FUDMA Journal of Sciences (FJS)
ISSN online: 2616-1370
ISSN print: 2645 - 2944
Vol. 9 No. 6, June, 2025, pp 156 – 164
DOI: https://doi.org/10.33003/fjs-2025-0906-3538

AUTOMATED EXAMINATION SEAT ALLOCATION AND CHART GENERATION SYSTEM

*[1]Zaharaddeen S. Iro, [2]Munir Salele Iro, [3]Muhammad Salisu and [3]Victor O. Olalekan

[1]Information Technology Department, Federal University Dutse.
[2]Computer Science Department, Capital City University Kano.
[3]Electrical Engineering Department, Bayero University Kano.

*Corresponding authors' email zs.iro@fud.edu.ng

ABSTRACT

Automated student seat allocation is a computerized process that utilizes algorithms to assign seats to students in educational institutions. This study provides an overview of automated student seat allocation, highlighting its benefits and challenges. We discussed the working mechanism of automated seat allocation systems, factors considered during the allocation process, and the algorithms employed to optimize the process. We emphasized the advantages of automated systems, such as speed, accuracy, and fairness, over traditional manual allocation methods. The study also addresses implementation challenges, including data privacy concerns and the importance of personnel training. The study developed a software using three programming stacks, each with its own unique function. The front-end is developed using React JS, the back-end utilizes Django, and the main software interface was built using Python programming language. Exam details are stored in the database held by the back-end and are presented to students through the front-end after they input their registration number and examination course code. By emphasizing the significance of automated student seat allocation systems, we underscored their role in ensuring a fair and efficient seat allocation process in educational institutions.

Keywords: Seat Allocation, React JS, Django, Python, Web Application, Database Management

INTRODUCTION

It is common for educational institutions to conduct examinations at fixed intervals. However, we have found that most Universities, do not utilize any software or platform for seat allocation during exams. The exam officers have to manually count the total number of registered students, select appropriate rooms/halls/theatres, and then allocate the students accordingly. Subsequently, he prepares a student list for each exam and creates seating arrangements based on student counts for each room. This process is extremely repetitive and prone to error due to its manual nature.

When it comes to conducting fair and secure examinations, it's essential to ensure that no opportunity for cheating or malpractice exists. Allowing students to arrange their seating order during exams may seem like a good idea, but it often leads to collaborative cheating and other forms of misconduct. To prevent this, a software program that randomly assigns seats to students is much more effective in promoting fairness and minimizing exam stress for all involved parties. The study aims to develop a system for universities that streamlines seat allocation process by automatically generating seats based on student registration details. This software uses three algorithms that sequentially introduce empty seats in contiguous chunks until all registered students are assigned seats. This approach not only prevents cheating but also accommodates accessible seating needs for physically challenged or visually impaired individuals. By implementing this innovative technology, we hope to foster an environment of integrity and academic excellence within our institution.

Literature Review

An examination is an assessment to measure student knowledge, fundamental ability, life skill, aptitude, physical fitness, or experience attained in some other topic. It is a set of questions mostly used to determine students' knowledge of various topics or fields(2009). Institutions of higher learning are constantly taking measures to ensure the smooth running of examinations and discouraging malpractice has been a concern for most higher learning institutions.

A good way to stop exam cheating is to assign seats at random. Porquet-Lupine *et al.* (2022)proposed an effective solution to combat exam cheating by implementing random seat assignments. They introduced LupSeat, a powerful seating chart generator software equipped with numerous features. The tool utilizes algorithms to ensure students are randomly assigned seats while maintaining an even distribution throughout the classroom. It operates based on a simple text representation of the classroom layout and a CSV-formatted student roster.

An automatic seating arrangement is a set of written or spoken instructions that determines where people should take their seats. It is usually used on diverse occasions. Seating plans have a wide range of purposes. Bougie *et al.* (2012)define space allocation as a process of allocating rooms or areas of space for specific functionality. Thus, since it is limited, it must be well managed by the faculties towards the availability and suitability required. The existing manual system has flaws and loopholes that are yet to be corrected, such as disorderliness and chaos, which prevents the system from being a typical recommendation for public or general use. Alvarez-Valdes, *et al.* (2002)used a set of heuristic algorithms in a program for solving course timetabling-related problems. Implementing a computerized system that handles the challenges of examination seating arrangements and comprehensive record-keeping for students in tertiary institutions will streamline the process of allocating exam seats and halls during examination periods.

