GROUP ASSIGNMENT

TECHNOLOGY PARK MALAYSIA

AAPP010-4-2-PWP

PROGRAMMING WITH PYTHON

UCDF2005(1)-ICT(DI)

HAND OUT DATE: 08TH APRIL 2021

HAND IN DATE: 18TH JUNE 2021

WEIGHTAGE: 100%

LECTURER: Mr. Sivaguru Subarmaniyan

GROUP MEMBER:

1. **Lim Jia Yong** **TP059192**
2. **Leanne Ooi Xin Ru** **TP059138**

Table of Contents

1.0 Introduction

As businesses grow, it is logical for traditional businesses to build an automated system or program to increase efficiency of business managements (S., 2018). Super Car Rental Services (SCRS) is an online car rental service based in Malaysia which allows customers to rent cars online. SCRS have decided to develop a Python program for three groups of users, namely admins, unregistered customers, and registered customers.

Unregistered customers may register for a new account or remain unregistered and view all available cars for rent. Conversely, registered customers can view available cars for rent, modify personal details such as password, email, and phone number, view personal rental history, and rent cars. Admins have higher privileges than customers as they are allowed to add new cars to the system, alter car details such as daily rental rate, review all or specific records of customer bookings and payments, as well as return a car.

1.1 Assumptions

Several assumptions have been made for this program. Firstly, users are assumed to cooperate with the system and are not in any way attempting to inject the system. Users are also assumed to insert information that corresponds to each input field.

Apart from that, new users registering for a new account are assumed to sign up as customer instead of admin. This is done to prevent regular users from modifying any crucial details in the system.

Furthermore, renting cars is on a per day basis and the currency used will be in Malaysian Ringgit (RM). Customers are assumed to book cars with a minimum duration of 1 day and pay on the same day upon the booking of car. The starting date of car rental is assumed to be 7 days from the date where customer booked the car. They are also assumed to be reasonable and not rent more than one car on the same day.

2.0 Program Design

2.1 Pseudocode

```
PROGRAM SCRS
BEGIN
    call menu()
END

=====================:

FUNCTION menu()
    Print "Welcome to Super Car Rental Service (SCRS)."

    Print "Do you want to continue? [‘0’ to Continue ‘-1’ to Terminate]: "
    Read option

    DOWHILE (option == 0)

        Print "Your options are: "
        Print "1.       Register as new customer."
        Print "2.       Log in as customer."
        Print "3.       Log in as admin."
        Print "4.       Remain as unregistered customer."

        Print "Enter your option: "
        Read option

        IF (option == 1) THEN
            username = False
            DOWHILE (username = False)
                username = signup()
            ENDDO
            call registeredCustView(username)

        ELSE IF (option == 2) THEN
            username = False
            DOWHILE (username = False)
                username = call login('customer')
            ENDDO
            call registeredCustView(username)


        ELSE IF (option == 3) THEN
            username = False
            DOWHILE (username = False)
                username = call login('admin')
            ENDDO
            call adminView(username)

        ELSE IF (option == 4) THEN
            call customerView()

        ELSE
            Print "Invalid Input"
        ENDIF

        Print "Do you want to continue? [‘0’ to   ‘-1’ to Terminate]: "
        Read option
    ENDDO

    Print "Thank you and have a nice day!"
ENDFUNCTION

=========================================================================
```

```
FUNCTION cars(availability)
    TRY
        Read bookingsFile From "bookings.txt"
            Declare availableCars[]
            Declare unavailableCars[]
            Declare bookedCars[]

            LOOP line IN bookingsFile
                SET bookingDetails TO line.rstrip().split(", ")
                IF (bookingDetails[6] == "False") THEN
                    Append bookingDetails[1] Into bookedCars
                ENDIF
                NEXT line
            ENDLOOP

            TRY
                Read carsFile From "cars.txt"
                    IF (availability == "all") THEN
                        Declare cars[]
                        LOOP line IN carsFile
                            SET carDetail TO line.rstrip().split(", ")
                            Append carDetail Into cars
                            NEXT line
                        ENDLOOP
                        RETURN cars
                    ELSE

                        LOOP line IN carsFile
                            SET carDetail TO line.rstrip().split(", ")
                            IF (carDetail[0] NOT IN bookedCars) THEN
                                Append carDetail Into availableCars
                            ELSE
                                Append carDetail Into unavailableCars
                            ENDIF
                            NEXT line
                        ENDLOOP
                    ENDIF
                CLOSEFILE "cars.txt"
            ENDTRY
            EXCEPT
                Print "Could not open 'cars.txt' file."
                CLOSEFILE "bookings.txt"
                RETURN
            ENDEXCEPT
        CLOSEFILE "bookings.txt"

    ENDTRY
    EXCEPT
        Print "Could not open 'bookings.txt' file."
        RETURN
    ENDEXCEPT

    IF (availability == 'available') THEN
        RETURN availableCars
    ELSE
        RETURN unavailableCars
    ENDIF
ENDFUNCTION

================================================================================

FUNCTION signup()
    Print "Please fill in the information below to signup."
    Read username
    Read email
    Read phoneNo
    Read password
    Read confirmation

    IF (password != confirmation) THEN
        Print "Passwords must match. Please register again."
        RETURN False
    ENDIF

    TRY
        Read file From "users.txt"
            LOOP line IN file
                SET usersDetail TO line.rstrip().split(", ")
                IF (usersDetail[0] == username) AND (userDetails[2] == 'customer') THEN
                    Print "Sorry, username has been taken. Please register with another username."
                    CLOSEFILE "users.txt"
                    RETURN False
                ENDIF
                NEXT line
            ENDLOOP

            OPENFILE "users.txt" FOR APPEND
                line = username, ",", password, ", customer, ", email, ",", phoneNo, "\n"
                Append line Into "users.txt"
                Print "Registeration successful, welcome ", username
            CLOSEFILE "users.txt"
        CLOSEFILE "users.txt"

    ENDTRY
    EXCEPT
        Print "Error occured in 'users.txt' file."
        RETURN
    ENDEXCEPT

    RETURN username
ENDFUNCTION

================================================================================
```

```
FUNCTION login(userType)
	Print "Please fill in the your credentials. Note: You are currently trying to login as", userType
	Read username
	Read password

	Try
		Read file From "users.txt"
			LOOP line IN file
				SET usersDetail TO line.rstrip().split(', ')
				IF (usersDetail[0] == username) AND (usersDetail[1] == password) AND (usersDetail[2] == userType) THEN
					Print "Welcome back", username
					RETURN user

				ENDIF
				NEXT line
			ENDLOOP

		CLOSEFILE "users.txt"

	ENDTRY
	EXCEPT
		Print "Could not open 'users.txt' file.\n"
		RETURN
	ENDEXCEPT

	Print "Invalid username and(or) password.\n"
	RETURN False

ENDFUNCTION
```

==========

```
FUNCTION registeredCustView(username)
	Print "Your options are: "
	Print "1. Modify personal details."
	Print "2. View personal rental history."
	Print "3. View detail of cars to be rented out."
	Print "4. Select and book a car for a specific duration."
	Print "5. Log out."

	Print "Enter your option: "
	Read option

	IF (option == 1) THEN
		call modifyCustDetails(username)

	ELSE IF (option == 2) THEN
		call displayRentalHistory(username)

	ELSE IF (option == 3) THEN
		call displayCars('available')


	ELSE IF (option == 4) THEN
		call bookCar(username)

	ELSE IF (option == 5) THEN
		RETURN

	ELSE
		Print "Invalid input."
	ENDIF

	call registeredCustView(username)

ENDFUNCTION
```

