JURNAL EMACS
[Engineering, MAthematics and Computer Science] Vol.7 No.1 January 2025: 9-16
e-ISSN: 2686-2573
DOI: 10.21512/emacsjournal.v7i1.12575

Parking System Application Using a Greedy Algorithm Approach

Hanis Amalia Saputri[1*], William Syaputra[2], Charles[3], Andreas Dwi Irawan[4], Ghinaa Zain Nabiilah[5]

[1-5] Computer Science Department, School of Computer Science,
Bina Nusantara University,
Jakarta, Indonesia 11480
hanis.saputri@binus.ac.id; william.syaputra@binus.ac.id; charles008@binus.ac.id; andreas.indrawan@binus.ac.id; ghinaa.nabiilah@binus.ac.id

*Correspondence: hanis.saputri@binus.ac.id

Abstract – *Indonesia has recently witnessed a significant increase in the number of automobiles, reaching an estimated 17.2 million units by the end of 2022, according to the Central Statistics Agency (BPS). Extensive ownership and usage of vehicles in public parking areas, including campuses, have created a high demand for parking spaces. However, challenges still exist within the parking system, such as longer search times for available parking spaces and the lack of technological regulation, leading to uncertainty. Our research focuses on addressing these issues by employing a priority-based greedy algorithm for the nearest lift, prioritizing convenience and speed. We utilize an SQL database to store parking data, leveraging its comprehensive features for efficient processing. The result of this research is a website where customers can input their license plate numbers, processed by our algorithm to generate parking tickets, granting access to designated parking areas. The algorithm works by providing parking slot locations from even-numbered floors first; when all even-numbered floors are filled, it will then allocate parking slots on odd numbered floors. The implementation of the greedy algorithm and SQL database has proven to be efficient in the context of the nearest lift in the Binus parking lot, handling a manageable amount of data and prioritizing data processing speed over achieving the optimal solution in all scenarios.*

Keywords: Parking System; Greedy Algorithm; Database SQL; Vehicle; Congestion

I. INTRODUCTION

Currently, a significant portion of the population owns vehicles as a means of transportation to various destinations (Sefriyadi et al., 2023). These vehicles can be categorized as either public or private. However, vehicles cannot operate continuously without intermittent stops. There are situations where vehicles need to be temporarily deactivated, such as during brief stops for shopping or longer periods for parking. Unfortunately, challenges frequently arise within existing parking systems, including prolonged searches for available parking spaces, confusing signage, and uncertainty in finding suitable locations (Elfaki et al., 2023).
High demand for parking among drivers can lead to congestion in parking areas (Tanaka, S., Ohno, S., & Nakamura, F., 2017). To address these challenges and improve parking availability, the implementation of smart parking systems is essential (Koumetio

Tekouabou et al., 2022). Such systems can quickly identify available parking spaces and allocate appropriate slots, thereby saving fuel and reducing the time spent searching for parking.

In the study titled *The Smart Parking Management System* by Elsonbaty and Shams (2020), the proposed solution focuses on integrating smart parking solutions by utilizing Internet of Things (IoT) technologies, specifically Arduino sensors and mobile phones, to detect vacant parking spaces. Conversely, our planned system development will employ a web-based greedy algorithm.

Our objective is to create an application system designed for parking lot management, starting with the Binus Kemanggisan parking area, specifically at the Anggrek campus. This system will streamline and simplify the parking process for drivers by incorporating data such as vehicle license plate numbers from parking attendants, enabling the system to identify and automatically allocate vacant parking spaces. Consequently, the system will generate outputs in the form of slot numbers and designated parking areas for customers.

In recent years, significant advancements have been made in smart parking systems utilizing various techniques and strategies. One prominent method is the greedy algorithm, which has been applied not only in smart parking management systems but also in other research domains. For instance, Fahmi et al. (2020) analyzed the use of Dijkstra's Algorithm in food production and achieved impressive results. Their analysis demonstrated that a trip from point A to point H, aiming to pass through as many cities as possible while minimizing distance, covered 648 kilometers and passed through five cities. Additionally, Al Aqel et al. (2019) proposed a modified iterated greedy algorithm for flexible job shop scheduling. While this approach yielded relatively optimal results, its effectiveness was considered less suitable for scheduling problems due to its emphasis on speed over accuracy.

