ASSUMPTION UNIVERSITY

Vincent Mary School of Science and Technology

CS3200

SENIOR PROJECT I

FINAL REPORT

ARMS: Assumption University Room Management System

Submitted By:	**Mr. Grid Kornsutatipkul**	**601-3732**
	Mr. Thanathas Chawengvorakul	**601-4586**
	Mr. Sokvathara Lin	**601-8002**

Project Advisor: **Dr. Songsak Channarukul**

Project Committee: **Chayapol Moemeng**

Asst.Prof.Dr. Paitoon Porntrakoon

Vincent Mary School of Science and Technology

Senior Project Approval

Project Title:	ARMS: Assumption University Room Management System
Members:	Mr. Grid Kornsutatipkul
	Mr. Thanathas Chawengvorakul
	Mr. Sokvathara Lin
Advisor:	Dr. Songsak Channarukul
Academic Year:	2/2019

The Department of Computer Science, Vincent Mary School of Science and Technology of Assumption University has approved the final report of the three credits course; the course title is CS3200 Senior Project I.

This senior project report is submitted in partial fulfillment of requirements of the degree of Bachelor of Science in Computer Science/Information Technology.

Approval Committee:

_____

Dr. Songsak Channarukul

(Project Advisor)

_____ _____

A. Chayapol Moemeng **Asst.Prof.Dr. Paitoon Porntrakoon**

(Committee Member) (Committee Member)

Acknowledgement

First and foremost, we cannot express enough gratitude to our advisor Dr. Songsak Channarukul for his continued reinforcement and encouragement. We offer our sincere appreciation for his recommendations and guidance. From the very start of this project, Dr. Songsak persistently assists us in many aspects until the completion of the project. We appreciate his patience, inspiration, and enthusiasm for our project.

We also would like to thank our project committee. Our completion of this project could not have been accomplished without the suggestions of the committee, A. Chayapol Moemeng, and Ass.Prof.Dr. Paitoon Porntrakoon. Their recommendations produce significant impacts on our project.

Finally, to our caring and supportive families. The encouragement from them when the times got rough is much appreciated and noted. It was a great comfort and relief to know that they were willing to provide management of our household activities while we completed our work. Our heartfelt thanks.

Abstract

Nowadays, automated systems are taking roles in assisting human life. It can perform many tasks in various kinds of field base on what it has programmed. Assumption University Room Management System or in short called ARMS. It can dramatically improve the performance of examination seat arrangement and as a result of saving more time and cost consumed. Moreover, it also diminishes the human errors which might occur during the process. In many universities' rooms, buildings, papers, documents, and electricity are heavily consumed during the examination period. Without proper management and allocating these resources, it might further raise the unnecessary expense of the university.

ARMS is a kind of automated system, which can automatically perform an examination seating arrangement. It provides the necessary features to automatically arrange all the eligible students to the specific seats in examination rooms easily. Moreover, the useful insights and reports of how the system arranged will also be provided. Furthermore, users can also modify and manually arrange some specialized courses of their desire. In conclusion, the ARMS system is almost a one-stop-solution for the examination seating management.

Table of Contents

List of Figures

Chapter 1

Introduction

1.1. Motivation

Examination Arrangement Process has always been the time-consuming process in educational institutes for a very long time. Especially, large institutions like universities or colleges could have numerous students, rooms, and facilities to be considered, which leads to a long process and waste of human resources.

Assumption University of Thailand's existing system will be discussed as an example of the traditional examination arrangement process as the following. First, the necessary data must be collected and validated before the arrangement process. For instance, the data of students eligible for examination, available examination rooms, courses, sections, and time slots for each course which should be pre-defined. Second, the arrangement process is done in the form of paper works. The rows of the table of arrangement show the list of the courses, as in Figure 1.1.

On the other hand, all the available rooms will be listed as columns. The capacity of each room is listed below the room's name. The numbers of students eligible for the examination are listed next to each course name. The assignment of seats used will be filled in the box at the coordinate of each room where the staff desire.

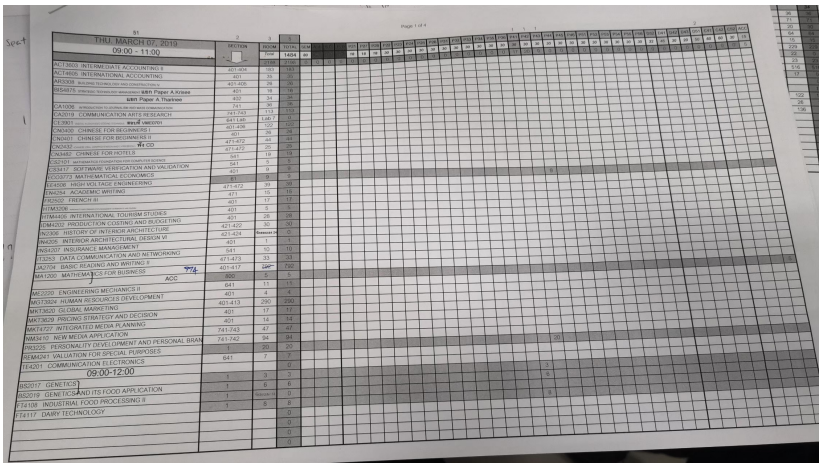

Figure 1.1: Section's seats allocation paper

After the arrangement process in the form of paper was done, the arrangement result will need to be input into a computer program manually. It was done just for the further uses in a printing process of examination arrangement results. It doubles the works for the staff because they already made the arrangement once in the paper.

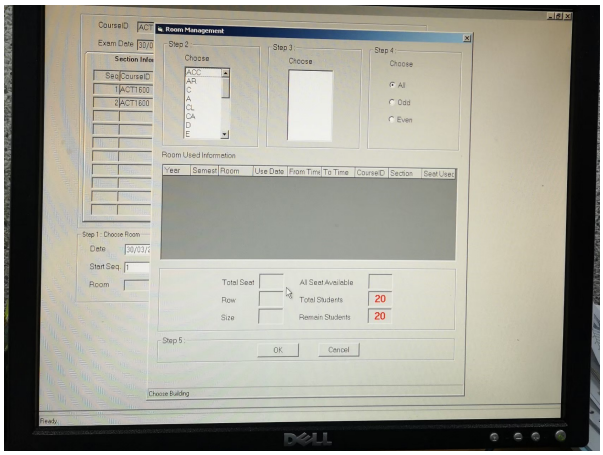

Figure 1.2: Examination Seat allocation program

Three main things should be printed for an announcement and acknowledgment to students that will take exams. First, the room announcement displays the list of courses with its sections and for an individual section where students should take exams shown as in Figure 1.3. Second, the seat announcement for each examination room shows the locations of the seat where an individual student needs to sit. Lastly, the exam paper envelope's sticker will be pasted in front of every exam paper envelopes to inform the proctors where and when the exams in the envelope will be taken.

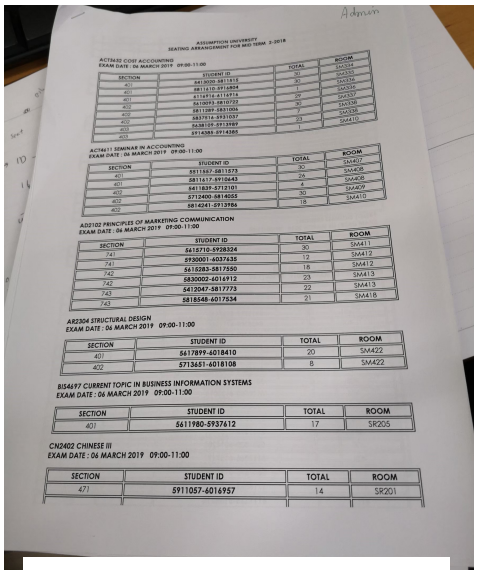

Figure 1.4: Room announcement

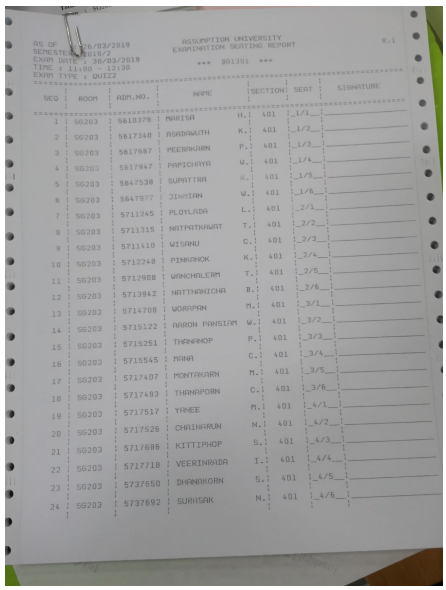

Figure 1.3: Seat announcement and attendance sheet

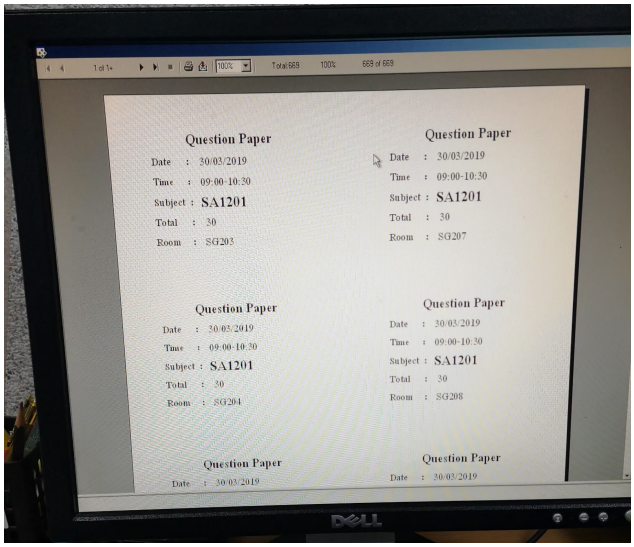

Figure 1.5: Exams paper envelope's sticker

As a result, the efficiency and quality of the mentioned procedure can become questionable due to the following disadvantages. First, when the arrangement must be made manually, it takes much time for preparing and assigning seats for each course in an examination. Moreover, maintaining the paperwork is complicated, and it will be more difficult according to the number of students and facilities constraints. Importantly, the whole mentioned process requires much time and workforce. The whole process needed to be restarted again for each examination period.