==========

```
FUNCTION modifyCustDetails(username)
        TRY
                Read file From "users.txt"
                        index = 0
                        lines = READLINES(file)

                        LOOP line IN file
                                index = index + 1
                                SET usersDetails TO line.rstrip().split(', ')
                                IF (userDetails[0] != username) THEN
                                        NEXT line
                                ENDIF

                                Print "These are your current details: "
                                Print "Email: ", userDetails[3]
                                Print "Phone Number: ", userDetails[4],"\n"

                                Print "Your options are: "
                                Print "1. Change password."
                                Print "2. Change email."
                                Print "3. Change phone number."
                                Print "4. Exit."

                                Print "Enter your option: "
                                Read option


                             IF (option == 1) THEN
                                    call modifyPassword(userDetails, lines, index)

                             ELSE IF (option == 2) THEN
                                    call modifyEmail(userDetails, lines, index)

                             ELSE IF (option == 3) THEN
                                    call modifyPhoneNo(userDetails, lines, index)

                             ELSE IF (option == 4) THEN
                                    RETURN

                             ELSE
                                    Print "Invalid input."
                             ENDIF

                             call modifyCustDetails(username)

                             NEXT line

                        ENDLOOP

                CLOSEFILE "users.txt"

        ENDTRY
        EXCEPT
                Print "Could not open "users.txt" file.\n"
                RETURN

        ENDEXCEPT

ENDFUNCTION

================================================================================================


FUNCTION modifyPassword(userDetails, lines, index)
        Read current_password
        password = userDetails[1]

        IF (current_password != password) THEN
                Print "Invalid password."
                RETURN modifyPassword(userDetails, lines, index)

        ENDIF
        Read new_password
        Read confirm_password

        IF(new_password != confirm password) THEN
                Print "Password must match."
                RETURN modifyPassword(userDetails, lines, index)

        ELSE
                call updateUserFile("password", userDetails, lines, index, new_password)
        ENDIF

ENDFUNCTION

================================================================================================
```

```
FUNCTION modifyEmail(userDetails, lines, index)
        Read new_email
        Read confirm_email

        IF (new_email != confirm_email) THEN
                Print "Emails must match."
                RETURN modifyEmail(userDetails, lines, index)

        ELSE
                call updateUserFile("email", userDetails, lines index, new_email)

        ENDIF

ENDFUNCTION

================================================================================

FUNCTION modifyPhoneNo(userDetails, lines, index)
        Read new_phoneNo
        Read confirm_phoneNo

        IF (new_phoneNo != confirm_phoneNo) THEN
                Print "Phone number must match."
                RETURN modifyPhoneNo(userDetails, lines, index)

        ELSE
                call updateUserFile("phone number", userDetails, lines index, new_phoneNo)

        ENDIF

ENDFUNCTION

============================================================================================

FUNCTION updateUserFile(key, userDetails, lines, index, information)
        username = userDetails[0]
        IF (key == "password") THEN
                currentInfo = userDetails[1]
                lines[index] = username, information, userDetails[2], userDetails[3], userDetails[4],"\n"

        ELSE IF (key == "email") THEN
                currentInfo = userDetails[3]
                lines[index] = username, userDetails[1], userDetails[2], information, userDetails[4],"\n"

        ELSE
                currentInfo = userDetails [4]
                lines[index] = username, userDetails[1], userDetails[2], userDetails[3], information,"\n"

        ENDIF

        Print username, ", your", key, "will be modified from", currentInfo, "to", information, "."
        Print "Do you agree to this modification? ['0' = No, '1'= Yes]: "
        Read answer

        IF (answer == 1) THEN
                TRY
`                       OPENFILE "users.txt" FOR WRITE
                                Write lines Into "users.txt"
                                Print username, ", you have successfully modified your", key, "."

                        CLOSEFILE "users.txt"

                ENDTRY

                EXCEPT
                        Print "Error occured in "users.txt" file.\n"

                ENDEXCEPT

        ELSE
                Print "No changes made."

        ENDIF

        RETURN

ENDFUNCTION

============================================================================================
```

```
FUNCTION displayRentalHistory(user)
	found = False
	TRY
		Read file From "bookings.txt"
			Print "Username  Car Plate  Start Date  End Date  Daily Rate (RM)  Pay Date  Return Status"
			LOOP line IN file
				SET bookingDetails TO line.rstrip().split(', ')
				IF (user == bookingDetails[0]):
					found = True
					Print bookingDetails[0], ",", bookingDetails[1], ",", bookingDetails[2], ",", bookingDetails[3], ",", bookingDetails[4], ",",
						bookingDetails[5], ",", bookingDetails[6]

				ENDIF
				NEXT line

			ENDLOOP
			IF (found == False) THEN
				Print "No personal rental history found."
			ENDIF

		CLOSEFILE "bookings.txt"

	ENDTRY

	EXCEPT
		Print "Could not open "bookings.txt" file.""\n"

	ENDEXCEPT

	RETURN

ENDFUNCTION
```

```
FUNCTION displayCars(availability)
	vehicles = cars(availability)
	Print availability, "cars for rent: "
	Print "Number Plate  Model Name  Number of Seats  Rental Price (RM)"

	FOR EACH vehicle IN vehicles
		Print vehicle[0], ",", vehicle[1], ",", vehicle[2], ",", vehicle[3]
		NEXT vehicle

	ENDFOR

	RETURN

ENDFUNCTION
```

```
FUNCTION bookCar(username)
	call displayCars("available")
	Read carNo

	TRY
		Read file From "cars.txt"
			found = False
			LOOP line IN file
				SET carDetails to line.rstrip().split(', ')
				IF (carNo != carPlate) THEN
					NEXT line

				ENDIF
				found = True

				validStartDate = False
				DOWHILE (validStartDate == False)
					Read firstDate
					Read todayDate
					numOfDaysFromToday = firstDate - todayDate
					IF (numOfDaysFromToday.days <= 7) and (numOfDaysFromToday.days > 0) THEN
						validStartDate = True

					ELSE
						Print "Please make sure the starting date is after today and within 7 days from today.\n"
						NEXT line

					ENDIF

					Read lastDate
					numberOfDays = lastDate - firstDate

					IF (numberOfDays <= 0) THEN
						Print "Start date has to be before end date."
						NEXT line

					ENDIF

					bookingPrice = numberOfDays * price
					payDate = todayDate
					return_status = "False"
```

```
                                Print username, ", you will be booking", carNo, "from", firstDate, "to", lastDate, "with the price of RM", bookingPrice, "."
                                Print "Do you agree to this booking? ['0'= No, '1'= Yes]: "
                                Read answer

                                IF (answer == 1) THEN
                                        TRY
                                                OPENFILE "bookings.txt" FOR APPEND
                                                        line = username, ",", carNo, ",", firstDate, ",", lastDate, ",", bookingPrice, ",", payDate,
                                ",",return_status, "\n"
                                                        Append line Into "bookings.txt"
                                                        Print "Booking successful, thank you", username

                                                CLOSEFILE "bookings.txt"

                                        ENDTRY

                                        EXCEPT
                                                Print "Error occured in "bookings.txt" file.""\n"
                                                BREAK

                                        ENDEXCEPT

                                ELSE
                                        Print "The booking process will not be carried out."

                                ENDIF

                        ENDDO
                        NEXT line

                ENDLOOP
                IF (found == False) THEN
                        Print "No match for car number plate -", carNo, "\n"
                ENDIF

        CLOSEFILE "cars.txt"

    ENDTRY

    EXCEPT
        Print "Could not open "cars. txt" file.""\n"

    ENDEXCEPT

    RETURN

ENDFUNCTION
```

```
FUNCTION adminView()
    Print "Your options are: "
    Print "1.   Add cars to be rented out."
    Print "2.   Modify car details."
    Print "3.   Display all details of cars rented out."
    Print "4.   Display all details of cars available for rent."
    Print "5.   Display all details of customer bookings."
    Print "6.   Display all details of customer payment for a specific time duration."
    Print "7.   Search specific record of customer booking."
    Print "8.   Search specific record of customer payment."
    Print "9.   Return a rented car."
    Print "10.  Log out."

    Print "Enter your option: "
    Read option

    IF (option == 1) THEN
        call addCars()

    ELSE IF (option == 2) THEN
        call modifyCarDetails()

    ELSE IF (option == 3) THEN
        call displayCars('unavailable')

    ELSE IF (option == 4) THEN
        call displayCars('available')


    ELSE IF (option == 5) THEN
        call displayCustBookings()

    ELSE IF (option == 6) THEN
        call displayTimePayment()

    ELSE IF (option == 7) THEN
        call searchCustBookings()

    ELSE IF (option == 8) THEN
        call searchCustPayment()

    ELSE IF (option == 9) THEN
        call returnCar()

    ELSE IF (option == 10) THEN
        Print "Successfully logged out."
        RETURN

    ELSE
        Print "Invalid Input."
    ENDIF

    call adminView()

ENDFUNCTION

============================================================

FUNCTION addCars()
    Print "Please fill in the information below to add a new car."
    Read carPlate
    Read carModel
    Read seaters
    Read price

    TRY
        Read file From "cars.txt"
            LOOP line IN file
                SET carDetail TO line.rstrip().split(", ")
                IF (carDetail[0] == carPlate) THEN
                    Print "Car with the number plate", carPlate, "already exist."
                    CLOSEFILE "cars.txt"
                    RETURN
                ENDIF
                NEXT line
            ENDLOOP

            Print "Car model", carModel, "(", carPlate, ")", "with", seaters,
                  "seaters and a daily rental rate of RM", price, "will be added."
            Print "Do you wish to add this car? ['0'= No, '1'= Yes]: "
            Read answer
```

```
            IF (answer == 1) THEN
                TRY
                    OPENFILE "cars.txt" FOR APPEND
                        line = carPlate, ",", carModel, ",", seaters, ",", price, "\n"
                        Append line Into "cars.txt"
                        Print "Car with the number plate", carPlate, "succesfully added."
                    CLOSEFILE "cars.txt"
                ENDTRY
                EXCEPT
                    Print "Error occured in 'cars.txt' file."
                ENDEXCEPT
            ELSE
                Print "No changes made."
            ENDIF
        CLOSEFILE "cars.txt"

    ENDTRY
    EXCEPT
        Print "Error occured in 'cars.txt' file."
    ENDEXCEPT

    RETURN
ENDFUNCTION

=============================================================================================

FUNCTION modifyCarDetails()
        call displayCars('all')

        TRY
                Read file From "cars.txt"
                        lines = READLINES(file)
                        Read carModel
                        Declare sameModelCars[]
                        found = False
                        index = 0

                        FOR EACH line IN lines
                                index = index + 1
                                SET carDetails TO line.rstrip().split(', ')
                                IF (carModel != carDetails[1]) THEN
                                        NEXT line
                                ENDIF

                                found = True


                                Append index Into carDetails
                                Append carDetails Into sameModelCars

                                NEXT line

                        ENDFOR

                        IF (found == False) THEN
                                Print "Car model - ", carModel, "does not exist."
                        ELSE
                                call modifyPrice(lines, sameModelCars)

                        ENDIF

                CLOSEFILE "cars.txt"

        ENDTRY

        EXCEPT
                Print "Could not open "cars.txt" file.""\n"

        ENDEXCEPT

        RETURN

ENDFUNCTION

=============================================================================
```

```
FUNCTION modifyPrice(lines, sameModelCars)
        Read new_price
        currentPrice = sameModelCars[0][3]
        modelName = sameModelCars[0][1]

        Print "The daily rental rate for all models of", modelName, "will be modified from RM", currentPrice, "to RM", new_price, "."
        Print "Do you agree to this modification? ['0'= No, '1'= Yes]: "
        Read answer

        IF (answer == 1) THEN
                FOR EACH car IN  sameModelCars
                        index = car[4]
                        carPlate = car[0]
                        lines[index] = carPlate, ",", car[1], ",", car[2], ",", new_price, "\n"
                ENDFOR

                TRY
                        OPENFILE "cars.txt" FOR WRITE
                                Write lines Into "cars.txt"
                                Print "You have successfully modified the daily rental rate for all models of", modelName, "."

                        CLOSEFILE "cars.txt"

                ENDTRY

                EXCEPT
                        Print "Error occured in "cars.txt" file.""\n"

                ENDEXCEPT

        ELSE
                Print "No changes made."

        RETURN

ENDFUNCTION

=============================================================================================================================================

FUNCTION displayCustBookings()
        TRY
                Read file From "bookings.txt"
                        Print "Username  Car Plate  Start Date  End Date  Daily Rate  Pay Date  Return Status"

                        LOOP line IN file
                                SET bookingDetails TO line.rstrip().split(', ')
                                Print bookingDetails[0], ",", bookingDetails[1], ",", bookingDetails[2], ",", bookingDetails[3], ",", bookingDetails[4]
                                NEXT line

                        ENDLOOP

                CLOSEFILE "bookings.txt"

        ENDTRY

        EXCEPT
                Print "Error opening "bookings.txt"."

        ENDEXCEPT

        RETURN

ENDFUNCTION

=============================================================================================================================================
```

```
FUNCTION displayTimePayment()
    Print "Fill in details below to search for specific record of customer payment."
    Read startDate
    Read endDate

    TRY
        Read file From "bookings.txt"
            found = False
            index = 0
            total = 0

            LOOP line IN file
                SET bookingDetails TO line.rstrip().split(", ")
                IF (bookingDetails[5] >= startDate) AND (bookingDetails[5] <= endDate) THEN
                    found = True
                    index = index + 1
                    total = total + bookingDetails[4]
                    Print index, bookingDetails[0], "paid RM", bookingDetails[4], "for car plate",
                                bookingDetails[1], "in", bookingDetails[5]
                ENDIF
                NEXT line
            ENDLOOP
        CLOSEFILE "bookings.txt"

        IF (found == False) THEN
            Print "No customer payment record found from", bookingDetails[2], "to", bookingDetails[3]
        ELSE
            Print "Total revenue generated from", bookingDetails[2], "to", bookingDetails[3], ": RM", total
        ENDIF
    ENDTRY
    EXCEPT
        Print "Could not open 'bookings.txt' file."
    ENDEXCEPT

    RETURN
ENDFUNCTION
```

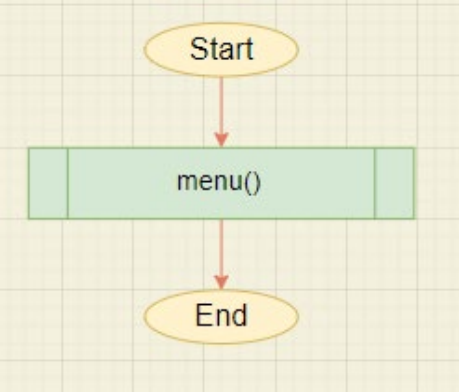
```
FUNCTION searchCustBookings()
    Print "Fill in details below to search for specific record of customer booking."
    Read username
    Read carPlate

    TRY
        Read file From "bookings.txt"
            found = False
            index = 0

            IF (carPlate == '') THEN
                LOOP line IN file
                    SET bookingDetails TO line.rstrip().split(", ")
                    IF (username == bookingDetails[0]) THEN
                        found = True
                        index = index + 1
                        Print index, username, "booked for car with number plate", bookingDetails[1],
                              "from", bookingDetails[2], "to", bookingDetails[3], "return status =", bookingDetails[6]
                    ENDIF
                    NEXT line
                ENDLOOP
            ELSE
                LOOP line IN file
                    SET bookingDetails TO line.rstrip().split(", ")
                    IF (username == bookingDetails[0]) AND (carPlate == bookingDetails[1]) THEN
                        found = True
                        index = index + 1
                        Print index, username, "booked for car with number plate", carPlate, "from",
                              bookingDetails[2], "to", bookingDetails[3], "return status =", bookingDetails[6]
                    ENDIF
                    NEXT line
                ENDLOOP
            ENDIF
        CLOSEFILE "bookings.txt"

        IF (found == False) THEN
            IF (carPlate == '') THEN
                Print "No bookings record found for", username
            ELSE
                Print username, "has not booked a car with number plate of", carPlate
            ENDIF
        ENDIF
    ENDTRY
    EXCEPT
        Print "Could not open 'bookings.txt' file."
    ENDEXCEPT

    RETURN
ENDFUNCTION
```

```
FUNCTION searchCustPayment()
    Print "Fill in details below to search for specific record of customer payment."
    Read username
    Read carPlate

    TRY
        Read file From "bookings.txt"
            found = False
            index = 0

            IF (carPlate == '') THEN
                LOOP line IN file
                    SET bookingDetails TO line.rstrip().split(", ")
                    IF (username == bookingDetails[0]) THEN
                        found = True
                        index = index + 1
                        Print index, username, "paid RM" bookingDetails[4], "for car plate",
                              bookingDetails[1], "in", bookingDetails[5]
                    ENDIF
                    NEXT line
                ENDLOOP
            ELSE
```