With the increasing adoption of Dijkstra's Algorithm, researchers have begun exploring its potential in parking management systems. Al Aqel et al. (2019) introduced a smart parking system that identifies the nearest parking slot based on entrance tickets. This system aims to enhance driver parking efficiency by minimizing the time spent searching for available slots. Furthermore, the Ant Colony Optimization (ACO) Algorithm has proven valuable in parking systems by facilitating the discovery of parking spots, reducing fuel consumption, and search time. Gupta and Rani (2020) developed a smart parking management system using machine learning and IoT concepts. This system utilizes Passive Infra-Red (PIR) sensors and RFID scanners to detect vacant parking slots, which are then shared with the server and displayed on user devices. Moreover, parking allocation models incorporating Directed Acrylic Graph (DAG) and distributed ledger technology based on peer-to-peer networks have shown promise in ensuring security and enabling adaptive pricing for parking requests, benefiting both users and parking space owners.

Thus, the implementation of the Greedy Algorithm has proven effective, yielding optimal results with fast computation time (Wayahdi, M., Ginting, S., & Syahputra, D., 2021). Leveraging the capabilities of this algorithm, we can address various real-life challenges, such as the search for available parking slots. The primary objective of these systems is to simplify the process of finding parking spaces, thereby reducing congestion and enhancing the overall experience and security for drivers.

II. METHODS

In this parking system application, we utilize a greedy algorithm and SQL database system in its implementation. This greedy algorithm aims to find the fastest solution for locating empty parking slots (Wang dan Xie., 2019).

A. Greedy Algorithm

An algorithm is a problem-solving technique commonly used in data calculation and computer operations, employing a step-by-step approach. Among various algorithm variants, some prioritize ease and speed in solving problems, one of which is the Greedy Algorithm. The Greedy Algorithm is an algorithm design used to solve various problems using greedy decision-making techniques, prioritizing speed and convenience for efficiency. Generally, the greedy algorithm can be applied when dealing with a numerical input "N".

Here, "N" represents the input value and determines several prerequisites or conditions that must be met. If the input value meets many conditions by minimizing and maximizing the required value, it can serve as a solution (Aung, S. L., 2019).

B. Database System

A database serves as a compilation of interconnected data intended for the manipulation of information. Within this database structure, tables are typically interlinked, with each table assuming a parent role in relation to the encompassed attributes. The relationships established among these tables are governed by key components of significance. The primary key, denoting the initial key, assumes the role of a unique identifier for attributes within a specific table. Typically manifested as an ID attribute, the primary key is customarily endowed with non-nullability, thereby ensuring the existence of only one instance of data within the table. By doing so, the primary key effectively safeguards against the occurrence of data duplication or processing errors within the database system (Susanto & Meiryani, 2019; Elmasri & Navathe, 2016),

Alternatively, the foreign key, constituting the subsequent key, facilitates the establishment of connections between the primary key table and other tables within a relational database. This interlinkage enables the primary key table to function as a point of reference for distinct data during queries. Unlike the primary key, the foreign key does not hold the primary key designation in other tables due to its repeated utilization. Furthermore, the data type assigned to the foreign key column must correspond with that of the primary key, ensuring data compatibility and integrity (Elmasri, R., & Navathe, S. B., 2015).

C. Flowchart of system Application

The program starts with a choice between searching for a registered ticket or registering a new user. When a parking space is available, the system will create one. If there is no parking space (Failure), the user cannot park his or her vehicle. However, if successful, the system will generate user data in the database (reservation) and allocate parking spaces using the specified algorithm. As a result, the user gets the ticket ID. By entering the ticket ID, users can find the ticket registered using the license plate. After the user provides the ticket ID, the admin can save car time and change the user's status to active. In the active state, the user can park the car in the designated parking lot according to the floor and area indicated on the ticket ID. After the user completes the activity and exits the parking lot, the admin will record the user's time out of the parking lot, and the system will charge the parking fee based on the ticket ID. Additionally, the system automatically deletes user data from the database.

During Flowchart analysis, several problems are identified, and appropriate solutions are implemented. For example, a user who doesn't register but immediately provides a ticket number to the dealer is now handled by a system that allows the administrator to validate the user's license plate number using a database. In case the registered user has not been assigned a parking space, they will be immediately notified that the parking space is no longer available. Also, if a user has already entered the parking space but tries to re-register their vehicle, they will be warned that they are already in the parking space and do not need to re-register. The system application flowchart is shown in Fig. 1.

