A logo of a globe with yellow rings around it

Description automatically generated

**GROUP ASSIGNMENT**

**TECHNOLOGY PARK MALAYSIA**

**AAPP015-4-1-PWP-LAB-3**

**Programming with Python**

**UCDF2304ICT(ITR)/UCDF2304ICT**

**HAND OUT DATE: 30th March 2024**

**HAND IN DATE: 20th May 2024**

**MODULE LECTURER: Wong Keng Tuck**

**INSTRUCTION:**

1. **Submit your assignment in the APU Moodle.**
2. **Students are advised to underpin their answers with the use of references (cited using the APA System of Referencing).**
3. **Late submission will be awarded zero (0) unless Extenuating Circumstances (EC) are upheld.**
4. **Cases of plagiarism will be penalized.**

|  |  |
| --- | --- |
| **Student Name** | **TP Number** |
| Ng Wei Jian | TP074500 |
| Tan Hui An | TP068270 |
| Eric Chan Yit Yuan | TP074253 |
| Ng Jun Yuan | TP067314 |

Table of Contents

[Introduction and assumptions 3](#_Toc167041939)

[Design of the program 4](#_Toc167041940)

[Pseudocode 4](#_Toc167041941)

[Flowcharts 25](#_Toc167041942)

[Programming concepts applied in program 32](#_Toc167041943)

[Screenshot of the program input and output 33](#_Toc167041944)

[Conclusion 43](#_Toc167041945)

[Workload matrix 44](#_Toc167041946)

[References 45](#_Toc167041947)

# Introduction and assumptions

This system is a car rental system and design by python as the programming language with txt file as acting the car rental system database.

The car rental system should be able to perform those function such as:

* Login and logout all staff and customer account.
* Register and update customer data records.
* Staff can update profile by themselves after login.
* Manager can view monthly revenue report based on car rental transactions.
* View a list of registered customers.
* Customer can check all car from availability to rental.
* Save and update the rental availability for all cars with rental retails from.
* Accept and record rental details from customers.
* Print receipt to customers who rental car.
* Register new car, update car information, and delete the car which is no exist.
* Update the rental availability for all cars.
* Use .txt and .json acting database which not need SQL sever.

# Design of the program

## Pseudocode

##manager register staff

start

print ("enter staff\_id, username, staff\_role, password, date\_of\_register")

input staff id, username, staff\_role, password, date\_of\_register

print (“role list”)

input role list number

print ("register complete")

end

## login page

start

print("input username, password")

get username, password

if username, password true and Staff Role == 'Manager'

print ongoing

if username, password true and Staff Role =='Customer Service Staff I'

login to Customer Service Staff II menu

if username, password true and Staff Role =='Customer Service Staff II’

login to Customer Service Staff II menu

if username, password true and Staff Role ==’Car Service Staff’

login to Customer Service Staff II menu

else:

Back to Main menu

end

## Main menu

Start

print("1. Login")

print("2. Register")

print("3. Exit")

get number

if number == "1":

open login page

elif choice == "2":

open register page

elif choice == "3":

print Goodbye!

else:

print ("Invalid choice. Please try again.")

end

##staff login in Customer Service Staff I

start

attempt=1

print ("enter staff\_id, username, staff\_role, password")

input staff\_id, username, staff\_role, password

if input data= "correct data"

return True

elif: attempt+1

print("please try again")

else attempt ==3

print("maximum attempt reached.")

return False

output choose staff\_role

input staff\_role

print ("Welcome Staff")

end

##Register Customer (staff 1)

start

input customer information

Customer\_ID = new\_id

Customer\_Name = input("Please Enter Your Name:")

Customer\_NRIC = input(int("Please Enter Your NRIC(Malaysian), example: xxxxxx-xx-xxxx. :"))

Customer\_PassportNo = input(int("Please Enter Your Passport Number(Foreigner Only), example: PA1234567890X. :"))

Customer\_LicenseNo = input(int("Please Enter Your Car Dring License Number, example: D12345678. :"))

Customer\_Address = input("Please Enter Your Address:")

Customer\_PhoneNo = input(int("Please Enter A Valid Phone Number, example: 011-41237894. :"))

Others\_info = input("Do let us know if there is other information:")

Customer\_Password = input("Please Create A Password:")

Registration\_Date = date.today()

get customer information

print ("Register Complete")

end

##customer request (staff 2)

start

print("Request form customer.")

get customer request

print("available car to rent.")

get available car

print("result of the request.")

end

##View rental transactions

start

print ("View rental transaction")

output list of the rental transaction

if transaction == true

print ("transaction detail")

else:

print ("No transaction found.")