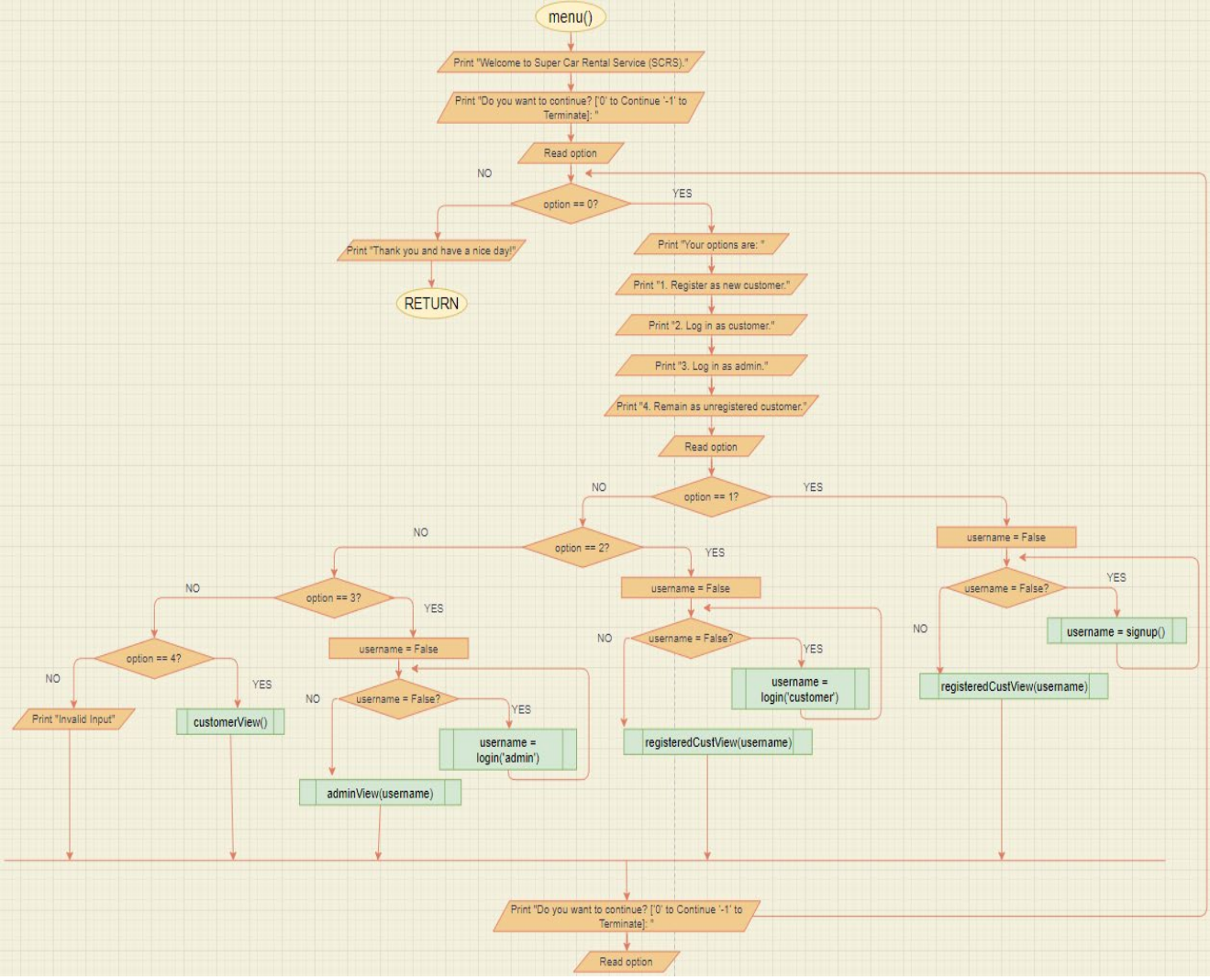
```
                LOOP line IN file
                    SET bookingDetails TO line.rstrip().split(", ")
                    IF (username == bookingDetails[0]) AND (carPlate == bookingDetails[1]) THEN
                        found = True
                        index = index + 1
                        Print index, username, "paid RM" bookingDetails[4], "for car plate",
                                bookingDetails[1], "in", bookingDetails[5]
                    ENDIF
                    NEXT line
                ENDLOOP
            ENDIF
        CLOSEFILE "bookings.txt"

        IF (found == False) THEN
            IF (carPlate == '') THEN
                Print "No payment record found for", username
            ELSE
                Print "No payment record found for", username, "with car plate", carPlate
            ENDIF
        ENDIF
    ENDTRY
    EXCEPT
        Print "Could not open 'bookings.txt' file."
    ENDEXCEPT

    RETURN
ENDFUNCTION

=============================================================================================

FUNCTION returnCar()
    Read carPlate

    TRY
        Read file From "bookings.txt"
            found = False
            index = 0
            lines = READLINES(file)

            FOR EACH line IN lines
                index = index + 1
                SET bookingDetails TO line.rstrip().split(", ")
                IF (carPlate == bookingDetails[1]) AND (bookingDetails[6] == 'False') THEN
                    found = True

                    Print username, "booked for car plate" bookingDetails[1], "from",
                               bookingDetails[2], "to", bookingDetails[3]
                    Print "Do you wish to return this car? ['0'= No, '1'= Yes]: "
                    Read answer

                    IF (answer == 1) THEN
                        lines[index] = username, ",", carPlate, ",", bookingDetails[2], ",", bookingDetails[3],
                                       ",", bookingDetails[4], ",", bookingDetails[5], "True\n"


                        TRY
                            OPENFILE "bookings.txt" FOR WRITE
                                Write lines Into "bookings.txt"
                                Print "Car with number plate", carPlate, "has been returned."
                            CLOSEFILE "bookings.txt"
                        ENDTRY
                        EXCEPT
                            Print "Error occured in 'bookings.txt' file."
                        ENDEXCEPT
                    ELSE
                        Print "Car with number plate", carPlate, "is not returned."
                    ENDIF
                    BREAK
                ENDIF
                NEXT line
            ENDFOR

            IF (found == False) THEN
                Print "No active booking record for car with number plate", carPlate"
            ENDIF
        CLOSEFILE "bookings.txt"
    ENDTRY
    EXCEPT
        Print "Could not open 'bookings.txt' file."
    ENDEXCEPT
    RETURN
ENDFUNCTION

=============================================================================================

FUNCTION customerView()
        call displayCars('available')

        RETURN

ENDFUNCTION

=======================================
```