MATERIALS AND METHODS

The study takes the form of system application software named "EigenSeat". The software is developed using the Python programming language due to its simplicity, performance, versatility, and compatibility with databases and web servers. It is designed to be cross-platform, ensuring compatibility with Microsoft Windows, MacOS, and Linux operating systems. The PyQT module is utilized to achieve this cross-platform functionality.

The software program is designed to automate the creation of randomized seating arrangements in exam venues. It addresses the need for an efficient and accurate method of assigning students to seats during exams. The software takes a textual description of a classroom and a CSV-formatted student roster as inputs.

The student roster contains essential information such as the student's full name, registration number, and additional fields for accessibility. The textual description of the classroom allows for the representation of various seating arrangements, including large lecture halls. The software generates three output files: a CSV file containing the students' seat numbers with their registration numbers, and two graphic files in JPEG and PDF formats.

The graphical output files visually depict the seating arrangement, with a PDF file showing the relationships between seat numbers and registration numbers, and a JPEG file representing the classroom layout. The software offers advanced features, including a graphical user interface, compatibility with multiple operating systems, the ability to specify seats for disabled students, and support for customizable seating preferences.

The software ensures privacy by allowing the display of only the last few digits of registration numbers on the seating chart. It offers flexibility and convenience in managing seating arrangements, saving time, and reducing the potential for errors associated with manual seat allocation.

The software is particularly valuable in the context of increasing numbers of individuals appearing for various examinations. It streamlines the allocation process, improves efficiency, and produces reliable results.

Its key feature is the ability to capture data related to students, rooms/halls/theatres, exam details, and invigilators/staff as input. The software interface comprises four segments as follows:

i. Input file: This segment encompasses the student and room files.
ii. Exam details: This segment covers essential examination information such as Date, Time, Course, and Invigilator.
iii. Settings: This segment includes options such as Format string, Sort by, Seed, Algorithm, Manner, and Seat Intervals.
iv. Output: This segment features checkboxes for generating the chart, image, PDF, and an option to publish online. It also allows for specifying the output directory for the generated files.

The input section provides the ability to select the file directory from which the software retrieves its input information. In the exam details section, all necessary details pertaining to exams, including invigilators, are entered. Administrators can customize seat allocation format and algorithm preferences in the settings section. In the final section, administrators use checkboxes to determine how the output is published (online or offline) and where it is saved. It is important to note that offline access is restricted to administrative use only.

Administrator Module (Offline)

This module offers a set of specific instructions to assist administrators in effectively operating this program. To begin, the administrator should initiate the software by double-clicking on its icon located on the computer system. Once launched, a highly detailed user interface will appear, allowing for easy navigation and interaction. From here, the administrator can interact with the interface by either making selections or entering data as needed.

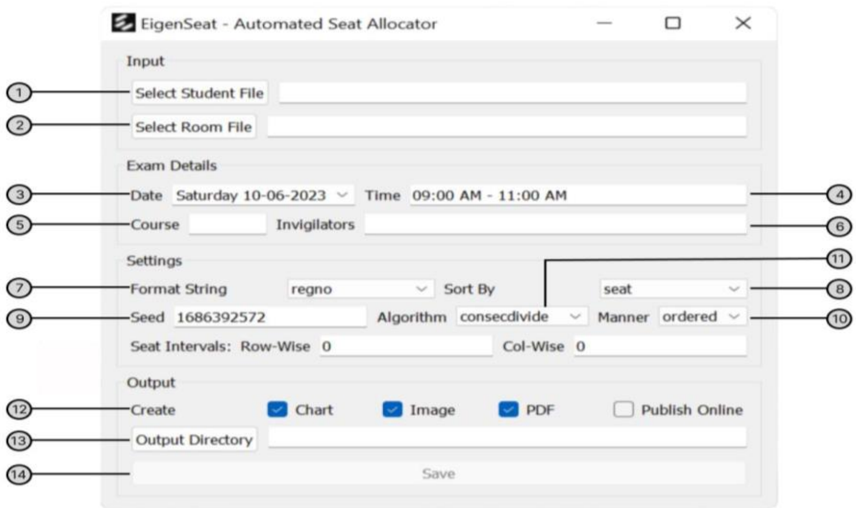

Figure 1: The User Interface of EigenSeat Software

Step 1: To select the students' data, the user can click on the "Students File" button and choose the .CSV file that contains the student data.