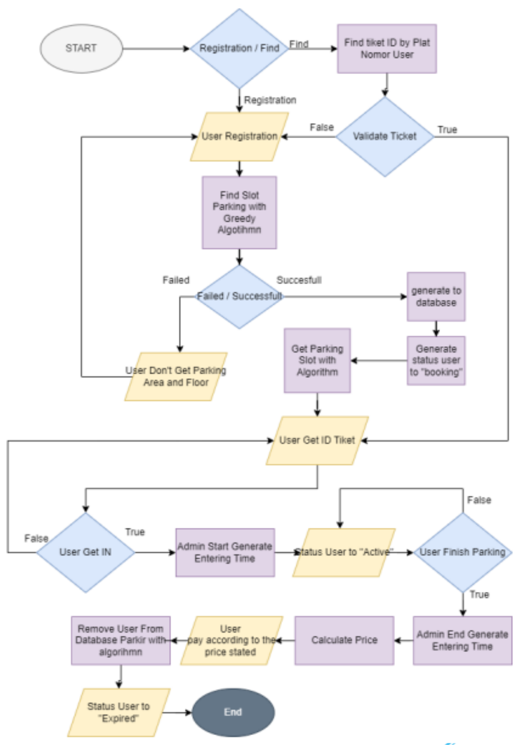

START
Registration / Find
Find
Find tiket ID by Plat Nomor User
Registration
User Registration
False
Validate Ticket
True
Find Slot Parking with Greedy Algotihmn
Failed
Succesfull
Failed / Successfull
generate to database
Generate status user to "booking"
Get Parking Slot with Algorithm
User Don't Get Parking Area and Floor
User Get ID Tiket
False
User Get IN
True
Admin Start Generate Entering Time
Status User to "Active"
User Finish Parking
False
True
Admin End Generate Entering Time
Calculate Price
User pay according to the price stated
Remove User From Database Parkir with algorihmn
Status User to "Expired"
End

Figure 1 Flowchart of process of system application

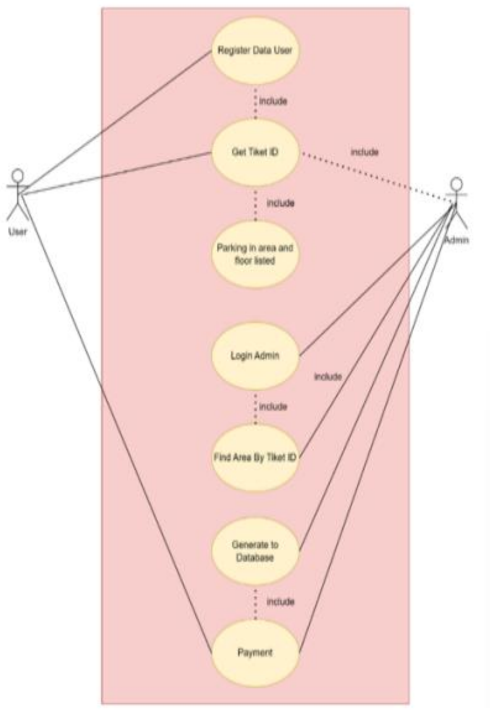


Figure 2 UML of application system design

D. UML Application system design

The initial step for the user involves registering their data. Upon validation of the data and the availability of a parking slot, the user will be issued an ID ticket. Subsequently, when the user intends to exit the parking lot, they will make a payment corresponding to the total price. On the other hand, administrators are required to log in before gaining access to the admin-only page. This page encompasses various functionalities essential for managing the application system, including searching for the area associated with the ticket ID, database management, and the payment system. Flowchart application system design shown in Fig. 2.

E. Database Design

The system incorporates three tables in the database system: the Parking Table, Customer Table, and Ticket Table. The Parking Table is responsible for storing information about parking slots in Binus, including the area, floor, driver ID (foreign key), and ticket ID (foreign key). Figure 3 illustrates the visual representation of the Parking Table. The ID serves as a unique primary key. A parking slot with a ticketID value of NULL indicates that it is unoccupied, while a non-NULL value indicates occupancy.