1.2. Overview of ARMS

As our system's full name, the program was developed for Assumption University to have a more efficient way to manage their rooms for an examination. With better room management, it would result in significant reduction of electricity and papers consumed. Most Importantly, the responsible staff in the examination will have to do significantly fewer works.

For ARMS to allocate the rooms for each examination, there are a few required data that need to be prepared. First, data of all the examination courses, including sections and the number of students. Second, the available rooms for an examination to use with its examination capacity and layout. Third, the arranged examination date and time for each course, which means our systems are excluded from managing the examination time arrangement. Finally, the student's information with the courses they take and the status of each course, whether they are eligible to take an exam.

1.3. Scope

ARMS will only arrange the examination seats based on the pre-defined time slot of each course. Moreover, the essential criteria of the system are to group the sections of the same courses to take exams next to each other for proctors to distribute the exam papers efficiently. Nevertheless, the benefit of using this system over the existing process is allocating the seats with the smallest number of buildings and rooms used. Lastly, this system will also generate a summarized report for further print out the rooms and seats announcements.

Chapter 2

Background

2.1. Examination Seating Arrangement System

There are many examination seating arrangement systems that exist nowadays. All of the similar systems are sharing the same concept. It programmatically allocates all the available seats to the eligible students based on the criteria and given constraints. The primary purpose of the system is to minimize the errors in the arrangement and the waste of resources in terms of physical and human resources.

2.2. Related Works

Many institutions have been using a similar system for a long time. Facing the same problems, many researchers have tried to develop solutions for this particular task and used them in the real application.

2.2.1. A case study at Universidade de Lisboa

The paper from Universidade de Lisboa [1] has mentioned about the timetabling problems in two categories: Examination Timetabling and Course Timetabling problem. These categories are characterized in the following paragraph.

Examination Timetabling focuses on creating the timetables for the examination. These timetables must ensure that students can attend all examinations for which they enrolled in each semester. Course Timetabling can be separated into two types. The first type is curriculum-based course timetabling. It focuses on creating timetable based on a pre-defined curriculum from the institute that students must follow. Another is post-enrollments timetabling. It deals with creating timetable based on the students' enrollment. Curriculum-based course timetabling has chance to product examination time conflict – one student has to take more than one examination at the same time, than the post-enrollments timetabling. Post-enrollments timetabling takes the students' enrollment and designs the examination timetables that would reduce the number of students with examination time conflict as much as possible [1].

The system proposed by Universidade de Lisboa did provide the solution for the timetabling process to prevent the examination time conflict. They also implemented the solution to minimize room

utilization. However, the solution is not suitable for the ARMS's problem because of the Hard Constraints and Soft Constraints. Constraints are requirements that either must be satisfied or should be satisfied to increase the quality of work. The constraints are going to be described in Chapter 4: Algorithm.

2.2.2. Research from the International Research Journal of Engineering and Technology

Another worth-mentioning work is the research published in the International Research Journal of Engineering and Technology (IRJET). They have proposed another solution for examination seat arrangement by focusing on optimum uses of seats. They did mention about the constraint of having a distance of students getting the same query set to prevent cheating in the examination, avoiding seat overlapping, and finally, a full seating arrangement that is comfortable both for the students and proctors. In their explanation, they mentioned that there were no fixed rules for examination seating. Different institutions follow the different seat allocation methodology, based on the number of students, seat capacity, environment, and examination type. Mostly, two types of seating methodologies are used for examination seating. These methodologies look like the English letter **'X',** and **'I'** and they are used when two courses have to take an examination in the same room [2].

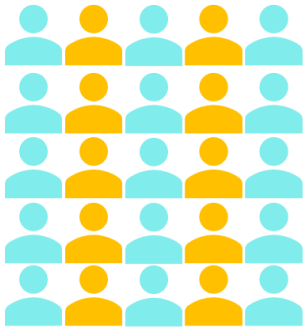

Figure 2.1: I-Methodology

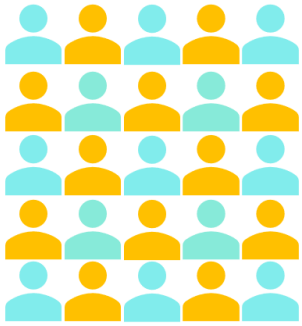

Figure 2.2: X-Methodology

From Figure 2.1 and Figure 2.2, they visualize two types of seating methodologies using Blue and Orange color as two different subjects having an examination in the same room. However, the most common methodology in Thailand would be I-Methodology, and Assumption University of Thailand also uses I-Methodology for exam seating arrangement.

2.3. Algorithms for Examination Assigning

Solutions and researches mentioned in the previous section have several similarities and differences. One of the essential components is algorithms, mathematical models, and approaches they

used to address the problems in the right manner. All the proposed models from researchers were chosen and used by the different researchers' points of view. Each of them has its advantages and disadvantages to be considered. The following section is going to discuss how each algorithm works and the reasons why they were chosen to be used in the problem.

2.3.1. Integer Linear Programming

From the definition [5], Integer Programming Problem seeks to minimize a linear cost function over all n-dimensional vectors to a set of linear equality and inequality constraints. In general, the main objective of Integer Linear Programming is to find the set of possible answers from constraints which were made from linear functions. After it can find the possible sets of answers, all values in the sets will be used to find the most optimal value. Linear functions that are used for constraining are generated from constraints that were set by the problem.

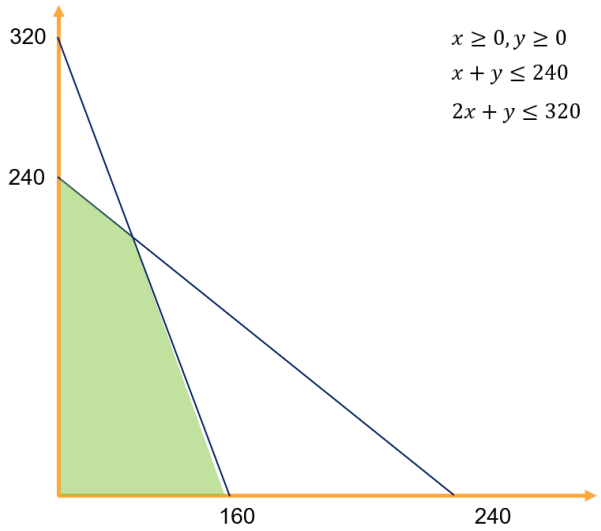

Figure 2.3: Linear Programming Graph

From the Figure 2.3, it has linear constraint functions that came from the requirements or constraints that the problem has. They will be transformed into linear functions and used to find the interested area or answer area. For the sake of explanation, Figure 2.3 also shows graph that is made from linear functions and highlights the area of answer already. The answer can be any real values inside the highlighted area. However, if the problem requires answer to be integer value such as the assigning problem, only integer value within the interested area will be included in the set of answers.

$$\text{maximize}: \sum_{l \in L} \sum_{t_l \in T_l} \sum_{r \in R} seated_{l,r} \times x_{l,r} \times A^l_{d_l,t_l} \quad (7)$$

$$\text{subject to}: \sum_{r \in R} x_{l,r} = 1 \quad \forall_{l \in L} \quad (8)$$

$$\sum_{l \in L} x_{l,r} \times A^l_{d,t} \leq 1 \quad \forall_{r \in R, d \in D, t \in T} \quad (9)$$

$$cap_r \geq x_{l,r} \times (students_l - students_l \times \alpha) \quad \forall_{r \in R, l \in L} \quad (10)$$

Figure 2.4: Linear functions from Universidade de Lisboa

Work from Universidade de Lisboa [1] implements the integer linear programming with their constraints into two steps, but the step that connects to the topic of this section is the first step only. The step is to maximize the number of students seated. As shown in Figure 2.4, objective of function (7) is to maximize the number of students seated. The problem needs two constraints to ensure the correct allocation of lectures. First, it is necessary to ensure that a lecture is allocated into exactly one room in the predefined time and day (function (8)). Constraint (9) is required to ensure that there is at most one lecture in each room in the specific slot. Finally, constraint (10) ensures that, in the worst case, the number of students enrolled that cannot be seated must be as less as possible.

In ARMS program, there are some general constraints that cannot be violated as well such as the number of students in the room cannot be greater than the number of examination seats of that room or the number of total used rooms must not exceed the total number of available rooms for examination in that exam period. Those constraints will be used to create the linear function. And the cost function that used for determining the optimal solution would be the number of rooms used and the number of unused seats from every room that has been assigned to examination. The main objective of this problem is to minimize the room used and the number of unused seats in each room to make sure that all seats will not be wasted.