Step 2: The user can select the exam room data by clicking on the "Select Room File" button, following the same procedure as in Step 1. The user then selects the .txt file that contains the exam room seat details.

Steps 3 and 4: In these steps, the user can edit the exam date and time.

Steps 5 and 6: The user is provided with the option to input the course code and the invigilator(s) for the examination.

Step 7: The user can utilize the drop-down toolbar to select the desired string format, such as registration number, full name, or username.

Step 8: The user can utilize the drop-down menu to select the desired sorting option for students, such as sorting by seat number, registration number, or seat arrangement.

Step 9: In this step, the user can select a unique reference character or code that can be used to regenerate exact outputs.

Step 10: The user has the choice to determine the manner in which students will be assigned seats. This can be either random or ordered, based on their preference.

Step 11: From the drop-down menu, the user can select the desired algorithm for seat allocation, choosing among options like chunk divide, consecutive divide, or custom algorithm. Note: Seat Intervals fields are only accessible when custom algorithm is selected.
Step 12: The user has the flexibility to select the desired outputs to be generated. By selecting the 'Charts' option, a CSV file will be generated, providing a list of students along with their assigned seat numbers. Additionally, selecting the 'Image' checkbox will generate an image file depicting the seating arrangement in a visual format. The 'PDF' option generates a PDF file that includes detailed information about the exams, including student names and their assigned seat numbers. Finally, by checking the 'Publish Online' checkbox, the software enables the publication of generated seat numbers online, allowing students to access them remotely.
Step 13: To specify the folder where the results will be saved, click on the "Output Directory" button. Then, browse through the file system and select the desired directory path to store the results.
Step 14**:** After correctly inputting all the necessary information, the user can proceed to click the "Save" button to generate the desired outputs.

Student Module (Online)
In this module, students have the convenience of remotely checking their seat number by visiting the host website. They can input their registration number along with their preferred course code to retrieve their exam details. Additionally, a visual format of the venue is attached for easy reference.

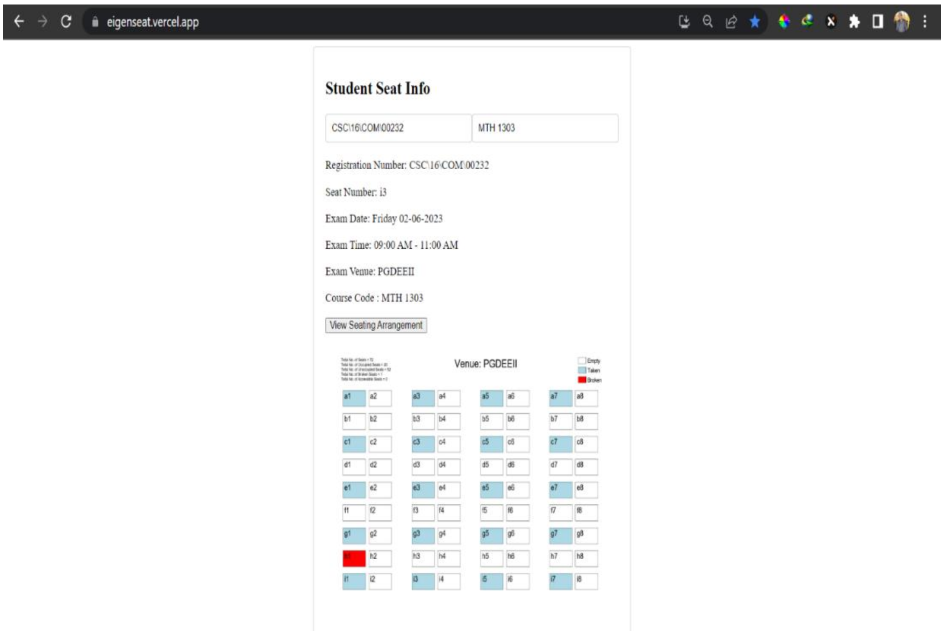

Figure 2: Pictorial view of the host website displaying a student's seat and exam details

The physical gathering team demonstrated great diligence in collecting comprehensive information about examination venues, including the number of seats available and identifying any broken seats in each venue. This valuable data was meticulously stored in a GitLab repository, providing online access to the database for convenient retrieval and utilization.

