"courseData.units": courseData.units,

"courseData.hours": courseData.hours,

},

}

);

} catch (err) {

win.webContents.send("edi t-data-course", {

success:

false,

message:

err,

});

}

});

ipcMain.on("edit-data- room", async (event, roomData, selectedData) => {

try {

let

roomNameExist = await RoomModel.find({

win.webContents.send("edi t-data-course", {

success:

true,

roomData.name,

});

if

name:

message: (roomNameExist.length >= 1) {

"Edit successfully.",

});

if (roomData.name != {

selectedData.name) {

$set: {

throw "Room already

exist.";

"roomData.name":

} roomData.name,

} },

}

await );

RoomModel.updateOne(

{ \_id:

roomData.\_id },

{

$set: {

name: roomData.name,

},

}

);

await ScheduleModel.updateMany(

{

"roomData.\_id": roomData.\_id },

win.webContents.send("edi t-data-room", {

success:

true,

message:

"Edit successfully.",

});

} catch (err) {

console.log(err);

win.webContents.send("edi t-data-room", {

success:

false,

message: err,

});

}

});

ipcMain.on(

"edit-data-professor", async (event,

professorData, selectedData) => {

try {

await

ProfessorModel.updateOne(

{

\_id: professorData.\_id },

{

"employment.status": professorData.employment.status,

"employment.hours": professorData.employment.hours,

gender: professorData.gender,

contact: professorData.contact,

},

}

);

$set: {

await

"name.first": professorData.name.first,

ScheduleModel.updateMany(

{

"professorData.\_id": professorData.\_id },

"name.last": {

professorData.name.last,

$set: {

address: professorData.address,

"professorData.name.first": professorData.name.first,

"professorData.name.last": professorData.name.last,

"professorData.address": professorData.address,

"professorData.employmen

t.status":

professorData.employment

.status,

"professorData.employmen t.hours": professorData.employment.hours,

"professorData.gender": professorData.gender,

"professorData.contact": professorData.contact,

},

}

);

win.webContents.send("edi t-data-professor", {

success: true,

message: "Edit successfully.",

});

} catch (err) {

win.webContents.send("edi t-data-professor", {

success: false,

message: err,

});

} programData.section !=

} selectedData.section

); ) {

ipcMain.on("edit-data-

program", async (event, programData, selectedData) => {

try {

let programExist = await ProgramModel.find({

name:

exist.";

throw "Program already

}

}

await

programData.name, ProgramModel.updateOne(

year: { \_id:

programData.year, programData.\_id },

section: {

programData.section,

});

if (programExist.length >= 1) {

$set: {

name: programData.name,

if (

acronym:

programData.name != selectedData.name ||

programData.year != selectedData.year ||

programData.acronym,

year: programData.year,

section: },

programData.section, }

}, );

}

);

win.webContents.send("edi

await ScheduleModel.updateMany(

t-data-program", {

success:

{

"programData.\_id": programData.\_id },

{

true,

"Edit successfully.",

message:

});

$set: {

"programData.name": programData.name,

"programData.acronym": programData.acronym,

"programData.year": programData.year,

"programData.section": programData.section,

} catch (err) {

win.webContents.send("edi t-data-program", {

success:

false,

message:

err,

});

}

});

ipcMain.on("edit-data-settings", async (event, settingsData) => {

try {

await SettingsModel.updateOne(

successfully.",

message: "Edit

});

} catch (err) {

is gold" },

$set: {

schoolYear:

{ uid: "time

{

win.webContents.send("edi t-data-settings", {

success:

false,

message:

err,

});

settingsData.schoolYear,

schoolSemester: settingsData.schoolSemester,

},

}

);

}

});

ipcMain.on("edit-data- password", async (event, settingsData)

=> {

try {

if (settingsData.systemPassword != settingsData.oldSystemPassword) {

win.webContents.send("edi t-data-settings", {

success:

true,

password did not match.";

}

if (

throw "Old

settingsData.newSystemP },

assword != } settingsData.confirmNewSystemPasswo );

rd

) {

throw "Your new password did not match.";

}

if (settingsData.newSystemPassword.leng th < 8) {

throw "Password must be 8 characters long.";

}

await SettingsModel.updateOne(

{ uid: "time

is gold" },

{

$set: {

systemPassword: settingsData.newSystemPassword,

win.webContents.send("edi t-data-password", {

success:

true,

message: "Password saved successfully.",

});

} catch (err) {

win.webContents.send("edi t-data-password", {

success:

false,

message:

err,

});

}

});

ipcMain.on("delete-schedule-data", async (event, tbdData) => {

try {

await ScheduleModel.deleteOne({

\_id:

tbdData,

});

console.log("Testing");

win.webContents.send("del ete-schedule-data", {

success:

true,

message:

"Deleted Successfully.",

});