However, there are some drawbacks from this algorithm. First, the algorithm can be computationally demanding when dealing with large problems such as resource allocation that has many recourses to be considered like the current problem. This drawback was also mentioned in the report from Universidade de Lisboa [1]. That is the reason why they also proposed another approach with greedy algorithm. Second, the algorithm itself is a mathematical-based model which is hard for understanding and maintain. Due to its complexity, it creates difficulties to modify when the program has to include new constraints or requirements. These drawbacks from the algorithm are enough to state that this algorithm is not suitable.

2.3.2. Greedy Algorithm

Greedy Algorithm is an algorithm paradigm that builds up a solution piece by piece, always choosing the next piece that offers the most obvious and immediate benefit [6]. The greedy algorithm is one of the simplest algorithms, its concept is very simple. The main concept of this algorithm is ordering the set of value into the way that when the program runs through the set of information, it will always meet the next best possible answer. With appropriate optimization, it can be used to find solutions for almost any problems with the following heuristic: “At each step of the journey, visit the nearest unvisited city.”

As mentioned that Integer Linear Programming is computationally demanding, the Universidade de Lisboa [1] has proposed this algorithm as an alternative method. They claimed that the result from using the greedy algorithm is faster than Integer Linear Programming when considering large data sets. The greedy algorithm requires less computational effort and be able to deliver a good result. Moreover, there exists research from E.H Kampke, W. de Souza Rocha, M.C.S Boeres, M.C. Rangle in A GRASP [3] algorithm with path relinking for the university course timetabling problem, shows that Greedy heuristics, such as Greedy Randomized Adaptive Search Procedure (GRASP), have successfully been applied to course timetabling, and examination timetabling.

The fact that it can deliver a solution without demanding high computation even in large data sets shows its potential. There is the case where one examination room can have more than one course or section. However, in order to minimize the number of used rooms and maximize the seat used for every room, the program cannot combine any courses together. The number of each course should be suitable and allow the program to arrange students of each course into ‘I’ seating methodology as well.

Chapter 3

ARMS Implementation

This chapter is about the implementation methods and procedures of the ARMS project in both frontend and backend environments. It will also clarify the entire flow of the program from the data that should be prepared until the arrangement result is revealed by the user. The step by step of how to use the ARMS program with its available features will also be discussed in the following sections.

3.1. System Diagram

The ARMS is implemented in microservices architecture which the client program is separated from the API to compute and process the data. So, the logic and queries are not heavily contained in the client program and to be more compatible with the low-performance hardware. The system diagram is shown in Figure 3.1

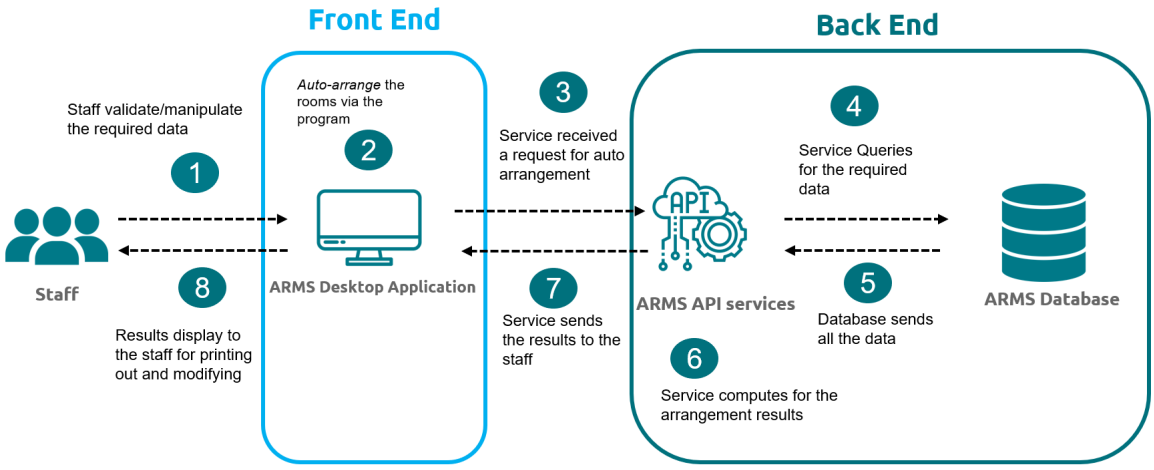

Figure 3.1: System Diagram

3.2. ARMS Data Structure

There are briefly four main categories of data that were stored in the ARMS backend environment. These categories are grouped by the type of data stored. When the arrangement process occurs, there will be a joining of tables between these four categories to generates the seat arrangement result as it was designed to keep all the transactional records for the arrangement. So, when the users reveal the arrangement result, there will not be any further joining across multiple tables. The design and examples of how ARMS stored the data in Figure 3.2.

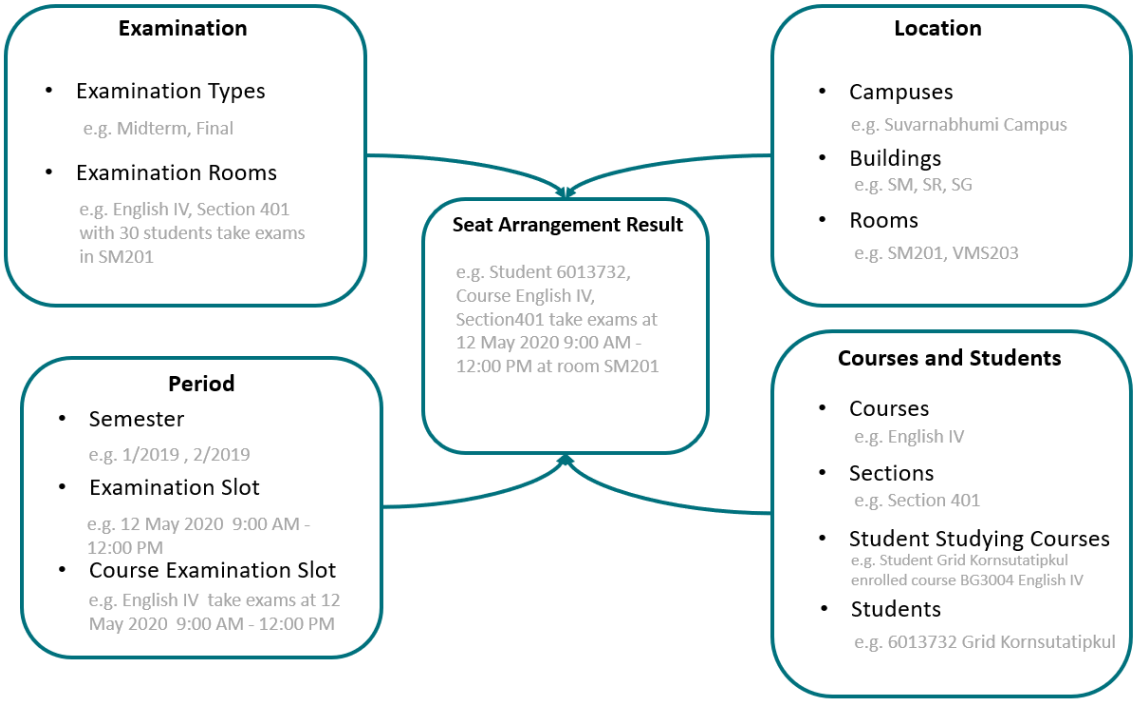

Figure 3.2: Data Structure Diagram

3.3. Functionality

3.3.1. Frontend System Overview

The ARMS App is a comprehensive desktop software solution for Assumption University's Department of Administration. It provides for exam assignment, dissolved assignment, searching, reporting, printing, recording, and administration. Human interaction with the desktop application software takes place via a secure standalone desktop app.

3.3.2. Executive Highlights

There are several functionalities that this desktop app will provide. First, one of the main functions is to provide a full comprehensive manual assign to assign any examination course to examination room(s) with good UX constraints, which prevents human error or ineligible course.

Second, this function is the most important feature which provides conveniently automate assignment to assign all available examination courses to the specific examination rooms that fit the protocol of the Department of Administration's requirement in a specific semester to ease Administrator's workload. Furthermore, it allows ARMS Administrators full access to all the information that they require to do their jobs, including comprehensive errors.

Third, the system shows fully comprehensive master tables as reports to comprehend users' workspace. Furthermore, this system provides flexibility for ARMS Administrators to dissolve any specific assigned examination course or whole assigned examination courses simultaneously in any selected semester that has been enabled to dissolve. As well as, the functionality to browse any specific student's schedule throughout his or her university's admission code. Likewise, ARMS also provide a comprehensive, enable to view a report of the conflict students list in any specific semester without having to do examination courses' assignments.

As showed above, every interaction with the system will issue alerts in any specific user activity, like the task has been completed or incomplete. Again, the system will record every single user's interaction as the system log with comprehensive activities of the specific users do with the system.

3.4. Microservices

3.4.1. Overview

Microservices[8] is a software development technique —a variant of the service-oriented architecture (SOA) structural style— that arranges an application as a collection of loosely coupled services. In a microservices architecture, services are fine-grained, and the protocols are lightweight.

