
# Programming with Python Group Assignment Documentation

**Intake Code: UCDF2304ICT(SE)**

**Subject: Programming with Python**

**Title: Programming with Python Group Assignment Documentation**

**Hand-Out Date: 25 March 2024**

**Hand-In Date: 1 June 2024**

---

## GROUP MEMBERS


| Name | TP Number |
|---|---|
| Wong Kap Onn (Group Leader) | TP074292 |
| Wong Zheng Han | TP074212 |
| Tan Yi Han | TP070378 |
| Timothy Tan Chern Tian | TP074658 |
| Tan Yen Hann | TP073629 |

# Table of Content

# 1.0 Introduction and Assumptions (Wong Zheng Han TP074212)

This system is for visitors or customers to view cars type, rental status and explore on variety of cars to rent and mostly it used by the staffs or the programmers. It is assumed that when a customer wants to rent a car, they will have to approach the staffs and rent the car through staffs. The staff will use the system to rent the car for the customer. Thus, the staff will ask customers for their name, username and password for any further login. The staff can also view the cars status and details of car plate, date of manufacture, etc, with these information staff will be able to provide customer information and if the car is available or the next available time. After everything are provided the staff will be contacting customers with the kind of car they are looking for and give them the time for rent start and insurance status.

# 2.0 Design of Program

# 2.1 Pseudocode

car_column = ["CarID", "CarPlate", "Brand", "Model", "ManuYr", "SeatCap", "LastServ", "InsuPolNo", "InsuExp","RoadTaxExp", "RentStart", "RentReturn", "RentRatePD", "Availability"]

cus_column = ["CusID", "CusName", "NRIC", "Passport", "License", "Address", "Phone", "RegDate"]

## Tan Yen Hann TP073629 (Manager)

```
TRY:
    OPEN "users.txt" in read mode
    READ data from file
    IF data is empty:
        OPEN "users.txt" in append mode
        WRITE empty string to file
    CLOSE file
EXCEPT FileNotFoundError:
    OPEN "users.txt" in append mode
```

```
    WRITE empty string to file
    CLOSE file

TRY:
    OPEN "cusinfo.txt" in read mode
    READ data from file
    IF data is empty:
        OPEN "cusinfo.txt" in append mode
        WRITE empty string to file
    CLOSE file
EXCEPT FileNotFoundError:
    OPEN "cusinfo.txt" in append mode
    WRITE empty string to file
    CLOSE file

TRY:
    OPEN "carinfo.txt" in read mode
    READ data from file
    IF data is empty:
        OPEN "carinfo.txt" in append mode
        WRITE empty string to file
    CLOSE file
EXCEPT FileNotFoundError:
    OPEN "carinfo.txt" in append mode
    WRITE empty string to file
    CLOSE file

FUNCTION main_page:
    PRINT welcome message
    PRINT options: 1. Login, 2. Register, 3. Exit
```

```
INPUT choice

IF choice == "1":
    allusers = []
    OPEN "users.txt" in read mode
    FOR each line in file:
        SPLIT line by comma
        ADD split data to allusers
    CLOSE file

    user = CALL login(allusers)
    IF user is not empty:
        IF user role == "car_s":
            CALL crss(user, allusers)
        ELSE IF user role == "cus_s_I":
            CALL cssI_page(user, allusers)
        ELSE IF user role == "cus_s_II":
            CALL cssI_page(user, allusers)
        ELSE IF user role == "manager":
            CALL update_profile(user, allusers)
    ELSE:
        PRINT "System has been disabled for 1 minute"
        SLEEP for 60 seconds

ELSE IF choice == "2":
    CALL register()

ELSE IF choice == "3":
    PRINT "You have exited the program."
    EXIT program
```

```
    ELSE:
        PRINT "Invalid Input."
        CALL main_page()

FUNCTION login(alluser):
    flag = False
    FOR cnt in range(3):
        INPUT username
        INPUT password

        FOR each user in alluser:
            IF username and password match user credentials:
                SET user
                SET flag to True
                BREAK loop

        IF flag is True:
            BREAK loop
        ELSE:
            PRINT "Incorrect username or password. Attempts left:" (2 - cnt)

    IF flag is False:
        SET user to empty list

    RETURN user

FUNCTION register:
    INPUT username
    INPUT name
    INPUT password
```

```
    INPUT reen_pass

    WHILE password != reen_pass:
        PRINT "Password not match"
        INPUT reen_pass

    PRINT role options
    INPUT role
    IF role == "1":
        SET user_role to "car_s"
    ELSE IF role == "2":
        SET user_role to "cus_s_I"
    ELSE IF role == "3":
        SET user_role to "cus_s_II"
    ELSE IF role == "4":
        SET user_role to "manager"
    ELSE:
        PRINT "Invalid Input."
        CALL register()

    GET current time

    OPEN "users.txt" in append mode
    WRITE username, name, password, user_role, current time to file
    CLOSE file
    PRINT "Account Registered"

FUNCTION update_profile(user, alluser):
    PRINT user details
    PRINT update options: 1. Username, 2. Name, 3. Password
```

```
INPUT choice
IF choice == "1":
    INPUT new username
    SET user[0] to new username
    OPEN "users.txt" in write mode
    FOR each user in alluser:
        IF current user:
            UPDATE username
        WRITE user to file
    CLOSE file
    PRINT "Username changed successfully"

ELSE IF choice == "2":
    INPUT new name
    SET user[1] to new name
    OPEN "users.txt" in write mode
    FOR each user in alluser:
        IF current user:
            UPDATE name
        WRITE user to file
    CLOSE file
    PRINT "Name changed successfully"

ELSE IF choice == "3":
    INPUT new password
    SET user[2] to new password
    OPEN "users.txt" in write mode
    FOR each user in alluser:
        IF current user:
            UPDATE password
```

```
            WRITE user to file
        CLOSE file
        PRINT "Password changed successfully"

    ELSE:
        PRINT "Invalid Input."
        CALL update_profile()

FUNCTION dlt_user(user, alluser):
    PRINT "Enter username to search"
    INPUT username
    OPEN "users.txt" in write mode
    FOR each user in alluser:
        IF username matches user:
            PRINT "User has been deleted"
        ELSE:
            WRITE user to file
    CLOSE file
    CALL update_profile(user, alluser)
```

## Tan Yi Han TP070378 (Customer Service Staff I)

```
#Customer Service Staff I
FUNCTION cssI_page
        PRINT("Welcome to the CSSI menu.")
                ("Please select the options below:")
                ("1. Register Customer Details 2. Update Customer Details 3. View
Customer      List 4. Delete Customer Details")
        INPUT cssI_Choice
        IF cssI_Choice = "1":
                FUNCTION reg_cus
        ELIF cssI_Choice = "2":
                FUNCTION upd_cus
        ELIF cssI_Choice = "3":
                FUNCTION view_cus
        ELIF cssI_Choice = "4":
                FUNCTION delete_cus
        ELSE:
                PRINT Invalid Input
                PRINT Please try again
                FUNCTION cssI_page
FUNCTION reg_cus
        cus_detail = list
        PRINT ("Please register the following customer details:")
        DECLARE cusID, cus_Name, cus_NRIC, cus_Passport, cus_License, cus_Address,
cus_Phone, cus_regDate
        INPUT Customer Name
        INPUT Customer NRIC No.(If foreigner enter null)
        INPUT Customer Passport No.(If local enter null
        INPUT Customer License No.
        INPUT Customer Address:
        INPUT Customer Phone No.
        INPUT Register Date
        PRINT Customer detail
```

```
        PRINT cus_detail
        DECLARE cfm_cusdetail
        INPUT Confirm Customer detail? (Y/N):
        IF cfm_cusdetail = "Y":
                cusinfo.append(cus_detail)
                        with open('cusinfo.txt','write') as cuslist:
                        cuslist.write
                cuslist.close()
     PRINT(“Customer detail is successfully added to the system.")
       PRINT ("Updated customer list:")
       FOR cus_list IN (cusinfo):
          PRINT(cus_list)
   ELSE:
       PRINT("Customer detail is not added to system.")
    returnCssIpage = cssI_page
    INPUT("Press enter to return to CSSI menu: ")
    RETURN returnCssIpage

FUNCTION upd_cus
    cusinfo = list
    cuslist = OPEN('cusinfo.txt', read)
    PRINT ("Welcome to customer detail update page.")
    PRINT ("Customer List: ")
    PRINT (cus_column)
    PRINT("Please enter the Cus ID of the customer that you wish the update.")
    INPUT Cus ID
    PRINT ("Customer detail:")
    PRINT (cus_column)
    PRINT (upd_cus)
```

```
    PRINT("Select the option you want to update:")
    PRINT ("1. Cus ID 2. Name 3. NRIC 4. Passport 5. License 6. Address 7. Phone 8.
RegDate")
    INPUT option
    cfm_cusdetail = INPUT("Confirm update customer detail into system?(Y/N): ")
    IF cfm_cusdetail() = "Y":
        with open('cusinfo.txt','w') as cuslist
            cuslist.write(',')
            cuslist.close()
        PRINT("Customer detail is successfully updated to the system.")
    ELSE:
            PRINT("Customer detail is not updated to system.")
    returnCssIpage = cssI_page(user, alluser)
    returnCssIpage = INPUT("Press enter to return to CSSI menu: ")
    RETURN returnCssIpage

FUNCTION view_cus
    PRINT("Welcome to the view menu.")
    PRINT("Select options:")
    PRINT("1. View Customer List 2. Back")
    view_choice = INPUT("Enter option: ")
   IF view_choice = "1":
        FUNCTION view_cuslist
   ELIF view_choice = "2":
        FUNCTION cssI_page

    ELSE:
        PRINT("Invalid Input")
        PRINT("Please try again. ")
        FUNCTION view_cus
```

```
FUNCTION view_cuslist
    cusinfo = list
    cuslist = open('cusinfo.txt', 'read')
    FOR line IN cuslist:
        line = line.rstrip()
        cusinfo.append(line.split(','))
    cuslist.close()
    FOR cus_list IN (cusinfo):
        PRINT (cus_list)
    returnCssIpage = cssI_page(user, alluser)
    returnCssIpage = INPUT("Press enter to return to CSSI menu: ")
    RETURN returnCssIpage

FUNCTION delete_cus
    cusinfo = list
    cuslist = open('cusinfo.txt', 'r')
    PRINT("Customer details to be deleted: ")
    PRINT(cus_column)
    FOR i IN range(len(cusinfo)):
        IF cusinfo  = "No Transaction":
            print(cusinfo)
    cfm_deletecus = INPUT("Are you sure you want to delete this customer? (Y/N): ")
    if cfm_deletecus.upper() == "Y":
        cusinfo.remove(cusinfo[i])
        with open('cusinfo.txt','w') as cuslist:
            for row in cusinfo:
                for item in row:
                    for chr in str(item):
```

```
                cuslist.write(chr)
            if item != row[-1]:
                cuslist.write(',')
        cuslist.write('\n')
    cuslist.close()

  print("\nCustomer detail is successfully deleted from the system.")
  print ("\nUpdated customer list:")
  for cus_list in (cusinfo):
      print(cus_list)
else:
  print ("Customer details failed to be deleted from system.")

returnCssIpage = cssI_page(user, alluser)
returnCssIpage = input("Press enter to return to CSSI menu: ")
return returnCssIpage
```