} catch (err) {

console.log(err);

win.webContents.send("del ete-schedule-data", {

success:

false,

message: err,

});

}

});

ipcMain.on("delete- professor-data", async (event, selectedData) => {

try {

await ScheduleModel.deleteMany({

"professorData.\_id": selectedData,

});

await ProfessorModel.deleteOne({

\_id:

selectedData,

});

win.webContents.send("del ete-professor-data", {

success:

true,

message: "Deleted Successfully.",

});

} catch (err) {

win.webContents.send("del

await ProgramModel.deleteOne({

selectedData,

});

\_id:

ete-professor-data", {

false,

err,

}

});

});

success:

message:

win.webContents.send("del ete-program-data", {

success:

true,

message:

"Deleted Successfully.",

});

} catch (err) {

ipcMain.on("delete-

program-data", async (event, selectedData) => {

try {

await ScheduleModel.deleteMany({

"programData.\_id": selectedData,

});

win.webContents.send("del ete-program-data", {

success:

false,

message:

err,

});

}

});

ipcMain.on("delete-room-data", async (event, selectedData) => {

try {

await ScheduleModel.deleteMany({

"roomData.\_id": selectedData,

});

await RoomModel.deleteOne({

\_id:

selectedData,

});

win.webContents.send("del ete-room-data", {

success:

true,

message:

"Deleted Successfully.",

});

} catch (err) {

win.webContents.send("del ete-room-data", {

success:

false,

message:

err,

});

}

});

ipcMain.on("delete-course- data", async (event, selectedData) => {

try {

await ScheduleModel.deleteMany({

"courseData.\_id": selectedData,

});

await CourseModel.deleteOne({

\_id:

selectedData,

});

win.webContents.send("del ete-course-data", {

success:

true,

password,

systemPassword:

});

if

"Deleted Successfully.",

});

} catch (err) {

message:

(passwordExist.length <= 0) throw "Wrong password";

win.webContents.send("log

win.webContents.send("del ete-course-data", {

success:

false,

message:

err,

});

}

in-validate", {

true,

"Logging in...",

});

} catch (err) {

success:

message:

}); win.webContents.send("log

in-validate", {

ipcMain.on("login-validate", async (event, password) => {

try {

let passwordExist

= await SettingsModel.find({

false,

err,

});

success:

message:

}

});

ipcMain.on("login-reset",

await RoomModel.remove({});

async (event) => {

try {

await ScheduleModel.remove({});

await SettingsModel.updateOne(

{ win.webContents.send("log

uid: in-reset", {

"time is gold", success:

}, true,

{

message:

"Your password has been changed to:

$set: { admin",

});

systemPassword: "admin",

},

}

);

in-reset", {

} catch (err) {

win.webContents.send("log

success:

await CourseModel.remove({});

await ProfessorModel.remove({});

await ProgramModel.remove({});

false,

err,

}

});

});

message:

ipcMain.on("export-to- excel", async (event, selectedData) => {

console.log(selectedData); let savePathSelected =

await dialog.showSaveDialog({

filters: [{ name: "Excel file", extensions: ["xlsx"] }],

});

console.log(savePathSelec

ted);

console.log(path.extname( savePathSelected.filePath));

console.log(path.parse(sav ePathSelected.filePath));

if (!savePathSelected.canceled) {

let scheduleData = await ScheduleModel.find({

"professorData.\_id": selectedData.\_id,

});

console.log(scheduleData);

let option = {

margins: {

left:

1.5,

right: 1.5,

},

sheetFormat: {

baseColWidth: 12,

defaultRowHeight: 22,

},

};

let wb = new

xl.Workbook();

let ws = wb.addWorksheet(

`${selectedData.name.last}

, ${selectedData.name.first}`,

option

);

let header =

wb.createStyle({

font: {

size: 11, {

bold: true,

},

alignment:

{

horizontal: ["center"],

vertical: ["center"],

wrapText: true,

},

fill: {

type: "pattern",

patternType: "solid",

bgColor: "#C2D59A",

fgColor: "#C2D59A",

},

border: {

left:

style: "thin",

color: "#000000",

},

right: {

style: "thin",

color: "#000000",

},

top: {

style: "thin",

color: "#000000",

},

bottom: {

style: "thin",

color: "#000000",

},

},

});

ws.cell(3, 1, 5, 1, true).style(header).string("7:30AM - 9:00AM");

ws.cell(6, 1, 8, 1, true).style(header).string("9:00AM - 10:30AM");

ws.cell(9, 1, 11, 1, true).style(header).string("10:30AM - 12:00PM");

ws.cell(12, 1, 14,

1, true).style(header).string("12:00PM - 1:30PM");

ws.cell(15, 1, 17,

1, true).style(header).string("1:30PM - 3:00PM");

ws.cell(18, 1, 20,

true).string(

ws.cell(1, 1, 1, 8,

`${selectedData.name.last}

1, true).style(header).string("3:00PM - 4:30PM");

ws.cell(21, 1, 23,

1, true).style(header).string("4:30PM -

, ${selectedData.name.first}`

);