A topic like microservices, it's very hard to come up with any kind of firm definition. This means it is just architecture, and it has many different approaches, which depend on the organizations that define the implementations of each process. However, this project is being implemented with the following architecture, which is a technology that has been adopted by many enterprises around the world, and with this kind of project, it is fit the design pattern to follow up, and it has good practices.

For ARMS web app, the frontend which is the blue unit display in Figure 3.3 will communicate data with its API service which is the orange unit. Again, ARMS API service will be a system

middleware which handles the computation and communication between frontend and data store service which is the red unit. Furthermore, data store service will only handle storing data in different instance. As shown above, each unit will have its own instance and independently process to handle the system tasks that have been defined.

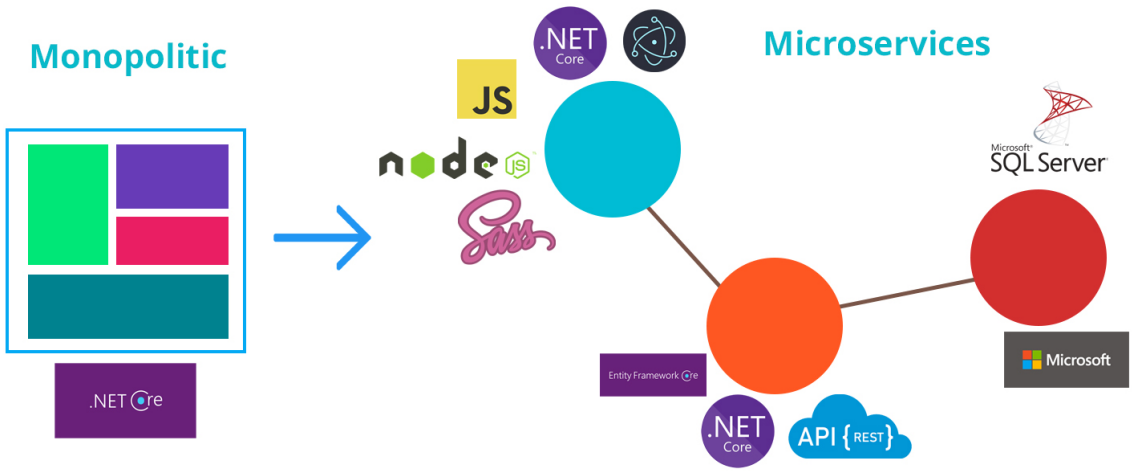

Figure 3.3: ARMS Microservices

3.4.2. Introduction

There are several reasons that this project has been opted for this Microservices Architecture. First, the development of this project can concentrate on small, independent units of working software, which the team can work with partials independently. For instance, in the Frontend part, which at first it was being implemented in Dotnet core 2.2, but it requires to implement as Electron.Net into the Cross-Platform Desktop App frontend, which is why it needs to migrate the whole project into Dotnet 3.1 to fix the Electron.Net SDK issues. Furthermore, when it comes to migration to dotnet 3.1, it is an independent development workspace, which the team do not have to worry about the changes. This separation of services might incur in less conflicts and communication issues. Moreover, a given unit can be implemented in more appropriated programming languages, which enables developers to be full to explore the benefits of other programming languages, frameworks, and tools. For instance, in the frontend part requires many libraries, languages, and frameworks like Nuget, NPM, JSON, C#, and JavaScript. Again, the frontend's unit needs to be deployed to Desktop App as standalone on the Administrator's machine, and for API web service that the team has been working on is going to be deployed on the cloud. Which clouds these days have functions and processes that use for deploying specific services like API Web Service. This benefit fosters rapid releases of features, configuration changes, or bug fixes in a sustainable and efficient way. Furthermore, there is no need to suspend the instance of the unit's process to deploy, to update, to fix, and to refactor at the "Subscription" unit.

Second, since deployments are independent, each process can be scaled independently. If any instance of the unit's process is demanding more than anther unit, such a unit can be scaled up

simultaneously with its need's configuration in an on-demand cloud platform, to handle the increased load. For instance, API Web Service, which can be very demanded tasks and that instance can be scaled up or down and in or out to facilitate the traffic workload of the client-side. Moreover, this kind of approach will also a monetary gain: since each unit does not need to scale up the entire application, and in the workload, which needs a fewer number of visual machines or containers of the instances might be required to meet demand.

As shown above, this project is being implemented with this architecture is not because it's so cool or a trending design architecture. With this practice, it fits the project and requires distributed computing and asynchronous communication. These significant boosts to project's services into various pieces independently, and with the upgradability, so does the deployment. Moreover, this Microservices approach gave the developers more flexibility, and this is important where they can deploy new update CI/CD.

3.5. UX/UI

3.5.1. Main Navigation

Most users tend to browse the application in an "F" pattern display in Figure 3.4, starting from the top left of the page and scanning down, then right. With this approach, users can perceive that important elements like navigations of this App are located across the top left of the page, and the partial pages are paging in the below header right.

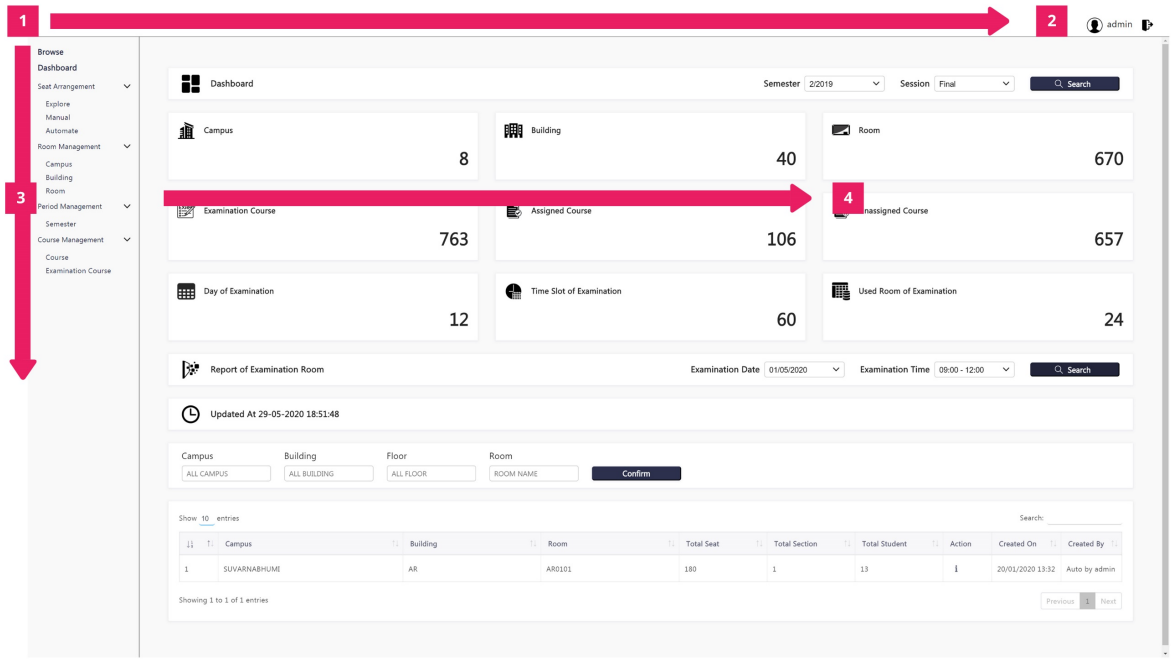

Figure 3.4: Main Navigation

There is an exception to the F Pattern rule in some cases where users know what they are looking for, which some users might abandon the F pattern of search behavior in on specific UI cues. With users that have experience with AU Keystone and other desktop application, this kind of landing pages are very similar to each other, which can indicate to user full perception of the pages' information.

3.5.2. Seamless Transition

Every user wants seamless transitions when interacts with the software. Implementing a seamless transition is very crucial, which can improve the product to be more attractive. In fact, ARMS App is just one main page that has a partial page that keeps paging when the user interacts with other navigations and reports. Also, this kind of design helps users to feel snappy when they interact with every function. There isn't even one page is reloading. Furthermore, the system has one main partial page, which is the page for partially paging. And, reloading animation to enhance the user's perception to alert specifically landing page is processing.

3.5.3. Perception

Design and implementation can have a tremendous effect on how potential user perceives the system, which ARMS App has many designs and patterns that indicate the user to have best experience ever, like having similar user's workflow and user's interaction to the other system like AU Keystone. And Nowadays, there are many desktop applications out there that use navigation patterns, segmented control, landing page, data table, and contextual actions. Moreover, with the content page, ARMS App uses different colored backgrounds to add organizational context and meaning to the sections and table of the report. These visual cues help present the data in a way is inclusive of viewing to understand in every aspect of the context within specific content.

3.5.4. UI

User Interface (UI) is referring to the graphical layout of an application, which consists of many elements and components within the layout or inside the whole application. For instance, the buttons for users to click on, the text that they read, the images, text field to input, sliders, and all other elements and components that the user interacts with. Moreover, it includes screen layout, transitions, interface animations, and every interface's interaction.