end

##cancel rental transaction

start

print ("Enter decision")

get decision

if 1 <= transaction\_number:

cancel\_transaction = rental\_car(transaction\_number - 1)]

car\_rental\_status = Available

print("Transaction cancel successful")

retuan true

else:

print ("Transaction not found")

return false

end

##add new car to system

start

input car details

car = {

"make": make,

"model": model,

"daily\_rate": daily\_rate,

"reg\_number": reg\_number,

"rental\_status": "Available"

}

print ("Car detail“: car)

end

##rent car

start

rent\_car(cars, rentals, car\_index, rental\_date, return\_date, Customer\_ID):

if car["rental\_status"] == "Reserved":

print("Car is reserved and cannot be rented at the moment.")

return none

elif car["rental\_status"] == "Available":

print("Car is available and ready to be rent.")

else:

print("No car is available for rent.")

return none

end

##return rented car

start

return\_car(cars, rentals, car\_index):

car = cars[car\_index]

for rental in rentals:

if rental["car"] == car:

car["rental\_status"] = "Available"

rentals.remove(rental)

return rental

print("Car was not rented out.")

return None

end

##staff login

##car service staff

start

attempt=1

print ("enter staff\_id, username, staff\_role, password")

input staff\_id, username, staff\_role, password

if input data= "correct data"

return True

elif: attempt+1

print("please try again")

else attempt ==3

print("maximum attempt reached.")

return False

output choose staff\_role

input staff\_role

print ("Welcome Staff")

end

##Register new car

start

print("input new car to system")

Car Registration Number= input("Car Registration Number: ")

Car Manufacturer= input("Car Manufacturer: ")

Car Model= input("Car Model:")

Year of Manufacturer= input(int(year:"))

Seating Capacity= input(int(Capacity:"))

Last Service Date= input(int(Date DD/MM/YYYY:“))

Insurance Policy Number= input(Insurance Policy Number:")

Insurance Expiry Date= input(int(Date DD/MM/YYYY:"))

Road Tax Expiry Date= intput(int(Date DD/MM/YYYY:"))

Get Information of the new car

print("Information of the new car.")

end

##update car detail

start

print("Update car information")

get car plate number from customer

if car\_plate\_no exist in database:

print("select field to update:"

Print "1. Insurance Policy Number"

Print "2. Insurance Expiry Date"

Print "3. Road Tax Expiry Date"

Print "4. Rental Availability"

Print "5. Exit, cancel update"

Get customer choice

if choice '1'

get new insurance policy number from customer

update insurance policy number for car\_plate\_no in database

print("New information update successful for car\_plate\_no.")

elif choice '2'

get new insurance expiry date from customer

update insurance expiry date for car\_plate\_no in database

print("New information update successful for car\_plate\_no.")

elif choice '3'

get new road tax expiry date from customer

update road tax expiry date for car\_plate\_no in database

print("New information update successful for car\_plate\_no.")

elif choice '4'

Print "Select the Rental Availability:"

Print "1. Available"

Print "2. Reserved"

Print "3. Rented"

Print "4. Under Service"

Print "5. Disposed"

get choice from customer

if availability\_choice is '1':

Update Rental Availability to "Available" in databse for car\_plate\_no

Print "Rental Availability updated to Available successfully!"

elif availability\_choice is '2':

Get Requested Date from customer

Update Rental Availability to "Requested Date: requested\_date" in database for car\_plate\_no

Print "Requested Date updated successfully!"

elif availability\_choice is '3':

Get Rent Date from customer

Get Returning Date from customer

Update Rental Availability to "Request Day Rented: rent\_date, Returning Date: return\_date" in database for car\_plate\_no

Print "Request Day Rented and Returning Date updated successfully!"

elif availability\_choice is '4':

Get New Last Service Date from customer

Update Rental Availability to "Under Service: last\_service\_date" in database for car\_plate\_no

Print "Last Service Date updated successfully!"

elif availability\_choice is '5':

Update Rental Availability to "Disposed" in database for car\_plate\_no

Print "Rental Availability updated to Disposed successfully!"

else: print("Invalid choice. Please enter number from 1-5.")

elif choice '5':

break loop and exit update function

else: print("Invalid choice, Please enter number 1-5."

else: print("Car plate number not found.")

update database

end

##view list of cars

start

print("View list of cars")

input car\_plate\_no

output car\_plate\_no information

print("information.")

end

##delete disposed car

start

print("delete car where availability is 'disposed'")

get car\_plate\_no

If car\_plate\_no exists in database:

If car rental\_availability is "Disposed":

Delete car from database

Print "Car with plate number car\_plate\_no deleted successfully!"

Else:

Print "Car is not in 'Disposed' status, cannot delete."