Table 1: Summary of collated data for selected venues

Venue	No. of Seats	No. of Broken Seats	Seat Arrangement
B09	128	7	Theatre
CPET	200	33	Theatre
EET	402	37	Theatre
TTA	320	7	Theatre
TTB	320	10	Theatre
NCL1	150	6	Hall
NCL2	150	8	Hall
CIV4	120	6	Room
CIV5	119	10	Room
COM4	80	30	Room
COM5	80	29	Room
ELE4	80	22	Room
ELE5	80	20	Room
TEL4	72	6	Room
TEL5	72	2	Room

Description of Student File

The student file required as input for the software should be in CSV format, comprising three columns: Full name, Registration Number, and Accessibility. For students requiring an accessible seat, the accessibility field should be marked with either the word "accessible" or the character "a", while leaving the rest of the field empty. These files should be obtained from the school administration for each registered course. Adhering to the specified format is essential to ensure compatibility and seamless utilization of the software. Below is a CSV-format sample of 20 randomly generated students, with 5 students requiring accessible seats.

Table 2: Sample of 20 Students in CSV-File Format

Full Name	Registration Number	Accessibility
Anas Bala Muhd	ENG/18/CIV/00491	
Sani Musa	CSC/18/INT/00301	
Bello Auwal	ENG/16/COM/00206	accessible
Musa Bala Sani	CSC/17/COM/00303	
Muhd Garba	CSC/17/CBS/00190	
Faith Daniel	CSC/18/SPF/00193	a
Bola Saminu	ENG/19/ELE/00219	
Atiku Danbala	ENG/19/COM/00435	
Shamsu Sani	CSC/18/SOF/00248	
Sunday Araga	CSC/18/INT/00246	accessible
Matthew Thompson	CSC/18/INT/00268	
Emma Martinez	CSC/17/COM/00315	
Josheph Rodriguez	ENG/17/TEL/00475	
Ava Hernandez	ENG/16/CIV/00316	
James Davis	CSC/16/COM/00498	accessible
Isabella Johnson	ENG/17/CIV/00275	
William Taylor	CSC/19/CBS/00382	
Mia Lee	ENG/19/CIV/00331	
Alexander Wilson	CSC/16/INT/00364	a

Description of Room File

The room file is structured into three segments: Seats, Specifiers, and Venue.

```
Seats:
a[1:1],a[3:5],a[6:9]
b[1:5],b[6:9]
c[1:5],c[6:9]
d[1:5],d[6:9]
e[1:5],e[6:9]

Specifiers:
b:c[2]
a:a[1]

Venue:
COM5
```

Figure 3: Overview of a Room file

In the "Seats" segment, each line represents a row labelled with lowercase characters from "a" to "z". These characters represent the row number when converted to their alphabetical indices and identify the seats from left to right within the row. Chunks of seats are indicated by enclosing the seat numbers in square brackets, separated by a colon. The numbers before and after the colon indicate the range of seat numbers within each chunk. A comma separating chunks denotes an aisle, which separates a group of seats.

The "Specifiers" segment uses the characters "a" and "b" as tags to specify accessible and broken seats within the venue.

Finally, the "Venue" segment provides the name of the specific exam room.

Consider a chunk denoted as "d [2:7]":

- The character "d" specifies the fourth row.
- The chunk "[2:7]" contains six seats ranging from seat number 2 to 7.

Description of Algorithm

From a programming perspective, the most intriguing aspect of this work is the seating algorithm. The objective is to maximize the number of empty seats between students while ensuring that all students are accommodated, given the number of students in a class and the number of seats in a classroom.

During the development of EigenSeat, we compared two distinct algorithms: chunk increase and consecutive divide. The chunk increase algorithm follows a bottom-up approach, whereas the consecutive divide algorithm adopts a top-down approach. In addition, we implemented a custom algorithm that allows users to determine seat intervals both row-wise and column-wise, providing them with flexibility and control.