## Wong Kap Onn TP074292 (Car Service Staff)

```
FUNCTION crss(user, alluser):
    DECLARE crss_choice
    PRINT "Welcome to the admin menu."
    PRINT "Please select the options below:"
    PRINT "1. Add car details 2. Modify car details 3. View car 4. Delete Car 5. Back"
    INPUT crss_choice
    IF crss_choice == "1":
        GO TO FUNCTION add_car(user, alluser)
    ELSE IF crss_choice == "2":
        GO TO FUNCTION modify_car(user, alluser)
    ELSE IF crss_choice == "3":
        GO TO FUNCTION view_car(user, alluser)
    ELSE IF crss_choice == "4":
        GO TO FUNCTION delete_car(user, alluser)
    ELSE IF crss_choice == "5":
        GO TO FUNCTION main_page()
    ELSE:
        PRINT "*******Invalid Input*******"
        PRINT "***Please try again.***"
        GO TO FUNCTION crss(user, alluser)
    ENDIF


FUNCTION add_car(user,alluser)
    DECLARE carinfo, carlist, line, car_detail, carid, Car_plate, Car_Manufacturer,
Car_Model, ManufactureRYear, Seat_Capacity, Last_Service_Date, InsurancePol_No,
InsuranceExP_Date, RoadTaxExp_Date, Rent_StartDate, Rent_ReturnDate,
Renting_Rate_Day, Availability, cfm_detail, row, item, chr, returnpage
    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
```

```
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
ENDFOR
CLOSE carlist
car_detail = []
PRINT "Please enter the following car details:"
carid = LENGTH OF carinfo + 1
carid = STR(carid)
INPUT Car_plate
ADD Car_plate TO car_detail
INPUT Car_Manufacturer
ADD Car_Manufacturer TO car_detail
INPUT Car_Model
ADD Car_Model TO car_detail
INPUT Manufacture_Year
ADD Manufacturer_Year TO car_detail
INPUT Seat_Capacity
ADD Seat_Capacity TO car_detail
INPUT Last_Service_Date
ADD Last_Service_Date TO car_detail
INPUT InsurancePol_No
ADD InsurancePol_No TO car_detail
INPUT InsuranceExp_Date
ADD InsuranceExp_Date TO car_detail
INPUT RoadTaxExp_Date
ADD RoadTaxExp_Date TO car_detail
INPUT Rent_StartDate
ADD Rent_StartDate TO car_detail
INPUT Rent_ReturnDate
ADD Rent_ReturnDate TO car_detail
```

```
INPUT Renting_Rate_Day
ADD Renting_Rate_Day TO car_detail
INPUT Availability
ADD Availability TO car_detail

PRINT "Car detail:"
PRINT car_detail

INPUT cfm_detail
IF cfm_detail == "Y":
        ADD car_detail TO carinfo
        OPEN WRITE TO ‘carlist.txt’ AS carlist:
                FOR row IN carinfo:
                        FOR item IN row:
                        FOR chr IN STR(item):
                                WRITE chr TO carlist
                        ENDFOR
                        IF item != row[-1]:
                                WRITE ‘,’ TO carlist
                        ENDIF
                ENDFOR
                        WRITE '\n' TO carlist
            ENDFOR
                CLOSE carlist
        PRINT “Car detail is successfully added to the system."
        PRINT “Updated car list:"
        FOR car_list IN carinfo:
                PRINT car_list
        ENDFOR
ELSE:
```

```
            PRINT "Car detail is not added to system."
        ENDIF
        INPUT returnpage
        GO TO FUNCTION crss

FUNCTION modify_car(user,alluser):
        DECLARE carinfo, carlist, line, car_column, i, car_location, mod_car, mod_cardetail,
cur_detail, upd_detail, cfm_detail, row, item, chr, returnpage
        carinfo = []
        carlist = OPEN READ 'carinfo.txt'
        FOR line IN carlist:
            line = REMOVE whitespaces FROM line
            ADD ‘,’  line split TO carinfo
        ENDFOR
        CLOSE carlist
        PRINT "Welcome to the modify page,”
        PRINT "Car List: “
        PRINT car_column
        FOR i IN carinfo:
            print i
        ENDFOR
        PRINT “Please enter the car id of the car that you wish the modify."
        INPUT car_location USING CARID FROM LIST
        car_location = INT(car_location) - 1
        mod_car = USING VALUE OF car_location AS index FOR carinfo
        PRINT "Car detail:"
        PRINT car_column
        PRINT mod_car

        PRINT "Select the option you want to change:”
```

```
PRINT "1. Car ID 2. Car Plate 3. Car Manufacturer 4. Car Model 5. Manufacture Year
6. Seat Capacity 7. Last Service Date 8. Insurance Policy No 9. Insurance Expiry Date 10.
Road Tax Expiry Date 11. Rent Start Date 12. Rent Return Date 13. Rent Rate Per Day 14.
Availability"

INPUT mod_cardetail

WHILE mod_cardetail != "1" AND mod_cardetail != "2" AND mod_cardetail != "3"
AND mod_cardetail != "4" AND mod_cardetail != "5" AND mod_cardetail != "6" AND
mod_cardetail != "7" AND mod_cardetail != "8" AND mod_cardetail != "9" AND
mod_cardetail != "10" AND mod_cardetail != "11" AND mod_cardetail != "12" AND
mod_cardetail != "13" AND mod_cardetail != "14":

        print ""

        INPUT mod_cardetail

mod_cardetail = INT(mod_cardetail) - 1

cur_detail = USING VALUE OF mod_cardetail AS index FOR mod_car

PRINT "Car detail:"

PRINT "Current Detail:", cur_detail

INPUT upd_detail

mod_car[mod_cardetail] = upd_detail

PRINT "Updated car detail:"

print mod_car

PRINT "Updated car list:"

FOR i IN carinfo:

        print i

ENDFOR

INPUT cfm_detail

IF cfm_detail == "Y":

        OPEN WRITE TO 'carlist.txt' AS carlist

                FOR row IN carinfo:

                        FOR item IN row:

                        FOR chr IN str(item):

                                carlist.write(chr)

                        ENDFOR

                        IF item != row[-1]:
```

```
                                        WRITE ‘,’ TO carlist
                                    ENDIF
                                ENDFOR
                                    WRITE '\n' TO carlist
                            ENDFOR
                                CLOSE carlist

            PRINT "Car detail is successfully updated to the system."
        ELSE:
            PRINT "Car detail is not updated to system."
        ENDIF
        INPUT returnpage
        GO TO FUNCTION crss

FUNCTION view_car(user,alluser):
        DECLARE view_choice
        PRINT "Welcome to the view menu."
        PRINT "Select options:"
        PRINT "1. All record 2. Available for Rent 3. Rented 4. Search 5. Back"

        INPUT view_choice

         IF view_choice == "1":
            GO TO FUNCTION view_allcar(user,alluser)
        ELSE IF view_choice == "2":
            GO TO FUNCTION view_availablecar(user,alluser)
        ELSE IF view_choice == "3":
            GO TO FUNCTION view_rentedcar(user,alluser)
        ELSE IF view_choice == "4":
            GO TO FUNCTION view_specificcar(user,alluser)
```

```
    ELSE IF view_choice == "5":
        GO TO FUNCTION crss(user,alluser)
    ELSE
        PRINT "*******Invalid Input*******"
        PRINT "***Please try again.***"
        GO TO FUCNTION view_car(user,alluser)
    ENDIF

FUNCTION view_allcar(user, alluser):

    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
    ENDFOR
    CLOSE carlist
    PRINT "Car List: "
    PRINT car_column
    FOR car_list IN carinfo:
        PRINT car_list
    ENDFOR
    INPUT returnpage
    GO TO FUNCTION view_car(user, alluser)

FUNCTION view_availablecar(user, alluser):
    DECLARE carinfo, carlist, line, car_column, i, returnpage
    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
```

```
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
    ENDFOR
    CLOSE carlist
    PRINT "Available Car to rent: "
    PRINT car_column
    FOR i IN RANGE OF LEN OF carinfo:
        IF carinfo[i][13] == "AVAILABLE":
                PRINT carinfo[i]
        ENDIF
    ENDFOR
    INPUT returnpage
    GO TO FUNCTION view_car(user, alluser)

FUNCTION view_rentedcar(user, alluser):
    DECLARE carinfo, carlist, line, car_column, i, returnpage
    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
    ENDFOR
    CLOSE carlist
    PRINT "Rented Car: "
    PRINT car_column
    FOR i IN RANGE OF LEN OF carinfo:
        IF carinfo[i][13] == "RENTED":
                PRINT carinfo[i]
        ENDIF
    ENDOFR
```

```
    INPUT returnpage
    GO TO FUNCTION view_car(user, alluser)

FUNCTION view_specificcar(user, alluser):
    DECLARE carinfo, carlist, line, search_info, car_column, i, returnpage
    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
    ENDFOR
    CLOSE carlist
    INPUT search_info Carplate
    PRINT "Car Detail:"
    PRINT car_column
    FOR i IN RANGE OF LEN OF carinfo:
        IF carinfo[i][1] == search_info:
                PRINT carinfo[i]
        ENDIF
    ENDFOR
    INPUT returnpage
    GO TO FUNCTION view_car(user, alluser)

FUNCTION delete_car(user, alluser):
    DECLARE, carinfo, carlist, line, car_column, i, cfm_delete, row, item, chr, car_list,
returnpage
    carinfo = []
    carlist = OPEN READ 'carinfo.txt'
    FOR line IN carlist:
        line = REMOVE whitespaces FROM line
        ADD ‘,’  line split TO carinfo
```

```
ENDFOR
CLOSE carlist
PRINT "Car to be disposed: "
PRINT car_column
FOR i IN RANGE OF LEN carinfo:
        IF carinfo[i][13] == "DISPOSED":
                PRINT carinfo[i]
        ENDIF
ENDFOR
INPUT cfm_delete
IF cfm_delete == "Y":
        REMOVE carinfo[i] FROM carinfo
        OPEN WRITE TO 'carlist.txt' AS carlist:
                FOR row IN carinfo:
                        FOR item IN row:
                        FOR chr IN str(item):
                                carlist.write(chr)
                        ENDFOR
                        IF item != row[-1]:
                                WRITE ',' TO carlist
                        ENDIF
                ENDFOR
                        WRITE '\n' TO carlist
            ENDFOR
                CLOSE carlist()
        PRINT "Car detail is successfully deleted from the system."
        PRINT "Updated car list:
        FOR car_list IN carinfo:
                PRINT car_list
        ENDFOR
```

ELSE:

PRINT "Car failed to delete from system."

ENDIF

INPUT returnpage

GO TO FUNCTION crss(user, alluser)

## Timothy Tan Chern Tian TP074658 (Car Service Staff II)

FUNCTION Load_Car_data:

Store the data in the empty list using []

Open the file and reach each line of the file.

For each line in the file, it will remove spaces and split them by commas to extract the data.

Close the file.

FUNCTION View_Rental Transaction_By_Date:

Retrieve the data from carinfo.txt.

Open the carlist.txt file.

Read each line, then remove spaces and add commas for each category.

Close the Carlist.txt file.

Print (“Enter Rent Start Date (DD/MM/YYYY): “),

Searches for the data with the same rent date.

Give some description and print the column of the car data that corresponds.

Only the column with the data will pop up or else it would be blank if there are nothing.

Return to the CSSI Page

FUNCTION Check_Car_Availaliblity:

Ask the Staff to input (Number of Passengers)

Check if there are cars with the number of passengers requested, and the car availability should only be “AVAILABLE”. Check from function to retrieve data from the text file.

If there are:

Print (“Available cars for {number of passengers} passengers: “)

Print (“All the car information with available seat”)

Else:

Print (There are no cars available {number of passengers} Passengers)

Back to home page

FUNCTION Record_Rental_Details:

Print("Please Enter Car Plate: ")

Read car_plate from user

Check if the car plate exists in the carinfo.txt file

If not, prompt staff to register the car

Check if the car availability is RESERVED, DISPOSED, or UNDERSERVICE

If unavailable, print("Car is not available for booking")

If available:

Print("Please input Customer ID: ")

Read customer_id from user

Check if customer_id is registered

If not, print("Please register the customer before renting the car")

If registered:

Prompt staff to input rent start and end dates

Calculate rental days

Get rent_per_day from carinfo.txt file

Calculate and print rental information

Prompt staff to input payment amount and give change

Print receipt

FUNCTION Accept_Payment:

Print("Please input payment amount: ")

Read payment_amount from user

If payment_amount is a valid number:

If payment_amount >= total_amount:

Calculate change

Print("Change: {change}")

Else:

Print("Insufficient funds")

Else:

Print("Invalid payment method")

FUNCTION Generate_Receipt:

Print receipt details: Car plate, customer ID, rental date, return date, rental days, and total rental amount

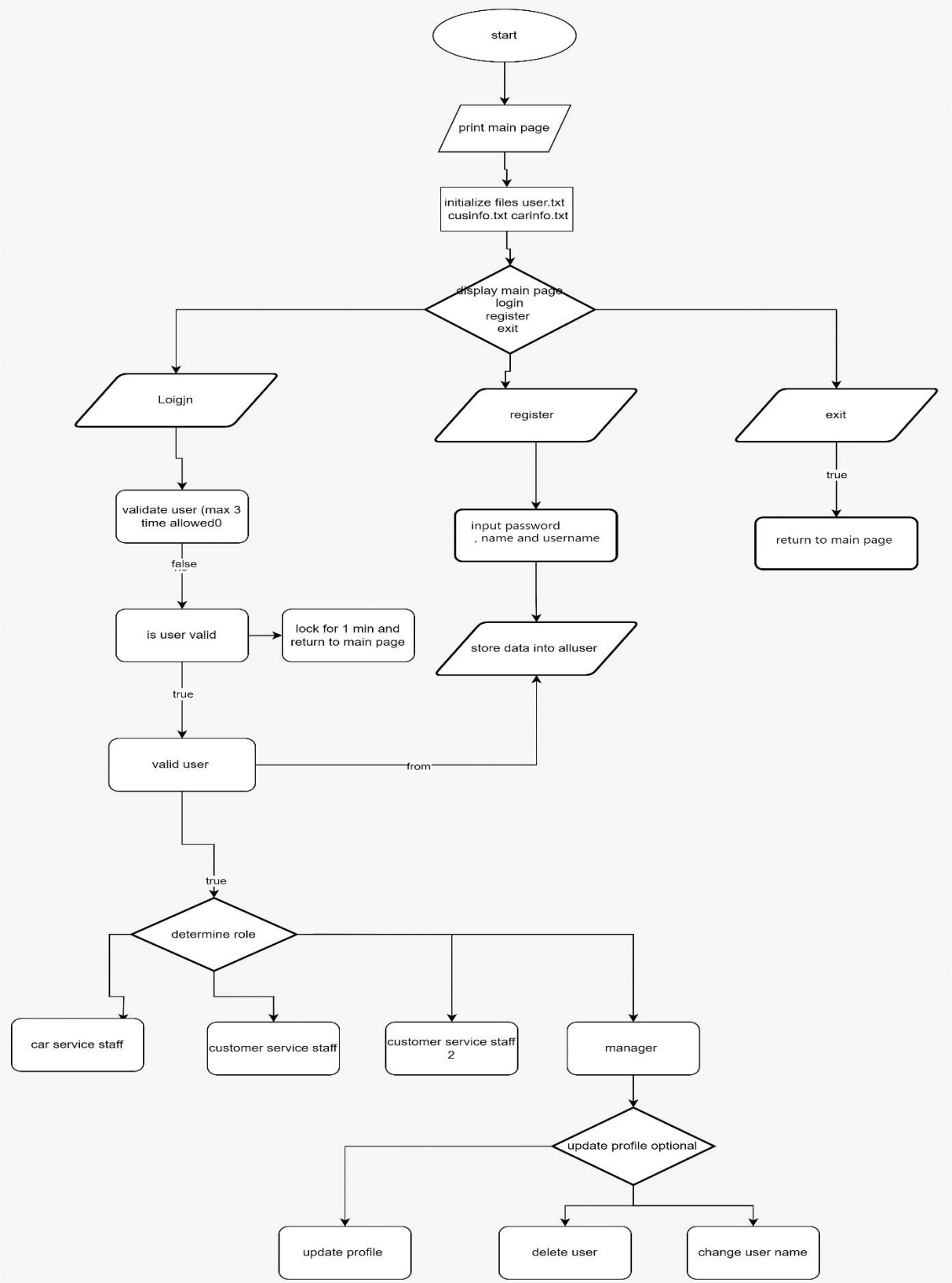
Calculate change

Print("Thank you for choosing our rental service")

# 2.2 Flowchart

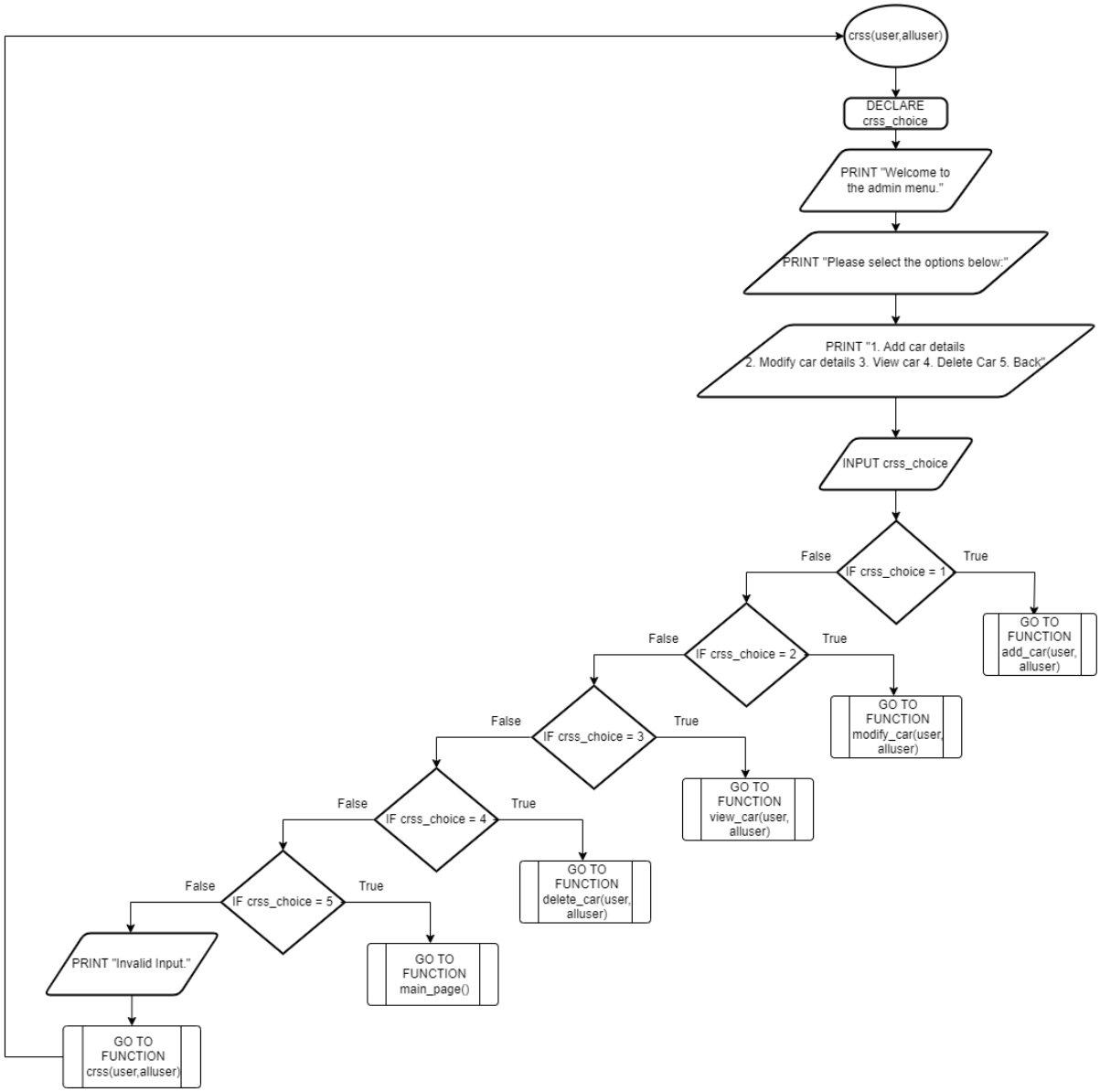
## Tan Yen Hann TP073629 (Manager)