Else:

Print "Car Plate Number not found."

update database with new data

end

start

import re

import datetime

# Create an empty text file for car\_database

set car\_db\_file TO "car\_database.txt"

open car\_db\_file AS file IN "write" mode

close file

set car\_database to empty dictionary

# Function to validate date format (mm/dd/yyyy)

function is\_valid\_date(date\_str):

try:

parse date\_str with format '%m/%d/%Y'

Return True

except ValueError:

Return False

# Specify the file path where you want to write the JSON data

set file\_path to "Car\_Database.txt"

# Function to check if a given plate is a valid Malaysian plate

function is\_valid\_malaysian\_plate(plate):

Return True IF plate MATCHES '[A-Z]{1,3}\d{1,4}[A-Z]{0,1}' ELSE False

# Function for registering a car

Function register\_car ():

print "Register Car"

input car\_plate\_no AS "Enter Car Plate Number: "

if not is\_valid\_malaysian\_plate(car\_plate\_no):

print "Invalid Malaysian car plate format."

return

if car\_plate\_no IN car\_database:

print "Car Plate Number already exists."

return

input manufacturer as “Enter Car Manufacturer: "

# Validate Year of Manufacture

while True:

input year\_of\_manufacture as "Enter Year of Manufacture (YYYY): "

if year\_of\_manufacture is not a four-digit number:

print "Invalid input."

else:

break

# Validate Seating Capacity

while True:

input seating\_capacity as "Enter Seating Capacity: "

if seating\_capacity is not a number:

print "Invalid input."

else:

break

# Validate Last Service Date

while True:

input last\_service\_date as "Enter Last Service Date (mm/dd/yyyy): "

if not is\_valid\_date(last\_service\_date):

print “Invalid input."

else:

parse last\_service\_date as date

parse year\_of\_manufacture as year\_date

if date is before year\_date:

print"Last Service Date cannot be before the Year of Manufacture."

else:

break

# Validate Insurance Expiry Date

whlie True:

input insurance\_expiry\_date as "Enter Insurance Expiry Date (mm/dd/yyyy): "

if not is\_valid\_date(insurance\_expiry\_date):

print "Invalid input."

else:

parse insurance\_expiry\_date as date

if date is before year\_date:

print "Insurance Expiry Date cannot be before the Year of Manufacture."

else:

break

# Validate Road Tax Expiry Date

whlie True:

input road\_tax\_expiry as"Enter Road Tax Expiry Date (mm/dd/yyyy): "

if not is\_valid\_date(road\_tax\_expiry):

print "Invalid input."

else:

pares road\_tax\_expiry as date

if date is before year\_date:

print "Road Tax Expiry Date cannot be before the Year of Manufacture."

else:

break

# Add Policy Number

input policy\_number as "Enter Policy Number: "

set car\_database[car\_plate\_no] to {

"Manufacturer": manufacturer,

"Year of Manufacture": year\_of\_manufacture,

"Seating Capacity": seating\_capacity,

"Last Service Date": last\_service\_date,

"Insurance Expiry Date": insurance\_expiry\_date,

"Road Tax Expiry": road\_tax\_expiry,

"Policy Number": policy\_number

}

# Write the car data to the "Car\_Database.txt" file

open "Car\_Database.txt" AS car\_file in "append" mode

write car\_plate\_no, manufacturer, year\_of\_manufacture, seating\_capacity, last\_service\_date, insurance\_expiry\_date, road\_tax\_expiry, policy\_number to car\_file

close car\_file

print"Car registration successful!"

# Function for updating car information

function update\_car():

while True:

print "Update Car Information"

input car\_plate\_no as "Enter Car Plate Number to update: "

if car\_plate\_no IN car\_database:

print "Select the field to update:"

print "1. Insurance Policy Number"

print "2. Insurance Expiry Date"

print "3. Road Tax Expiry Date"

print"4. Car Renting Rate per day"

print"5. Rental Availability"

print "6. Last Service Date"

print"7. Requested Date (for Reserved)"

print"8. Request Day Rented and Returning Date (for Rented)"

print "9. Exit"

input choice as "Enter your choice: "

if choice is '1':

input insurance\_policy\_no as "Enter New Insurance Policy Number: "

set car\_database[car\_plate\_no]["Insurance Policy Number"] to insurance\_policy\_no

print "Insurance Policy Number updated successfully!"

else if choice is '2':

input insurance\_expiry\_date as "Enter New Insurance Expiry Date: "

set car\_database[car\_plate\_no]["Insurance Expiry Date"] to insurance\_expiry\_date

print "Insurance Expiry Date updated successfully!"

else if choice is '3':

input road\_tax\_expiry as "Enter New Road Tax Expiry Date: "

set car\_database[car\_plate\_no]["Road Tax Expiry"] to road\_tax\_expiry

print "Road Tax Expiry Date updated successfully!"