Table 1 Parking Database

parkirID	tiketID	pengemudiID	area	lantai
1.	5	5	A1	Lantai 1
2.	Null	Null	A1	Lantai 1
3.	Null	Null	A1	Lantai 1
5.	Null	Null	A2	Lantai 2
6.	Null	Null	B2	Lantai 2
7.	Null	Null	B1	Lantai 3
8.	Null	Null	B1	Lantai 3
9.	Null	Null	B2	Lantai 4
10	Null	Null	B2	Lantai 4

e-ISSN: 2686-2573
DOI: 10.21512/emacsjournal.v7i1.12575

The second table, the Customer Table, records driver information provided in the registration form, such as name, license plate, and ticketID (foreign key). This table is presented in Table 2. It captures the details of all successfully registered drivers for available parking slots.

Table 2 Driver Database

pengemudiID	name	platNomor	tiket ID
1.	andre	B 1111 HGD	1
2.	Adi Susanto	B 1111 HGD	2
3.	nicky	B 3333 THW	3
4.	tea	B 1811 UJW	4
5.	raymond	B 1234 QUA	5
Null	Null	Null	Null

The third table, the Ticket Table, stores information pertaining to the driver's entry date, exit date, and ticket status using the ticket ID as the primary key to ensure data integrity. The table is depicted in Table 3. The entry date is recorded when the admin initiates the 'Generate IN' button in the application, while the exit date is captured through the 'Generate OUT' button. The status field changes dynamically based on different situations. The price calculation is derived from the date data, as explained in the working section.

Table 3 Ticket Database

tiketID	tgl_masuk	tgl_keluar	status	harga
1.	3/13/2023 10:3	3/13/2023 1:37	expired	9000
2.	3/13/2023 10:4	3/13/2023 2:41	expired	12000
3.	3/13/2023 11:4	3/13/2023 2:51	expired	9000
4.	3/13/2023 11:5	3/13/2023 2:54	expired	9000
5.	3/13/2023 11:5	Null	active	Null
6.	Null	Null	booking	Null

F. Greedy Algorithm Pseudocode

```
// find parking slots based on floor that have elevator
        For (int I = 0; I < floorHaveElevator; i++)
            If(parking[i] == null)
            Return parking[i];
        Return null;

// input to slot parking & set datetime
        For (int I = 0; I < sizeparkir; i++)
            If(parking[i].tiketID == tiketID)
                Set parking[i] = driver;
        Return null;

// finished parking
        For (int I = 0; I < sizeparkir; i++)
            If(parking[i].tiketID == tiketID)
                CalculateParkingPrice ()
                setUpParkingAreaToAvailable();
```

First, drivers can register themselves in the application by providing their name and license plate details on the registration form. This enables drivers to promptly determine if a parking slot is available for them. The algorithm prioritizes floors with elevators over those without elevators. Second, upon successful registration, the application will display a ticket number on the driver's screen. The driver's information will be stored directly in the parking table, and the ticket status will be updated to "booking." Unsuccessful registrants will be directed to a page notifying them that the parking slots are full. Third, drivers who have successfully registered can inform the officer of their ticket ID. The officer will immediately validate the ticket number against the license plate recorded in the database. If the validation is successful, the attendant will provide information about the designated area and floor for the driver to park. Subsequently, the driver's ticket status will change to "active," indicating their presence in the parking area, and the entry time will commence. When the driver completes parking and intends to leave, they must inform the ticket number to the admin gate. The admin will enter the code into the system, automatically calculating the parking fee as the difference between the entry and exit times multiplied by the parking fee rate. Finally, upon completion of all these steps, the driver's data in the parking system will be reset to NULL, indicating an available parking slot

III. RESULTS AND DISCUSSION

First, the customer must register their name and license plate to obtain their ticket ID. If a parking slot is still available, the ticket ID will be displayed on the screen. If the customer forgets their ticket ID, they can access it by inputting their license plate.

The admin can search for floor locations and customer parking areas based on the ticket ID provided by the user. If the license plate listed in the system matches the customer's, the admin can confirm the validity and proceed to "Action IN" to start the customer's parking time.