Function Main Page and manger page:

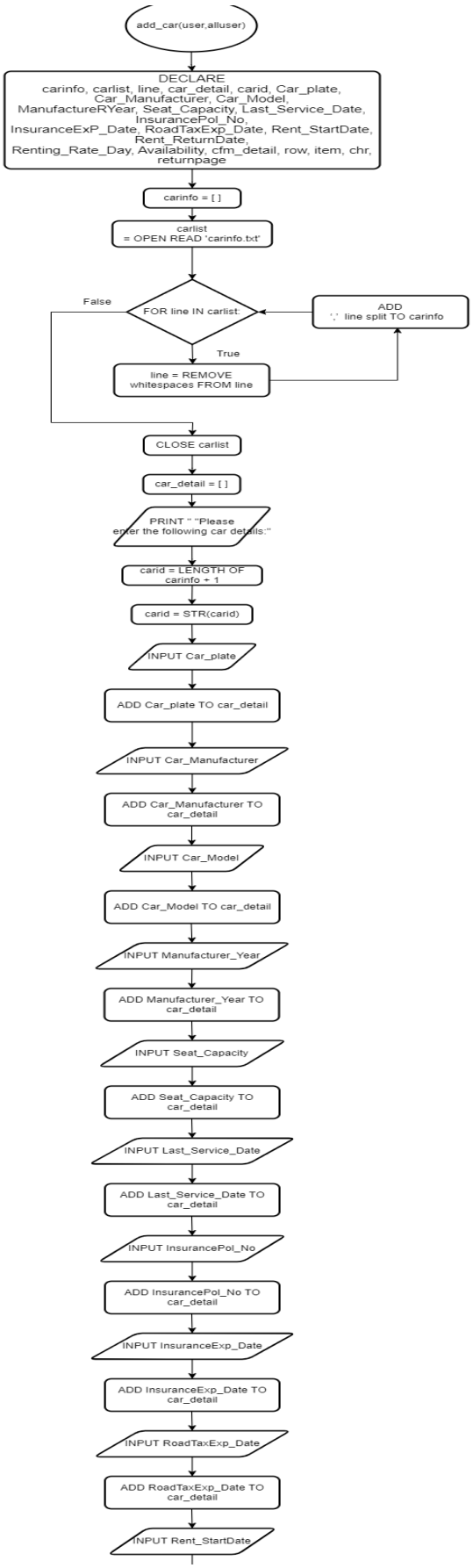


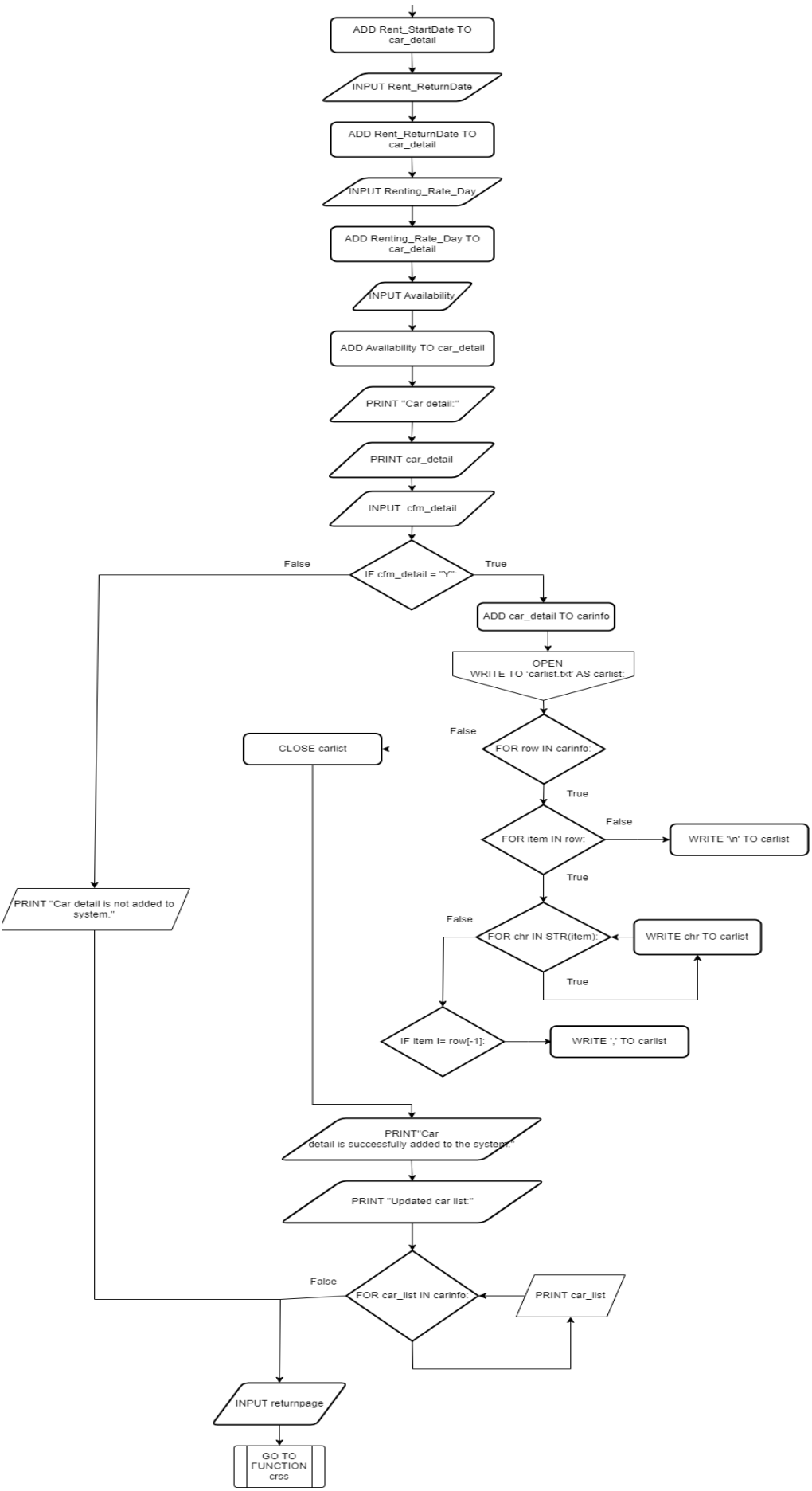
## Wong Kap Onn TP074292 (Car Service Staff)

Function car service stuff page:

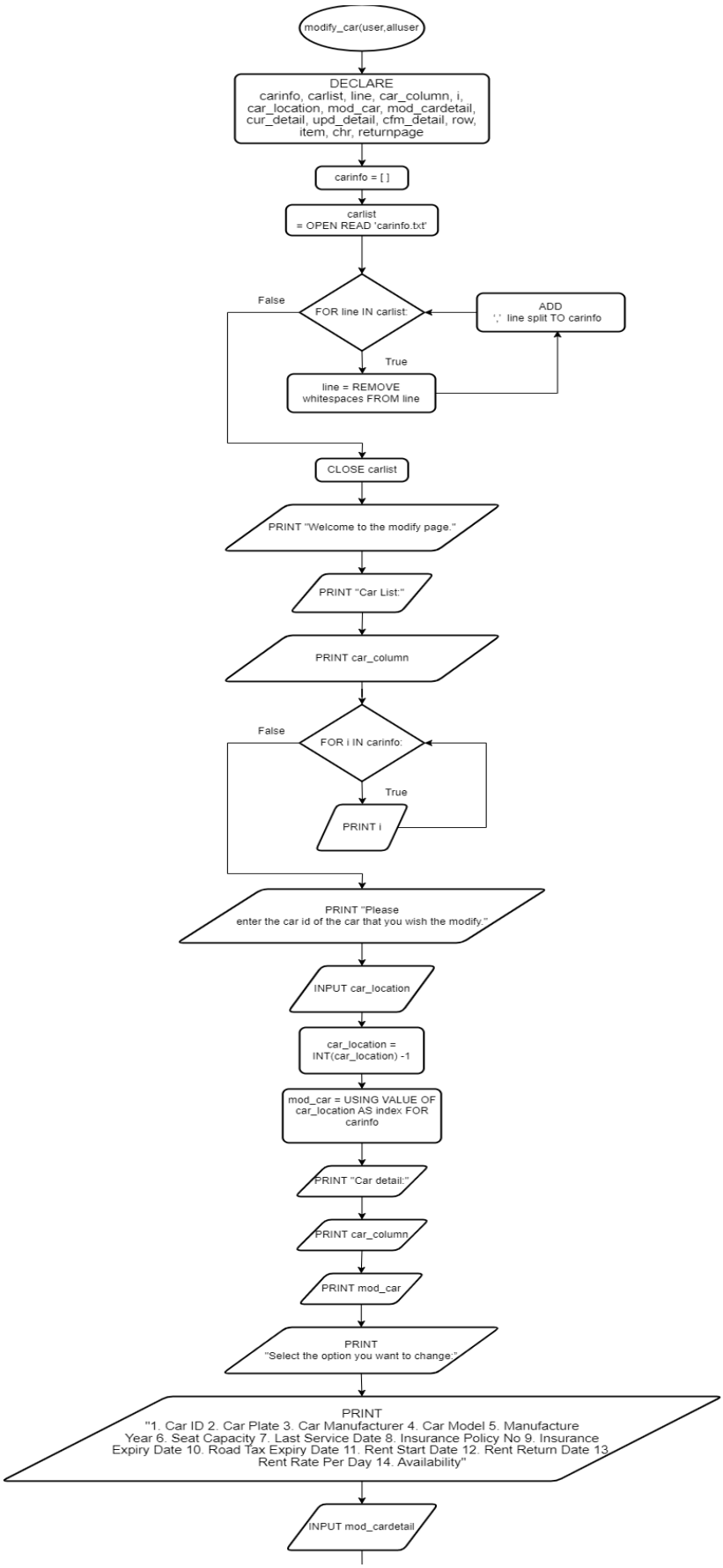


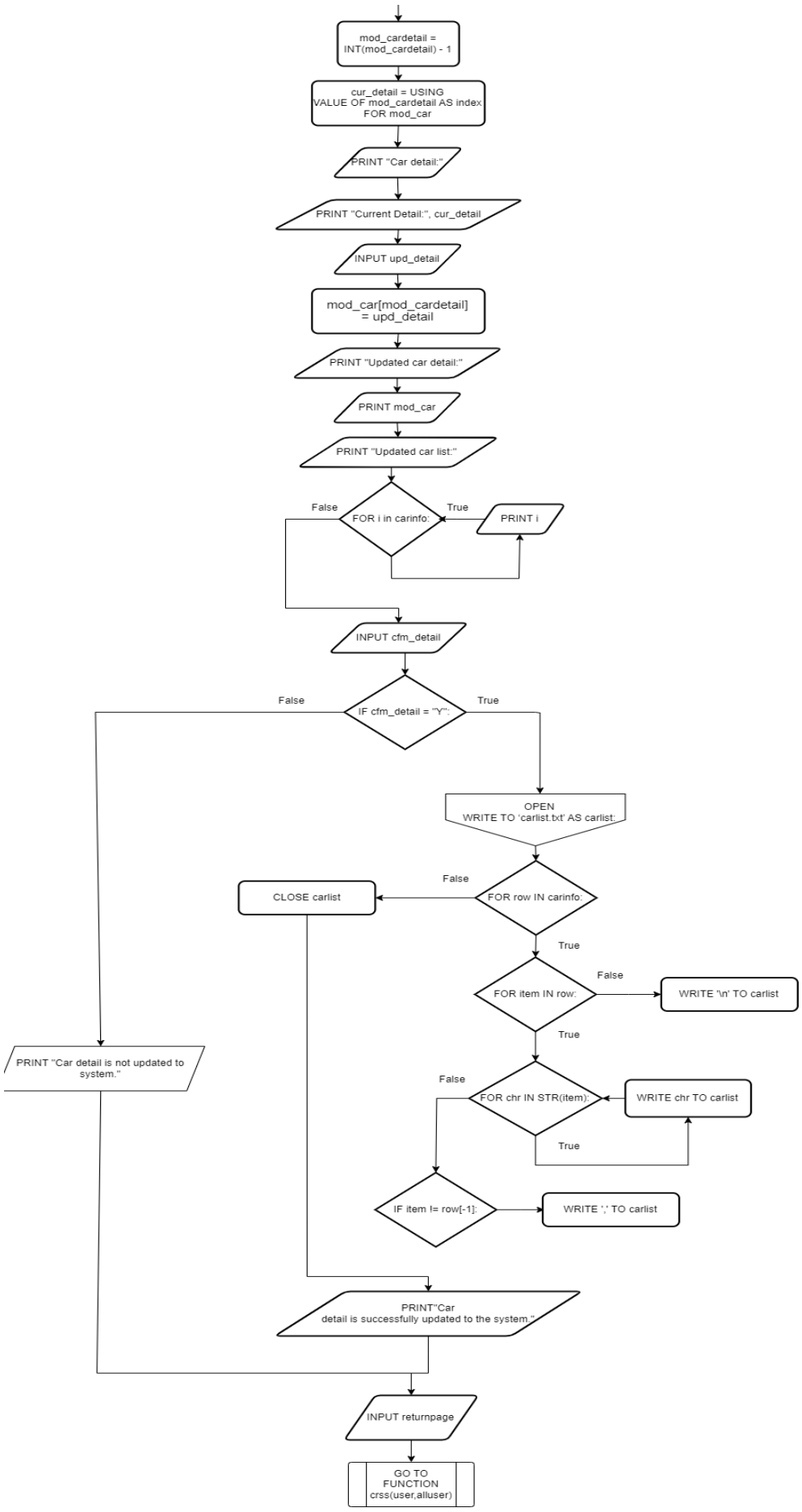
Function Add Car:



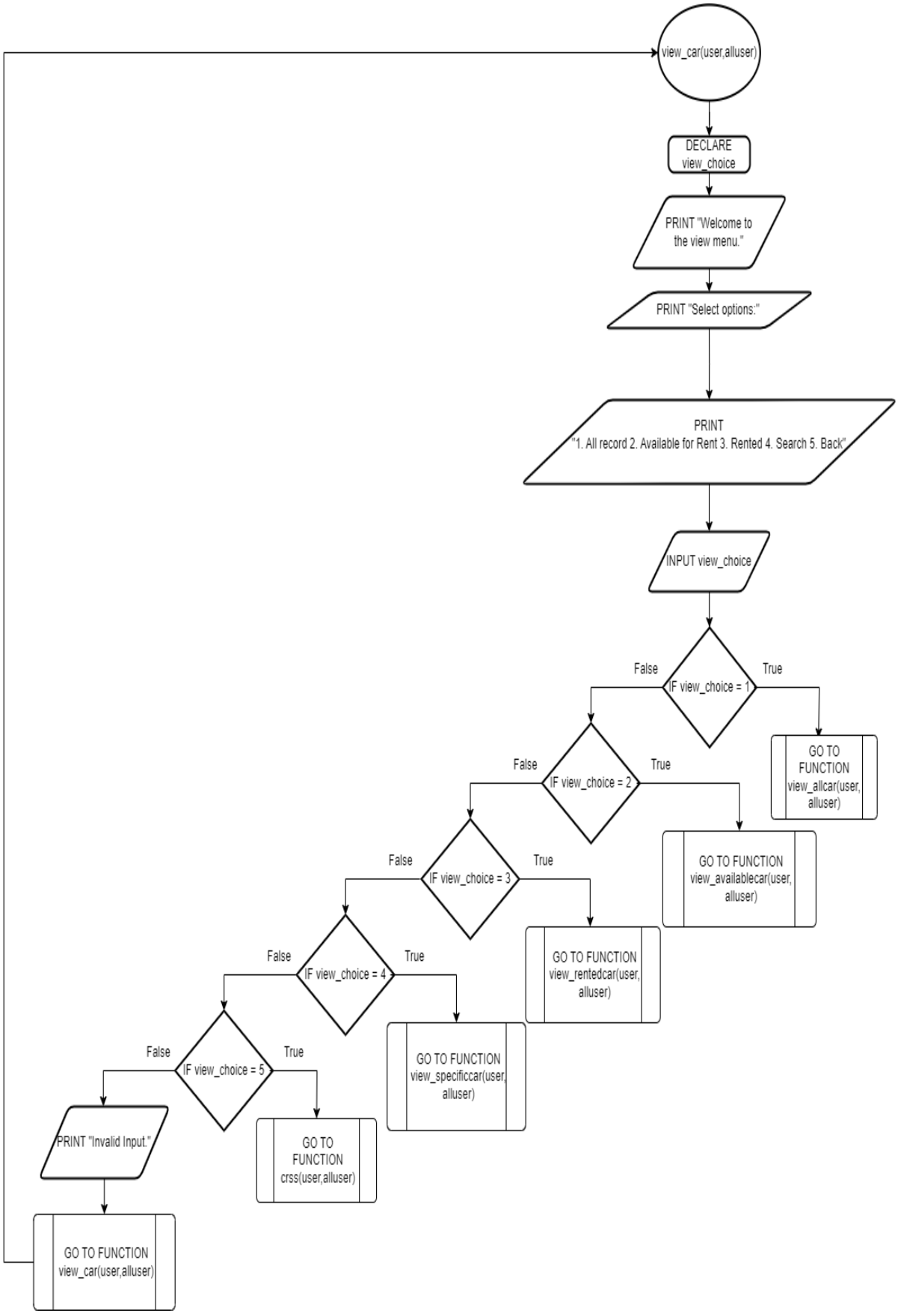

ADD Rent_StartDate TO car_detail
INPUT Rent_ReturnDate
ADD Rent_ReturnDate TO car_detail
INPUT Renting_Rate_Day
ADD Renting_Rate_Day TO car_detail
INPUT Availability
ADD Availability TO car_detail
PRINT "Car detail:"
PRINT car_detail
INPUT cfm_detail
IF cfm_detail = "Y":
False
True
ADD car_detail TO carinfo
OPEN
WRITE TO 'carlist.txt' AS carlist:
FOR row IN carinfo:
False
CLOSE carlist
True
FOR item IN row:
False
WRITE '\n' TO carlist
True
FOR chr IN STR(item):
False
WRITE chr TO carlist
True
IF item != row[-1]:
WRITE ',' TO carlist
PRINT "Car detail is not added to system."
PRINT"Car detail is successfully added to the system."
PRINT "Updated car list:"
FOR car_list IN carinfo:
False
PRINT car_list
INPUT returnpage
GO TO FUNCTION crss

Function Modify Car:

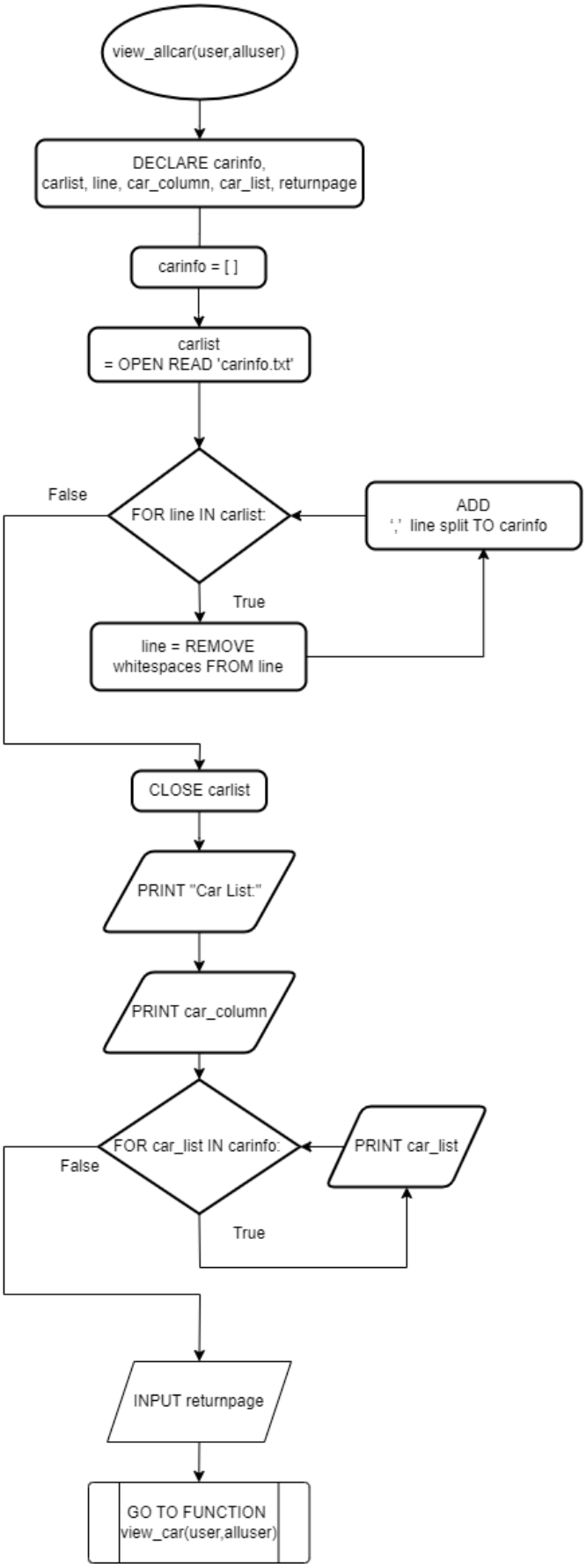



mod_cardetail = INT(mod_cardetail) - 1
cur_detail = USING VALUE OF mod_cardetail AS index FOR mod_car
PRINT "Car detail:"
PRINT "Current Detail:", cur_detail
INPUT upd_detail
mod_car[mod_cardetail] = upd_detail
PRINT "Updated car detail:"
PRINT mod_car
PRINT "Updated car list:"
FOR i in carinfo:
False
True
PRINT i
INPUT cfm_detail
IF cfm_detail = "Y":
False
True
OPEN WRITE TO 'carlist.txt' AS carlist:
FOR row IN carinfo:
False
CLOSE carlist
True
FOR item IN row:
False
WRITE '\n' TO carlist
True
FOR chr IN STR(item):
False
WRITE chr TO carlist
True
IF item != row[-1]:
WRITE ',' TO carlist
PRINT "Car detail is not updated to system."
PRINT"Car detail is successfully updated to the system."
INPUT returnpage
GO TO FUNCTION crss(user,alluser)

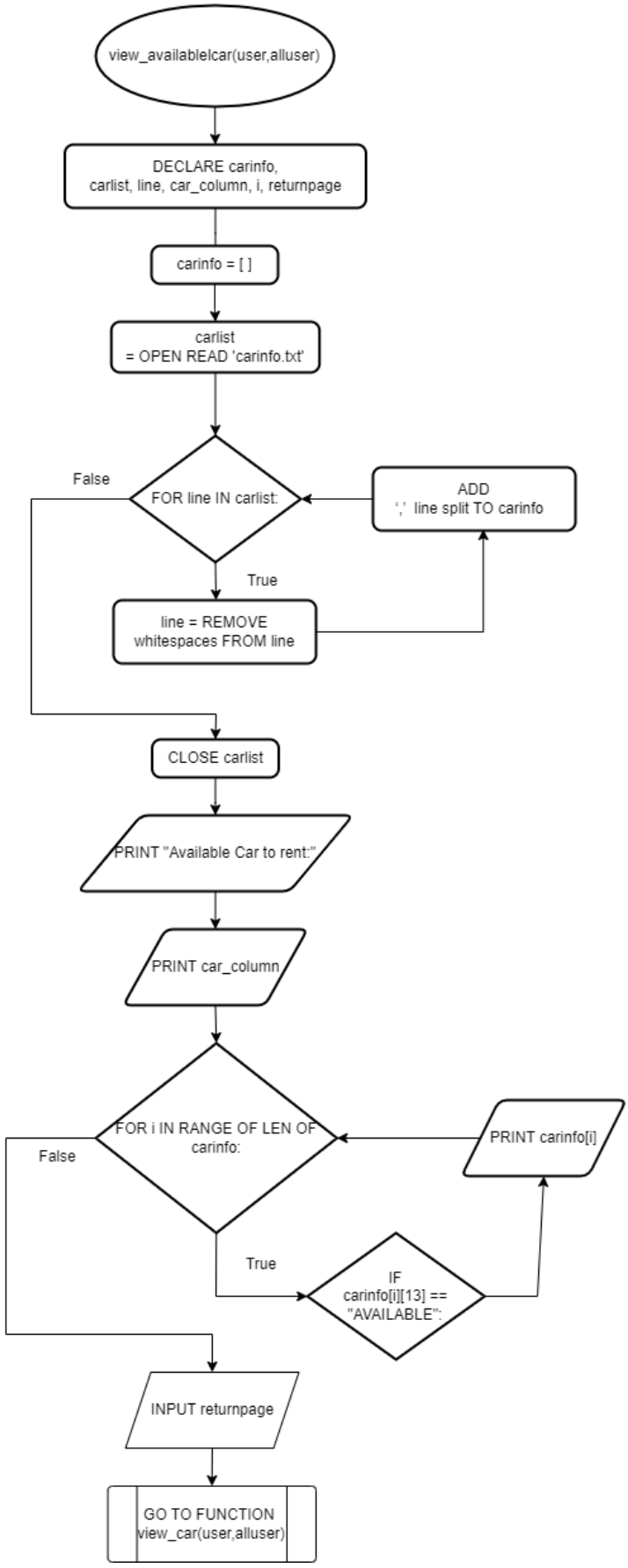
Function View Car Menu:



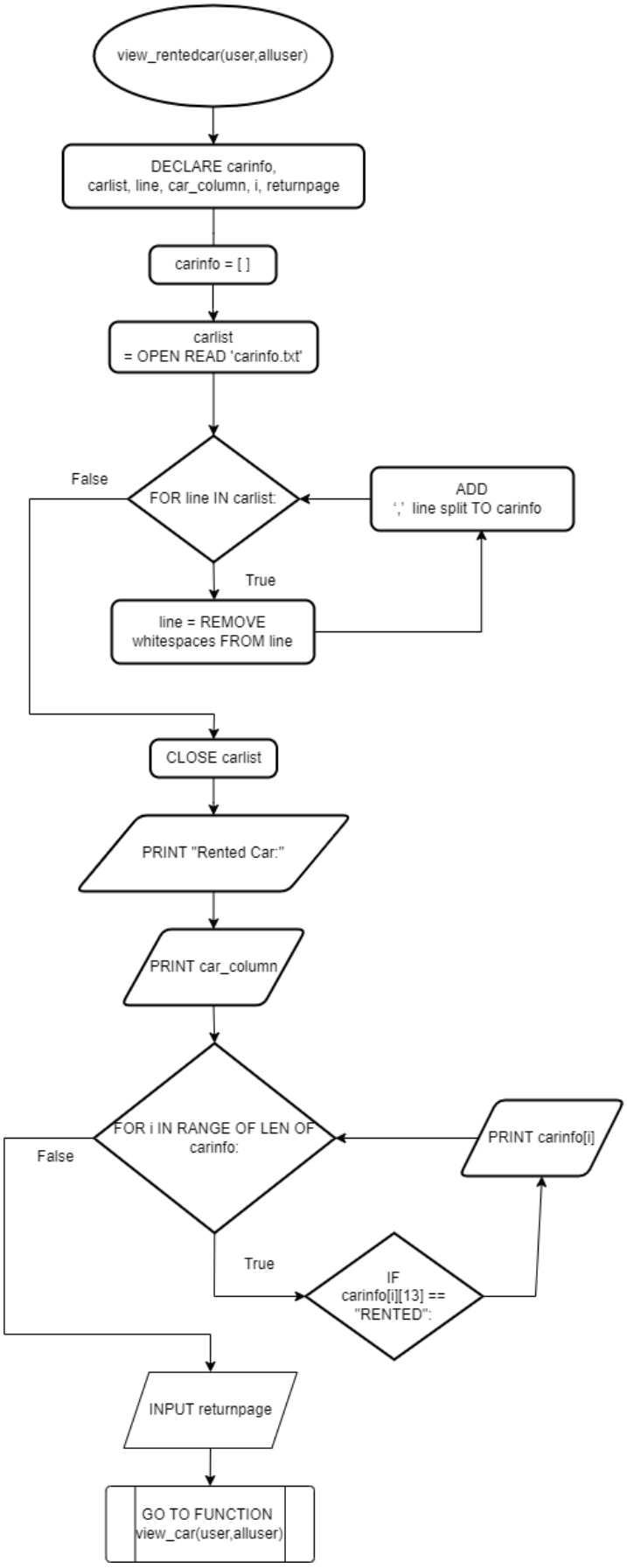
Function View All Car:



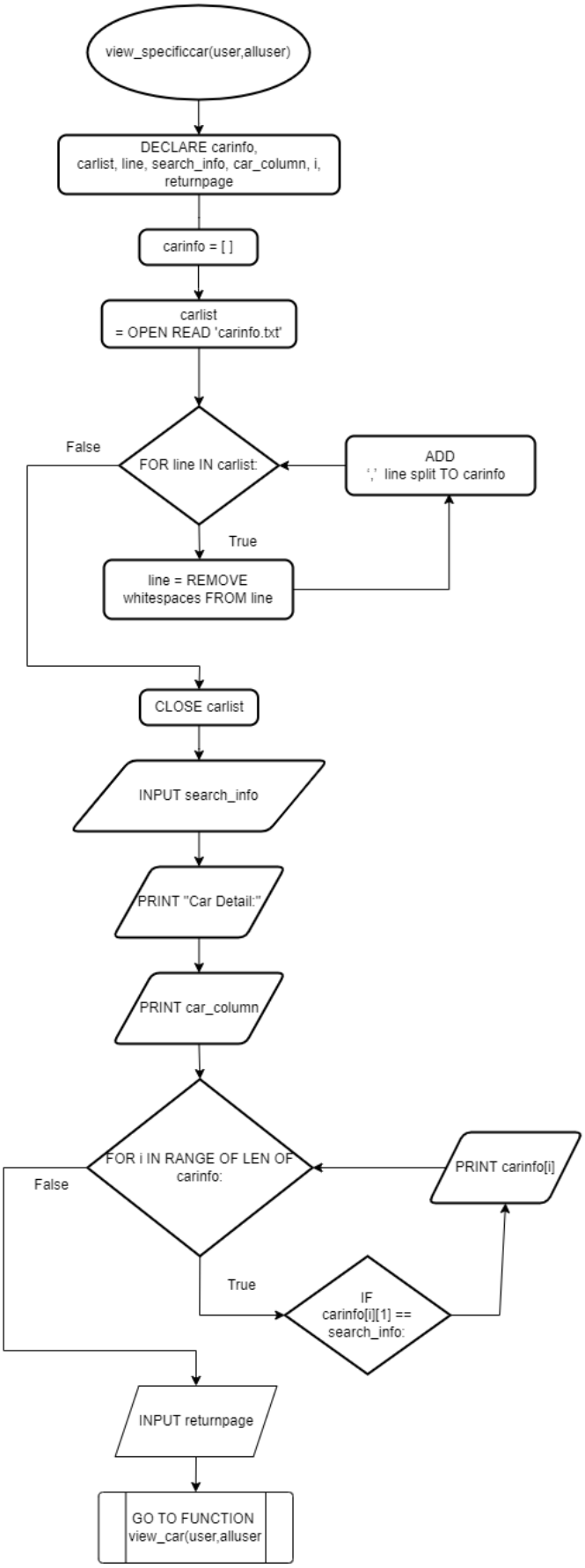
Function View Available Car:



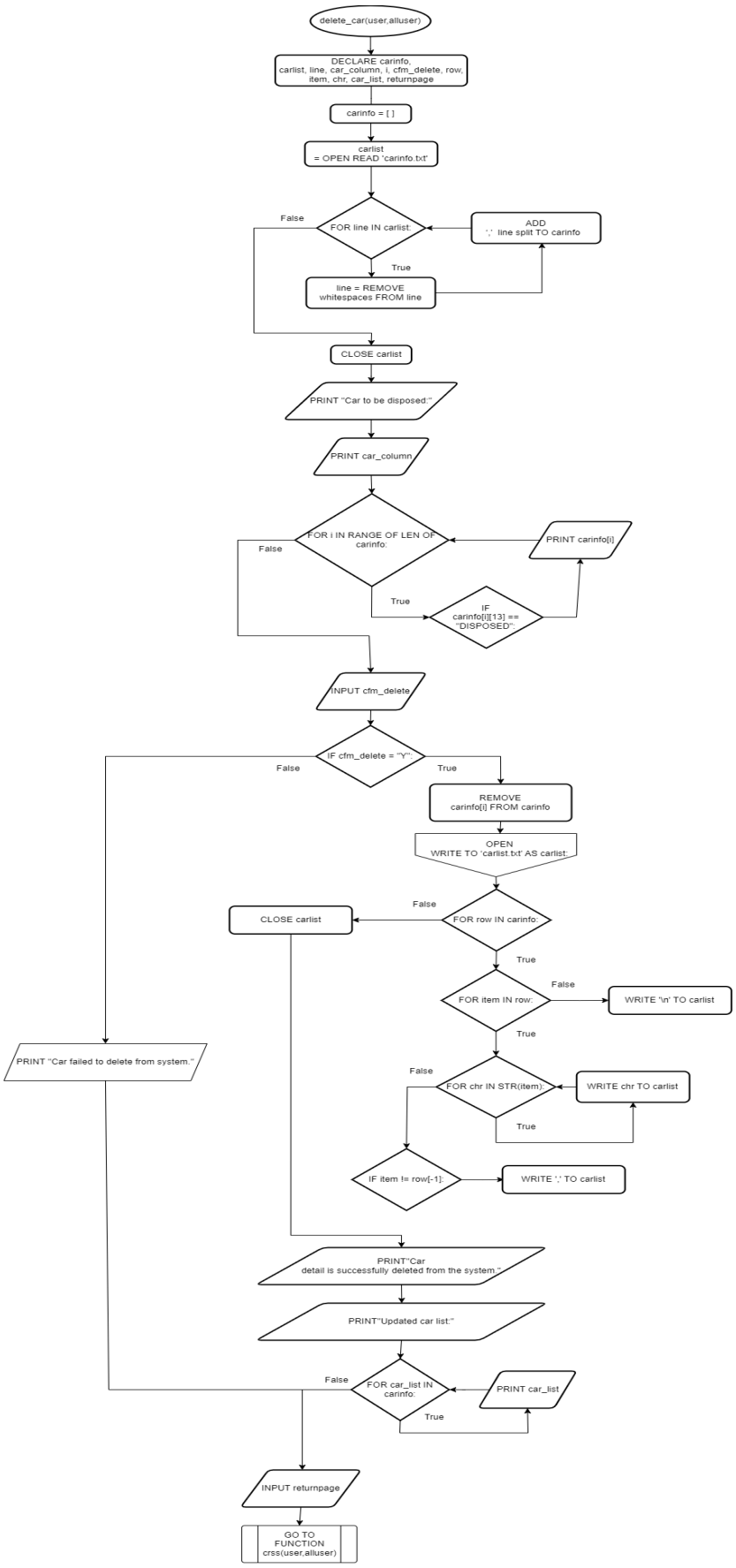
Function View Rented Car:



Function View Specific Car:

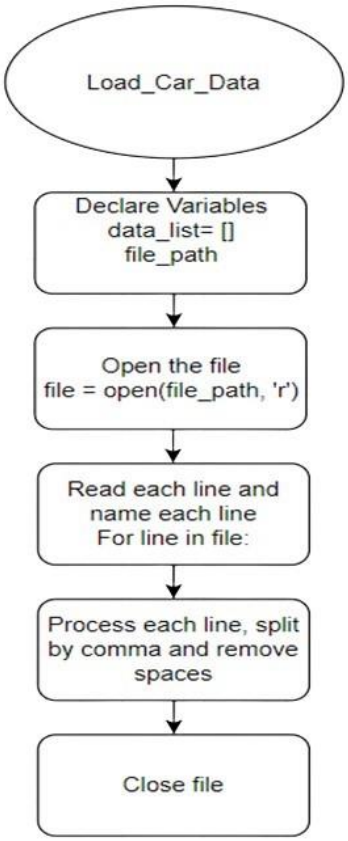


Function Delete Car:

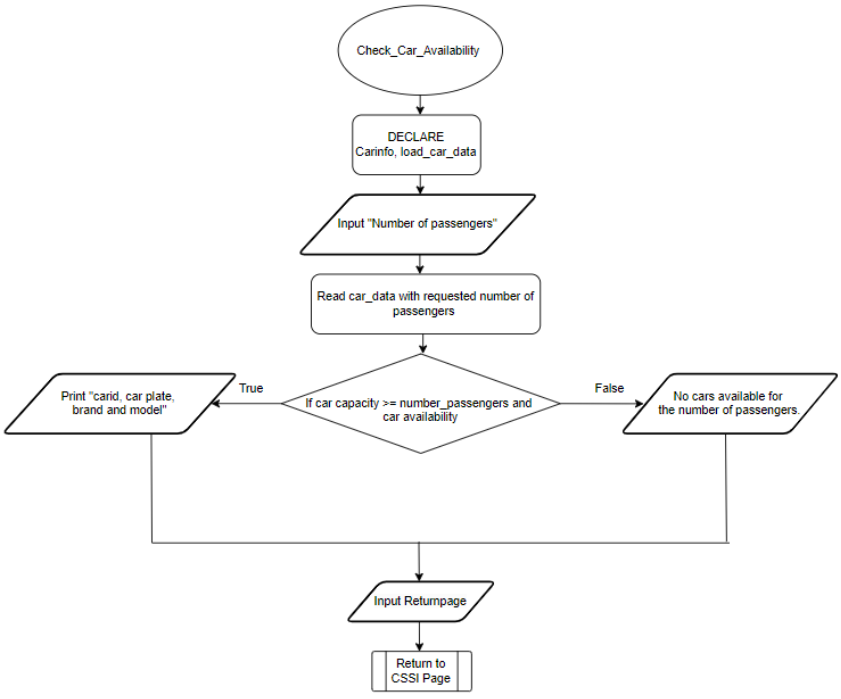


## Timothy Tan Chern Tian TP074658 (Customer Service Staff II)

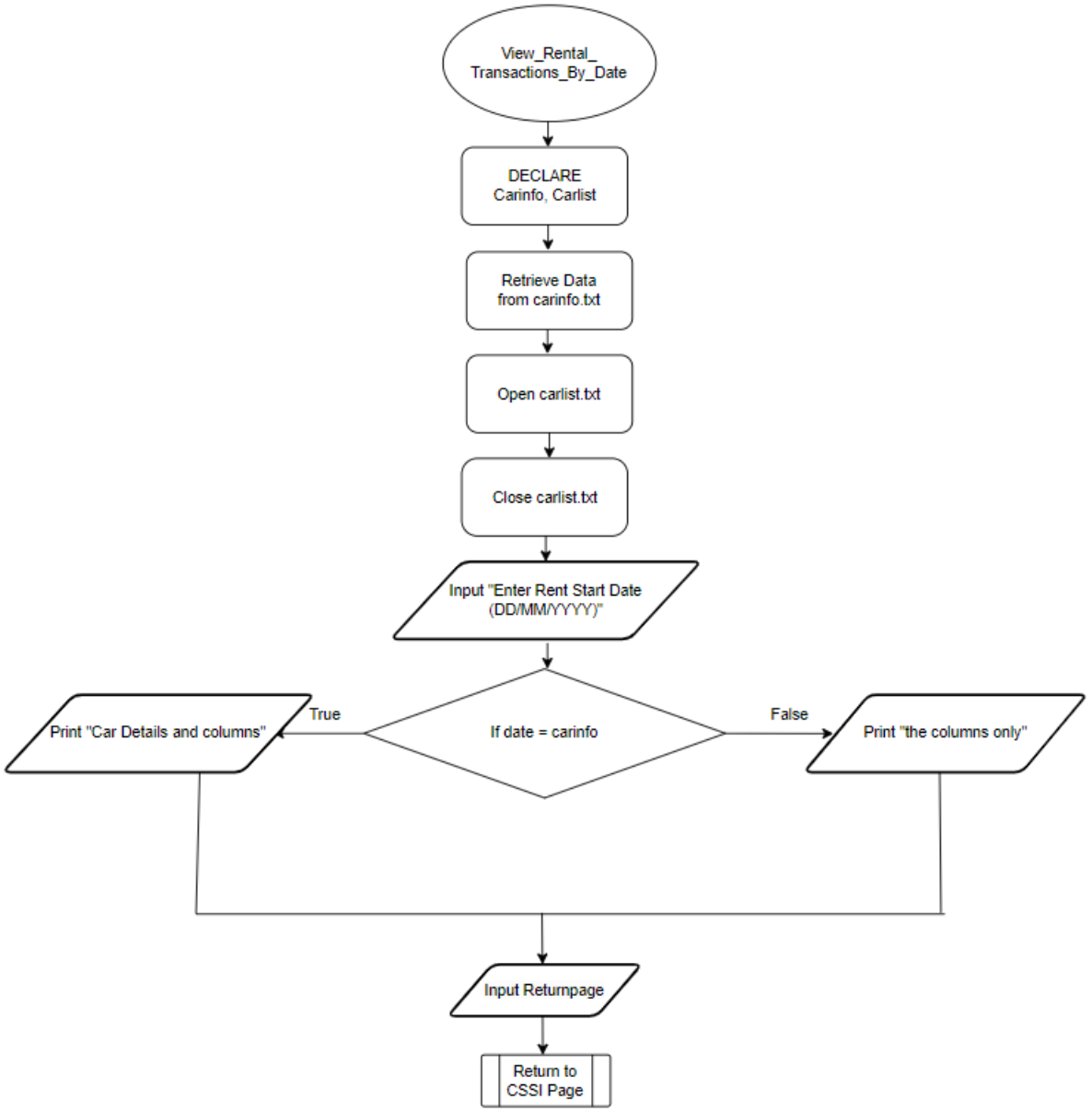
Function to Load_Car_Data:



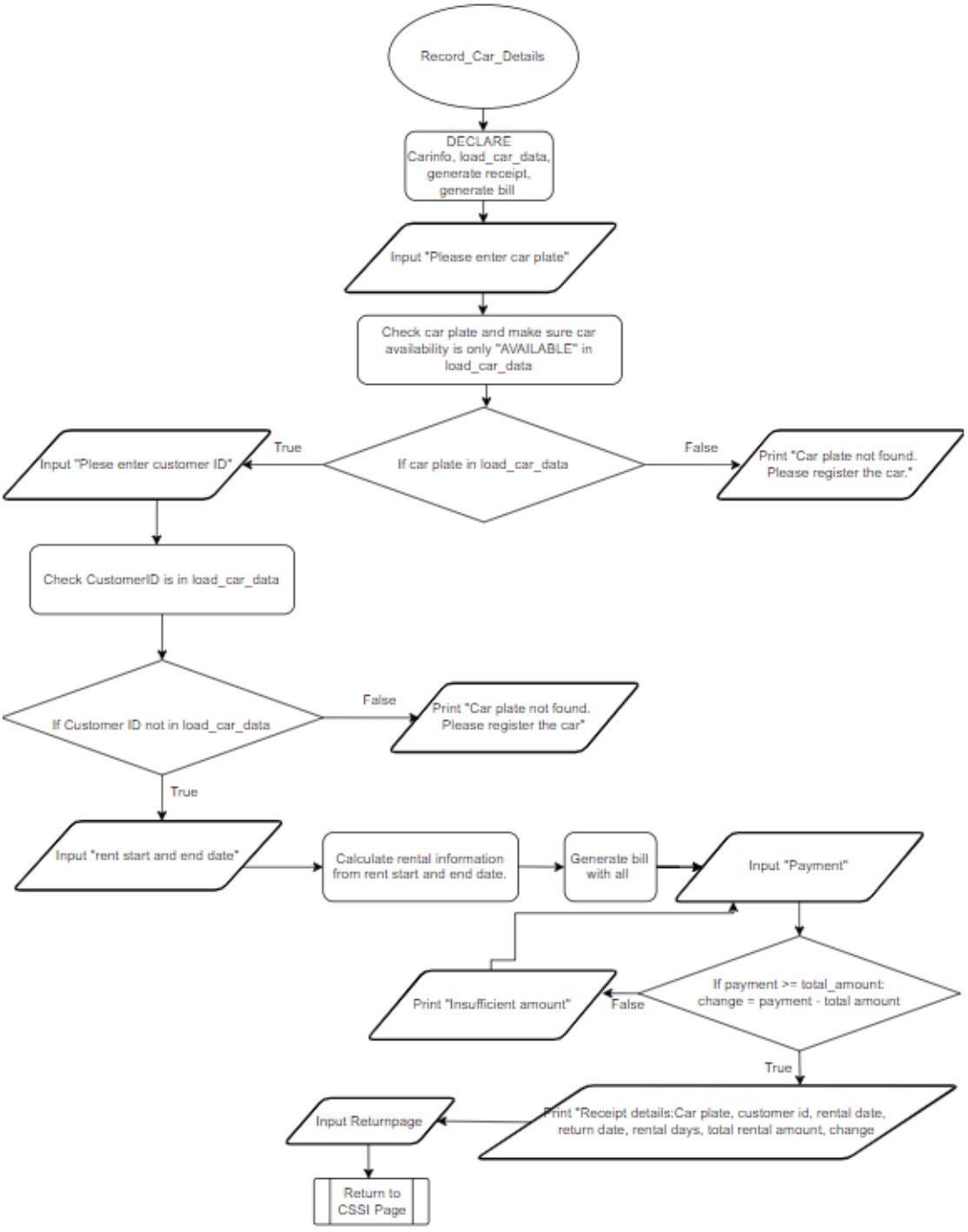

Function to Check_Car_Availability



Function to View_Rental_Transaction_By_Date

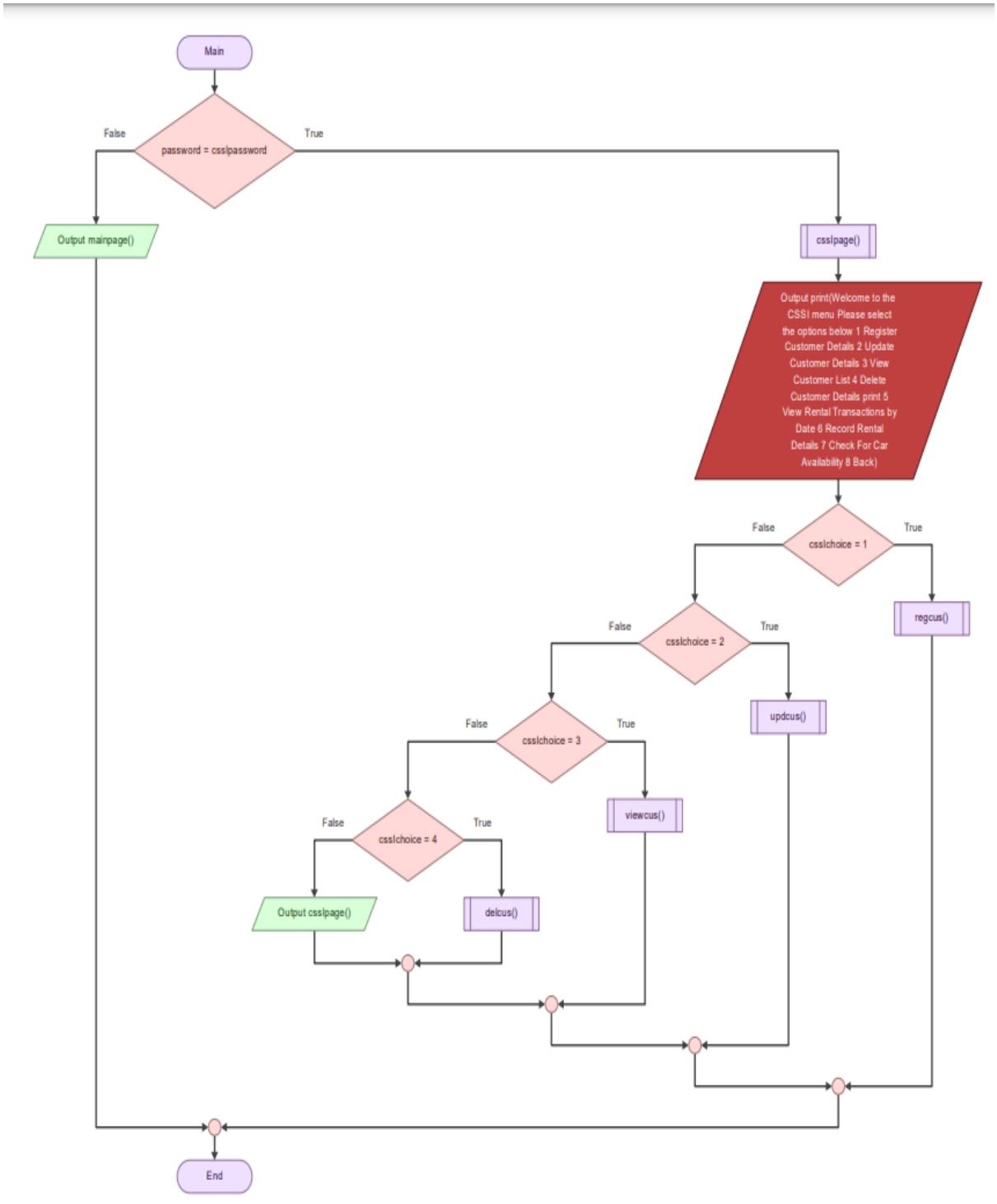


Function to Record_Rental_Details

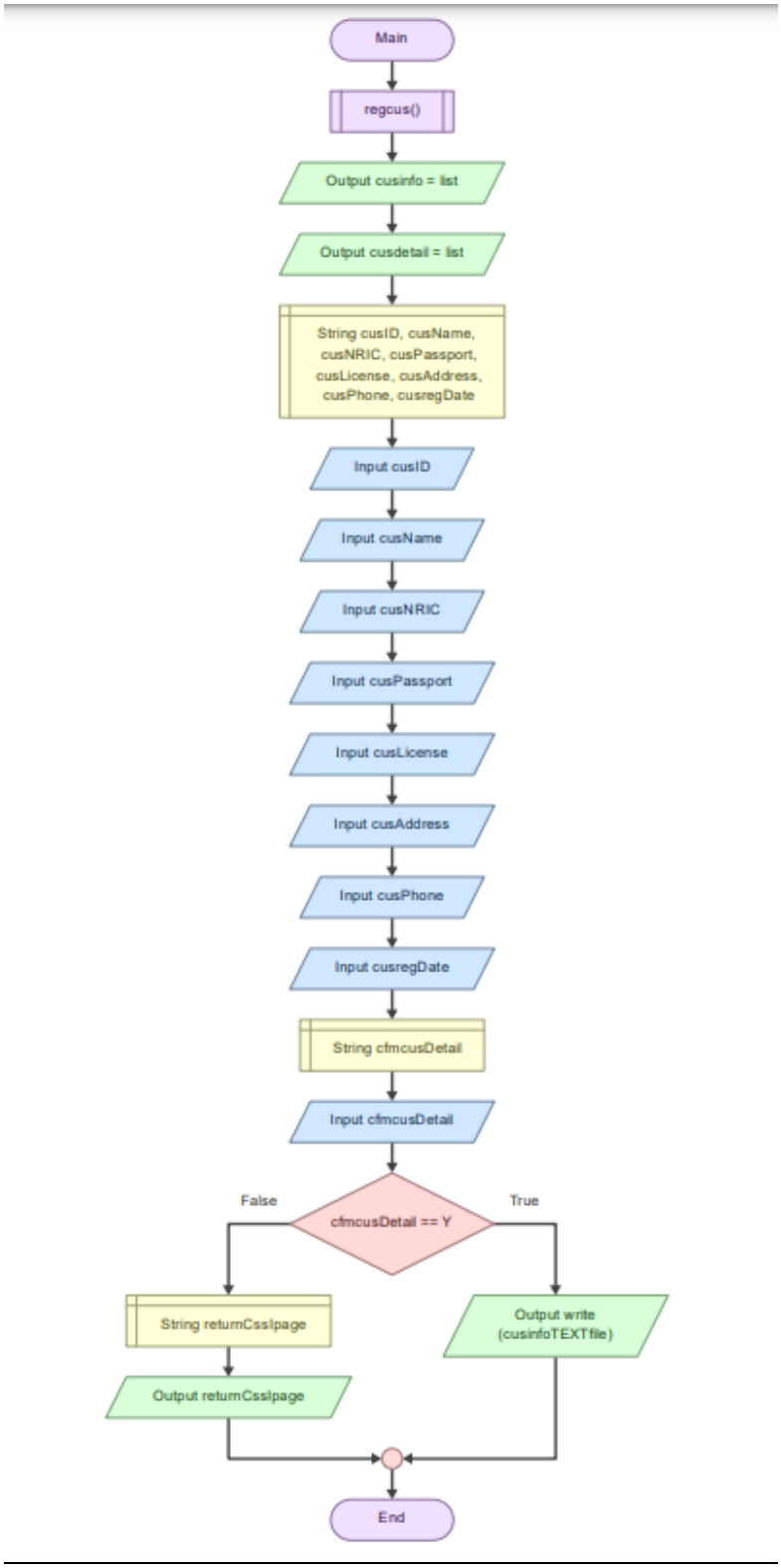


## TAN YI HAN TP070378 (Customer Service Staff I)

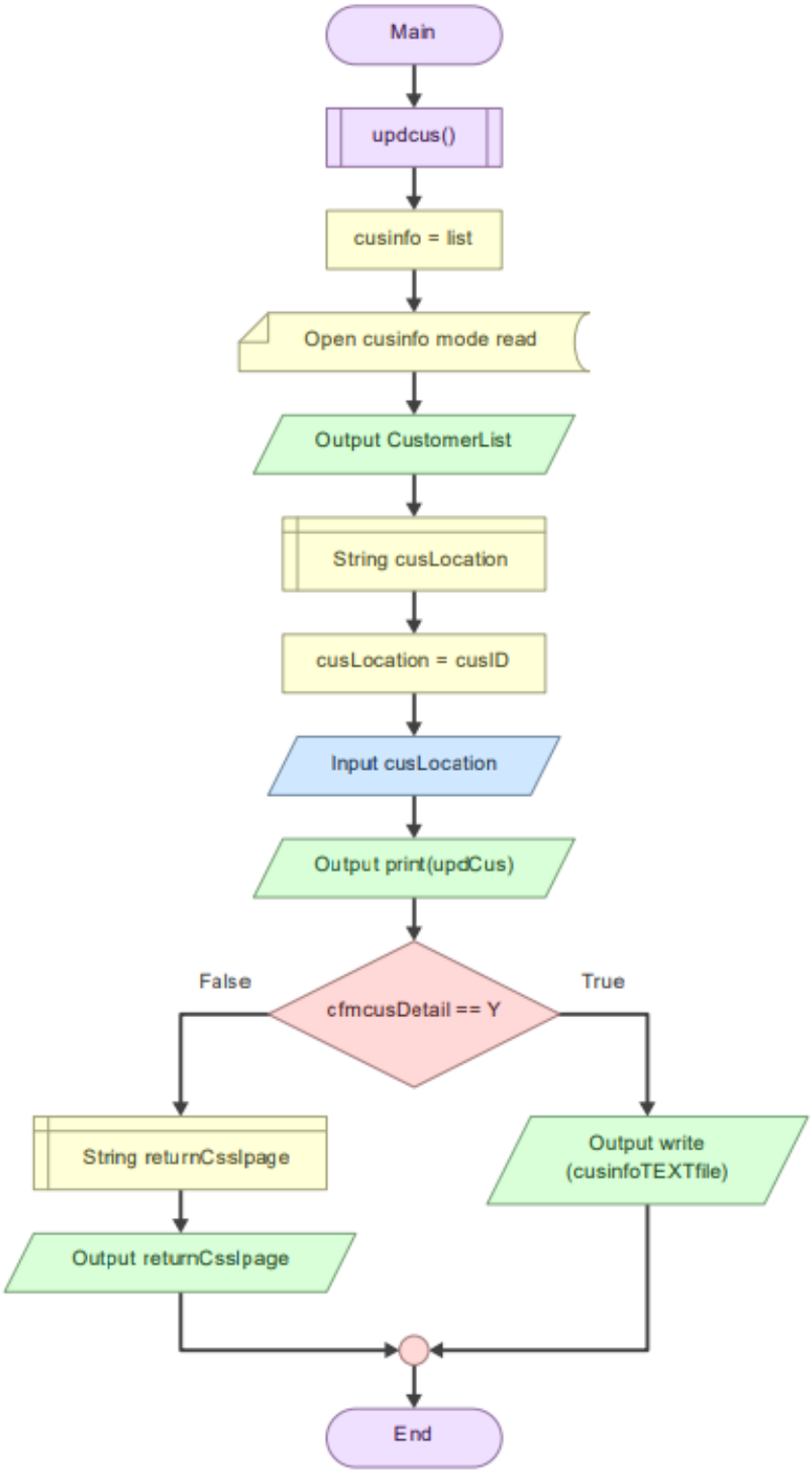
Function to Customer Service Staff I



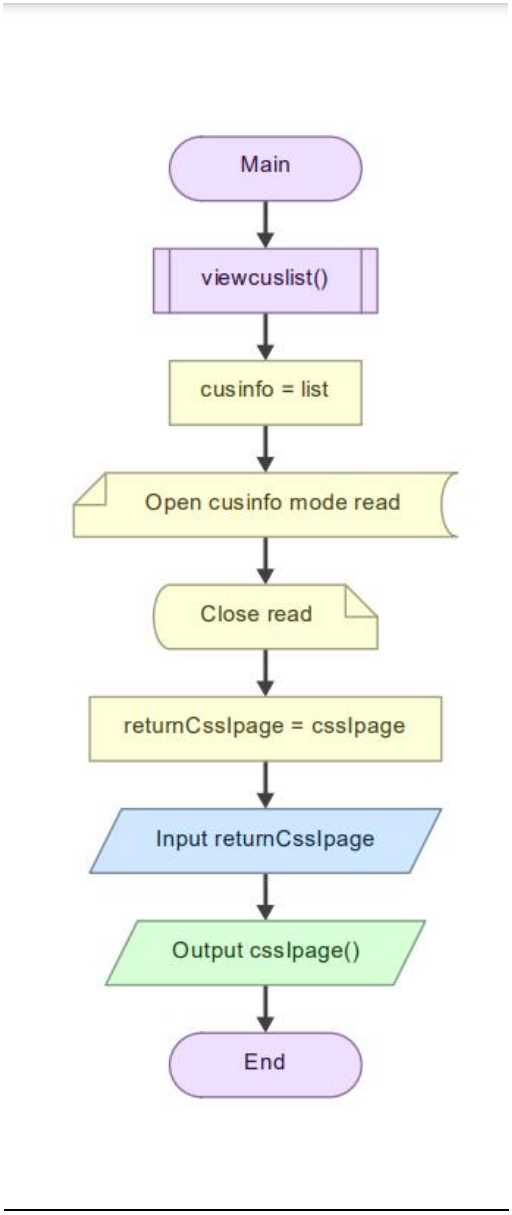
Function to Register Customer



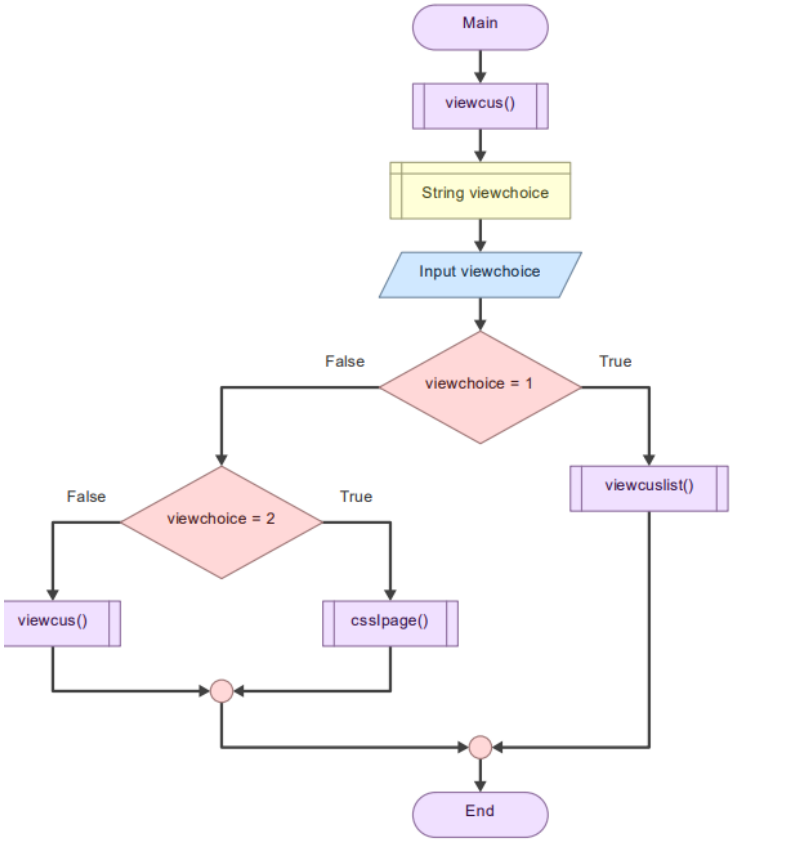
Function to Update Customer



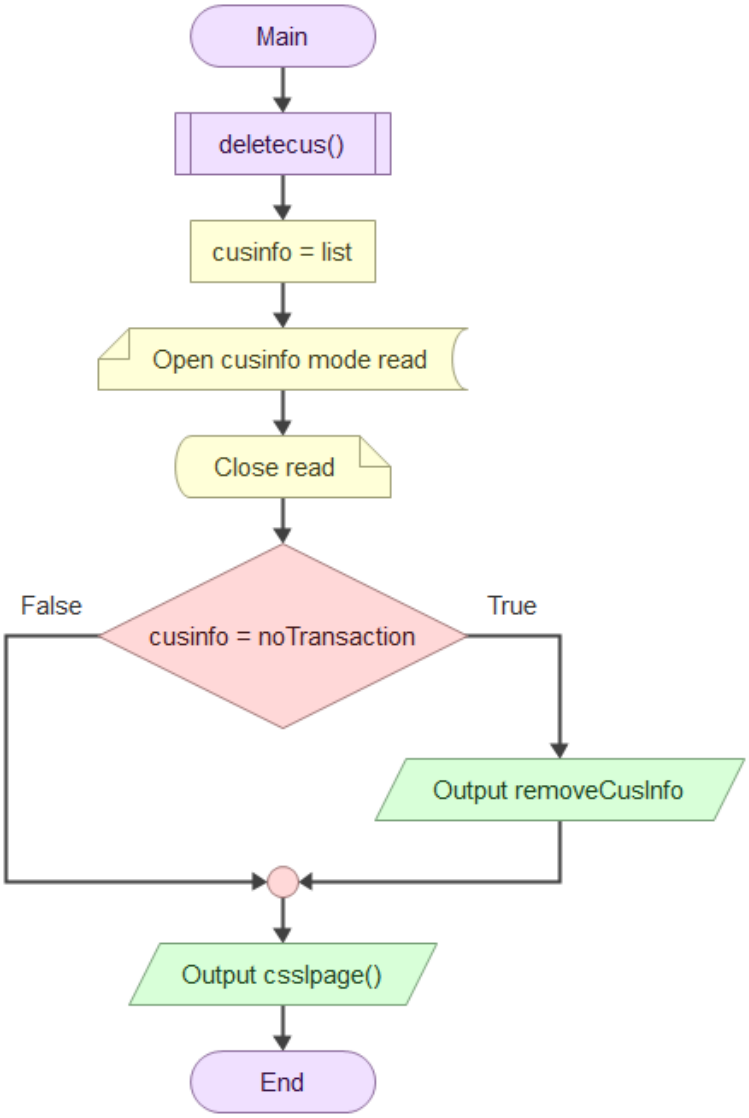
Function to View Customer List



Function to View Customer



Function to Delete Customer



## 3.0 Explanation of programming concept (Wong Zheng Han TP074212)

File handling in python allows users to open, read, write and close file, this is mainly use as data storage, configuration management and logging.

```
from datetime import datetime
import time

# User File
try:
    with open("users.txt", "r") as user_file:
        data = user_file.read()
        if not data:
            with open("users.txt", "a") as user_file:
                user_file.write("")
                user_file.close()
except FileNotFoundError:
    with open("users.txt", "a") as user_file:
        user_file.write("")
        user_file.close()
```

‘import’ used to bring in modules and packages from python library or 3rd party libraries into the current space.

‘open( )’ function is used to open a file, here it opens a file called users.txt.

‘with’ statement ensure the file is properly managed,

‘read( )’ reads the whole file,

‘write( )’ writes a string to file,

‘try’ and ‘except’ blocks potential errors.

A main menu shows the user with a set of options to choose from and direct them to different functions based on their action.

```
def main_page():
    print("#########################################################")
    print ("\n*******WELCOME TO Super Crazy CAR RENTAL SERVICE*******\n")
    print ("#########################################################")
    print ()
    print("1. Login")
    print("2. Register")
    print("3. Exit")
    choice = input("Enter choice : ")
    if choice == "1":

        allusers = []
        with open("users.txt", "r") as user_file:
            for data in user_file:
                user_data = data.strip().split(",")
                allusers.append(user_data)
            user_file.close()
        user = login(allusers)

        if len(user) > 0:
            if user[3] == "car_s":
                crss(user, allusers)
            elif user[3] == "cus_s_I":
                cssI_page(user, allusers)
            elif user[3] == "cus_s_II":
                cssI_page(user, allusers)
            elif user[3] == "manager":
                update_profile(user, allusers)
        else:
            print("System has been disable for 1 minutes")
            time.sleep(60)
```

‘print( )’ display of main menu, option and inputs,

‘input’ identify user’s choice,

‘if,else,elif’ direct the program flow based on the user, here if user choice is 1, it will open log in panel after if user enter log in credential = car_s thenn login elif = cus_s_I, cus_s_II, manager else system disable.

User authentication verifies the identity of the user by validating the credentials.

```
# Login
def login(alluser):

    flag = False
    for cnt in range(3):
        username = input("Enter username : ")
        password = input("Enter password : ")

        for i in range(len(alluser)):
            if username == alluser[i][0] and password == alluser[i][2]:
                user = alluser[i]
                flag = True
        if flag == True:
            break
        else:
            print("Incorrect username or password. Attempts left:", 2 - cnt)

    if flag == False:
        user = []
    return user
```

‘for loop’ allows user to re-attempt if the credentials are incorrect up to 3 times,

‘flag’ indicates that the authentication is successful.

‘return’ returns the authenticated user data if successful.

```
while password != reen_pass:
    print("Password not match")
    reen_pass = input("Reenter Password : ")
```

‘while’ repeatedly execute code until specified condition is true.

```
with open("users.txt", "w") as user_file:
    for i in range(len(alluser)):
        if alluser[i][0] == prev_un:
            alluser[i][0] = new_un
            user_file.write(",".join(alluser[i]) + "\n")
        else:
            user_file.write(",".join(alluser[i]) + "\n")
print("Username changed sucessfully")
```

'in' is used to check if the value exists in the sequence.

```
def record_rental_details(user, alluser):
    car_plate = input("Please enter the Car Plate: ")
    car_index = None
    for i, car in enumerate(car_data):
        if car["carplate"] == car_plate:
            car_index = i
            break
```

'break' is used to terminate the current loop. Whereas 'except' manages error that occurs during the execution of the program.

**Car Service Staff** (Wong Kap Onn TP074292)

Car Service Staff Menu:

```
def crss(user, alluser):
    print("Welcome to the Car Service Staff menu.")
    print("Please select the options below:")
    print("\t1. Add car details \n\t2. Modify car details \n\t3. View car \n\t4. Delete Car \n\t5. Back")


    crss_choice = input("Please enter the option shown above: ")

    if crss_choice == "1":
        print()
        add_car(user, alluser)

    elif crss_choice == "2":
        print()
        modify_car(user, alluser)

    elif crss_choice == "3":
        print()
        view_car(user, alluser)

    elif crss_choice == "4":
        print ()
        delete_car(user, alluser)

    elif crss_choice == "5":
        print()
        main_page()

    else:
        print("\n*******Invalid Input*******\n")
        print("***Please try again.***")
        print()
        crss(user, alluser)
```

This is the car service menu where the car service staff will be directed to after login. The IF-ELSE statement is used so user will be brought to the function of the option entered and for validation if user enter options that are not shown.

Add Car Function:

```
def add_car(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    car_detail = []
    print ("Welcome to the add car page.")
    print ("Please add the following car details:")
    carid = len(carinfo) + 1 #auto generate carid based on the length of the list.
    carid = str(carid)
    car_detail.append(carid)
    Car_plate = input ("Car Registration No: ")
    car_detail.append(Car_plate.upper())
    Car_Manufacturer = (input ("Car Manufacturer: "))
    car_detail.append(Car_Manufacturer.upper())
    Car_Model = input ("Car Model: ")
    car_detail.append(Car_Model.upper())
```

First, the content of the carinfo text file is read and loaded into the master list (carinfo). The car id is auto generated by counting the length of the master list and adding value 1. The new car details are added to the car_detail list. All car details are capitalized for ease of manipulating and finding item in master list (carinfo).

```
cfm_detail = input("Confirm add car detail into system?(Y/N): ")

if cfm_detail.upper() == "Y":
    carinfo.append(car_detail)
    with open('carinfo.txt','w') as carlist:
        for row in carinfo:
            for item in row:
                for chr in str(item):