The *chunk-increase* algorithm operates by initially assigning a chunk size of 1, ensuring that each student has an empty seat, aisle, or wall on both sides. If the room capacity is insufficient to accommodate all students with this chunk size, the algorithm is iterated, gradually increasing the chunk size by 1 each time. The process continues until all students can be accommodated within the room.

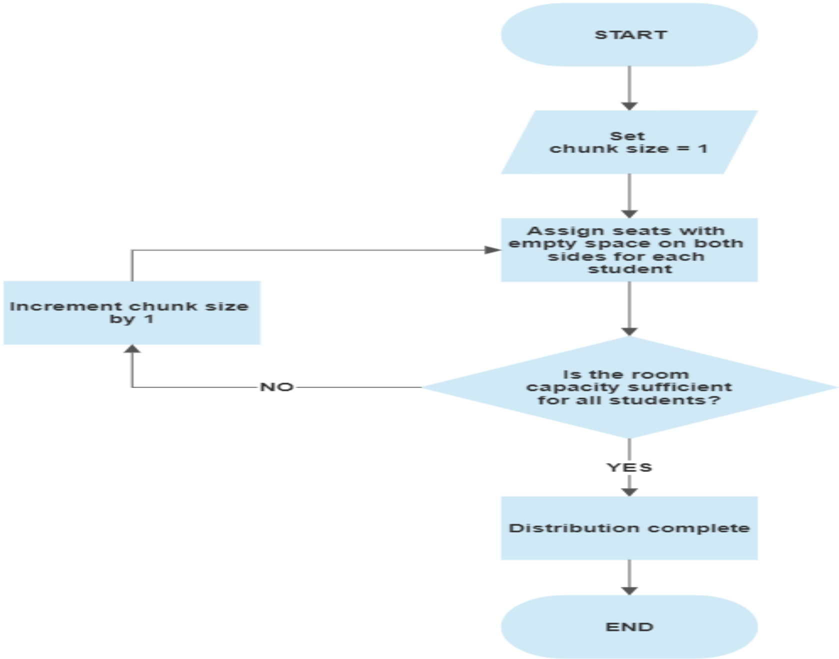

Figure 4: Flowchart of Chunk-Increase Algorithm

The *consecutive-divide* algorithm is designed to allocate seats in a classroom using a top-bottom approach. It begins by considering all the available seats. When there are more seats than students, the algorithm proceeds to divide the largest adjacent seat groups by gradually introducing empty seats. To ensure balanced chunks within the same row and avoid creating smaller, unbalanced chunks, a backtracking phase is implemented. Prior to introducing an additional empty seat, the entire row is reset to its original configuration and then evenly divided once again. This process continues until the number of available seats matches the number of students, resulting in a fair and balanced distribution of seats.

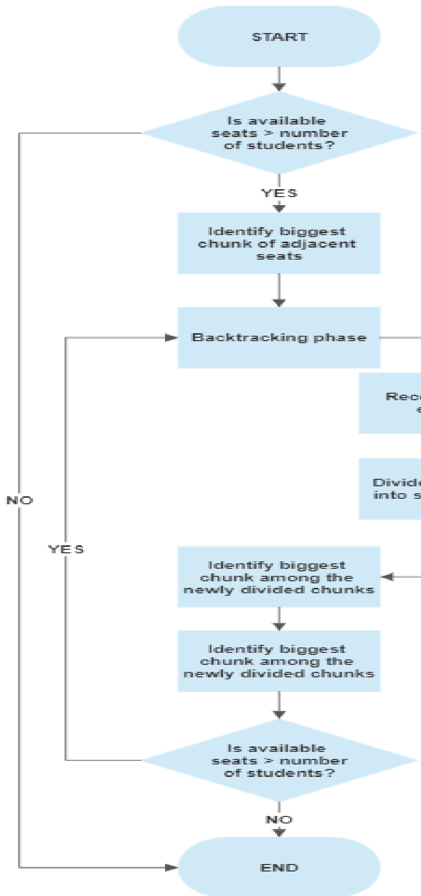

Figure 5: Flowchart of Consecutive-Divide Algorithm

The *Custom Algorithm* is a distinct seating allocation method that offers flexibility and customization. In this algorithm, the administrator has the freedom to specify the spacing between seats, both row-wise and column-wise, according to their preferences. This algorithm relies on the administrator's judgment and knowledge to allocate specific seats based on predetermined factors and justifications. The Custom Algorithm empowers the user to make informed decisions regarding seat allocation, considering various elements and factors relevant to the seating arrangement.