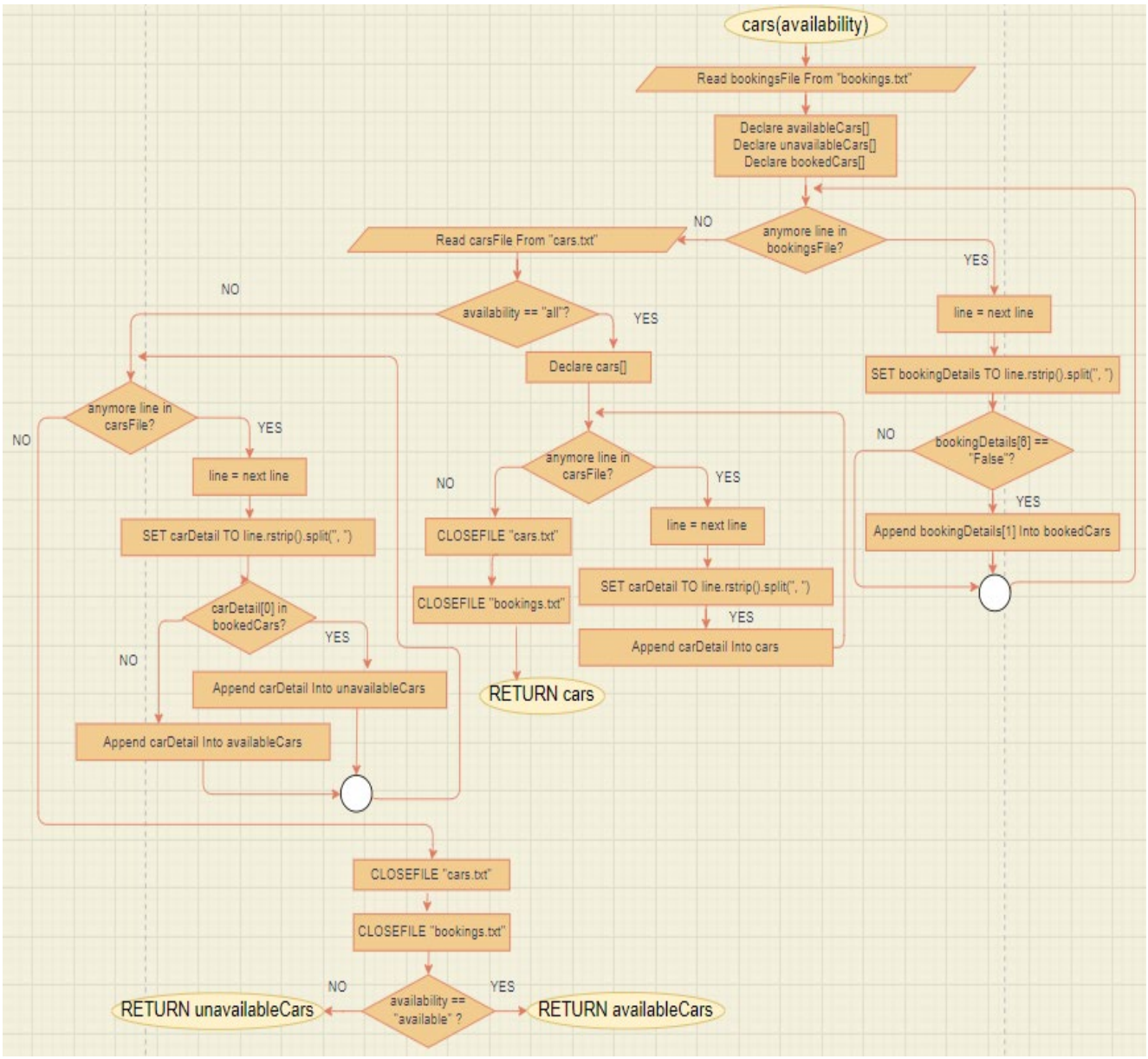
2.2 Flowchart

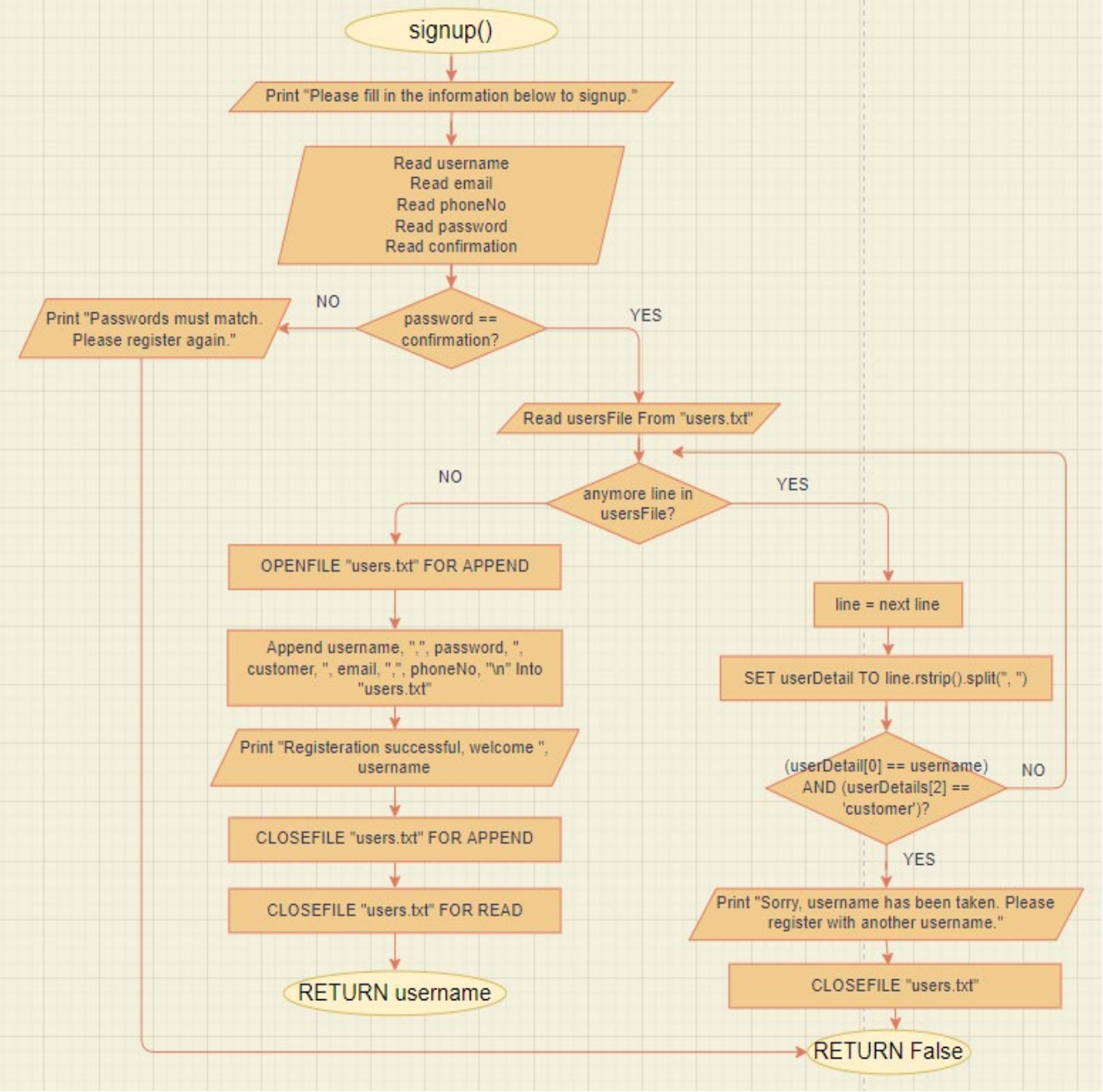
Main Program

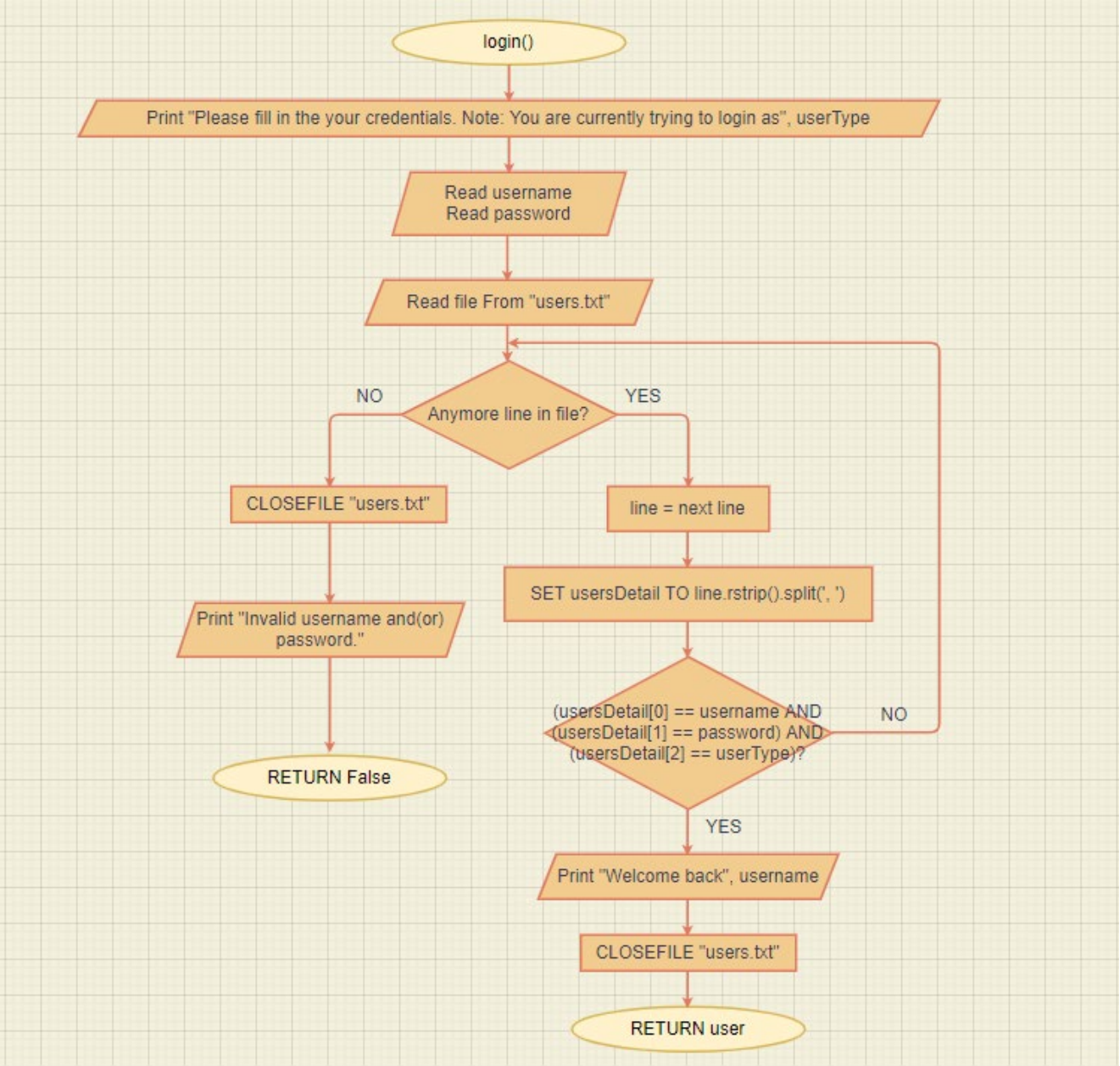


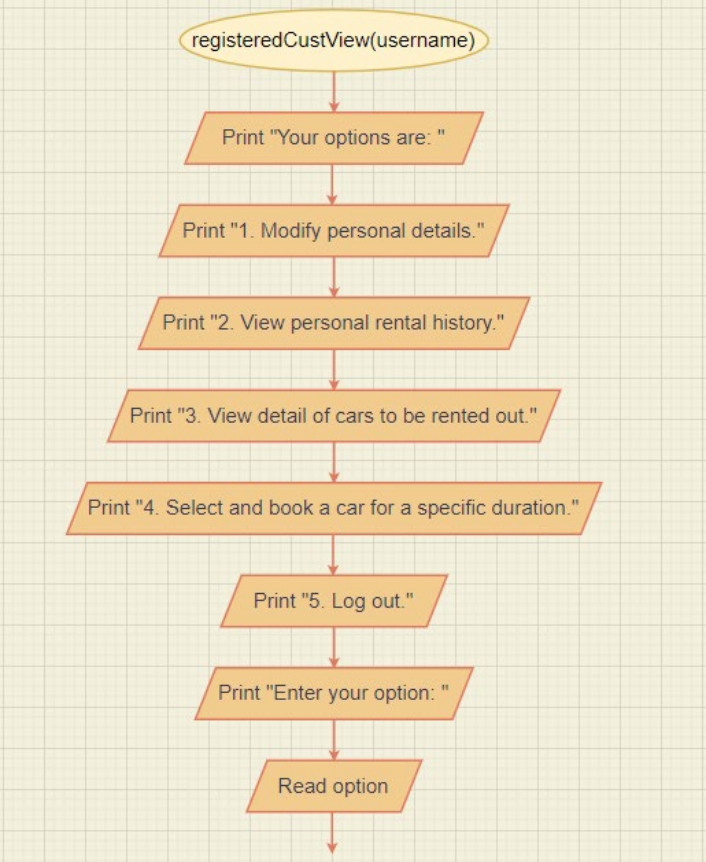
Menu Function

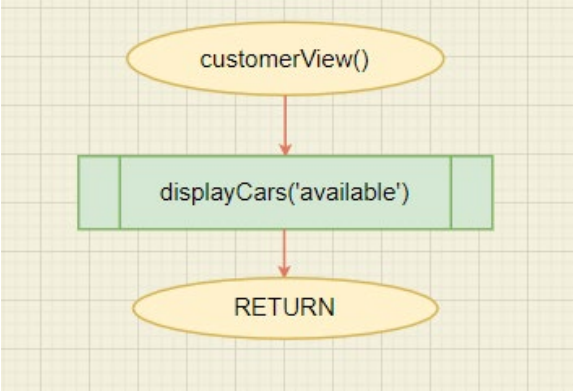


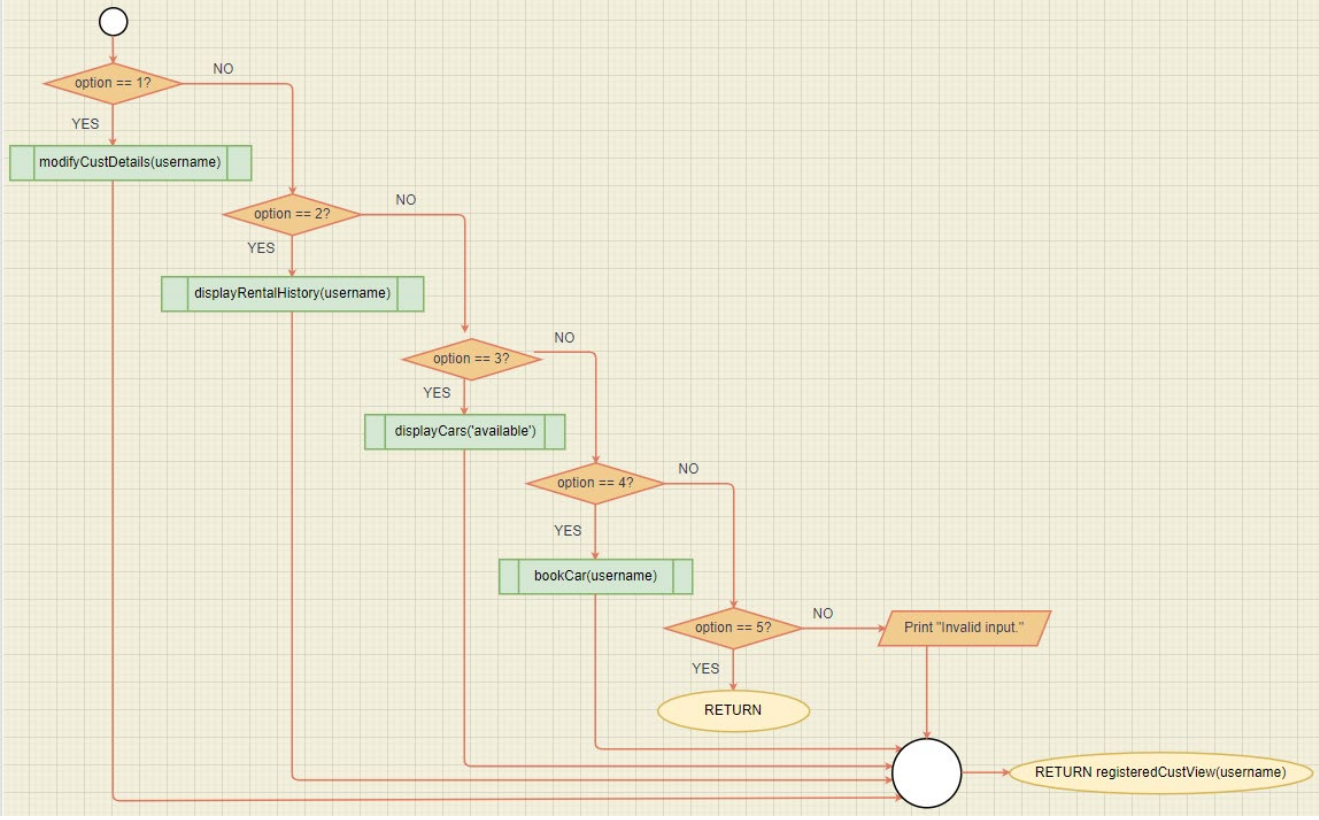

cars(availability)
Read bookingsFile From "bookings.txt"
Declare availableCars[]
Declare unavailableCars[]
Declare bookedCars[]
anymore line in bookingsFile?
NO
YES
Read carsFile From "cars.txt"
line = next line
SET bookingDetails TO line.rstrip().split(", ")
bookingDetails[6] == "False"?
NO
YES
Append bookingDetails[1] Into bookedCars
availability == "all"?
NO
YES
Declare cars[]
anymore line in carsFile?
NO
YES
CLOSEFILE "cars.txt"
CLOSEFILE "bookings.txt"
RETURN cars
line = next line
SET carDetail TO line.rstrip().split(", ")
YES
Append carDetail Into cars
anymore line in carsFile?
NO
YES
line = next line
SET carDetail TO line.rstrip().split(", ")
carDetail[0] in bookedCars?
NO
YES
Append carDetail Into availableCars
Append carDetail Into unavailableCars
CLOSEFILE "cars.txt"
CLOSEFILE "bookings.txt"
availability == "available" ?
NO
YES
RETURN unavailableCars
RETURN availableCars

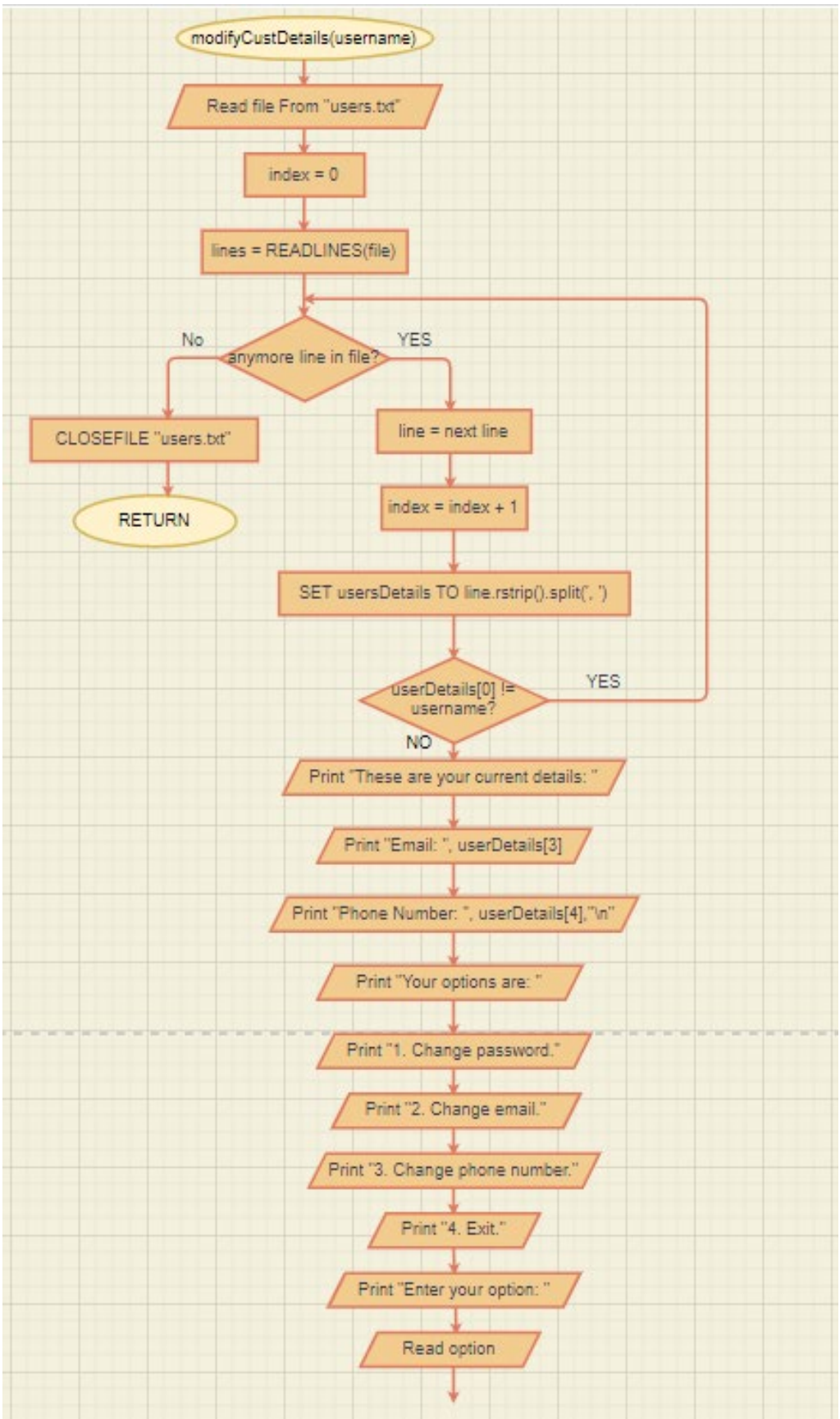

signup()
Print "Please fill in the information below to signup."
Read username
Read email
Read phoneNo
Read password
Read confirmation
password == confirmation?
NO
Print "Passwords must match. Please register again."
YES
Read usersFile From "users.txt"
anymore line in usersFile?
NO
YES
OPENFILE "users.txt" FOR APPEND
Append username, ",", password, ", customer, ", email, ",", phoneNo, "\n" Into "users.txt"
Print "Registeration successful, welcome ", username
CLOSEFILE "users.txt" FOR APPEND
CLOSEFILE "users.txt" FOR READ
RETURN username
line = next line
SET userDetail TO line.rstrip().split(", ")
(userDetail[0] == username) AND (userDetails[2] == 'customer')?
NO
YES
Print "Sorry, username has been taken. Please register with another username."
CLOSEFILE "users.txt"
RETURN False