else if choice is '4':

input renting\_rate as "Enter New Car Renting Rate per day: "

set car\_database[car\_plate\_no]["Renting Rate"] to renting\_rate

print "Car Renting Rate per day updated successfully!"

else if choice is '5':

print "Select the Rental Availability:"

print "1. Available"

print "2. Requested Date (for Reserved)"

print"3. Request Day Rented and Returning Date (for Rented)"

print "4. Under Service"

print "5. Disposed"

input availability\_choice as "Enter your choice: "

if availability\_choice is '1':

set car\_database[car\_plate\_no]["Rental Availability"] to "Available"

print "Rental Availability updated to Available successfully!"

else if availability\_choice IS '2':

input requested\_date AS "Enter Requested Date (for Reserved): "

set car\_database[car\_plate\_no]["Rental Availability"] to "Requested Date: " + requested\_date

print “Requested Date updated successfully!"

else if availability\_choice IS '3':

input rent\_date as "Enter Rent Date: "

input return\_date as "Enter Returning Date: "

set car\_database[car\_plate\_no]["Rental Availability"] to "Request Day Rented: " + rent\_date + ", Returning Date: " + return\_date

print "Request Day Rented and Returning Date updated successfully!"

else ifavailability\_choice is '4':

input last\_service\_date as"Enter New Last Service Date: "

set car\_database[car\_plate\_no]["Rental Availability"] to "Under Service: " + last\_service\_date

print "Last Service Date updated successfully!"

else if availability\_choice IS '5':

set car\_database[car\_plate\_no]["Rental Availability"] to "Disposed"

print "Rental Availability updated to Disposed successfully!"

else:

print "Invalid choice. Please enter a number from 1 to 5."

else if choice IS '6':

input last\_service\_date AS "Enter New Last Service Date: "

set car\_database[car\_plate\_no]["Last Service Date"] to last\_service\_date

print"Last Service Date updated successfully!"

else if choice is '7':

Input requested\_date as "Enter Requested Date (for Reserved): "

set car\_database[car\_plate\_no]["Rental Availability"] to "Requested Date: " + requested\_date

print "Requested Date updated successfully!"

else if choice IS '8':

input rent\_date AS "Enter Rent Date: "

input return\_date AS "Enter Returning Date: "

set car\_database[car\_plate\_no]["Rental Availability"] to "Request Day Rented: " + rent\_date + ", Returning Date: " + return\_date

print "Request Day Rented and Returning Date updated successfully!"

else if choice IS '9':

open user\_menu()

else:

print "Invalid choice. Please enter a number from 1 to 8."

# Update the Car\_Database.txt file

open "Car\_Database.txt" AS car\_file IN "write" MODE

for car\_plate\_no, car\_info IN car\_database:

write car\_plate\_no, car\_info["Manufacturer"], car\_info["Year of Manufacture"], car\_info["Seating Capacity"], car\_info["Last Service Date"], car\_info["Insurance Policy Number"], car\_info["Insurance Expiry Date"], car\_info["Road Tax Expiry"], car\_info.get("Renting Rate", ""), car\_info.get("Rental Availability", "") to car\_file

close car\_file

# Function for viewing car information

function view\_car():

print "View Car Information"

# Read data from the Car\_Database.txt file

try:

open "Car\_Database.txt" as car\_file in "read" mode

for each line in car\_file:

set line by ',' into car\_data

set car\_plate\_no to car\_data[0]

set manufacturer to car\_data[1]

set year\_of\_manufacture to car\_data[2]

set seating\_capacity to car\_data[3]

set last\_service\_date to car\_data[4]

set insurance\_policy\_no TO car\_data[5]

set insurance\_expiry\_date TO car\_data[6]

set road\_tax\_expiry TO car\_data[7]

set renting\_rate TO car\_data[8] IF car\_data LENGTH > 8 ELSE "Not Available"

set rental\_availability TO car\_data[9] IF car\_data LENGTH > 9 ELSE "Not Available"

print "Car Plate Number: " + car\_plate\_no

print “Manufacturer: " + manufacturer

print "Year of Manufacture: " + year\_of\_manufacture

print "Seating Capacity: " + seating\_capacity

print "Last Service Date: " + last\_service\_date

print "Insurance Policy Number: " + insurance\_policy\_no

print "Insurance Expiry Date: " + insurance\_expiry\_date

print "Road Tax Expiry: " + road\_tax\_expiry

print "Renting Rate per day: " + renting\_rate

print "Rental Availability: " + rental\_availability

print ""

close car\_file

except FileNotFoundError:

print "Car database file not found."