The assignment of accessible students to available seats is the primary step in any algorithm employed. Through extensive testing on randomly generated student lists and a thorough evaluation of seating arrangements in various venues, it has been concluded that the consecutive-divide algorithm consistently surpasses other algorithms, especially in venues with non-standard seating layouts like theatres. Consequently, the consecutive-divide algorithm has been designated as the default seating assignment strategy. This algorithm prioritizes the initial assignment of accessible students to available seats, thereby optimizing the seating arrangement for optimal efficiency and suitability.

RESULTS AND DISCUSSION

The software generates three separate output files: Chart, Image, and PDF as shown in Figure 6 a,b, and c respectively below.

	A	B	C	D	E	F
1	a1	CSC\16\COM\00232				
2	a3	CSC\16\SOF\00789				
3	a5	ENG\17\TEL\00291				
4	a7	CSC\18\INT\00451				
5	c1	CSC\19\CBS\00411				
6	c3	CSC\17\INT\00234				
7	c5	CSC\17\SOF\00349				
8	c7	ENG\19\TEL\00291				
9	e1	CSC\19\SOF\00488				
10	e3	FST\17\OFS\01114				
11	e5	CSC\16\CBS\00240				
12	e7	FST\17\OFS\01435				
13	g1	CSC\17\SOF\00356				
14	g3	ENG\17\TEL\00232				
15	g5	CSC\18\COM\00350				
16	g7	ENG\19\TEL\00345				
17	i1	ENG\18\TEL\01131				
18	i3	CSC\19\COM\00232				
19	i5	FST\17\OFS\02343				
20	i7	ENG\16\TEL\01234				
21						
22	Seed:1685717874					
23						

Figure 6(a): Sample of Generated Chart File

Figure 6(b): Sample of Generated Venue Image File

Date	Time	Course	Enrolment	Venue	Invigilators
Friday 02-06-2023	09:00 AM - 11:00 AM	MTH 1303	20	PGDEEII	DR. NAS

a1	CSC\16\COM\00232
a3	CSC\16\SOF\00789
a5	ENG\17\TEL\00291
a7	CSC\18\INT\00451
c1	CSC\19\CBS\00411
c3	CSC\17\INT\00234
c5	CSC\17\SOF\00349
c7	ENG\19\TEL\00291
e1	CSC\19\SOF\00488
e3	FST\17\OFS\01114
e5	CSC\16\CBS\00240
e7	FST\17\OFS\01435
g1	CSC\17\SOF\00356
g3	ENG\17\TEL\00232
g5	CSC\18\COM\00350
g7	ENG\19\TEL\00345
i1	ENG\18\TEL\01131
i3	CSC\19\COM\00232
i5	FST\17\OFS\02343
i7	ENG\16\TEL\01234

Figure 6(c): Sample of Generated PDF File

The Chart file is in CSV format and provides the student seating information, including their seat numbers and registration numbers. It also includes a seed number at the end of the CSV file for reference purposes. The Image file displays a visual representation of the venue-generated seat arrangements, for easy access by students. It also provides essential information regarding the total number of seats, the count of occupied and unoccupied seats and the total number of broken and accessible seats. Finally, the PDF file contains comprehensive exam details, such as the exam date, time, enrollment information, course code, venue and name of invigilators assigned to the venue.

To provide a comprehensive understanding of the software's functionality within a real system, we generated visual representations of various venues accommodating varying numbers of students. Figure 7a and b below shows the seating arrangement using Custom algorithm with two seat interval row-wise and with 3 seat intervals row-wise by 1 seat interval col-wise respectively. Figure 8 and 9 shows the seating arranging using Consecutive Divide and Chunk Increase algorithms respectively. These visualizations will demonstrate the effectiveness of the software when utilizing the three main algorithms.