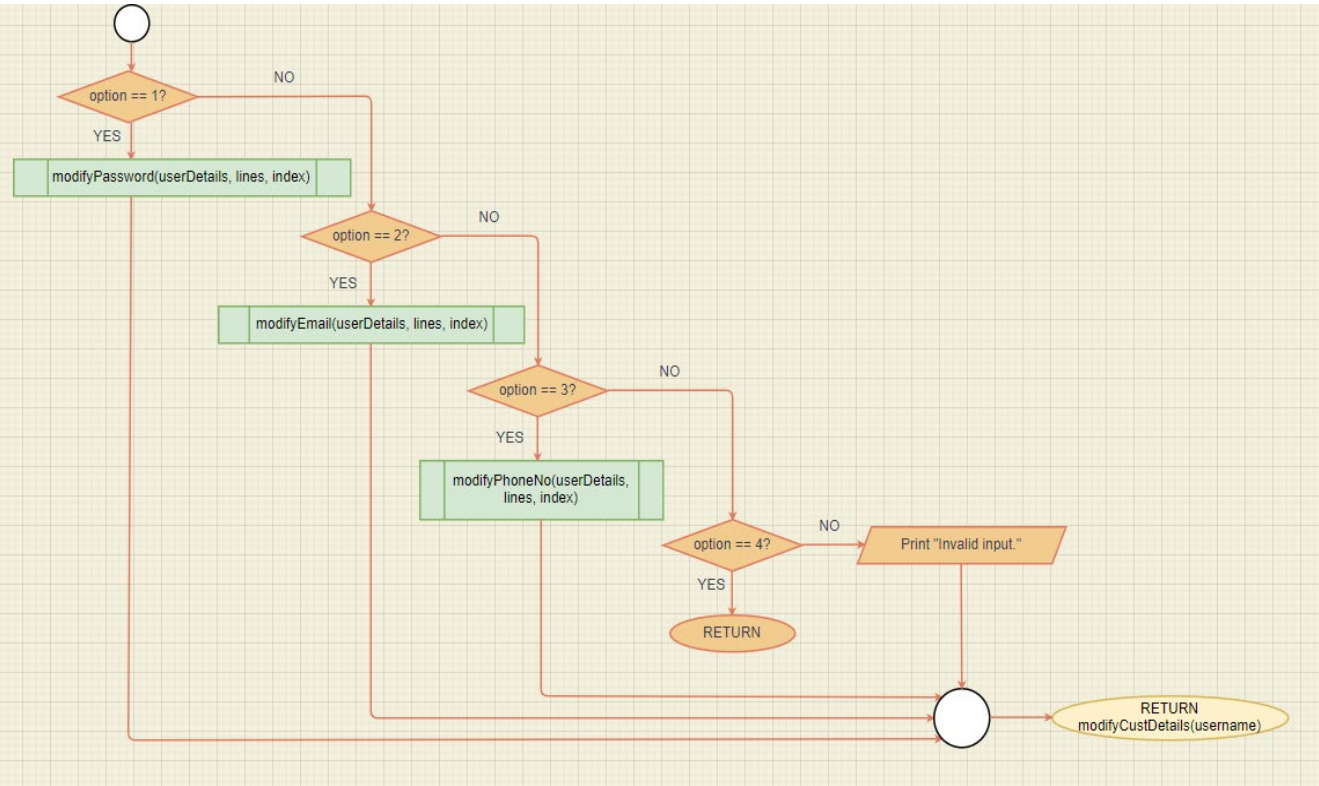

login()
Print "Please fill in the your credentials. Note: You are currently trying to login as", userType
Read username
Read password
Read file From "users.txt"
NO
Anymore line in file?
YES
CLOSEFILE "users.txt"
line = next line
Print "Invalid username and(or) password."
SET usersDetail TO line.rstrip().split(', ')
(usersDetail[0] == username AND (usersDetail[1] == password) AND (usersDetail[2] == userType)?
NO
RETURN False
YES
Print "Welcome back", username
CLOSEFILE "users.txt"
RETURN user

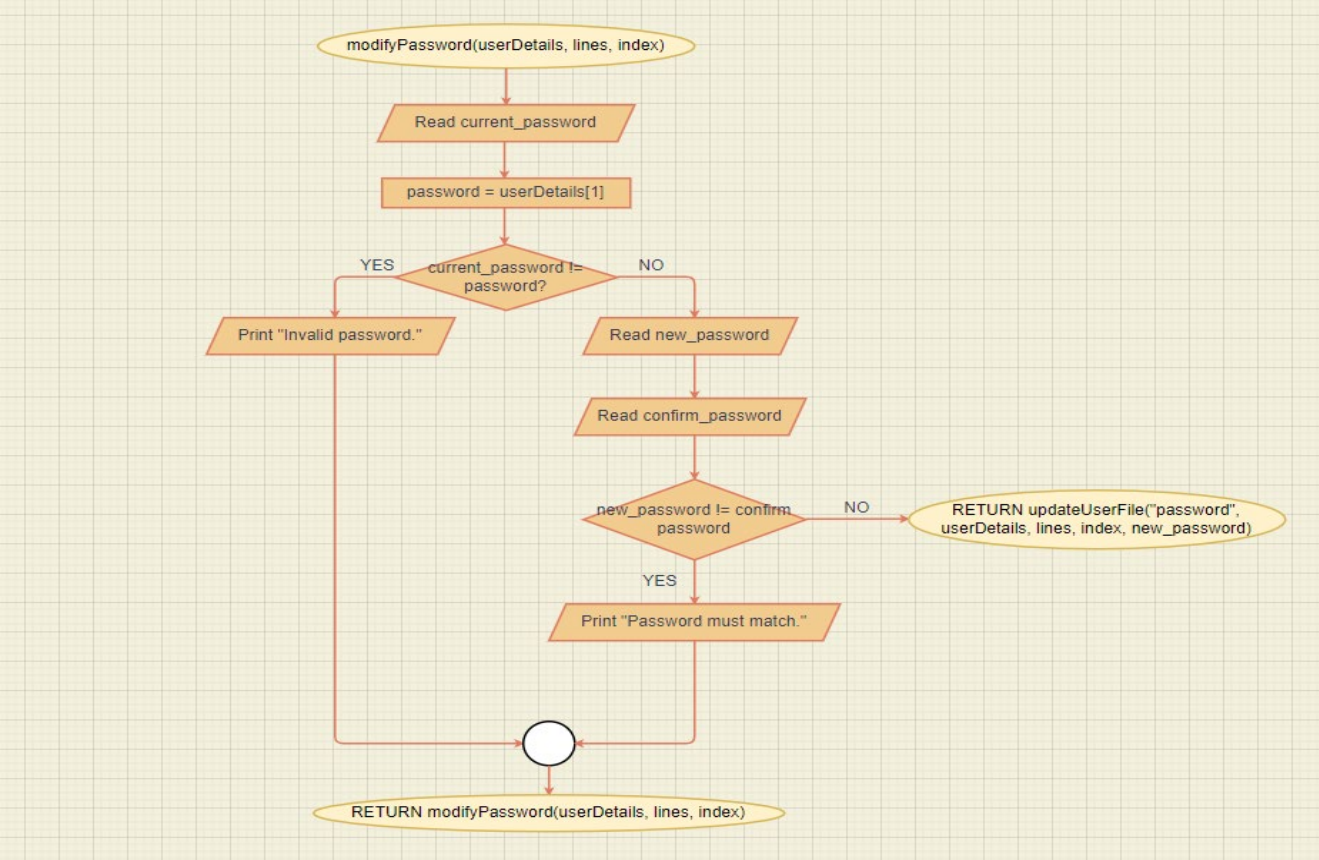

registeredCustView(username)
Print "Your options are: "
Print "1. Modify personal details."
Print "2. View personal rental history."
Print "3. View detail of cars to be rented out."
Print "4. Select and book a car for a specific duration."
Print "5. Log out."
Print "Enter your option: "
Read option

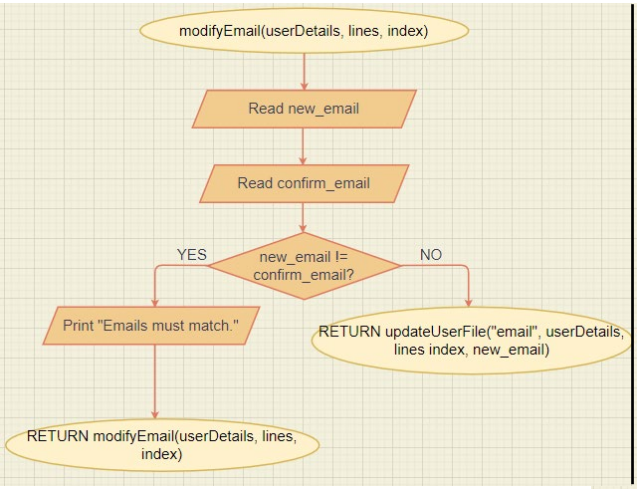

customerView()
displayCars('available')
RETURN

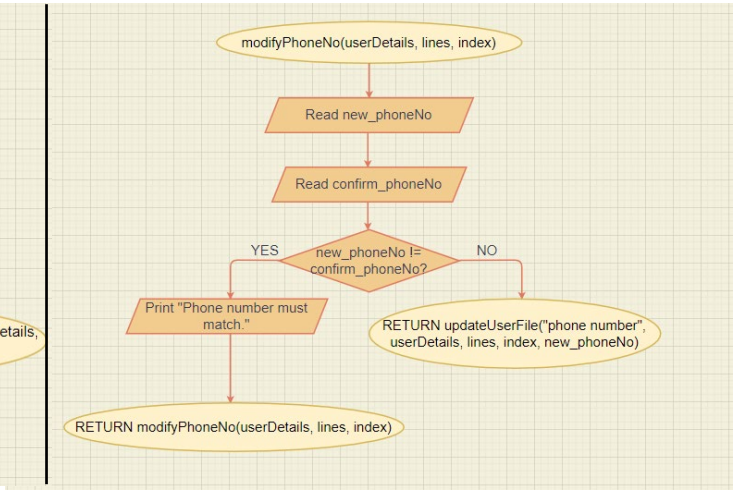

option == 1?
NO
YES
modifyCustDetails(username)
option == 2?
NO
YES
displayRentalHistory(username)
option == 3?
NO
YES
displayCars('available')
option == 4?
NO
YES
bookCar(username)
option == 5?
NO
Print "Invalid input."
YES
RETURN
RETURN registeredCustView(username)

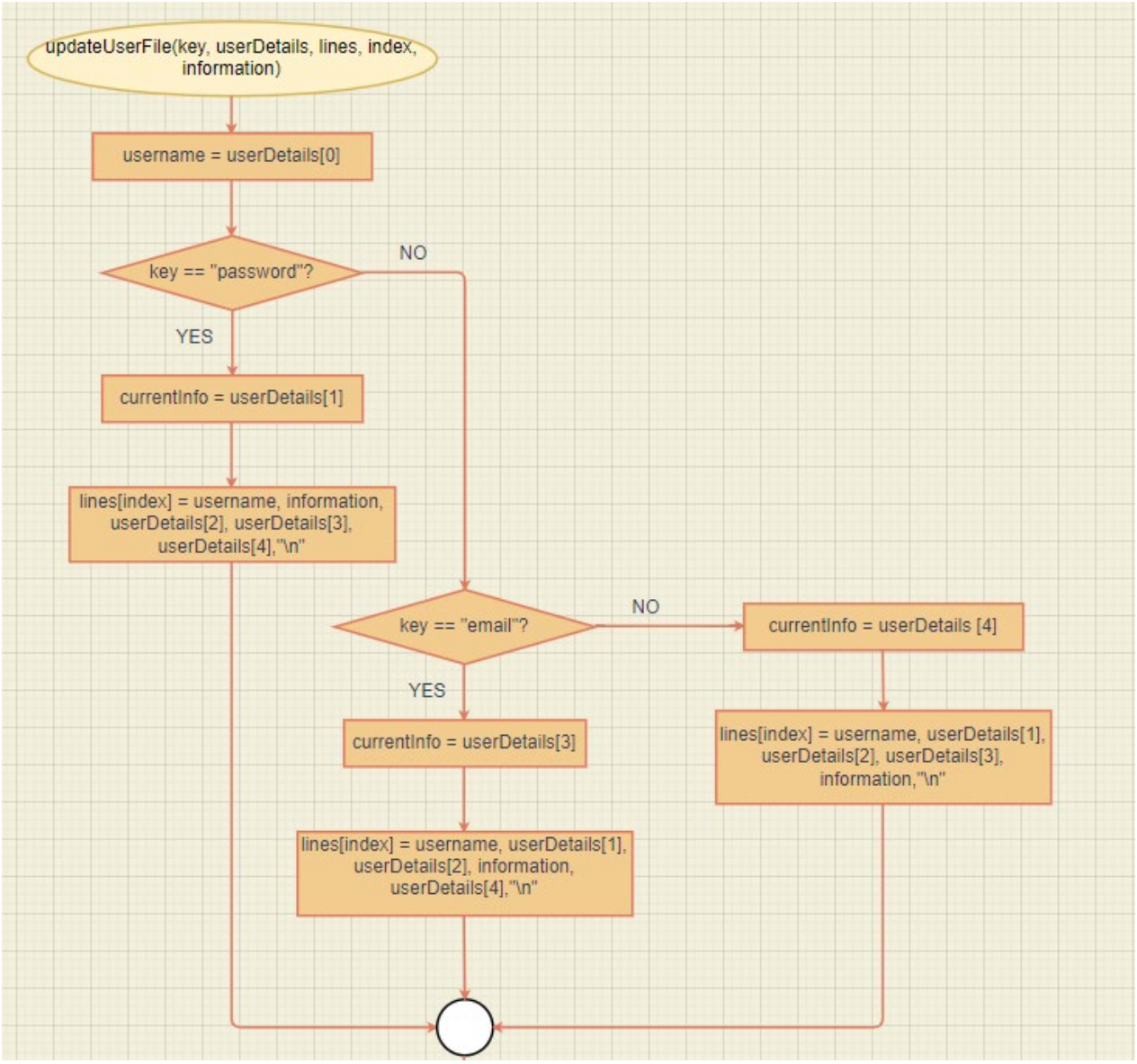

modifyCustDetails(username)
Read file From "users.txt"
index = 0
lines = READLINES(file)
anymore line in file?
No
YES
CLOSEFILE "users.txt"
RETURN
line = next line
index = index + 1
SET usersDetails TO line.rstrip().split(', ')
userDetails[0] != username?
YES
NO
Print "These are your current details: "
Print "Email: ", userDetails[3]
Print "Phone Number: ", userDetails[4],"\n"
Print "Your options are: "
Print "1. Change password."
Print "2. Change email."
Print "3. Change phone number."
Print "4. Exit."
Print "Enter your option: "
Read option