# Function for deleting a car

function delete\_car():

print "Delete Car"

input car\_plate\_no AS "Enter Car Plate Number to delete: "

if car\_plate\_no IN car\_database:

if car\_database[car\_plate\_no]["Rental Availability"] IS "Disposed":

delete car\_database[car\_plate\_no]

print "Car " + car\_plate\_no + " deleted successfully!"

else:

print "Car " + car\_plate\_no + " cannot be deleted because it is not in 'Disposed' status."

# Update the Car\_Database.txt file

open "Car\_Database.txt" as car\_file in “write" mode

for car\_plate, car\_info in car\_database:

write car\_plate, car\_info["Manufacturer"], car\_info["Year of Manufacture"], car\_info["Seating Capacity"], car\_info["Last Service Date"], car\_info["Insurance Policy Number"], car\_info["Insurance Expiry Date"], car\_info["Road Tax Expiry"], car\_info.get("Renting Rate", ""), car\_info.get("Rental Availability", "") TO car\_file

cllose car\_file

else:

print "Car Plate Number not found."

# Main function for user menu

function user\_menu ():

while True:

print "User Menu:"

print "1. Register Car"

print "2. Update Car"

print "3. View Car"

print "4. Delete Car"

print "5. Logout"

input choice as "Enter your choice: "

if choice ‘1':

open register\_car ()

else ifchoice '2':

open update\_car()

else if choice '3':

open view\_car()

else if choice '4':

open delete\_car()

else if choice ‘5':

print"Logging out. Goodbye!"

break

else:

print “Invalid choice. Please enter a number from 1 to 5."

open user\_menu() again

end

## Flowcharts

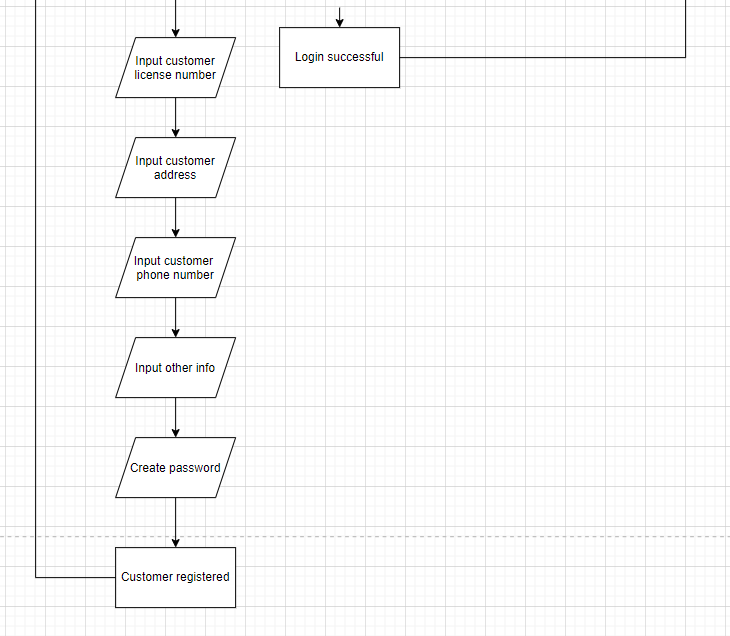
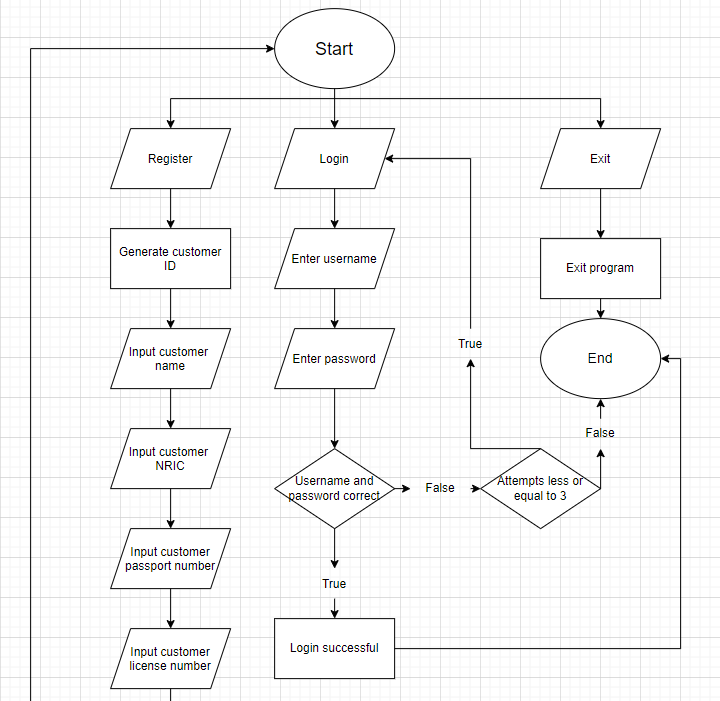
Manager