3.5.5. Page Section

When it comes to the design page of the UI, the main goal is to prioritize readability and remove any visual clutter that might distract the users' interaction. In ARMS App, every page has sections, and

each section contains context to show user information on that specific page. Moreover, the background in each partial page is using different colors from the section by adding contrast to the page section to help the user while interacting with the page section. First, in the first-page section within the partial page is the semester selection section which is indicated as blue outline in Figure 3.5, it also showed the page name of a specific page view in the left side. Moreover, on the right side of section has two main selections for user to select a specific semester and examination type with the button to interact as search. Further, this is the search page section to expand the information on a specific page that users interact with. Second, which is the purple outline in the Figure, indicates the segmented control page section. And, this page is for users to switch pages that disclose different functions and reports to their needs. Third, the green outline in the Figure which is the date-time section that indicates the report's updated date time. Fourth, which is the orange outline section showing as the report filters with button confirmation interaction. Given these point, the red outline section is the table to display a report of a specific task. Moreover, most column data are left-aligned. This helps make the data conveniently scannable, readable and comparable.

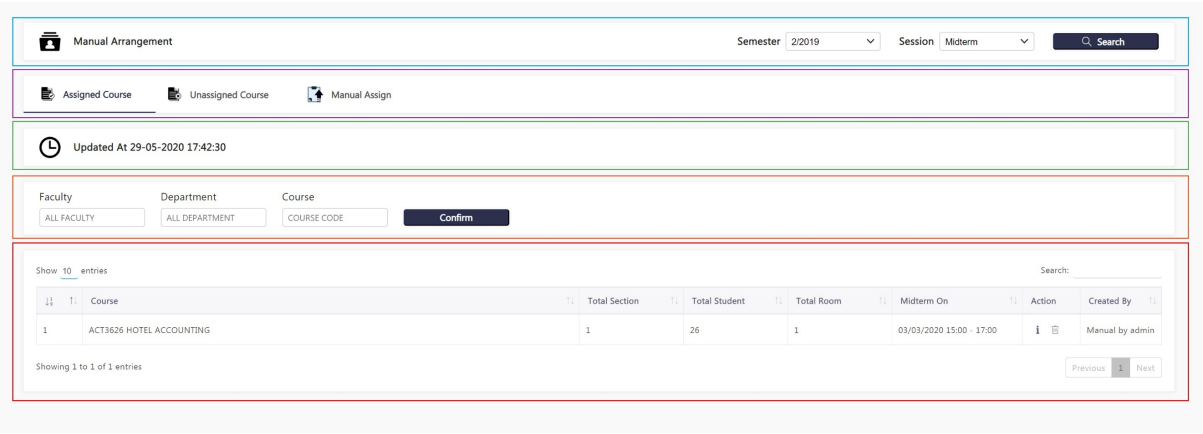

Figure 3.5: Page's Sections

3.5.6. Consistency

Consistency is a core principle in life and in design. It has abilities to enhance the works more persistent in a manner. And ARMS App has been implemented with mightiest principle like consistency to improve user experience (UX) and user interface (UI) as businesslike working system.

When it comes to design, consistency is one of the keys of the Design DNA. ARMS app is being improved with intuitive design to make it highly useful and better. Moreover, usability and learnability have come from consistency which most users have learned to understand by this kind of perception.

There are various consistencies that ARMS app has been accomplished. First, ARMS app has Visual Consistency that helps perceiving users to increase learn abilities to the system. Additionally, every element like font, button, label, text field, icon, color, selection, and other components are always be consistent across the product to enhance Visual Consistency to the users' perception. Second, the similarities of control are always functioning across the product that is Functional Consistency. Furthermore, it can increase the users' perception to predict the product's function, and predictability leads to users feeling safe and secure. For instance, in Manual Assignment and Automate Assignment, after users have clicked any event of button to trigger action, it will alert the confirmation dialog to disclose confirmation that he or she is agreed to that specific action. Third, when the usability and learnability of the product have been improved by both various consistencies above, it will be convenient to users when new features/pages are introduced to the ARMS app. With that mentioned, it leads users to keep along the path of how they use the system and that consistency is Internal Consistency. As showed above, users' knowledge and perception of the product that are archived by the consistency across multiple systems/products can be reused again with another. In addition, this good example of external consistency helps user to gain greater user experience and eliminate a lot of the friction. These consistencies can be found in the Figure 3.6 below.

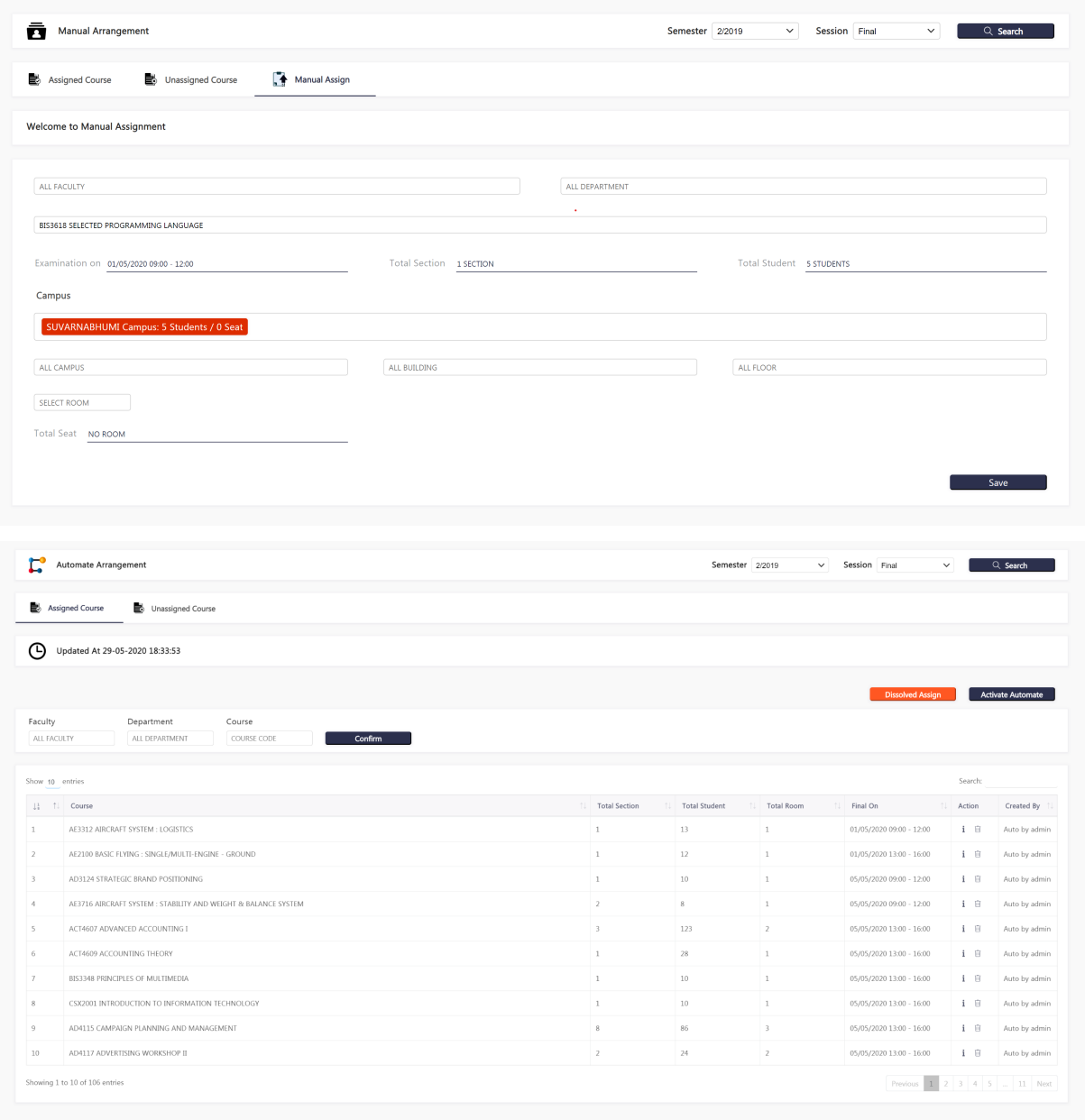

Figure 3.6: Design Consistency

Consistency is very important, and ARMS app has been implemented with these four types of consistency above to help user gain better usability and experience.

3.5.7. Examination Seating Report

Examination Seating Report is a report that is used to indicate an specific examination room to comprehend students' information like seat row, number, and section for students to view their information before entering the examination room. Meanwhile, the current examination seating report has an examination room's name column which is not efficient due to waste of column. Again, one examination seating report means it belongs to a specific room and it cannot be belong to more than one room. Likewise, with ARMS examination seating report the report has been designed and generated

with consistent and efficient format to facilitate examination students' information as simplicity as possible. And, the ARMS examination seating report design layout will as in a Figure 3.7.