                    carlist.write(chr)
                if item != row[-1]:
                    carlist.write(',')
            carlist.write('\n')
        carlist.close()

    print("\nCar detail is successfully added to the system.")
    print ("\nUpdated car list:")
    for car_list in (carinfo):
        print(car_list)

else:
    ("\nCar detail is not added to system.")

print()
returnpage = input("Press enter to return to admin menu: ")
crss(user, alluser)
```

After receiving confirmation from user to enter car detail into system, the car_detail list will be added to the master list (carinfo) and the whole master list (carinfo) will be written and replace into the carinfo text file. The content of the master list (carinfo) is written character by character. If the item is not the last item in the individual car list, then a ‘,’ is added after each item. After each individual car list is added, then the next individual car list is written on a new line.

Modify Car Function:

```
def modify_car(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print ("Welcome to the modify page.\n")
    print ("Car List: \n")
    print (car_column)
    for i in (carinfo):
        print (i)
    print("Please enter the car id of the car that you wish the modify.")
    car_location = input("Enter Car Id:")
    car_location = int(car_location) - 1 #Since car id is length of list + 1.
    mod_car =  (carinfo[car_location]) #To find the car through the use of index.
    print ("\nCar detail:\n")
    print (car_column)
    print (mod_car)

    print("\nSelect the option you want to change:")
    print ("1. Car ID\n2. Car Plate\n3. Car Manufacturer\n4. Car Model\n5. Manufacture Year\n6. Seat Capacity\n7. Last Service Date\n8. Insurance Po
    mod_cardetail = input("Enter option: ")
    while mod_cardetail != "1" and mod_cardetail != "2" and mod_cardetail != "3" and mod_cardetail != "4" and mod_cardetail != "5" and mod_cardetail
        print ()
        mod_cardetail = input("Enter option provided: ")
    mod_cardetail = int(mod_cardetail) - 1 #As the first element of the list starts at 0.
    cur_detail = mod_car[mod_cardetail] #To find the specific element of the specific car list.
    print ("\ncurrent detail:", cur_detail)
    upd_detail = input("Enter new value:")
    upd_detail = upd_detail.upper()
    mod_car[mod_cardetail] = upd_detail #Replace the current value of the index of the list mod_car to the new value which is variable upd_detail.
    print ("\nUpdated car detail:")
    print (mod_car)
    print ("\nUpdated car list:")
```

First, the content of the carinfo text file is read and loaded into the master list (carinfo). The master list will be displayed to user, then user will be asked to enter the car id of the car to be modified (car_location). car_location will be used as the index value for the master list (carinfo) to display the individual car list (mod_car = carinfo[car_location]), mod_car will display the individual car list. car_location will be deducted by value 1 as the index of a list starts with 0, then user will be asked to input the detail option that is to be modified (mod_cardetail). Firstly, the current item will be displayed which is found using the index value for the individual car list (cur_detail = mod_car[mod_cardetail]). Secondly, user will be asked to enter a new item which will be capitalized and used to replace the current item (mod_car[mod_cardetail] = upd_detail). Then like the add car function, user will be asked to confirm the changes and the master list (carinfo) will be updated. Once changes are confirmed, the content updated master list (carinfo) will write to and replace the content in the carinfo text file.

View Car Menu:

```
def view_car(user, alluser):
    print()
    print("Welcome to the view menu.")
    print("\nSelect options:")
    print("\t1. All record \n\t2. Available for Rent \n\t3. Rented \n\t4. Search \n\t5. Back")

    view_choice = input("Enter option: ")

    if view_choice == "1":
        print()
        view_allcar(user, alluser)

    elif view_choice == "2":
        print()
        view_availablecar(user, alluser)

    elif view_choice == "3":
        print()
        view_rentedcar(user, alluser)

    elif view_choice == "4":
        print()
        view_specificcar(user, alluser)

    elif view_choice == "5":
        print()
        crss(user, alluser)

    else:
        print("\n*******Invalid Input*******\n")
        print("***Please try again.***")
        print()
        view_car(user, alluser)
```

The view car menu is similar to the car service staff menu where the IF-ELSE statement is used so user will be brought to the function of the option entered and for validation if user enter options that are not shown.

View all car function:

```
def view_allcar(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print ("\nCar List:\n")
    print (car_column)
    for car_list in (carinfo):
        print (car_list)

    returnpage = input("Press enter to return to view menu: ")
    view_car(user, alluser)
```

First, the content of the carinfo text file is read and loaded into the master list (carinfo). Then, using the FOR loop, the content of the master list (carinfo) is displayed list by list of the individual car list.

View available and rented car function:

```
def view_availablecar(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print("\nAvailable Car to rent:\n")
    print(car_column)
    for i in range(len(carinfo)):
        if carinfo[i][13] == "AVAILABLE": #To only print the list that contains AVAILABLE in the 13th index of the list.
            print(carinfo[i])

    returnpage = input("Press enter to return to view menu: ")
    view_car(user, alluser)
```

```
def view_rentedcar(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print("\nRented Car:\n")
    print(car_column)
    for i in range(len(carinfo)):
        if carinfo[i][13] == "RENTED": #To only print the list that contains RENTED in the 13th index of the list.
            print(carinfo[i])
```

First, the content of the carinfo text file is read and loaded into the master list (carinfo). Then, using the FOR loop, for each individual car list that is being searched through, if the 13$^{th}$ index item of the individual car list is equal to “AVAILABLE” / “RENTED”, the individual car list will be displayed.

View specific car function:

```
def view_specificcar(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print()
    print ("Welcome to view specific car page.")
    search_info = input("Enter Car Plate: ")
    print("\nCar Detail: \n")
    print(car_column)
    for i in range(len(carinfo)):
        if carinfo[i][1] == search_info: #To only print the list that contains the user input data in the 1st index of the list.
            print(carinfo[i])

    print ()
    returnpage = input("Press enter to return to view menu: ")
    view_car(user, alluser)
```

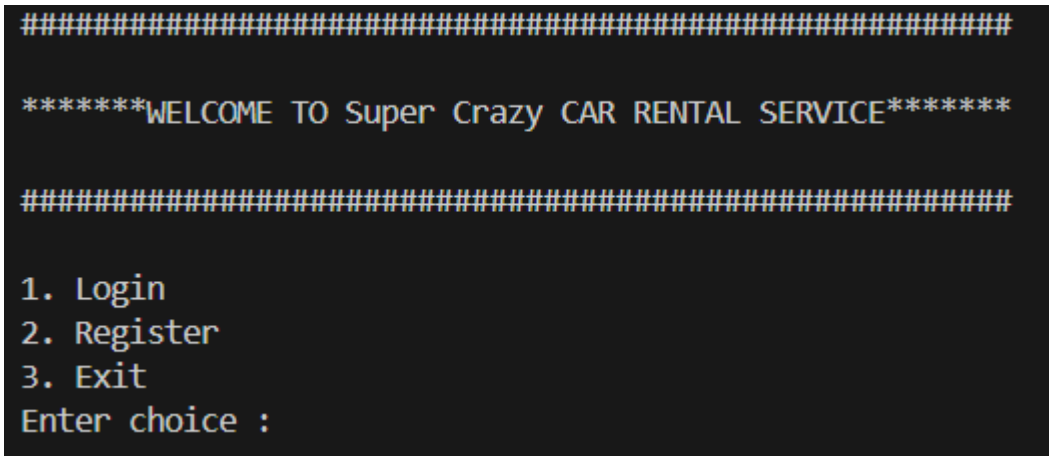
First, the content of the carinfo text file is read and loaded into the master list (carinfo). User will be asked to enter the car plate of the car they want to view. Then, using the FOR loop, for each individual car list that is being searched through, if the 1st index item of the individual car list is equal to the user input, then the individual car list will be displayed.