Using Custom Algorithm

Figure 7(a): Custom Algorithm seating arrangement with two seat intervals row-wise

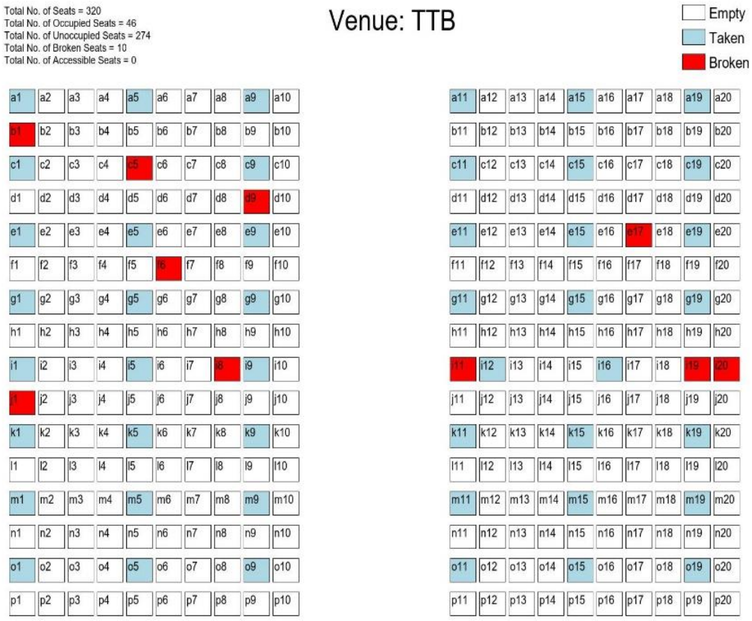

Figure 7(b): Seating arrangement with 3 seat intervals row-wise by 1 seat interval col-wise