6:00PM");

ws.cell(24, 1, 26,

1).style(header);

ws.cell(2,

1, true).style(header).string("6:00PM - 7:30PM");

ws.cell(27, 1, 29, 1, true).style(header).string("7:30PM - 9:00PM");

ws.cell(2, 2).style(header).string("Monday");

ws.cell(2, 3).style(header).string("Tuesday");

ws.cell(2, 4).style(header).string("Wednesday");

ws.cell(2, 5).style(header).string("Thursday");

ws.cell(2, 6).style(header).string("Friday");

ws.cell(2, 7).style(header).string("Saturday");

ws.cell(2, 8).style(header).string("Sunday");

for (let i = 2; i <= 8;

i++) {

for (let j =

3; j <= 29; j++) {

ws.cell(j, i).style({

border: {

left: {

},

right: {

},

top: {

},

style: "thin",

color: "#000000",

style: "thin",

color: "#000000",

style: "thin",

color: "#000000",

bottom: {

style: "thin",

wrapText: true,

bold: true,

},

color: "#000000", };

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | },  }, |  | }); | let mondayLoad =  0;  let tuesdayLoad =  0; |
|  |  | } | let |
|  | } |  | wednesdayLoad = 0; |
|  |  |  | let thursdayLoad = |
|  | let | scheduleStyle | 0; |
| = { |  |  |  | let fridayLoad = 0; |
|  |  |  | font: { | let saturdayLoad = |
|  |  |  |  | 0; |
|  | size: 11, |  |  | let sundayLoad = |
|  |  |  | }, | 0; |
|  |  |  | alignment: |  |
| { |  |  |  |  |
|  |  |  |  | scheduleData.forEach((sch |

horizontal: ["center"], edule) => {

let x1 =

vertical: ["center"], schedule.day + 2;

let x2 = schedule.day + 2;

let y1, y2;

switch (

parseInt(`${schedule.time.s tart.hours}${schedule.time.start.minutes}

`)

) {

case 730:

y1 = 3;

break;

case 800:

y1 = 4;

break;

case 830:

y1 = 5;

break;

case 900:

y1 = 6;

break;

case 930:

y1 = 7;

break;

case 1000:

y1 = 8;

break;

case 1030:

y1 = 9;

break;

case 1100:

y1 = 10;

break;

case 1130:

y1 = 11;

break;

case 1200:

y1 = 12;

break;

case 1230:

y1 = 13;

break;

case 1300:

y1 = 14;

break;

case 1330:

y1 = 15;

break;

case 1400:

y1 = 16;

break;

case 1430:

y1 = 17;

break;

case 1500:

y1 = 18;

break;

case 1530:

y1 = 19;

break;

case 1600:

y1 = 20;

break;

case 1630:

y1 = 21;

break;

case 1700:

y1 = 22;

break;

case 1730:

y1 = 23;

break;

case 1800:

y1 = 24;

break;

case 1830:

y1 = 25;

break;

case 1900:

y1 = 26;

break;

case 1930:

y1 = 27;

break;

case 2000:

y1 = 28;

break;

case 2030:

y1 = 29;

break;

default:

y1 = 29;

break;

end.hours}${schedule.time.end.minutes}

`)

) {

case 730:

y2 = 3;

break;

case 800:

y2 = 3;

break;

case 830:

y2 = 4;

break;

}

switch (

parseInt(`${schedule.time.

case 900:

y2 = 5;

break;

case 930:

y2 = 6;

break;

case 1000:

y2 = 7;

break;

case 1030:

y2 = 8;

break;

case 1100:

y2 = 9;

break;

case 1130:

y2 = 10;

break;

case 1200:

y2 = 11;

break;

case 1230:

y2 = 12;

break;

case 1300:

y2 = 13;

break;

case 1330:

y2 = 14;

break;

case 1400:

y2 = 15;

break;

case 1430:

y2 = 16;

break;

case 1500:

y2 = 17;

break;

case 1530:

y2 = 18;

break;

case 1600:

y2 = 19;

break;

case 1630:

y2 = 20;

break;

case 1700:

y2 = 21;

break;

case 1730:

y2 = 22;

break;

case 1800:

y2 = 23;

break;

case 1830:

y2 = 24;

break;

case 1900:

y2 = 25;

y2 = 27;

break;

case 2030:

y2 = 28;

break;

case 2100:

y2 = 29;

break;

}

break;

switch

case 1930:

y2 = 26;

break;

case 2000:

(schedule.day) {

case 0:

mondayLoad =

mondayLoad +

parseInt(schedule.courseData.units);

break;

case 1:

tuesdayLoad =

tuesdayLoad +

parseInt(schedule.courseData.units);

break;

case 2:

wednesdayLoad = wednesdayLoad +

parseInt(schedule.courseData.units);

break;

case 3:

thursdayLoad =

thursdayLoad +

parseInt(schedule.courseData.units);

break;

case 4:

fridayLoad = fridayLoad + parseInt(schedule.courseData.units);

break;

case 5:

saturdayLoad =

saturdayLoad +

parseInt(schedule.courseData.units);

break;

case 6:

sundayLoad = sundayLoad

+ parseInt(schedule.courseData.units);

break;

}

console.log(schedule);

console.log(y1, x1, y2, x2);

ws.cell(y1,

x1, y2, x2, true)

.style(scheduleStyle)

.string(

`${schedule.courseData.co de} ${schedule.courseData.name} ―

${schedule.programData.acronym}

${schedule.programData.year}-

${schedule.programData.section} ―

${schedule.roomData.name}`

);

});

console.log(

mondayLoad +

tuesdayLoad +

wednesdayLoad +

thursdayLoad +

fridayLoad +

saturdayLoad +

sundayLoad

);

ws.cell(30, 1)

.style({

font: { bold: true } })

.number(

mondayLoad +

tuesdayLoad +

wednesdayLoad +

thursdayLoad +

fridayLoad +

saturdayLoad +

sundayLoad

);

.number(fridayLoad);

ws.cell(30, 7)

ws.cell(30, 2)

.style({

font: { bold: true } })

.style({

font: { bold: true } })

.number(saturdayLoad);

.number(mondayLoad);

ws.cell(30, 3)

.style({

font: { bold: true } })

font: { bold: true } })

ws.cell(30, 8)

.style({

.number(sundayLoad);

.number(tuesdayLoad);

ws.cell(30, 4)

.style({ wb.write(savePathSelected

font: { bold: true } }) .filePath);

}

.number(wednesdayLoad);

ws.cell(30, 5)

.style({

font: { bold: true } })

.number(thursdayLoad);

ws.cell(30, 6)

.style({

font: { bold: true } })

});

ipcMain.on("export-to- excel-room", async (event, selectedData) => {

console.log(selectedData); let savePathSelected =

await dialog.showSaveDialog({

filters: [{ name: "Excel file", extensions: ["xlsx"] }],

});

let option = {

margins: {

left:

1.5,

console.log(savePathSelec

ted); right: 1.5,

},

console.log(path.extname(

savePathSelected.filePath));

console.log(path.parse(sav ePathSelected.filePath));

if (!savePathSelected.canceled) {

let scheduleData = await ScheduleModel.find({

"roomData.\_id": selectedData.\_id,

});

console.log(scheduleData);

sheetFormat: {

baseColWidth: 12,

defaultRowHeight: 22,

},

};

let wb = new

xl.Workbook();

let ws = wb.addWorksheet(`${selectedData.nam e}`, option);

let header =

wb.createStyle({

font: {

size: 11,

bold: true,

left: {

style: "thin",

},

alignment: color: "#000000",

{ },

horizontal: ["center"],

vertical: ["center"],

wrapText: true,

},

fill: {

right: {

style: "thin",

color: "#000000",

},

top: {

type: "pattern",

style: "thin",

patternType: "solid",

color: "#000000",

bgColor: "#C2D59A",

fgColor: "#C2D59A",

},

border: {

},

bottom: {

style: "thin",

color: "#000000",

},

},

});

ws.cell(18, 1, 20, 1, true).style(header).string("3:00PM - 4:30PM");

ws.cell(21, 1, 23,

1, true).style(header).string("4:30PM -

ws.cell(1, 1, 1, 8, true).string(`${selectedData.name}`);

6:00PM");

ws.cell(24, 1, 26,

1).style(header);

ws.cell(2,

ws.cell(3, 1, 5, 1,

1, true).style(header).string("6:00PM - 7:30PM");

ws.cell(27, 1, 29,

true).style(header).string("7:30AM - 9:00AM");

ws.cell(6, 1, 8, 1, true).style(header).string("9:00AM - 10:30AM");

ws.cell(9, 1, 11, 1, true).style(header).string("10:30AM - 12:00PM");

ws.cell(12, 1, 14,

1, true).style(header).string("12:00PM - 1:30PM");

ws.cell(15, 1, 17,

1, true).style(header).string("1:30PM - 3:00PM");