Assumption University

Examination Seating Report

Semester: 2/2019
Examination on: 02/03/2020 09:00 - 11:00
Examination Type: Midterm
Examination Campus: SUVARNABHUMI
Examination Room: AR0402

SEQ	Student Code	Student Name	Section	Seat	Signature
1	591194BAE05CD11EAD2	559EDD30AE2DD62D8DB5D4DEDB119BF7 3.	401	1/1	
2	59133D97C3A70EDF021	E09A0BC05A0564135032AAA969C1B384 B.	401	1/2	
3	5915183F65A04E57D06	4F2B4FCE527A286B35DDB9E353C2DC22 9.	401	1/3	
4	5917B4F77467BC94200	125E1A7E3705C265D2030FAE9EB110BE A.	401	1/4	
5	591A4C573DBFEE51B96	1858B3E8DCB1654FD4557788EA05F8C2 8.	401	1/5	
6	591BB463BD4E962F4C5	D32FF9358B1CB096518F43705A9279FD B.	401	1/6	
7	591BE0BB8AC0F2D98BF	EFC8A3328EF920E086E3D5EF89FFD545 6.	401	1/7	
8	591E09D49D54733C27B	5D595B7511475230F5622BEFD7F1CBFB 3.	401	1/8	
9	591FE89AA6214672D6E	52F3DB8B5CA19E4ADCEC9B2F98274050 7.	401	2/1	
10	601032BFB6B1B71C388	94B3F184B11C0F551C2F9BA9F7CB7F8B 8.	401	2/2	
11	6014AEA3243E47C1D02	619F49B556655DC451264D17C3823F48 6.	401	2/3	
12	60154660980CDFBE274	09D31520ACE8B8BEBAF20C9DF105B2A7 F.	401	2/4	
13	60175D367652F049DD4	632CD86E9EE1A56E6340334D1CEB3B71 D.	401	2/5	
14	601812B4046ED6D2CF8	404A910D817701D2F3DC573496EC3F7F 3.	401	2/6	
15	601C4D81E296EA417D2	0CFF1B606A51BD97A42B8757FD9FCD22 1.	401	2/7	
16	601D898A37564E075B6	8C07E169D0D09732EFD06EB38BEDA951 D.	401	2/8	
17	601DE40C69522588A1F	79DDE6D037671E1D3C6E479B0F01ABBD 5.	401	3/1	
18	601E29A88AD1AFDEBD7	A319D040ED0943124CEDA98B37386AB3 D.	401	3/2	
19	601E3E6ACD94CD10CEE	F1CF93480D9C7BABA70C2164943DAD0E B.	401	3/3	
20	601E64CBD6736710656	180FFEBE02FEBB4AEE3B71AF4219E8DB 9.	401	3/4	
21	601EC12EDB96F6C1E09	8724A9685B0ABAEA91DD6F2A43FB7B28 D.	401	3/5	
22	601FF038D92A783831E	A01EE89075FE69DB0046CBEC71402C0B 6.	401	3/6	
23	603154AEE69D27B1D24	ECD2E0CECAF3174BB8234F201AED336E 3.	401	3/7	
24	60323E8FA12E6DFEEC6	7488D7D82DEF4D3E82447AD545FF343B 5.	401	3/8	
25	611664C24BED9DFC7C4	3E02C3E3258A3CB8B63C3A926FE3191A 4.	401	4/1	

Figure 3.7: ARMS Examination Seating Report

Futhermore, if any specific examination room has more than 25 students it will generate more than one report because the report only allows to facilate 25 students' information per report. For instance, if examination room A has 124 students the system will generate 5 examination seating reports and the last page of report will show the last 24 examination students. As showed above, in the ARMS examination seating report has been removed the room's name column due to inefficient which was exists in the current examination report of Assumption University.

Chapter 4

Algorithm

In the examination arrangement process, the most expensive step is assigning courses to examination rooms and students to their seats according to the study course. Chapter 2: Background has given an example of the Assumption University of Thailand of which processes are done by humans and given the questionable results. It also becomes a big workload for employees, implementing the program to do such redundant steps can reduce the cost of time and workforce. However, assigning examination rooms and examination seats are huge tasks, it can be computationally demanding, so the program should be constructed and optimized based on that in mind. Therefore, researchers who published their works on similar topics have faced the same issues and considered to use algorithms to solve the problem. However, the program cannot be implemented with the same method that exists works have used. Considering the constraints on the arrangement process of Assumption University of Thailand is a must since each educational institution could use different rules and constraints. Thus, the program must follow those rules and constraints.

4.1. Problem Description

Before getting to the rules and constraints that the Assumption University of Thailand is using, there are some factors that are involved in the constraints. To be informative, the following sections are going to explore each factor and how they affect the program.

4.1.1. Timeslot

Timeslot is the period of time which contains date, started time and ended time of each examination. In Assumption University, there are two main types of examination: midterm examination and final examination. There are also other types of examinations such as laboratory, practical, quiz and some specific examination for other faculties. The differences between these examinations and midterm, final examinations are the period that they are scheduled and the rooms that they use.

Midterm and final examinations are usually scheduled within a period of two to four weeks. The midterm period could be shorter than the final period since many courses can choose not to have a midterm examination. While other types of examination are going to be scheduled anytime within the semester depends on lecturers.

Moreover, regular midterm and final examinations are usually assigned in the normal lecture room such as St. Raphael, St. Michael and St. Gabriel buildings for Assumption University

Suvarnabhumi Campus. While other types of examinations such as laboratory and practical examination must be assigned to special room for the examination that requires computers or other equipment.

Lastly, timeslots for each examination can be different, but all of them are going to be within the period of examination. However, there are examinations from students who have different academic level or academic program as well.

Assumption University of Thailand uses curriculum-based timetabling, so the timeslots of every examination are pre-defined by the Registrar Office. Thus, the program will not be able to interfere that process, which means the program will mostly focus on performing redundant tasks that are time-consuming such as assigning examinations to rooms, assigning students to seats, visualize important information, and printing examination room papers.

4.1.2. Campus

Assumption University has four campuses: Suvarnabhumi Campus, Huamak Campus, City Campus, and ACC. Many courses at Assumption University can have some sections that study in the different campus as well. Additionally, students can choose to study any courses within their curriculum from any campuses. However, they must take the examination of that course on the campus that they chose to study. For example, Student A studies Computer Programming I in Suvarnabhumi campus and Selected Topic of Object-Oriented Concept in Huamak campus. For the examination, Student A must take the examination for Computer Programming I in Suvarnabhumi campus and Selected Topic of Object-Oriented Concept in Huamak campus.

4.1.3. Room Type

Assumption University of Thailand has many types of rooms: lecture room, seminar room, laboratory room etc. Lecture rooms are normally used for examination because of their seating format is suitable for examination seating. However, there are some courses that can not have the examination in the normal lecture room, for example, that course might require computer or specific equipment to do the examination. In addition, some courses that are counted as general courses that every student must take such as English or World Civilization, can have a number of sections and students. In that case, the Department of Administrative would prefer to assign those courses to take examination in rooms that have big seat number such as St. Michael 5^{th} floor.

Such cases that courses require specific rooms cannot be assigned by the program. The program chooses to have an option for user to choose specific rooms for specific courses.

4.1.4. Seating Methodology

It is normal to see more than one course or section having an examination in the same room. To avoid the prohibit actions such as cheating or copying from other students, seating methodology is used. As mentioned earlier that Assumption University of Thailand uses 'I' Methodology. But in order to arrange two courses in such way, the number of students in each course must be … For example, a room has 30 examination capacity, 5 rows and 6 seats for each row. To assign two courses into this room with 'I' Methodology, first course must have the number of students around 13 – 18 to be filled in 3 rows and the second course must have the number of students around 7 – 12 to be filed within 2 rows.

However, there are cases where the number of one course is less than the other and they cannot form the 'I' Methodology. In this case, another similar method will be used. Instead of having each course uses all rows alternately, one section with a smaller number of students will take one row and other rows will be used by the rest. This method requires one course or section to have the number of students less than or equal to the number of seats in one row of that room.

4.2. Constraints

Constraints are base rules that the program supposes to follow, it could be the requirement from the users or optimizations which can improve the program's performance. Constraints can be separated into two types. Hard Constraint is compulsory or constraint that must be satisfied, so the final result of the program should have the result that does not violate rules grouped as Hard Constraint. On the other hand, Soft Constraint is the opposite of Hard Constraint. It is non-compulsory requirement – it is optional but completing it would improve the quality of the work. These constraints usually depend on the requirements of the institution, which have been mentioned and described in the Problem Description section.

4.2.1. Hard Constraints

H1. For each examination, the number of total students must be less than or equal to the number of seats in each examination room.

H2. The total number of examination rooms in each examination period must be less than or equal to the number of available rooms in that examination period.

H3. The maximum number of courses in one examination room is two unless that room is allowed to have more than one course by the user.

H4. The student must take the exams based on the campus they study that course, for example, Programming Language class takes place in Suvarnabhumi campus, the examination of the said subject must be in Suvarnabhumi campus as well.

4.2.2. Soft Constraints

S1. The number of seat utilization in each room should be maximized to avoid wasting unused seats in each room.
S2. Minimizing room usage is also considered since it could save resources that university will have to pay for maintaining the building such as janitors, electricity, etc.
S3. Users or staff should be able to specifically assign courses into the specific rooms, in the occasion of having courses that require some special equipment for the examination.
S4. Having two courses in the same examination room is acceptable, but the seating format should be 'I' methodology. In order to form that format, the number of each course must be considered and choose the best course with suitable number of students to fill the room.
S5. Every examination with the same course should be assigned next to each other in a consecutive way to facilitate staff while they were distributing the examination paper or announcing information to the specific course or section.

These are Hard Constraints, and Soft Constraints in the system, that prevent some algorithms proposed by researchers likes Integer Linear Programming cannot be used, according to the soft constraint S4

4.3. Techniques

Information from previous sections has stated the reasons why the program cannot use the same set of algorithms and methods in the previous section of Problem Description and Constraints. This section will continue with techniques or methods the program will propose as a solution, as well as describe their functionalities and benefits to the program.