Using Consecutive Divide Algorithm

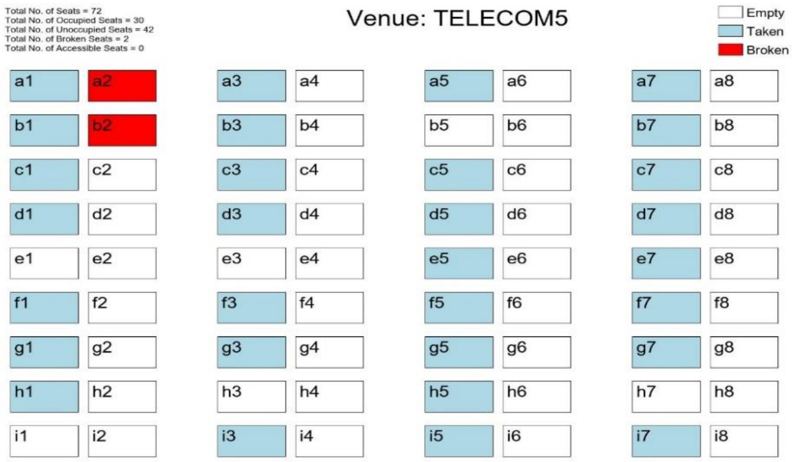

Figure 8: Consecutive Divide Algorithm seating arrangement

Using Chunk Increase Algorithm

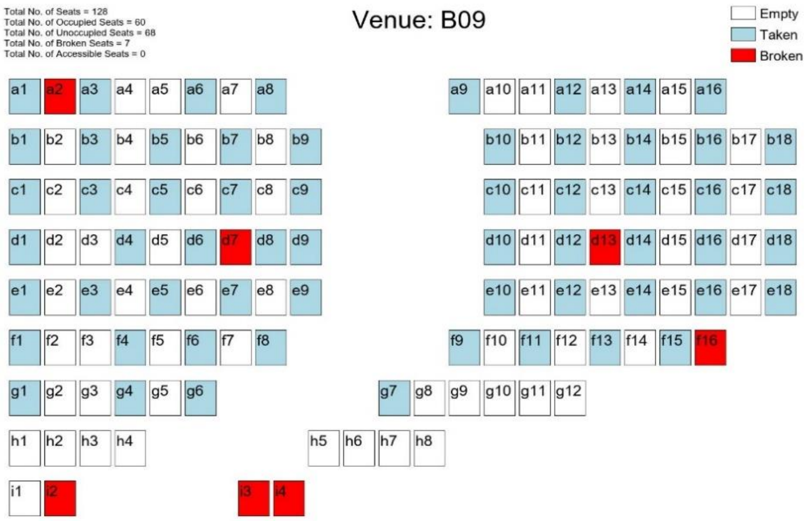

Figure 9: Chunk Increase Algorithm seating arrangement

CONCLUSION

The study has developed a software system called "EigenSeat" for seat allocation to be used during exams in Universities. Currently, universities lack automated system for seat allocation, leading to a manual and error-prone process. The study streamlined the seat allocation process by automatically generating seats based on student registration details. The software uses three algorithms to assign seats, ensuring fairness and accommodating accessible seating needs. The work included creating a central database system, generating seat allocation charts, and ensuring accessibility for disabled students. By automating the allocation of students to their desired locations, the software reduces the workload for institutions, improving efficiency and productivity. It provides time-saving benefits during examination periods and serves as a useful tool for exam management.

REFERENCES

S. Caroline, “What is Assessment and its Societal Importance.,” South Carolina, College of Education, South Carolina, 2009, p. 10.

J. Porquet-Lupine, H. Gojo, and P. Breault., “LupSeat: A Randomized Seating Chart Generator to Prevent Exam Cheating.,” in International Journal Of Engineering Applied Sciences And Technology, New York, NY, USA, 2022.

Bougie, J., Gangopadhyaya, A., Mallow, J., & Rasinariu, C., Supersymmetric quantum mechanics and solvable models., 2012, pp. Symmetry, 4(3), 452-473..

Alvarez-Valdes, R., Crespo, E., & Tamarit, J. M., “Design and implementation of a course scheduling system using tabu search,” European Journal of Operational Research,, vol. 137(3), pp. 512-523, 2002.

Ashti Fatima Alam, ““ Seating Arrangemnet Tools for Examinations”,,” International Journal Of Engineering Applied Science and Technology, 2015.

C. Anuradha, S. Pothumani,, ““Effective Seat Allocation for Examination Using Android Application”,” Eurasian Journal of Analytical Chemistry, vol. 13, no. (4), pp. 775-782, 2018.

R. Gokila, Antony Rohan Dass , ““Examination Hall and Seating Arrangement Application using PHP”,” International Journal of Engineering Science and Computing, vol. 8, no. 2, 2018.

L. T. G. Merlot, N. Boland, B. D. Hughes, and P. J. Stuckey, “A Hybrid Algorithm for the Examination Timetabling Problem,” Gent, Belgium, 2003.

E.K.Burke, B.McCollum, and P.McMullan,, ““Examination Timetabling: A New Formulation,”,” in International Conference on the Practice and Theory of Automated Timetabling (PATAT), Brno, Czech Republic, 2006.

Kahar, M. N. M., & Kendall, G., “The Examination Timetabling Problem : Comparison of a constructive heuristic with an existing software solution.,” European journal of operational research, vol. 207(2), pp. 557-565., 2010.

O’David Sklar, Adam Trachtenberg,, PHP COOK BOOK, Reilly Publications,, 2002 Edition.

Prof S.S.Aravinth, G. Pavithra, “Exam hall seating arrangement System using PHP”, vol. 1, AP/CSE Knowledge Institute of Technology, 2014.

“w3schools,” [Online]. Available: http://www.w3schools.com/php/default.asp. [Accessed 4 May 2023].

P. K. Chaki, “Algorithm for efficient seating plan for centralized exam system,” in Shikha Anirban 2016 International Conference on Computational Techniques in Information and Communication Technologies (ICCTICT), 2016.

Apurva Inamdar, Anand Gangar, Arun Gupta, Varsha Shrivastava, “Automatic Exam Seating & Teacher Duty Allocation System,” in Second International Conference on Inventive Communication and Computational Technologies (ICICCT), 2018.

D. D.Dhotre, S. Makwane and a. P. Lahase, “Automatic Exam Seating Arrangement System,” International Research Journal of Engineering and Technology (IRJET), vol. 6, no. 05, pp. 625-631, May 2019.

S. Anjum, M. D. Chodey and &. M. Afzal, “Automation of Exam Hall Alllotment and Seating Arrangement,” International Journal Of Engineering Research & Technology, vol. 10, no. 06, June 2021