1, true).style(header).string("7:30PM - 9:00PM");

ws.cell(2, 2).style(header).string("Monday");

ws.cell(2, 3).style(header).string("Tuesday");

ws.cell(2, 4).style(header).string("Wednesday");

ws.cell(2, 5).style(header).string("Thursday");

ws.cell(2, 6).style(header).string("Friday");

ws.cell(2, 7).style(header).string("Saturday");

ws.cell(2, 8).style(header).string("Sunday");

for (let i = 2; i <= 8;

i++) {

for (let j =

3; j <= 29; j++) {

ws.cell(j, i).style({

},

top: {

},

style: "thin",

color: "#000000",

border: {

left: {

style: "thin",

color: "#000000",

bottom: {

style: "thin",

color: "#000000",

},

}, },

});

right: { }

}

style: "thin",

let scheduleStyle

color: "#000000", = {

font: {

let fridayLoad = 0;

let saturdayLoad =

size: 11, 0;

},

alignment: 0;

let sundayLoad =

{

horizontal: ["center"], scheduleData.forEach((sch

edule) => {

vertical: ["center"], let x1 =

schedule.day + 2;

wrapText: true, let x2 =

schedule.day + 2;

bold: true, let y1, y2;

},

};

let mondayLoad =

0;

let tuesdayLoad =

0;

switch (

parseInt(`${schedule.time.s tart.hours}${schedule.time.start.minutes}

`)

) {

let

wednesdayLoad = 0; case 730:

let thursdayLoad =

0; y1 = 3;

break;

case 800:

y1 = 4;

break;

case 830:

y1 = 5;

break;

case 900:

y1 = 6;

break;

case 930:

y1 = 7;

break;

case 1000:

y1 = 8;

break;

case 1030:

y1 = 9;

break;

case 1100:

y1 = 10;

break;

case 1130:

y1 = 11;

break;

case 1200:

y1 = 12;

break;

case 1230:

y1 = 13;

break;

case 1300:

y1 = 14;

break;

case 1330:

y1 = 15;

break;

case 1400:

y1 = 16;

break;

case 1430:

y1 = 17;

break;

case 1500:

y1 = 18;

break;

case 1530:

y1 = 19;

break;

case 1600:

y1 = 20;

break;

case 1630:

y1 = 21;

break;

case 1700:

y1 = 22;

break;

case 1730:

y1 = 23;

break;

case 1800:

y1 = 24;

break;

case 1830:

y1 = 25;

break;

case 1900:

y1 = 26;

break;

case 1930:

y1 = 27;

break;

case 2000:

y1 = 28;

break;

case 2030:

y1 = 29;

break;

default:

y1 = 29;

break;

break;

case 830:

y2 = 4;

break;

}

switch (

parseInt(`${schedule.time. end.hours}${schedule.time.end.minutes}

`)

) {

case 730:

y2 = 3;

break;

case 800:

y2 = 3;

case 900:

y2 = 5;

break;

case 930:

y2 = 6;

break;

case 1000:

y2 = 7;

break;

case 1030:

y2 = 8;

break;

case 1100:

y2 = 9;

break;

case 1130:

y2 = 10;

break;

case 1200:

y2 = 11;

break;

case 1230:

y2 = 12;

break;

case 1300:

y2 = 13;

break;

case 1330:

y2 = 14;

break;

case 1400:

y2 = 15;

break;

case 1430:

y2 = 16;

break;

case 1500:

y2 = 17;

break;

case 1530:

y2 = 18;

break;

case 1600:

y2 = 19;

break;

case 1630:

y2 = 20;

break;

case 1700:

y2 = 21;

break;

case 1730:

y2 = 22;

break;

case 1800:

y2 = 23;

break;

case 1830:

y2 = 24;

break;

case 1900:

y2 = 25; break;

}

break;

switch

case 1930:

y2 = 26;

break;

case 2000:

y2 = 27;

break;

case 2030:

y2 = 28;

break;

case 2100:

y2 = 29;

(schedule.day) {

case 0:

mondayLoad =

mondayLoad +

parseInt(schedule.courseData.units);

break;

case 1:

tuesdayLoad =

tuesdayLoad +

parseInt(schedule.courseData.units);

break;

case 2:

wednesdayLoad =

wednesdayLoad +

parseInt(schedule.courseData.units);

break;

case 3:

thursdayLoad =

thursdayLoad +

parseInt(schedule.courseData.units);

break;

case 4:

fridayLoad = fridayLoad + parseInt(schedule.courseData.units);

break;

case 5:

saturdayLoad =

saturdayLoad +

parseInt(schedule.courseData.units);

break;

case 6:

sundayLoad = sundayLoad

+ parseInt(schedule.courseData.units);

break;

}

console.log(schedule);

console.log(y1, x1, y2, x2);

ws.cell(y1,

x1, y2, x2, true)

.style(scheduleStyle)

.string(

`${schedule.courseData.co de} ${schedule.courseData.name} ―

${schedule.programData.acronym}

${schedule.programData.year}-

${schedule.programData.section} ―

${schedule.professorData.name.last},

${schedule.professorData.name.first}`

);