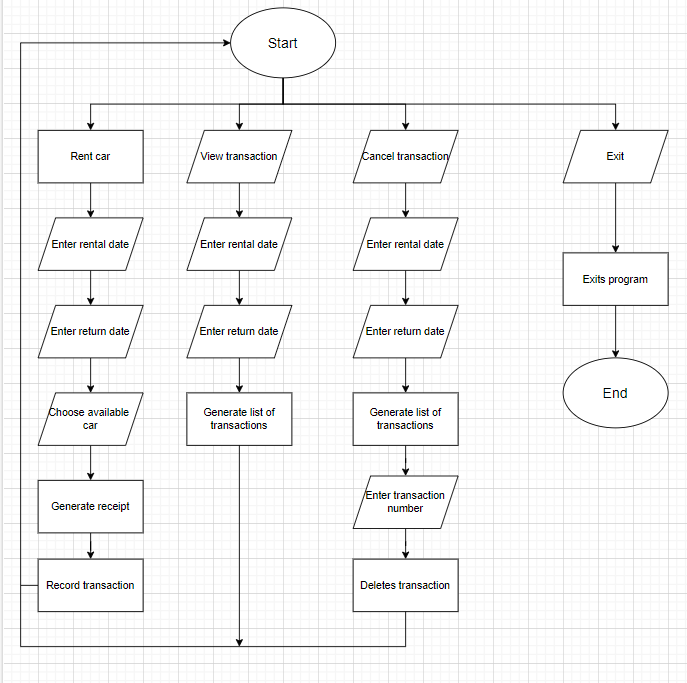
A diagram of a process

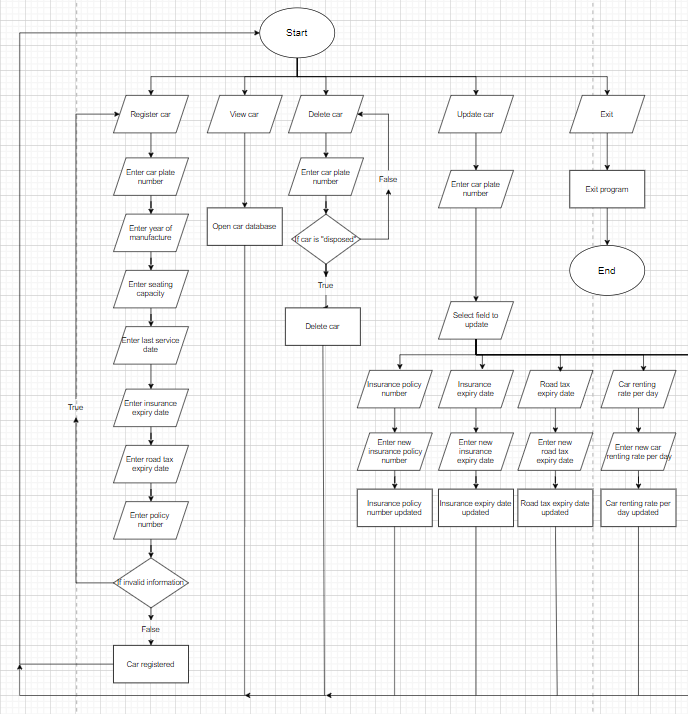
Description automatically generated

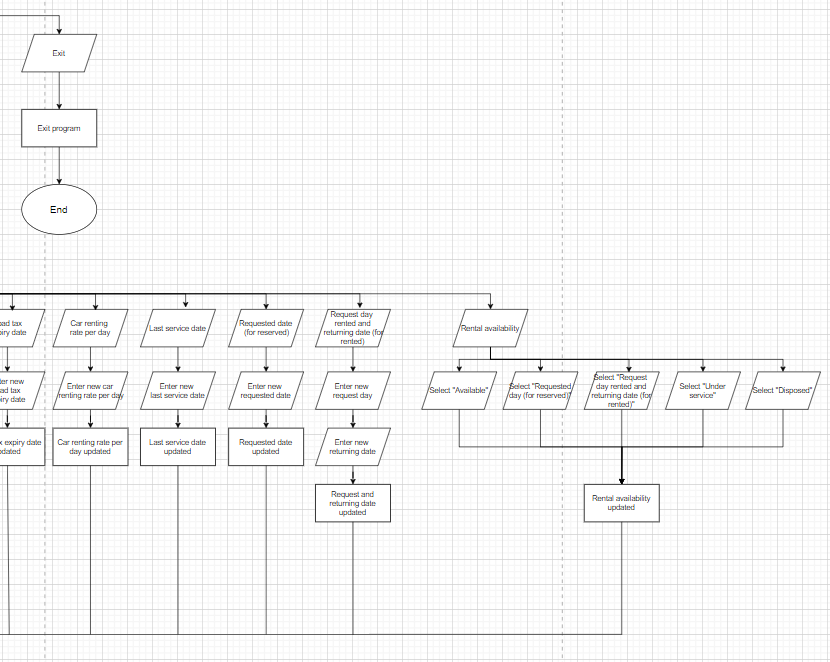
A diagram of a service

Description automatically generated

Customer Service I

Customer Service 2

Car Service



# Programming concepts applied in program

The programming concepts for the program are:

1. Set the date and time automatic from datetime using import date, timedelta.

2. Set txt file acting as database.

3. Using dictionary form in database.

4. View database information using read function for the corresponding database. Example: ("Staff\_record.txt", "r")

5. Write database information into the for the corresponding database. Example: ("Staff\_record.txt", "a").

6. Using re.fullmatch function the set the format of the car plate. Example: def is\_valid\_malaysian\_plate(plate: str) -> bool: return re.fullmatch(r'[A-Z]{1,3}\d{1,4}[A-Z]{0,1}', plate) is not None

7.Using strftime to set the format of the date. Example: Date\_of\_register.strftime("%Y-%m-%d")

8.Using strptime to set the format of the date. Example: insurance\_expiry\_datetime = datetime.strptime(insurance\_expiry\_date, "%m/%d/%Y")

9.Use if statement to filer the terms of the function to different action. Example:

if int(Staff\_Role)>3:

print("Invalid Staff role, please try again.")

register()

elif int(Staff\_Role)<0:

print("Invalid Staff role, please try again.")

register()

else:

Staff\_Role = Role\_list[Staff\_Role]

10. Use compare function to filer the terms of the function to different action. Example

if choice == "1":

login()

# Screenshot of the program input and output

A screenshot of a computer

Description automatically generated

Figure 1: Register staff function in Manager.

A screenshot of a computer

Description automatically generated

Figure 2: The record of the staff in Staff\_record.txt

A screenshot of a computer

Description automatically generated

Figure 3: Invalid staff role outcome retry register new staff.

When the manager register staff, manager need set a Staff ID, Staff name, Staff role, and password. Date of register will be generated by itself.

To avoid spelling errors causing program problems, numbers are used here to select different Roles of staff. If the numbers chosen out for the value in Role\_list, then manager get the message for invalid staff role and retry register the staff again which show in figure 3.

After input all information needed, then the information will by write into staff\_record.txt like figure 2 and info manager staff created successfully like figure 1.

A screenshot of a computer program

Description automatically generated

Figure 4: Customer registration in Customer Service Staff I (did not work)

The customer ID will be generated by itself and the other Customer Name, Customer NRIC, Customer Passport NO, Customer License No, Customer Address, Customer, Phone No, Others info and Customer Password need set by Customer Service Staff I’s staff. Date of register will be generated by itself.

After input all information needed, then the information will by write into staff\_record.txt and info Customer Service Staff I’s staff created successfully.

A computer screen with a white background

Description automatically generated

Figure 5: Customer need use customer account to login with 3-time limits. (did not work)

When the customer login, the program counts one attempt by default. If customer input the right username and password, that customer can into the system. If customer input the wrong username or password, the value for the attempts will be plus one. If the user input the wrong username or password three times, that customer cannot attempts anymore to login my customer themself.

A computer screen with text

Description automatically generated

Figure 6: The main menu of Customer Service Staff I

In figure 6, staff need input a number to choose the function on the main menu.

A computer screen with a white background

Description automatically generated

Figure 7: The program ask input again due to value out of range (did not work)

In figure 7, If the value of the number out of the main menu show, the program will ask input again.

A screenshot of a computer

Description automatically generated

Figure 8: The record for the transactions.

A screenshot of a computer

Description automatically generated

Figure 9: The Customer Service Staff II’s staff can input the date for view the transactions.

In figure 9, staff can input any date to check which date have transactions.

A screenshot of a computer

Description automatically generated

Figure 10: No transaction in that date.

In figure 10, it occurs when the Figure 8 have not any transaction record.

A screenshot of a computer

Description automatically generated

Figure 11: staff can choose a car base form list displayed.

A screenshot of a computer

Description automatically generated

Figure 12: The CustomerSerivce2.txt can add new transaction record.

Form Figure 11, after done the rent car registration, it will show the rental transaction record with car registration number, Customer ID, Rental Date, Return Date, Rental Periods and Total Rental. That information also saves into Figure 12 at the same time.