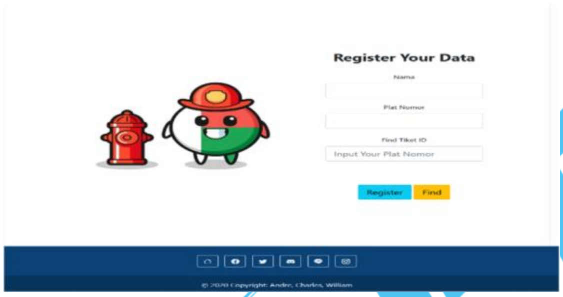
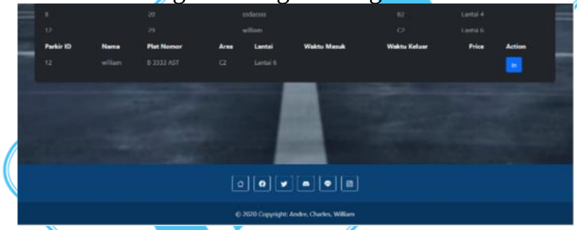


Figure 3 Register Page

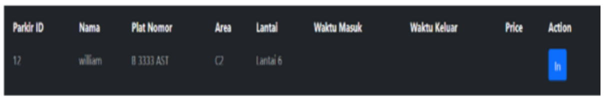


8	20	indarsss		B2	Lantai 4			
17	29	william		C7	Lantai 6			

Parkir ID	Nama	Plat Nomor	Area	Lantai	Waktu Masuk	Waktu Keluar	Price	Action
12	william	B 3333 AST	C2	Lantai 6				In

Figure 4 Admin Page

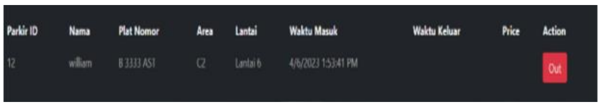
If you click "Action IN" then the customer's parking start time will begin at that time. If you click "Action OUT", the customer's parking time.



Parkir ID	Nama	Plat Nomor	Area	Lantai	Waktu Masuk	Waktu Keluar	Price	Action
12	william	B 3333 AST	C2	Lantai 6				In

Figure 5 Admin Page (Action IN)

Will be calculated according to (parking price x (difference between entry and exit time)).



Parkir ID	Nama	Plat Nomor	Area	Lantai	Waktu Masuk	Waktu Keluar	Price	Action
12	william	B 3333 AST	C2	Lantai 6	4/6/2023 1:53:41 PM			Out

Figure 6 Admin Page (Action IN)

EXAMPLE SIMULATION CASE GREEDY ALGORITHMN PROCESSING.

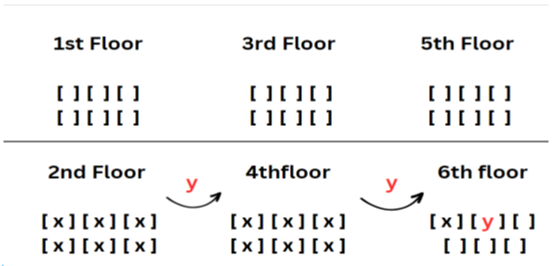


Figure 7 Simulation Case Even Floor

The Greedy algorithm will prioritize even floors first. Initially, it will begin searching from the 2nd floor. However, since the slots on the 2nd floor are already filled, it will proceed to the 4th and 6th floors. It will then check how many seats are available on the 6th floor, and if there are available slots, the car will be parked there. Even floors are prioritized to start the search from lower levels because lower levels have higher priority, enabling quicker access to the parking area. This simulation shown in Fig. 7.

Odd floor Case

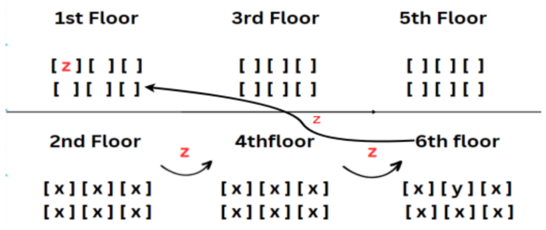


Figure 8 Simulation Case Odd Floor

When the even floor is empty, the greedy algorithm will automatically move to the next odd floor that still has room for parking. However, if even the odd parking situation is filled, the system will return a message indicating that the parking slot is no longer available at this time. This simulation shown in Fig. 8.