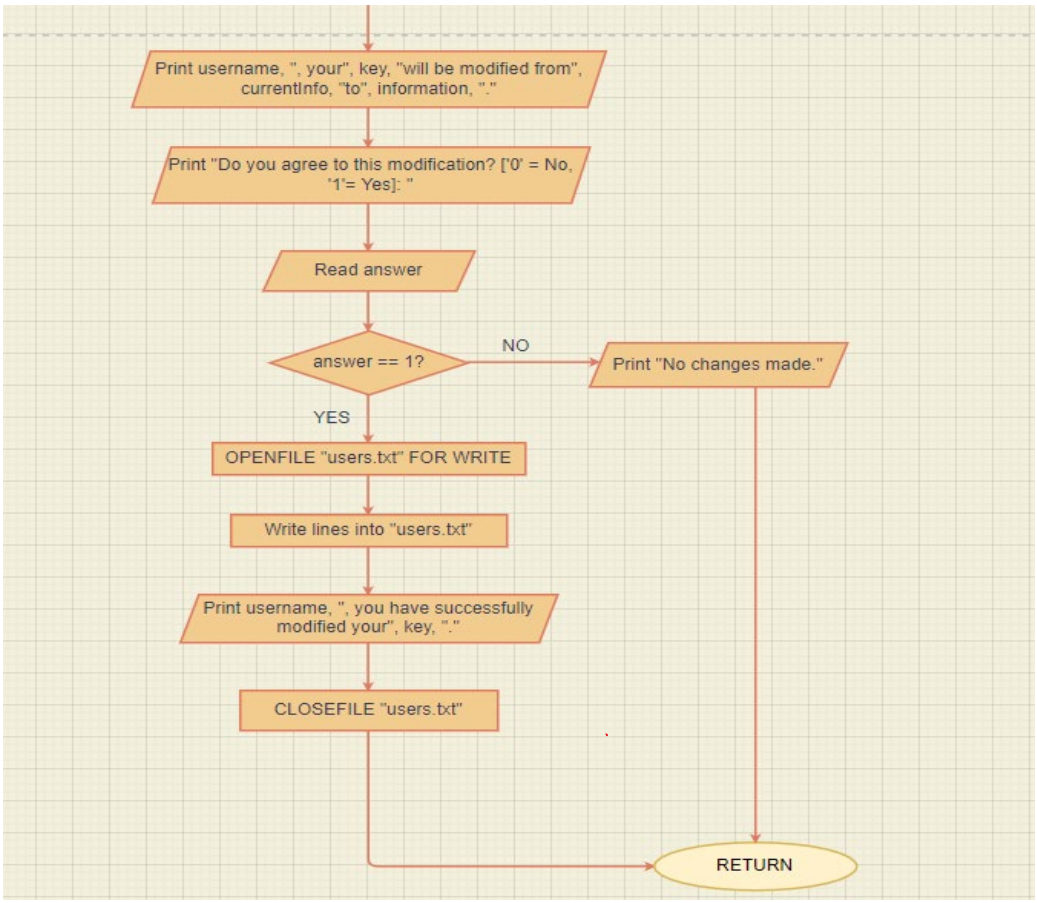

option == 1?
NO
YES
modifyPassword(userDetails, lines, index)
option == 2?
NO
YES
modifyEmail(userDetails, lines, index)
option == 3?
NO
YES
modifyPhoneNo(userDetails, lines, index)
option == 4?
NO
Print "Invalid input."
YES
RETURN
RETURN modifyCustDetails(username)

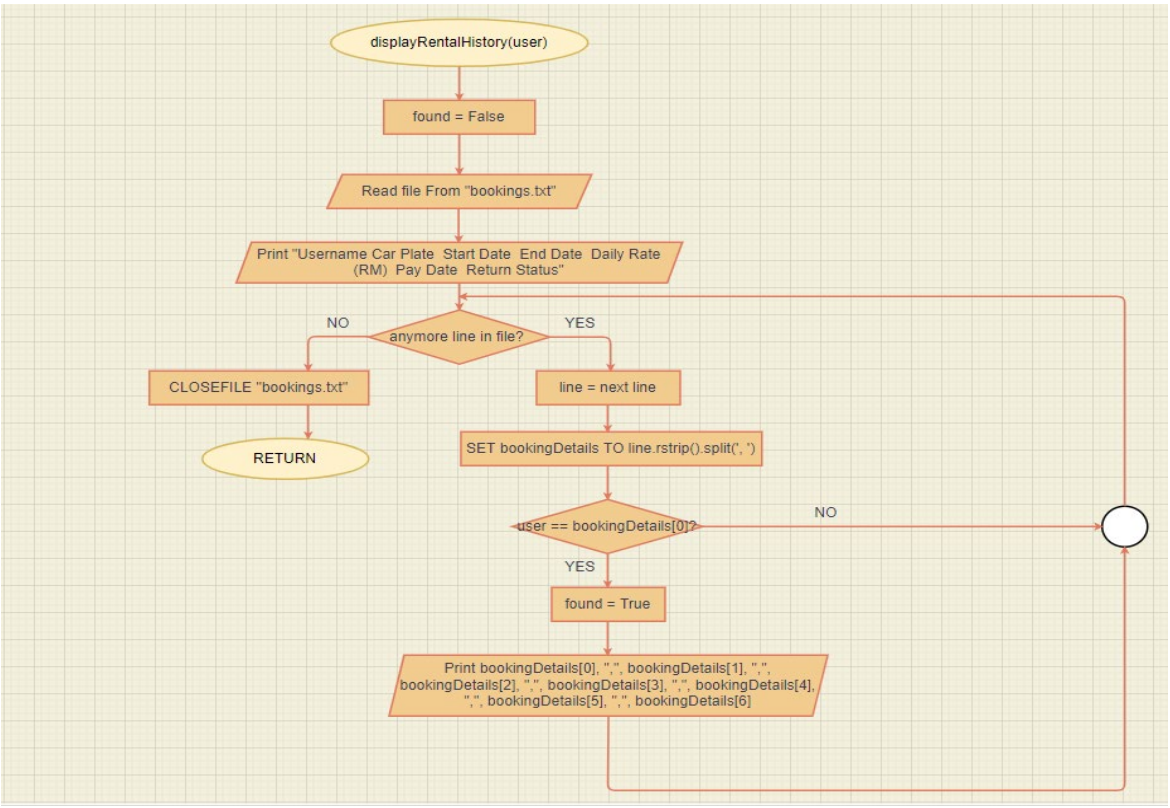

modifyPassword(userDetails, lines, index)
Read current_password
password = userDetails[1]
current_password != password?
YES
NO
Print "Invalid password."
Read new_password
Read confirm_password
new_password != confirm password
NO
RETURN updateUserFile("password", userDetails, lines, index, new_password)
YES
Print "Password must match."
RETURN modifyPassword(userDetails, lines, index)

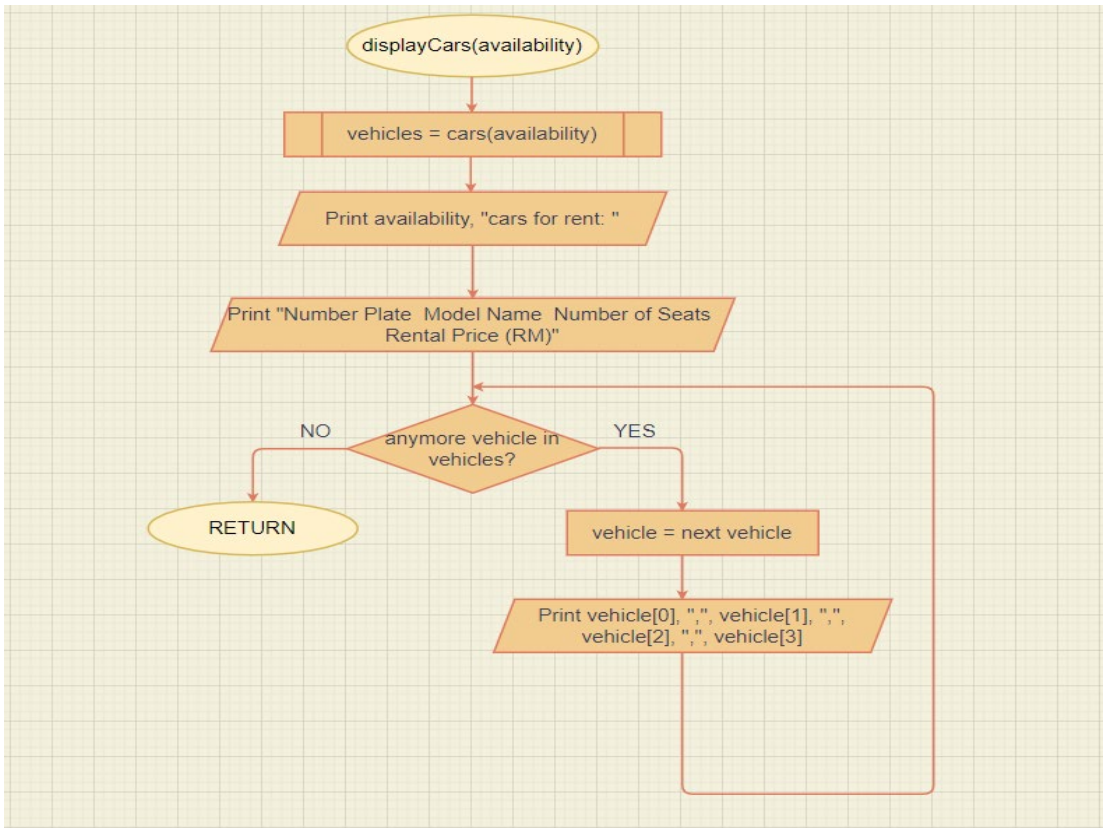

modifyEmail(userDetails, lines, index)
Read new_email
Read confirm_email
YES
new_email != confirm_email?
NO
Print "Emails must match."
RETURN updateUserFile("email", userDetails, lines index, new_email)
RETURN modifyEmail(userDetails, lines, index)

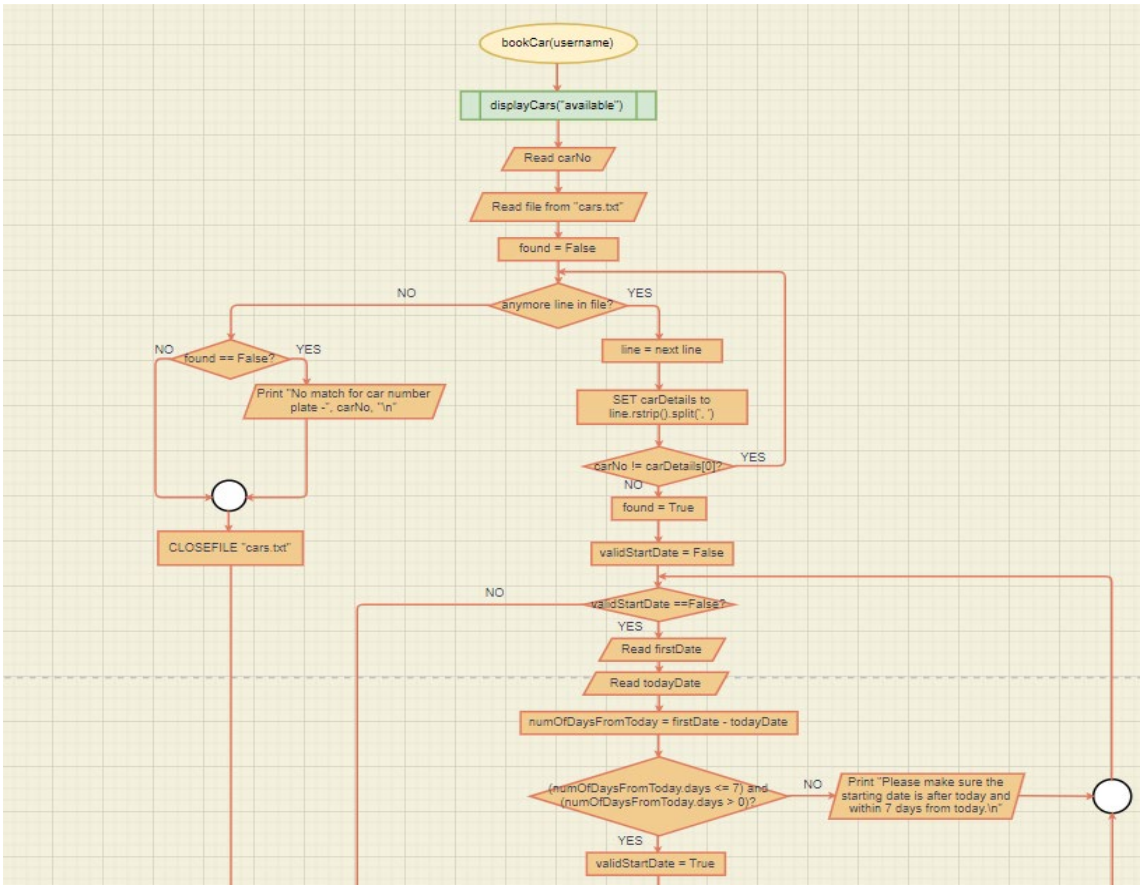

modifyPhoneNo(userDetails, lines, index)
Read new_phoneNo
Read confirm_phoneNo
YES
new_phoneNo != confirm_phoneNo?
NO
Print "Phone number must match."
RETURN updateUserFile("phone number", userDetails, lines, index, new_phoneNo)
RETURN modifyPhoneNo(userDetails, lines, index)