});

console.log(

mondayLoad +

tuesdayLoad +

wednesdayLoad +

thursdayLoad +

fridayLoad +

saturdayLoad +

true } })

.style({ font: { bold:

.number(

mondayLoad +

tuesdayLoad +

wednesdayLoad +

thursdayLoad +

fridayLoad +

saturdayLoad +

sundayLoad

);

ws.cell(30, 2)

.style({

font: { bold: true } })

sundayLoad

);

ws.cell(30, 1)

.number(mondayLoad);

ws.cell(30, 3)

.style({ font: { bold: true } }) .style({ font: { bold:

true } })

.number(tuesdayLoad);

ws.cell(30, 4)

.style({

.number(sundayLoad);

font: { bold: true } })

wb.write(savePathSelected

.number(wednesdayLoad);

ws.cell(30, 5)

.style({

font: { bold: true } })

.filePath);

}

});

ipcMain.on("export-to-

excel-program", async (event,

.number(thursdayLoad);

ws.cell(30, 6)

.style({

font: { bold: true } })

.number(fridayLoad);

ws.cell(30, 7)

.style({

font: { bold: true } })

.number(saturdayLoad);

ws.cell(30, 8)

selectedData) => {

console.log(selectedData); let savePathSelected =

await dialog.showSaveDialog({

filters: [{ name: "Excel file", extensions: ["xlsx"] }],

});

console.log(savePathSelec

ted);

console.log(path.extname( savePathSelected.filePath));

console.log(path.parse(sav ePathSelected.filePath));

if (!savePathSelected.canceled) {

let scheduleData = await ScheduleModel.find({

"programData.\_id": selectedData.\_id,

});

console.log(scheduleData);

let option = {

margins: {

left:

1.5,

right: 1.5,

},

sheetFormat: {

baseColWidth: 12,

defaultRowHeight: 22,

},

};

let wb = new

xl.Workbook();

let ws =

wb.addWorksheet(

`${selectedData.acronym}

${selectedData.year}-

${selectedData.section}`,

option

);

let header =

wb.createStyle({

font: {

size: 11,

bold: true, style: "thin",

},

alignment: color: "#000000",

{ },

horizontal: ["center"],

vertical: ["center"],

wrapText: true,

},

fill: {

right: {

style: "thin",

color: "#000000",

},

top: {

type: "pattern",

style: "thin",

patternType: "solid",

color: "#000000",

bgColor: "#C2D59A",

fgColor: "#C2D59A",

},

border: {

left:

{

},

bottom: {

style: "thin",

color: "#000000",

},

},

true).string(

});

ws.cell(1, 1, 1, 8,

ws.cell(15, 1, 17,

1, true).style(header).string("1:30PM - 3:00PM");

ws.cell(18, 1, 20,

1, true).style(header).string("3:00PM - 4:30PM");

`${selectedData.name}

${selectedData.year}-

${selectedData.section}`

);

ws.cell(2,

1).style(header);

ws.cell(3, 1, 5, 1, true).style(header).string("7:30AM - 9:00AM");

ws.cell(6, 1, 8, 1, true).style(header).string("9:00AM - 10:30AM");

ws.cell(9, 1, 11, 1, true).style(header).string("10:30AM - 12:00PM");

ws.cell(12, 1, 14,

1, true).style(header).string("12:00PM - 1:30PM");

ws.cell(21, 1, 23,

1, true).style(header).string("4:30PM - 6:00PM");

ws.cell(24, 1, 26,

1, true).style(header).string("6:00PM - 7:30PM");

ws.cell(27, 1, 29,

1, true).style(header).string("7:30PM - 9:00PM");

ws.cell(2, 2).style(header).string("Monday");

ws.cell(2, 3).style(header).string("Tuesday");

ws.cell(2, 4).style(header).string("Wednesday");

ws.cell(2, 5).style(header).string("Thursday");

ws.cell(2, 6).style(header).string("Friday");

ws.cell(2, 7).style(header).string("Saturday");

ws.cell(2, 8).style(header).string("Sunday");

for (let i = 2; i <= 8;

i++) {

for (let j =

3; j <= 29; j++) {

ws.cell(j, i).style({

},

top: {

},

style: "thin",

color: "#000000",

style: "thin",

color: "#000000",

border: {

left: {

style: "thin",

color: "#000000",

bottom: {

style: "thin",

color: "#000000",

},

}, },

});

right: { }

} let

wednesdayLoad = 0;

let scheduleStyle let thursdayLoad =

= { 0;

font: {

let fridayLoad = 0; let saturdayLoad =

size: 11, 0;

},

alignment: 0;

let sundayLoad =

{

horizontal: ["center"], scheduleData.forEach((sch

edule) => {

vertical: ["center"], let x1 =

schedule.day + 2;

wrapText: true, let x2 =

schedule.day + 2;

bold: true, let y1, y2;