IV. CONCLUSION

e-ISSN: 2686-2573
DOI: 10.21512/emacsjournal.v7i1.12575

Based on the findings and discussions, the results indicate that the implementation of the greedy algorithm exhibits notable efficacy in identifying parking slots at Binus Anggrek. This effectiveness can be attributed to the manageable scale of the Binus Anggrek parking facility, comprising an approximate range of 15-16 floors. Moreover, our approach includes a parameterization strategy that prioritizes even floors equipped with elevators, thereby providing expedited access to the campus. Furthermore, the utilization of a SQL database as the designated data storage system significantly enhances the workflow of the parking application, owing to the comprehensive functionality afforded by SQL, including proficient querying and indexing capabilities. However, it is imperative to acknowledge that the greedy algorithm may not yield optimally favorable outcomes in scenarios where the campus encompasses a substantial quantity of parking lots or when drivers possess diverse destinations. Thus, it is crucial to duly consider the specific characteristics of the problem at hand, along with the magnitude of additional data requiring processing.

REFERENCES

Abu-Alsaad, H., & Al-Taie, R. (2024). Smart parking system using IoT. *Proceedings of the 2024 European Conference on Artificial Intelligence (ECAI)*. https://doi.org/10.1109/ECAI61503.2024.10607419.

Al Aqel, G., Li, X., & Gao, L. (2019). A modified iterated greedy algorithm for flexible job shop scheduling problem. *Chinese Journal of Mechanical Engineering, 32*(1). https://doi.org/10.1186/s10033-019-0337-7

Ata, K., Che Soh, A., Ishak, A., Jaafar, H., & Khairuddin, N. (2019). Smart indoor parking system based on Dijkstra's algorithm. *International Journal of Electrical Engineering and Applied Sciences, 2*(1)

Aung, S. L. (2019). Comparative study of dynamic programming and greedy method. *International Journal of Computer Applications Technology and Research, 8*(8), 327–330. https://doi.org/10.7753/ijcatr0808.1007

Elmasri, R., & Navathe, S. B. (2015). *Database systems* (7th ed.). Pearson.

Elsonbaty, A., & Shams, M. (2020). The smart parking management system. *arXiv*. https://doi.org/10.48550/arXiv.2009.13443

Elfaki, A. O., Messoudi, W., Bushnag, A., Abuzneid, S., & Alhmiedat, T. (2023). A smart real-time parking control and monitoring system. *Sensors, 23*(24), 9741. https://doi.org/10.3390/s23249741

Fahmi, H., Zarlis, M., Nababan, E. B., & Sihombing, P. (2020). Implementation of the greedy algorithm to determine the nearest route search in distributing food production. *IOP Conference Series: Materials Science and Engineering, 769*(1). https://doi.org/10.1088/1757-899X/769/1/012005

Gupta, R. K., & Rani, G. (2020). Machine learning and IoT-based real-time parking system: Challenges and implementation. *3rd International Conference on Innovative Computing and Communication (ICICC-2020)*. Retrieved from https://ssrn.com/abstract=3563377

Koumetio Tekouabou, S. C., Abdellaoui Alaoui, E. A., Cherif, W., & Silkan, H. (2022). Improving parking availability prediction in smart cities with IoT and ensemble-based model. *Journal of King Saud University -*

Computer and Information Sciences, 34(3), 687–697. https://doi.org/10.1016/j.jksuci.2020.01.008

Sefriyadi, I., Andani, I. G. A., Raditya, A., Belgiawan, P. F., & Windasari, N. A. (2023). Private car ownership in Indonesia: Affecting factors and policy strategies. *Transportation Research Interdisciplinary Perspectives, 19*, 100796. https://doi.org/10.1016/j.trip.2023.100796

Susanto, A., & Meiryani. (2019). Database management system. *International Journal of Scientific & Technology Research.*

Tanaka, S., Ohno, S., & Nakamura, F. (2017). Analysis on drivers' parking lot choice behaviors in expressway rest area. *Transportation Research Procedia, 25*, 1342–1351. https://doi.org/10.1016/j.trpro.2017.05.158

Wang, B., & Xie, K. (2019). Design of open parking lot based on grid-segmentation and greedy algorithm. *World Scientific Research Journal, 5*(9). https://doi.org/10.6911/WSRJ.201909_5(9).0030

Wayahdi, M., Ginting, S., & Syahputra, D. (2021). Greedy, A-Star, and Dijkstra's algorithms in finding shortest path. *International Journal of Advances in Data and Information Systems, 2*(1), 45–52. https://doi.org/10.25008/ijadis.v2i1.1206