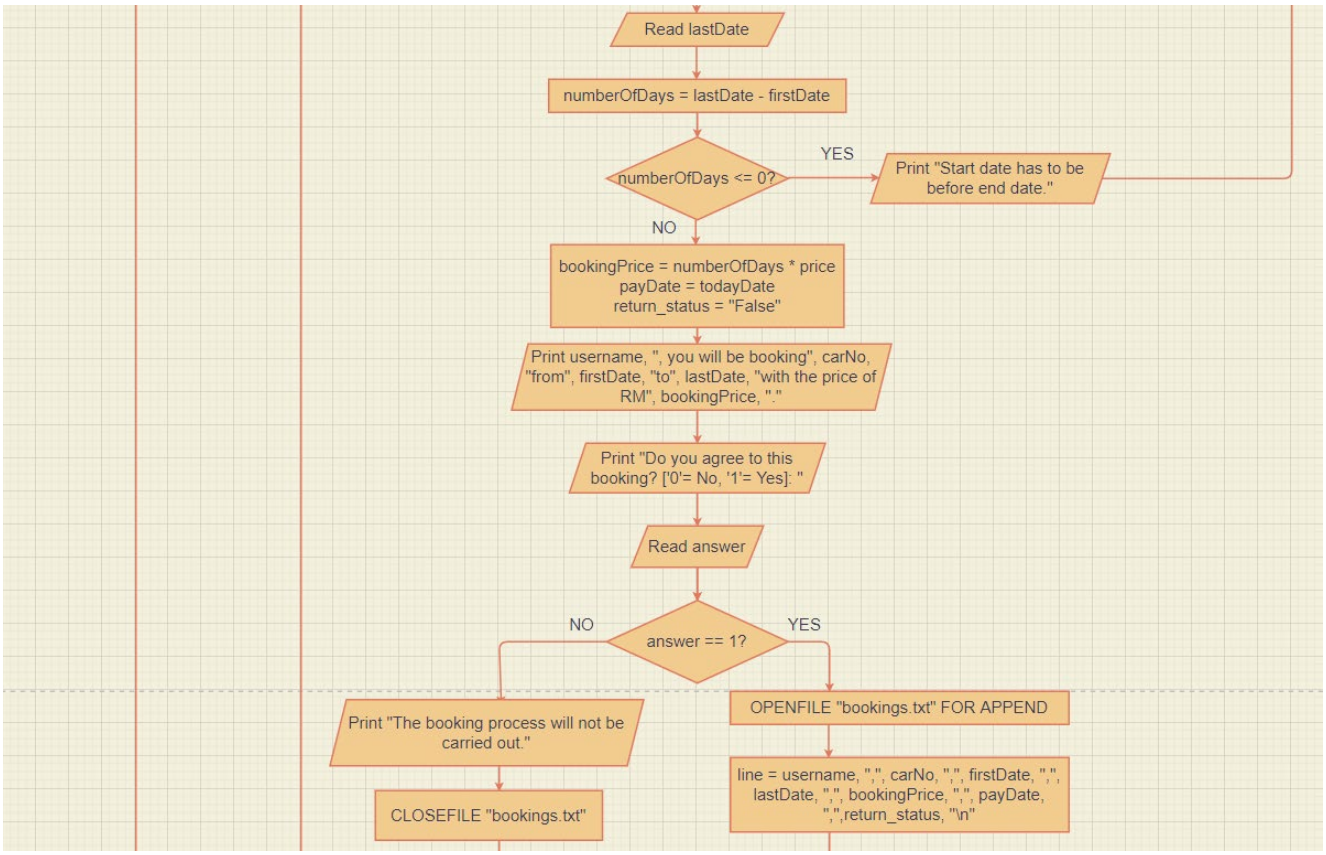

updateUserFile(key, userDetails, lines, index, information)
username = userDetails[0]
key == "password"?
NO
YES
currentInfo = userDetails[1]
lines[index] = username, information, userDetails[2], userDetails[3], userDetails[4],"\n"
key == "email"?
NO
currentInfo = userDetails [4]
YES
currentInfo = userDetails[3]
lines[index] = username, userDetails[1], userDetails[2], userDetails[3], information,"\n"
lines[index] = username, userDetails[1], userDetails[2], information, userDetails[4],"\n"

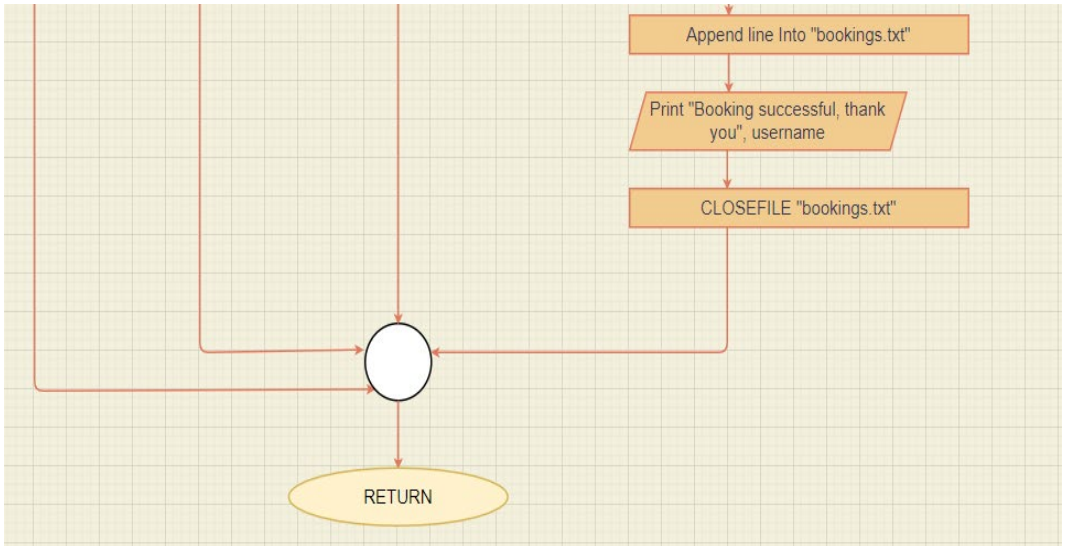

Print username, ", your", key, "will be modified from", currentInfo, "to", information, "."
Print "Do you agree to this modification? ['0' = No, '1'= Yes]: "
Read answer
answer == 1?
NO
Print "No changes made."
YES
OPENFILE "users.txt" FOR WRITE
Write lines into "users.txt"
Print username, ", you have successfully modified your", key, "."
CLOSEFILE "users.txt"
RETURN

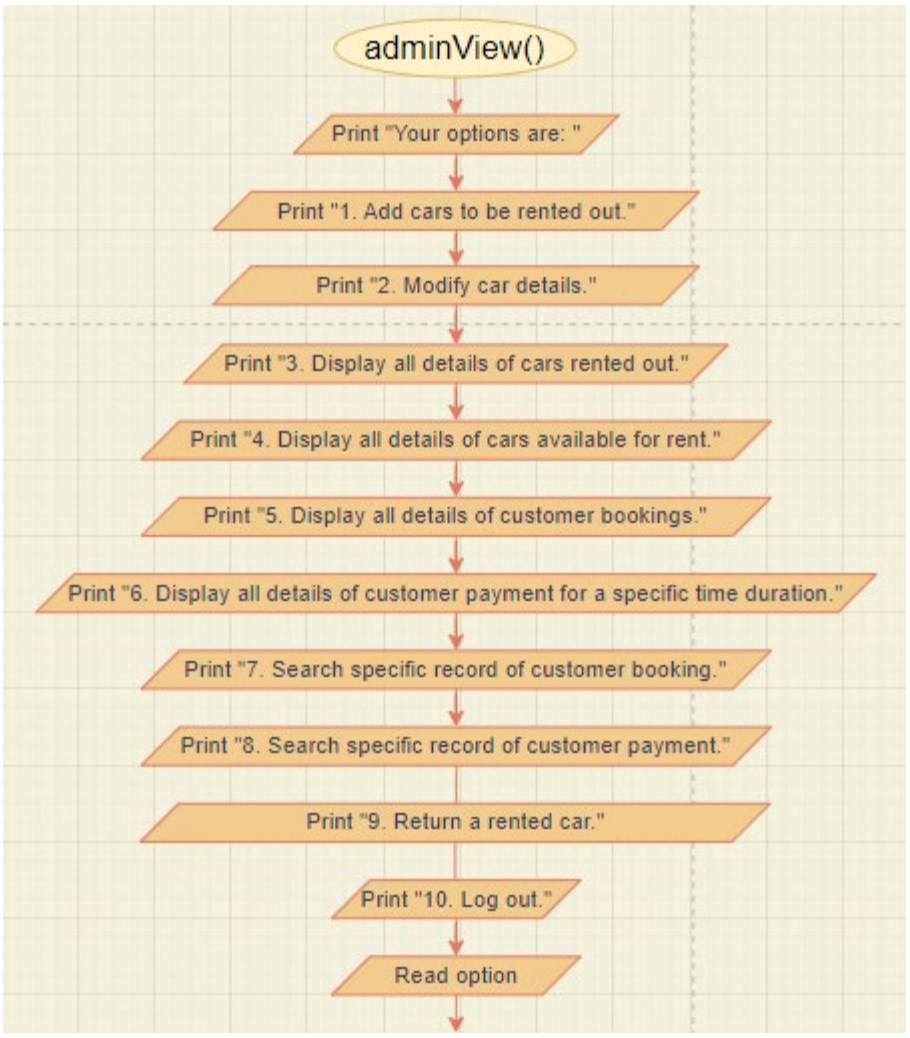

displayRentalHistory(user)
found = False
Read file From "bookings.txt"
Print "Username Car Plate Start Date End Date Daily Rate (RM) Pay Date Return Status"
NO
anymore line in file?
YES
CLOSEFILE "bookings.txt"
line = next line
RETURN
SET bookingDetails TO line.rstrip().split(', ')
user == bookingDetails[0]?
NO
YES
found = True
Print bookingDetails[0], ",", bookingDetails[1], ",", bookingDetails[2], ",", bookingDetails[3], ",", bookingDetails[4], ",", bookingDetails[5], ",", bookingDetails[6]

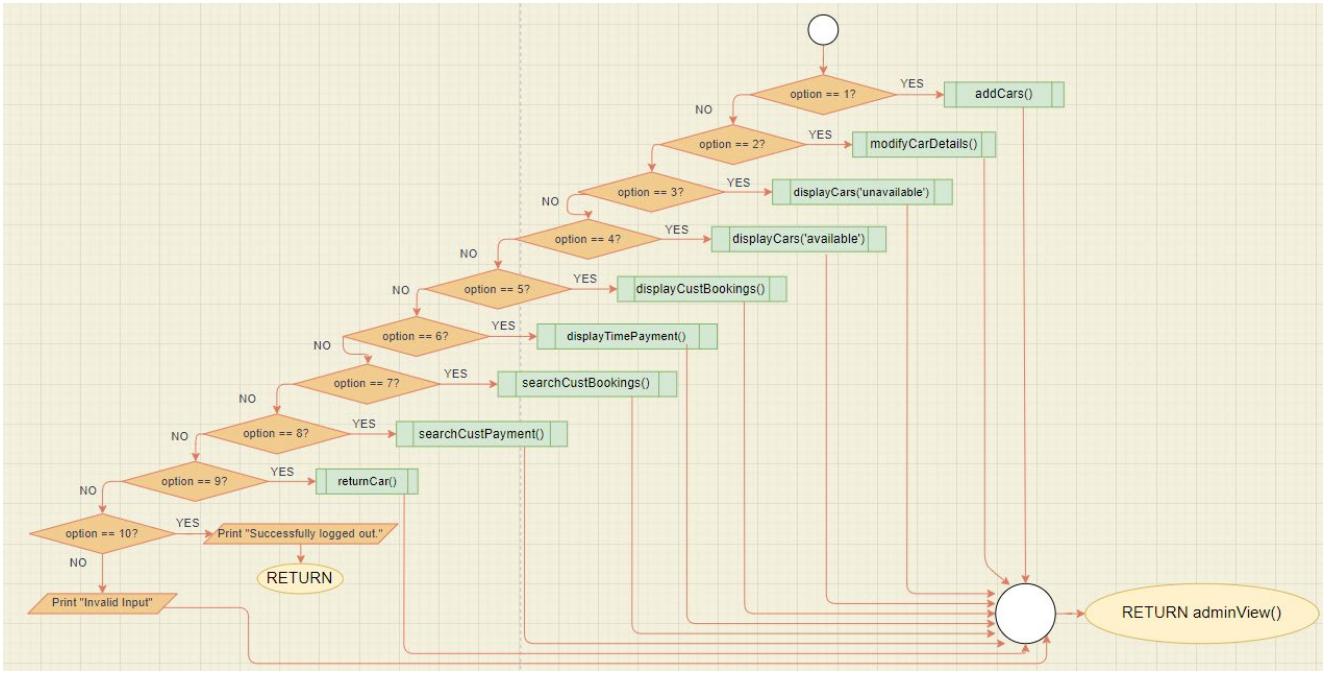

displayCars(availability)
vehicles = cars(availability)
Print availability, "cars for rent: "
Print "Number Plate Model Name Number of Seats Rental Price (RM)"
anymore vehicle in vehicles?
NO
YES
RETURN
vehicle = next vehicle
Print vehicle[0], ",", vehicle[1], ",", vehicle[2], ",", vehicle[3]