A screenshot of a computer

Description automatically generated

Figure 13: The cancellation of the rental transaction function

A screenshot of a computer

Description automatically generated

Figure 14: CustomerService2.txt after the rental transaction cancellation

In Figure 13, staff need input the date of the rental transaction which needs to be cancelled, then staff input the number of the transaction want to be cancelled. If the transaction can be cancelled, program will show the message for cancelled successfully and the rental transaction in CustomerService2.txt Figure 14 will be deleted.

A screenshot of a computer

Description automatically generated

Figure 15: The register of the new car function in Car Service

A screenshot of a computer

Description automatically generated

Figure 16: The car database after register

Car Service Staff’s staff can add a new car record by Register Car in menu show in Figure 15, as the program only can input car plate in the right format in Malaysia. Last service date, insurance expiry date, road tax expiry date can be accepted by program if the date is after the car manufacture year. Figure 16 show the car information in Figure 15 is write and save into car\_database.txt.

A screenshot of a computer

Description automatically generated

Figure 17: The update function in Car Service (did not work)

The Update Car Information design to update Insurance Policy Number, Insurance Expiry Date, Road Tax Expiry Date, Car Renting Rate per day, Rental Availability, Last Service Date, Requested Date (for Reserved) and Request Day Rented and Returning Date (for Rented).

A screenshot of a computer screen

Description automatically generated

Figure 18: View car in Car Service menu

In Figure 18, it shows the recorded car information in car\_database.txt.

A screenshot of a computer program

Description automatically generated

Figure 19: Delete car function in View car in Car Service menu. (did not work)

The delete car function is delete the car recorded in car\_database.txt.

# Conclusion

In this assignment, we learn about the how to complete design program more efficient. At the same time, it is also very important for everyone to understand the codes of their respective roles.

The knows issues in our program are:

* The Update Car function did not work on the Car Service, only show the info to charge.
* Customer registration in Customer Service Staff I cannot be use due to customer ID issue.
* Rent a Car in Customer Service Staff II, the date can set without logic.
* Customer login in Customer Service Staff I did not work as Customer registration in Customer Service Staff I cannot run and save account.
* In Customer Service Staff II and Manager, if input get the wrong data type in some input, the program will get error.
* Delete Car in Car Service cannot run and get error.
* car\_database.txt data will be erasing every time when the program rerun. (Can save after exit the program, but when next time start the program data will be clear.)

The following function in our program we do not have:

* All roles have not function to update own profile.
* Manager cannot view monthly revenue report based on car rental transactions.
* The database forms each role cannot attach those data needed in each role due to design issues.
* The status for the staff cannot be check.
* Customer Service Staff II car record is not shared with Car Service. (Renting Rent per day and Rental Availability data is come from Customer Service Staff II)
* Car Service car record also not shared car record to Customer Service Staff II (After Car Service’s Staff add a new car, Customer Service Staff II did not information about the new car)
* Rental availability update in program is not designed.
* The print receipt to customers who rental car did not design.

From the above summary, we still have a lot to improve in our python language expression and the design on this program.

# Workload matrix

|  |  |  |
| --- | --- | --- |
| Student name | TP number | Workload |
| Ng Wei Jian | TP074500 | Code for Customer Service Staff II and flowchart in document. |
| Tan Hui An | TP068270 | Code for Customer Service Staff I and pseudocode in document. |
| Eric Chan Yit Yuan | TP074253 | Code for Car Service Staff and Screenshot of the program input and output in document. |
| Ng Jun Yuan | TP067314 | Code for Manger, Introduction and assumptions Programming concepts applied in program and Conclusion in document. |

# References

geeksforgeeks. (16 March, 2021). Retrieved from re.fullmatch() function in Python: https://www.geeksforgeeks.org/re-fullmatch-function-in-python/

hundredvisionsguy. (11 February, 2015). *Youtube*. Retrieved from Python List Tricks A Menu of Items Part 1: https://www.youtube.com/watch?v=bflhiO5oJn4

Paul, O. (19 May, 2021). *Youtube*. Retrieved from Python Login System: Using a text file (Beginners Project): https://www.youtube.com/watch?v=dR\_cDapPWyY&t=1766s

Principles, P. (n.d.). *Python Principles*. Retrieved from Python Programming Reference Sheet: https://pythonprinciples.com/reference/

*PythonGuides.com*. (1 March , 2023). Retrieved from Python select from a list + Examples: https://pythonguides.com/python-select-from-a-list/

*stackoverflow*. (8 April, 2020). Retrieved from Reading operations line by line and making eval() for each line: https://stackoverflow.com/questions/61097144/reading-operations-line-by-line-and-making-eval-for-each-line

Team, B. S. (2 February, 2023). *Better Stack*. Retrieved from How can I randomly select an item from a list in Python?: https://betterstack.com/community/questions/python-how-to-randomly-select-list-item/