Along the way, this paper has given the information about algorithms and methods that existed works have been using, and reasons why they may or may not fit to the program. Moreover, in the beginning of this chapter, constraints have been explained to state the clear requirements of the program. The next section is going to use those information to explain and discuss about the algorithms and methods that ARMS chose to implement.

4.3.1. Iteration

Iteration is one of the simplest programming techniques that everyone should know. However, designing bad iteration functions could make the program runs extremely slow and needs more computation power than it should. This problem normally happens to the program that deal with huge among of data. In this problem, the program will have to handle a huge amount of data from database

such as examination courses, sections, rooms, buildings, campuses, students, registration courses and examination slots. For your information, those data have to be fetched from database, but opening connection to the database and query is very expensive process and performing such process repeatedly in a short time could lead to big performance issues. Thus, creating a query that can fetch all required information from database and format them to suitable format is very important process.

The reason why the information must be formatted into suitable way is because of the constraints that the program has. The constraints such as H1 – optimize the number of student in room, H2 – optimize the number of examination room, H4 – campus constraint and S5 – consecutive assigning, the iteration will be made over the examinations and campuses to separate the list of examination courses and available rooms for each campus in each time slot. Information in each iteration will contain the information of campus and examination slot such as an examination from 9 a.m. to 12 a.m. at Huamak campus, and the list of courses and available rooms ordered in consecutive way to maintain the S5. The assignment happens inside each iteration of campus and examination slot by performing another iteration over the available rooms and sections. Due to H2 constraint, the program must assign the students from each course to the available rooms without exceeding the resource that it has.

```
WHILE current section and room are not the last DO
      Assigning examination room and seat
END
```

Figure 4.1: Iteration Condition

From the Figure 4.1, in order to prevent H2 from being violated, the iteration with sections and available rooms will use the above pseudocode. Moreover, each room assignment it will use the number of students in each section from field ‘SeatUsed’ in section table from database compare to ‘ExaminationCapacity’ field in rooms table. The condition will prevent the violation in H1 constraint. There are more detail in assigning process, but the process uses different method than iteration, so it will be further more explained in the next section.

4.3.2. Loop Unrolling

Loop Unrolling is a looping transformation technique that researchers use its function to optimize the execution time of a program. However, the main propose of using this method is not to optimize the program efficiency, but to maintain the constraints. Even so, before going to the real

application that it is used, this section will start with the basic understanding of this methods and go to real usage along with reasons.

```
def normalLoop():
    for i in range(5):
        print("Hello World!")
```

Figure 4.2 : Regular Looping in Python

```
def loopUnrolling():
    print("Hello World!")
    print("Hello World!")
    print("Hello World!")
    print("Hello World!")
    print("Hello World!")
```

Figure 4.3 : Loop Unrolling in Python

To implement a loop unrolling into the system, it removes or reduces iterations as the code shown in Figure 4.3**.** Code in Figure 4.3 is more efficient than the code in Figure 4.2 because code in Figure 4.2 must check the value of 'I' and increase the value of 'I' every time round the loop before performing the function inside each iteration. However, the difference will not be obvious due to its size of iteration is very small. But using this technique to the program that has to deal with huge data and iteration, reducing the loop overhead can give a better performance. From the Figure 4.3, it shows the simple version of loop unrolling, but when the data sets are big, writing redundant code will not give a very good result to program both maintaining propose and readability. Moreover, the code size can be hideous, and it can give the slower result when meet code that has branches such as recursive.

Luckily that the main reason of using is not to optimize the runtime, but to optimize the program to satisfy H3 – maximum number of courses in each room, S1 – maximize the number of used seat in each examination room, S2 – minimize room usage and S4 – having 'I' seating methodology, constraints. There are two assigning process that must be done. First is assigning courses to examination room, and another is assigning the students in each course into examination seats. However, the examination seat cannot be assigned unless the assigning of examination room is successfully assigned or the number of courses in one examination room is exceed the H3 constraint, since the assigning seat process requires to know the total number of the students inside each course. These processes suppose to be finished before assigning the next course or room. In order to fulfil the H3, the program can either

get next section or course to fill the empty slot left from assigning the first course or find some course that has only one section to fill in the gap. Using section from courses that has only one section can help to complete S1 without losing S5 constraint. If the filling process uses any sections with the examination period, the results may not be able to maintain S5, since it will not be consecutive when one of the section in some courses is used to fill the gap from other course, that may assign to a totally different building or floor.

From the above optimization, the result could maintain the H3, S1 and S2 constraints, but not for S4. Even though the program can find the best fit or other courses to fill the gap inside the examination room from the above condition, but it is possible that the number of students from each course cannot satisfy the S4 constraint. Due to S4, the program must find the filled-in course that have the suitable number of students that can construct the 'I' seating methodology. The process of finding the most suitable course or section to fill the gap will be explained for in the next algorithm.

4.3.3. Greedy Algorithm

In the Chapter 2: Background, the information of greedy algorithm such as how does it works, and the basic properties has been explained in conceptual level already. The application of the greedy algorithm in the program is finding the most suitable filled-in course for the seat left in each examination room. The suited courses must be courses that have only one section for the same reason that has already been mentioned in the previous section. In order to separate the sections that met the condition, it is more efficient to perform the filter process in SQL to prevent the addition looping in the program. This can be done by grouping the section in the examination slot by their 'CourseId', filter group that has only one section inside and order them by the total number of students in section in descending way. The reason why these single section courses have to be order in descending way is to allow greedy algorithm to perform finding the most suitable section. The function that does the finding will get the number of seats or rows of section that can be used to form 'I' seating methodology.

For the sake of explanation, the following section will use term of section (A) to represent the course that has already been assigned to the examination room, and section (B) to represent the single section course that is going to be used to fill the seat left from section (A).

For example, the section (A) has been assigned to the examination room with 30 examination seats. Section (A) has 16 students in the section, means there are 14 seats left. Without the S4 constraint, any section or course that has number of students within 1 – 14 students can be used to fill the gap. However, the constraint S1 is also implemented, means section that has 14 students would be more preferable than section with less student. That's the reason why list of single section course is ordered by the number of students in descending way, since the main concept of greedy algorithm is

encountering each step with optimal choice. The pseudocode in Figure 4.4 can be used to further explain the above statement.

```
numberNeeded = to number of student needed to fill the space without
violating `I` Methodology
courses = courses that has only one section ordered in decending way

FOREACH course in courses DO
      IF number of student in course can satify numberNeeded DO
            RETURN course
END
RETURN null
```

Figure 4.4: Pseudocode for finding a suitable course

However, there are cases that available seats from section (A) is 16 and the most suitable single section course has only 3 students. Even the section can be used to fill the gap, but it will violate that S1 constraint – maximize the number of seats used in each room, since there will be 11 unused seats in that examination room. In this case, it would be more preferable to not using single section course and use the course from the next iteration instead. In the example above, there is 14 seats available, there are only 3 students in filled-in course. The course after section (A) will assigned to fill the gap instead. In the end, after assigning all the normal courses, if there are any single section course left, they will be assigned through the same method.

4.4. Outcomes

Implementing a program to the tasks that is usually done by a human would give some obvious benefits such as improving speed or reducing errors created by humans. This section will not discuss those obvious benefits because they can be found and emphasized in another Chapter. The following section is going to discuss the outcomes from optimization that the program has made.

In the beginning of this Chapter, it has the parts that explain the requirements of the program in term of constraints. Completing hard constraints will not be included as outcomes since those are compulsory requirement which are needed to be satisfied. Failing to complete the basic requirement means the program cannot be used in a real application. On the other hand, soft constraints are non-compulsory, and completing them can be additional outcomes from the program and can be used to improve the quality of the result. This method of evaluation has been proposed by Manar Hosny and Muhrah Al-Olayan [4].

Due to the method of finding filled-in courses from the seat left in each examination room, it has completed the S1 – maximizing seat used in each room and S4 – maintaining 'I' seating methodology. Moreover, the courses in each examination slot has been grouped and ordered by the course code and section number to satisfy the constraint S5 – assigning in consecutive way. All of the implementation for S1, S4 and S5 has given the result of S2 – minimize the number of used rooms. However, S3 constraint – allowing staff to arrange courses to rooms, is not completed by the algorithm in this chapter. The constraint has been satisfied by the manual assign feature, which allows user to choose a specific to be assigned into the set of rooms that user has chosen. From some cases that the automate program cannot design for when some courses require special tools such as computer or laboratory. Additionally, this feature can be used to handle unexpected examination outside the schedule such as examination for conflict students, quiz and practical examination.

Chapter 5

Evaluation

The project must be proven by the process of justifying the quality of works to evaluate the result and outcome. Evaluation should be crafted to address the specific goals and objectives. The following sections are going to discuss how this project is validated in different aspects.

5.1. Algorithm Quality

Algorithm Quality can be proven in many ways, according to the chosen algorithm. The mathematical algorithm models are normally evaluated by the cost function, but programs' algorithms are chosen based on the constraints it has. Constraints can be separated into two types: Hard Constraints and Soft Constraint. The program's constraints have been mentioned in Chapter 4: Algorithm, the outcomes of the program can be evaluated on satisfied constraints. To begin with, hard constraints are compulsory, and achieving it is the base requirement of the algorithm. On the other hand, soft constraints are non-compulsory, satisfying them is not necessary, but outcomes are beneficial to the program as well.