},

};

let mondayLoad =

0;

let tuesdayLoad =

0;

switch (

parseInt(`${schedule.time.s tart.hours}${schedule.time.start.minutes}

`)

) {

case 730:

y1 = 3;

break;

case 800:

y1 = 4;

break;

case 830:

y1 = 5;

break;

case 900:

y1 = 6;

break;

case 930:

y1 = 7;

break;

case 1000:

y1 = 8;

break;

case 1030:

y1 = 9;

break;

case 1100:

y1 = 10;

break;

case 1130:

y1 = 11;

break;

case 1200:

y1 = 12;

break;

case 1230:

y1 = 13;

break;

case 1300:

y1 = 14;

break;

case 1330:

y1 = 15;

break;

case 1400:

y1 = 16;

break;

case 1430:

y1 = 17;

break;

case 1500:

y1 = 18;

break;

case 1530:

y1 = 19;

break;

case 1600:

y1 = 20;

break;

case 1630:

y1 = 21;

break;

case 1700:

y1 = 22;

break;

case 1730:

y1 = 23;

break;

case 1800:

y1 = 24;

break;

case 1830:

y1 = 25;

break;

case 1900:

y1 = 26;

break;

case 1930:

y1 = 27;

break;

case 2000:

y1 = 28;

break;

case 2030:

y1 = 29;

break;

default:

y1 = 29;

break;

case 800:

y2 = 3;

break;

case 830:

y2 = 4;

break;

}

switch (

parseInt(`${schedule.time. end.hours}${schedule.time.end.minutes}

`)

) {

case 730:

y2 = 3;

break;

case 900:

y2 = 5;

break;

case 930:

y2 = 6;

break;

case 1000:

y2 = 7;

break;

case 1030:

y2 = 8;

break;

case 1100:

y2 = 9;

break;

case 1130:

y2 = 10;

break;

case 1200:

y2 = 11;

break;

case 1230:

y2 = 12;

break;

case 1300:

y2 = 13;

break;

case 1330:

y2 = 14;

break;

case 1400:

y2 = 15;

break;

case 1430:

y2 = 16;

break;

case 1500:

y2 = 17;

break;

case 1530:

y2 = 18;

break;

case 1600:

y2 = 19;

break;

case 1630:

y2 = 20;

break;

case 1700:

y2 = 21;

break;

case 1730:

y2 = 22;

break;

case 1800:

y2 = 23;

break;

case 1830:

y2 = 24;

break;

case 1900:

y2 = 25;

case 2100:

y2 = 29;

break;

}

break;

switch

case 1930:

y2 = 26;

break;

case 2000:

y2 = 27;

break;

case 2030:

y2 = 28;

break;

(schedule.day) {

case 0:

mondayLoad =

mondayLoad +

parseInt(schedule.courseData.units);

break;

case 1:

tuesdayLoad =

tuesdayLoad +

parseInt(schedule.courseData.units);

break;

case 2:

wednesdayLoad = wednesdayLoad +

parseInt(schedule.courseData.units);

break;

case 3:

thursdayLoad =

thursdayLoad +

parseInt(schedule.courseData.units);

break;

case 4:

fridayLoad = fridayLoad + parseInt(schedule.courseData.units);

break;

case 5:

saturdayLoad =

saturdayLoad +

parseInt(schedule.courseData.units);

break;

case 6:

sundayLoad = sundayLoad

+ parseInt(schedule.courseData.units);

break;

}

console.log(schedule);

console.log(y1, x1, y2, x2);

ws.cell(y1,

x1, y2, x2, true)

.style(scheduleStyle)

.string(

`${schedule.courseData.co de} ${schedule.courseData.name} ―

${schedule.roomData.name} ―

${schedule.professorData.name.last},

${schedule.professorData.name.first}`

);

});

console.log(

ws.cell(30, 1)

font: { bold: true } })

mondayLoad +

tuesdayLoad +

.style({

.number(

wednesdayLoad +

mondayLoad +

thursdayLoad +

tuesdayLoad +

fridayLoad +

wednesdayLoad +

saturdayLoad +

thursdayLoad +

sundayLoad

fridayLoad +

saturdayLoad +

);

ws.cell(30, 2)

.style({

font: { bold: true } })

sundayLoad

); .number(mondayLoad);

ws.cell(30, 3)

.style({ font: { bold: true }

})

.number(tuesdayLoad);

ws.cell(30, 4)

.style({

.number(fridayLoad);

ws.cell(30, 7)

.style({

font: { bold: true } })

.number(saturdayLoad);

font: { bold: true } })

.number(wednesdayLoad);

ws.cell(30, 5)

font: { bold: true } })

ws.cell(30, 8)

.style({

.style({ .number(sundayLoad);

font: { bold: true } })

.number(thursdayLoad);

ws.cell(30, 6)