Delete Car Function:

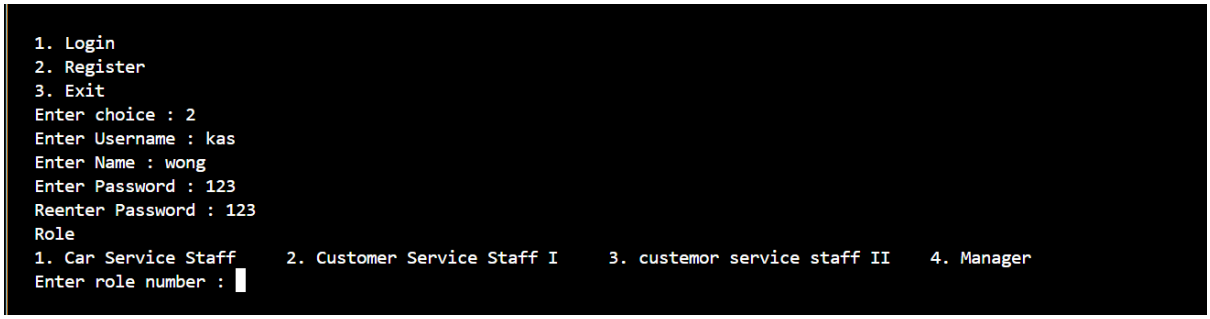
```
def delete_car(user, alluser):
    carinfo = []
    carlist = open('carinfo.txt', 'r')
    for line in carlist:
        line = line.rstrip()
        carinfo.append(line.split(','))
    carlist.close()
    print()
    print("\nCar to be disposed:\n")
    print(car_column)
    for i in range(len(carinfo)):
        if carinfo[i][13] == "DISPOSED": #To only print the list that contains DISPOSED in the 13th index of the list.
            print(carinfo[i])
    cfm_delete = input("Are you sure you want to delete this car? (Y/N): ")
    if cfm_delete.upper() == "Y":
        carinfo.remove(carinfo[i]) #To remove the list from the master list.
        with open('carinfo.txt','w') as carlist:
            for row in carinfo:
                for item in row:
                    for chr in str(item):
                        carlist.write(chr)
                    if item != row[-1]:
                        carlist.write(',')
                carlist.write('\n')
            carlist.close()

        print("\nCar detail is successfully deleted from the system.")
        print ("\nUpdated car list:")
        for car_list in (carinfo):
                print(car_list)

    else:
        print ("Car failed to delete from system.")

    returnpage = input("Press enter to return to admin menu: ")
    crss(user, alluser)
```

First, the content of the carinfo text file is read and loaded into the master list (carinfo). Then, using the FOR loop, for each individual car list that is being searched through, if the 13th index item of the individual car list is equal to “DISPOSED”, the individual car list will be displayed. Then user will be asked whether to delete car from system. If user choose yes, then the individual car list will be removed from the master list (carinfo.remove(carinfo[i])). Then the updated master list (carinfo) will be written and replace to the carinfo text file. If user choose no, then the message “Car failed to delete from system.” is displayed.

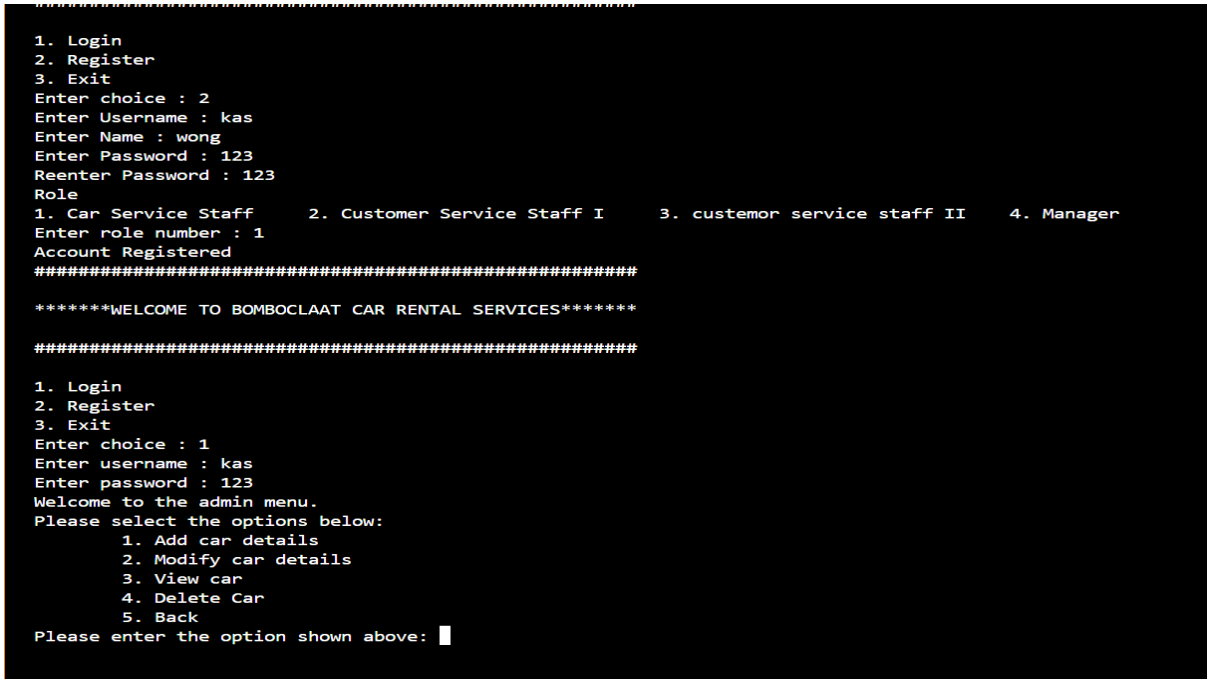
## 4.0 Sample of Input/Output (Wong Zheng Han TP074212)



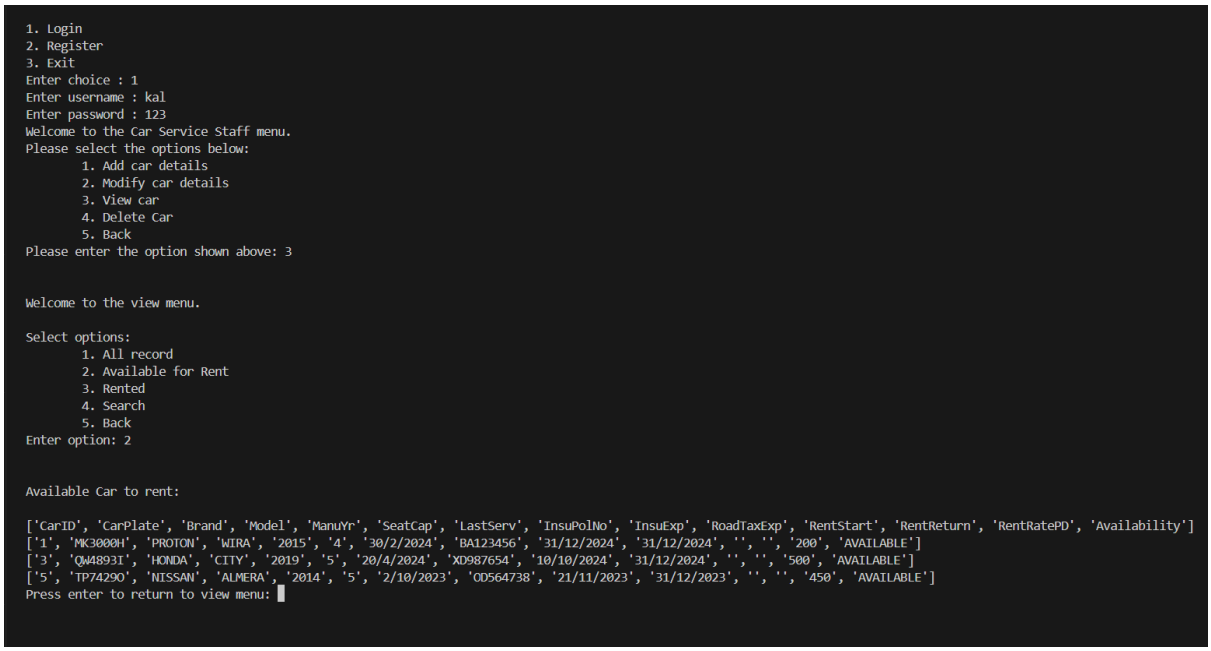

User/Staff interacts with system to register, login or exit.



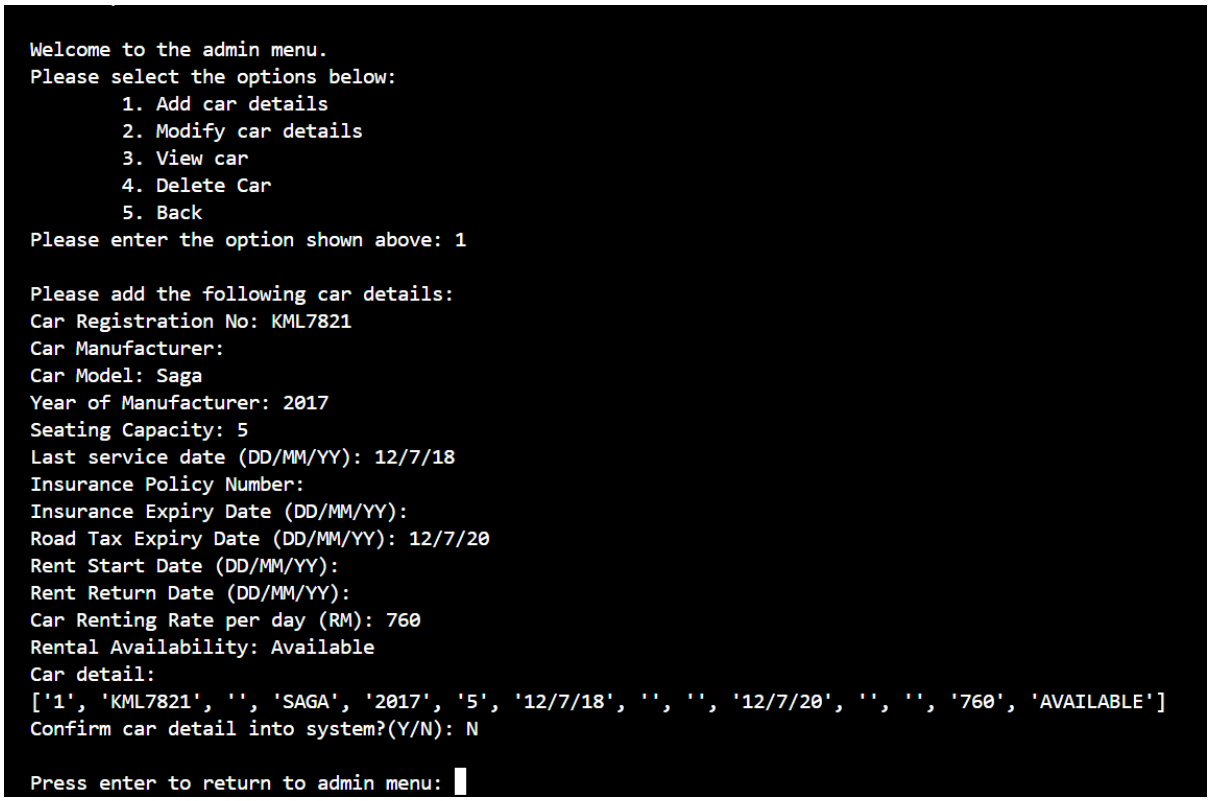

Choice 2 register account: user needs to input a username, their name and password with confirmation then they will proceed to choose which role they are registering for.



After account registered, user will be redirected to main menu. User will select option 1, login by entering username and password then user will be brought to the menu of the user's role.




After entering option 3 'view car', user will be directed to the view menu. When user select option 2, 'Available for Rent', user will be shown which car is available now, after that user can press enter to return to the last menu.




Entering 1 allows user to add new car details, after selecting the option, user will be directed

to enter series of detail for the car ranging from car registration numbers to rental availability, after all details are entered, user is required to confirm if the data you entered to be stored into the system.

```
Please select the options below:
        1. Add car details
        2. Modify car details
        3. View car
        4. Delete Car
        5. Back
Please enter the option shown above: 4



Car to be disposed:

['CarID', 'CarPlate', 'Brand', 'Model', 'ManuYr', 'SeatCap', 'LastServ', 'InsuPolNo', 'InsuExp', 'RoadTaxExp', 'RentStart', 'RentReturn', 'RentRatePD', 'Availability']
['6', 'WYL607', 'PERODUA', 'ALZA', '2015', '8', '31/12/2023', 'RT123456', '31/12/2024', '31/12/2024', '', '', '500', 'DISPOSED']
Are you sure you want to delete this car? (Y/N): Y

Car detail is successfully deleted from the system.

Updated car list:
['1', 'MK3000H', 'PROTON', 'WIRA', '2015', '4', '30/2/2024', 'BA123456', '31/12/2024', '31/12/2024', '', '', '200', 'AVAILABLE']
['2', 'SC5674J', 'PERODUA', 'MYVI', '2018', '5', '15/3/2024', 'CD789123', '20/11/2024', '31/12/2024', '4/7/2024', '6/7/2024', '400', 'RESERVED']
['3', 'QW4893I', 'HONDA', 'CITY', '2019', '5', '20/4/2024', 'XD987654', '10/10/2024', '31/12/2024', '', '', '500', 'AVAILABLE']
['4', 'PO5438Q', 'TOYOTA', 'VIOS', '2017', '5', '8/11/2023', 'FT019238', '3/9/2024', '31/12/2024', '', '', '600', 'UNDERSERVICE']
['5', 'TP7429O', 'NISSAN', 'ALMERA', '2014', '5', '2/10/2023', 'OD564738', '21/11/2023', '31/12/2023', '', '', '450', 'AVAILABLE']
Press enter to return to admin menu:
```

When user selects option 4 ‘Delete Car’, the car detail that has ‘DISPOSED’ for Availability will be displayed and user will be given the option to confirm the removal of the car detail in the system. Then the updated list will be displayed to user.

## 5.0 Conclusion

In this documentation, we have explored the main functionality of the car rental application developed by using python. This application includes user authentication, customer management, car rental details, and rental processing. User authentications are used for user login and registration using file handling to ensure user data to be secured. Customers management allows users to modify customer detail and handle customer information. Car rental details this is used for user to modify car detail like number plate, road tax expiry date, and change rental availability, etc. Rental processing calculates the rental duration and bills of the car.

## 6.0 Reference


Fish, D. T. (2024, April 27). *Modify an item within a list in Python*.

https://datatofish.com/modify-list-python/

*Python Tutorial*. (n.d.). https://www.w3schools.com/python/

# 7.0 Workload Matrix

| Name | Student ID | Tasks |
|---|---|---|
| Wong Kap Onn (Group Leader) | TP074292 | • Car Service Staff (Code)<br>• Pseudocode and Flowchart (Car Service Staff)<br>• carinfo.txt<br>• Explanation of programming concepts (Car Service Staff) |
| Wong Zheng Han | TP074212 | • Introduction and Assumption<br>• Explanation of programming concepts<br>• Sample Input/Output |
| Tan Yi Han | TP070378 | • Customer Service Staff I (Code)<br>• Pseudocode and Flowchart (Customer Service Staff I)<br>• cusinfo.txt |
| Timothy Tan Chern Tian | TP074658 | • Customer Service Staff II (Code)<br>• Pseudocode and Flowchart (Customer Service Staff II) |
| Tan Yen Hann | TP073629 | • Manager (Code)<br>• Pseudocode and Flowchart (Manager)<br>• users.txt |