From Chapter 4: Algorithm, there were explanations of how the program has achieved all hard constraints. Hard Constraints can be satisfied by the specific condition of the program, such as the condition in the iteration and assigning process. The soft constraints which focus on creating better examination room arrangement and seat allocation more efficiently are also achieved. Since all hard and soft constraints have been achieved, the system could guarantee quality and efficiency. Comparing the program efficiency to the original system used by Assumption University's Department of Administrative might not be appropriate since comparing the human and machine is not equitable. Moreover, there are existed solutions, as mentioned in Chapter 2: Background, comparing to them would be more appropriate. The comparison will not include the constraints between this program and their program, because it is subjective to different institutes that maintain different rules.

To begin with, there are two systems mentioned, and they focus on different problems and solutions. Universidade de Liboa's Work focuses on examination and course timetabling. Research from the International Research Journal of Engineering and Technology also mentioned the timetabling, but they said that the objective of their work is to find out the solution of exam seating and room assigning problem. Even this project did not include the timetabling feature, but the system provides the convenience of an automated system. Users can perform the room and seat allocation with one-click action that does not included in other works. In conclusion, other than the satisfied requirement, another thing to mentioned is its convenience and usability.

5.2. Data Quality

The data stored in ARMS are the data which built to be further integrated with the university's primary data sources. So, the quality of the data stored in ARMS will be quiet depends on the data sources it integrated with. Nevertheless, these are the data quality's verifications criteria, which we considered based on the current integration with Assumption University data sources. First, the availability of the data, as we are the program built for a specific purpose like arranging the examination, so all the data required to arrange the real examinations are already verified by multiple arrangement testing. Second, accessibility of data, ARMS' data was storing the microservices architecture with all the securities implemented, so the data will be delivered to the client program only when there is a secured request. So, it will not densely store in the client program. Third, Usability of the data, the usability of the data store in ARMS is already proved by the uses in various kinds of features the ARMS application such as visualization, examination report, arrangement algorithm, and printing system. Fourth, Structure of data, ARMS data was designed to store as a formal database structure, but the normalization is not much implemented in this design because we need to prevent the overhead of joining across multiple tables. Finally, the Reliability of data, the data sources which ARMS integrated with was the data that used the real university system, so the data is reliable.

5.3. Business Logic

In ARMS development, there are multiple steps included in the preparations for the best suit in the real business implementation. There was a requirement gathering from the university's Administration Department who been responsible for the university's examination arrangement task, so it will ensure that the program we will build will be able to solve their pain points. Furthermore, all the constraints and requirements of the university's examination also being collected to ensure that we will not violate the university's regulations. Moreover, the result of examinations arrangement via ARMS also be compared with the result of the existed system, so we can verify that ARMS provide a better result with a more efficient solution.

5.4. Architecture Quality

The main architecture of this project has been mentioned in Chapter 3.4, which is the Microservices Architecture. There are several characteristics of Microservices that make this project accomplish the standard. First, it is all about the componentization via service, which means the software should be broken into components and various pieces to work with. Moreover, it is independently replaceable and independently upgradeable. Second, to organized business capabilities, and each team can have some

elements right always through and ideally focus on specific workspace. Third, the unit project can be decentralized data management. This means every service should be responsible for its own data and its own persistence to maintain the unit. Again, the frontend unit will not be allowed to communicate through a service data store; it can only communicate with another service through its API. As showed above, this Microservices' architecture is the best practice of the enterprise organization's architecture to adopt.

5.5. Source Code Quality

There are various technologies and architecture that this project is being implemented. First, the main design pattern in each unit in this project is Model View Controller (MVC)[9]. Moreover, this architecture is widely being used in many enterprises around the world. Additionally, MVC is a design pattern that divides an application into three major aspects: Model, View, and Controller. In MVC, the input is directed at the Controller first, which is not the view. That specific input might be coming from a user interacting with a page; however, it could also be from simply entering a specific URL into a browser. In either case, its controller is interfaced with to trigger the action. Again, when it comes to MVC design pattern architecture, it is challenging not to write spaghetti code in C#. Equally important, C# is an Object-Oriented Programming Language[10], which developers must write in an excellent manner to accomplish the best practice. Besides, with constraints and Microsoft's Software Development Kit framework like Dotnet core, always come with documentation to facilitate clean code in the workspace.

5.6. Open-Sources Use

There are several open sources that being used by this project. First, Entity Framework[11] is being used in this project to ease the workload with data communication to replace "SQL String." Moreover, it is very suited for .Net Core applications that is developed by Microsoft. It enables developers with many flexibilities to work with data using objects of domain-specific classes without focusing on the underlying databases tables and columns where the data are stored. Second, jQuery[12], which is the JavaScript framework to ease the frontend workload. Further, to accomplish a seamless transition with the UI, this project requires this framework to reduce less vanilla JavaScript, which is very complicated to use vanilla JavaScript. With that being mentioned, it is essential to reduce the spaghetti JavaScript code by using jQuery. Third, when it comes to CSS's framework, Bootstrap[13] is another main option for the developer to opt this popular framework to style components in the web frontend. As can be seen, the frontend unit requires to deploy as a standalone cross-platform desktop which needs Electron.Net framework[14] to encapsulate the unit into the container to run as Chromium desktop app. However, the Electron framework is mostly focusing on JavaScript's community, which C# just had to

happen a couple of years ago, also it was developed and maintained by GitHub. This framework allows the JavaScript code workspace or C# .Net Core workspace combines with the Chromium rendering engine and the Node.js runtime. Further, which means one based code application can run multiple platform in a single written code.

Chapter 6

Conclusion

ARMS is going to change the Assumption University's examination's seat arrangement system tremendously. Each step existed in arranging or allocating the examination seats will be more efficient and automated by ARMS. This program provides the easy to perceived result with the flexibility for further modification, visualization, and printing. Importantly, all the steps included in the existed arrangement process regularly take 1-2 weeks. It will be diminished to 1 day or even a few hours of the automatic arrangement for any examination.

6.1. Limitations

As the current implementation of ARMS, the examination slot or examination period management is not available, which means users can only arrange the examination seats based on the pre-defined time slot for each course. Nevertheless, there is the preview of examination time conflict provided where users can use to reveal any conflicts occurred. In ARMS, the only priority on grouping the students together are their sections and courses. It means that the algorithm will only group the students with the same sections and subjects.

6.2. Future Work

We are looking forward to implementing the system to be able to handle more comprehensive data structure, in order to make this system thoroughly fit to the Assumption University's management system's ecosystem. For instance, the abilities to arrange the examination rooms for multiple academic levels and academic programs' sections where there are multiple time slots occur in the same course. Furthermore, there would also more software verification protocols applied to this system likes unit testing, integration testing, system testing, acceptance testing and Release testing. We would also implement the examination slot management and further included in the automatic arrangement so that it generates the least examination time conflict as possible. Last but not least, ARMS could be further improved to arrange the normal classrooms and schedules. This could give a significant impact on the students' university life and create a better learning quality.

References

[1] Alexandre Lemos, Francisco S. Melo and Pedro T. Monteiro Ines Lynce, 2018. *Room usage optimization in timetabling: A case study at Universidade de Lisboa.* Retrieved May 9, 2020 from https://www.sciencedirect.com/science/article/pii/S2214716018301696

[2] Prof. Nilima Nikam, Akshata Jagdale, Gunjan Patil and Prachi Patil , 2017. *Algorithm for efficient seat allocation process in college exam system,* pp. 2-3.
Retrieved from https://www.irjet.net/archives/V4/i3/IRJET-V4I3717.pdf

[3] Edmar Hell Kampke, Walace de Souza Rocha, Maria Claudia Silva Boeres and Maria Cristina Rangel, 2017. *A GRASP algorithm with Path Relinking for the University Courses Timetabling Problem,* pp. 1-7.
Retrieved from https://proceedings.sbmac.emnuvens.com.br/sbmac/article/view/1007

[4] Manar Hosny and Muhrah Al-Olayan, 2014. *A Mutation-Based Genetic Algorithm for Room and Proctor Assignment in Examination Scheduling,* pp. 1-2
Retrieved May 11, 2020 from
https://www.researchgate.net/publication/318224682_A_mutation-based_genetic_algorithm_for_room_and_proctor_assignment_in_examination_scheduling

[5] Integer Linear Programming (2019) Retrieved May 14, 2020 from
https://neos-guide.org/content/integer-linear-programming

[6] Greedy Algorithm (2017) Retrieved from
https://www.geeksforgeeks.org/greedy-algorithms/

[7] Loop Unrolling (2017) Retrieved from https://www.geeksforgeeks.org/loop-unrolling/

[8] Definition of Microservices Architecture (May 2020) Retrieved May 18, 2020 from
https://en.wikipedia.org/wiki/Microservices

[9] .Net MVC Pattern (February, 2020) Retrieved May 18, 2020 from
https://dotnet.microsoft.com/apps/aspnet/mvc

[10] C# Object-Oriented Programming (May 2020) Retrieved May 20, 2020 from
https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/object-oriented-programming

[11] Microsoft Corporation, C# Entity Framework (September, 2019) Retrieved May 20, 2020 from https://docs.microsoft.com/en-us/ef/

[12] JQuery (May 2020) Retrieved from
https://jquery.com/

[13] Bootstrap (May 2020) Retrieved May 20, 2020 from https://getbootstrap.com/

[14] Electron .Net (May 2020) Retrieved May 24, 2020 from
https://github.com/ElectronNET/Electron.NET