.style({

font: { bold: true } })

.filePath);

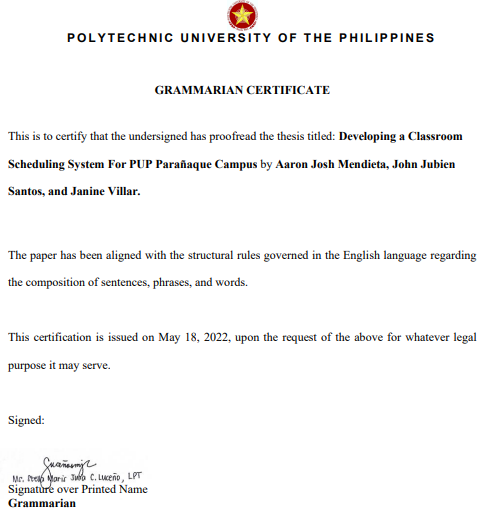
wb.write(savePathSelected

}

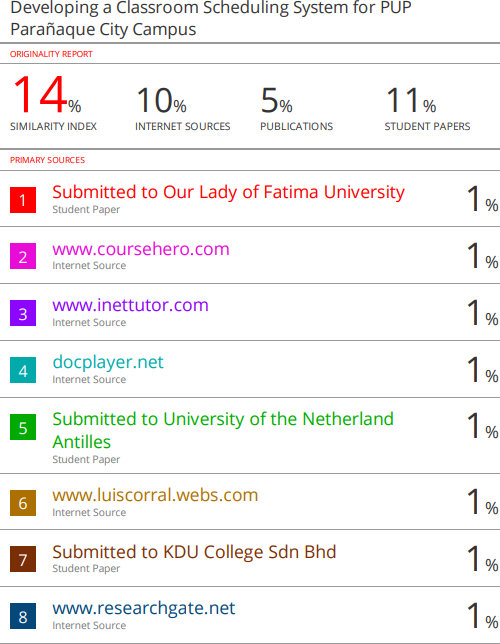
});

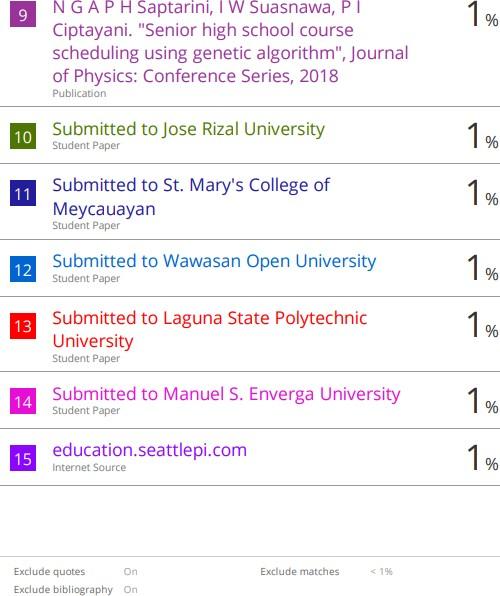
};

**Appendix 8 Certificate from Grammarian**



**Appendix 9 Plagiarism Check Result**





**Appendix 10 Biographical Statement**



#### Mendieta, Aaron Josh F.

21 Medalla Milagrosa St. SAV3 Sucat, Paraɦaque City Contact No. : 0997-240-5320

Email : [aaronjoshfmendieta@gmail.com](mailto:aaronjoshfmendieta@gmail.com)

Aaron Josh F. Mendieta, is a student of Bachelor of Science in Information Technology at the Polytechnic University of the Philippines. He received his Bachelors Degree in Information Technology at the PUP Parañaque Campus in the year 2021-2022. His recent work with his group mates, is developing a Classroom Scheduling System that has been implemented to the school to help the campus to organize its schedules for the professors.



#### Santos, John Jubien A.

Blk 6 Lot 18, Phase 1 Anx B Carmel Village Majada Out Calamba, Laguna City

Contact No. : 0999-334-8003

Email : [jubien15.js@gmail.com](mailto:jubien15.js@gmail.com)

John Jubien A. Santos, a graduate student from Polytechnic University of the Philippines Parañaque Campus, with the program of Bachelor in Science in Information Technology. His current research is about Classroom Scheduling System, which helps the school to organize it scheduling of the classroom



#### Villar, Janine D.

51 Street Ninoy Aquino Ave Sto Niño Paranaque City Contact No. : 0976-067-8355

Email : [j](mailto:jubien15.js@gmail.com)[villarjanine05@gmail.com](mailto:villarjanine05@gmail.com)

Janine D. Villar was a graduate student from Polytechnic University of the Philippines Parañaque Campus, with the program of Bachelor in Science in Information Technology. Her current research is on a Classroom Scheduling System, which she and her group developed and implemented to assists the school in organizing classroom scheduling.