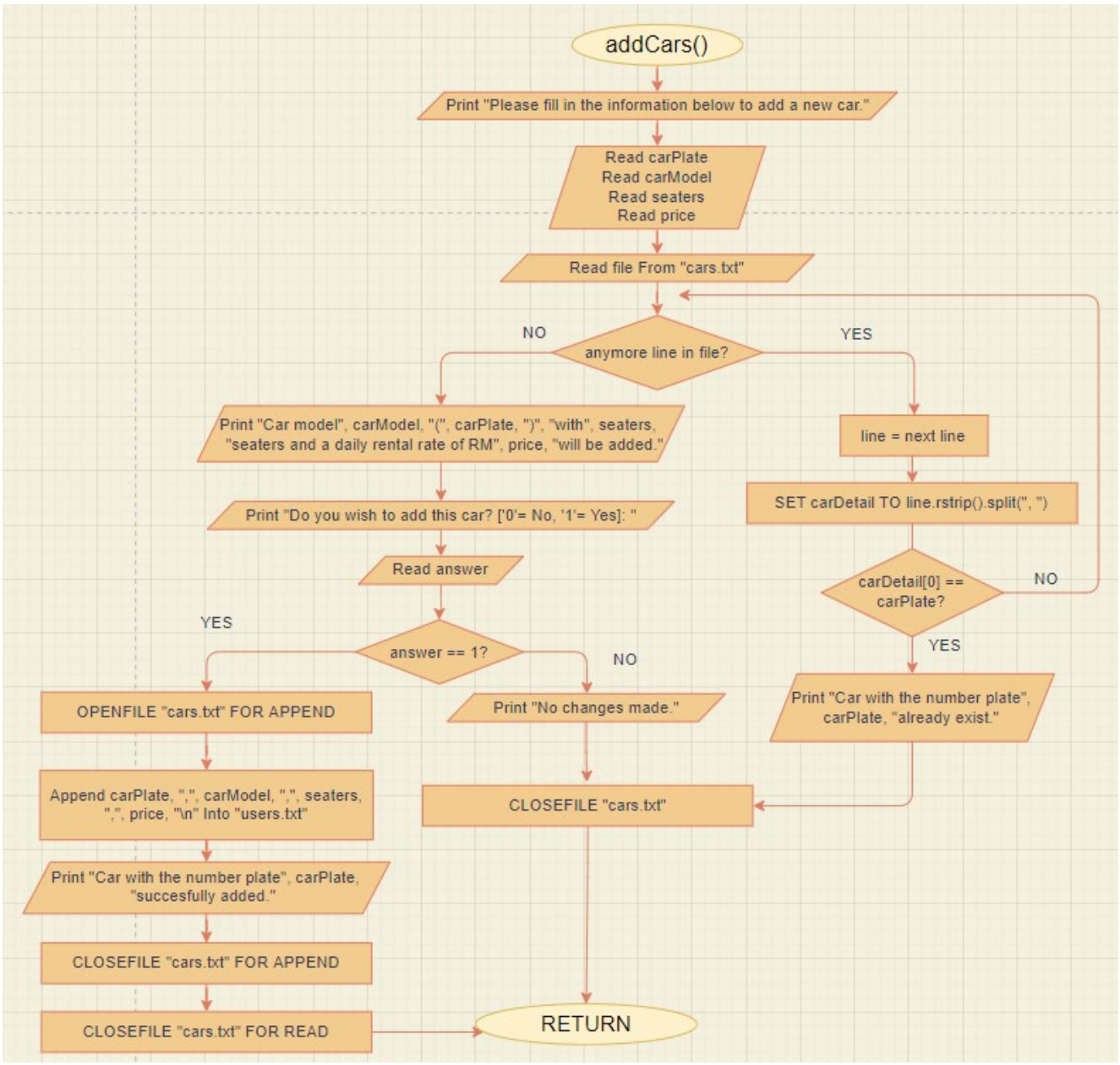

bookCar(username)
displayCars("available")
Read carNo
Read file from "cars.txt"
found = False
anymore line in file?
NO
YES
found == False?
NO
YES
Print "No match for car number plate -", carNo, "\n"
line = next line
SET carDetails to line.rstrip().split(',')
carNo != carDetails[0]?
YES
NO
found = True
CLOSEFILE "cars.txt"
validStartDate = False
validStartDate ==False?
NO
YES
Read firstDate
Read todayDate
numOfDaysFromToday = firstDate - todayDate
(numOfDaysFromToday.days <= 7) and (numOfDaysFromToday.days > 0)?
NO
Print "Please make sure the starting date is after today and within 7 days from today.\n"
YES
validStartDate = True

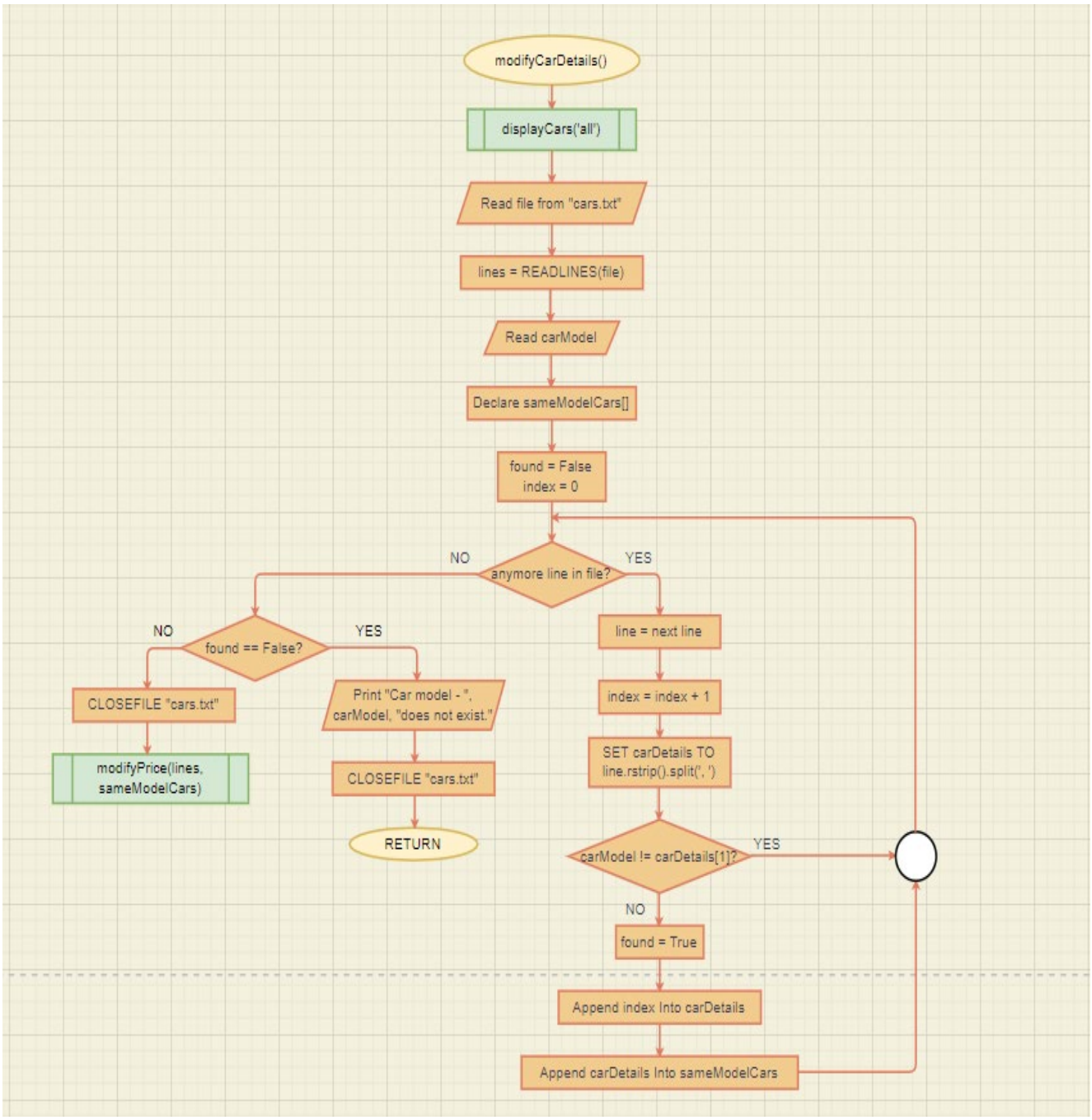

Read lastDate
numberOfDays = lastDate - firstDate
numberOfDays <= 0?
YES
Print "Start date has to be before end date."
NO
bookingPrice = numberOfDays * price
payDate = todayDate
return_status = "False"
Print username, ", you will be booking", carNo, "from", firstDate, "to", lastDate, "with the price of RM", bookingPrice, "."
Print "Do you agree to this booking? ['0'= No, '1'= Yes]: "
Read answer
NO
answer == 1?
YES
Print "The booking process will not be carried out."
CLOSEFILE "bookings.txt"
OPENFILE "bookings.txt" FOR APPEND
line = username, ",", carNo, ",", firstDate, ",", lastDate, ",", bookingPrice, ",", payDate, ",",return_status, "\n"

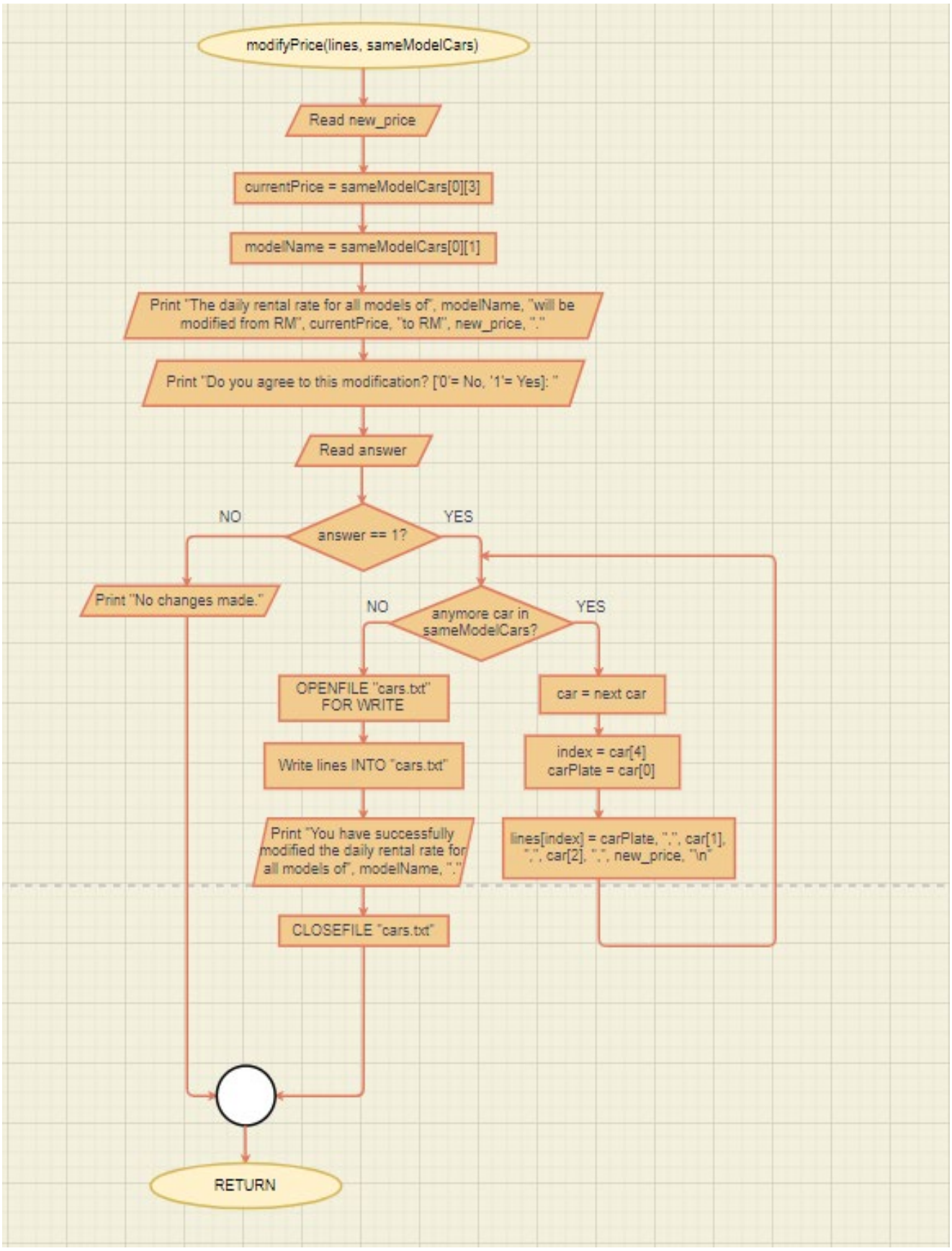

Append line Into "bookings.txt"
Print "Booking successful, thank you", username
CLOSEFILE "bookings.txt"
RETURN

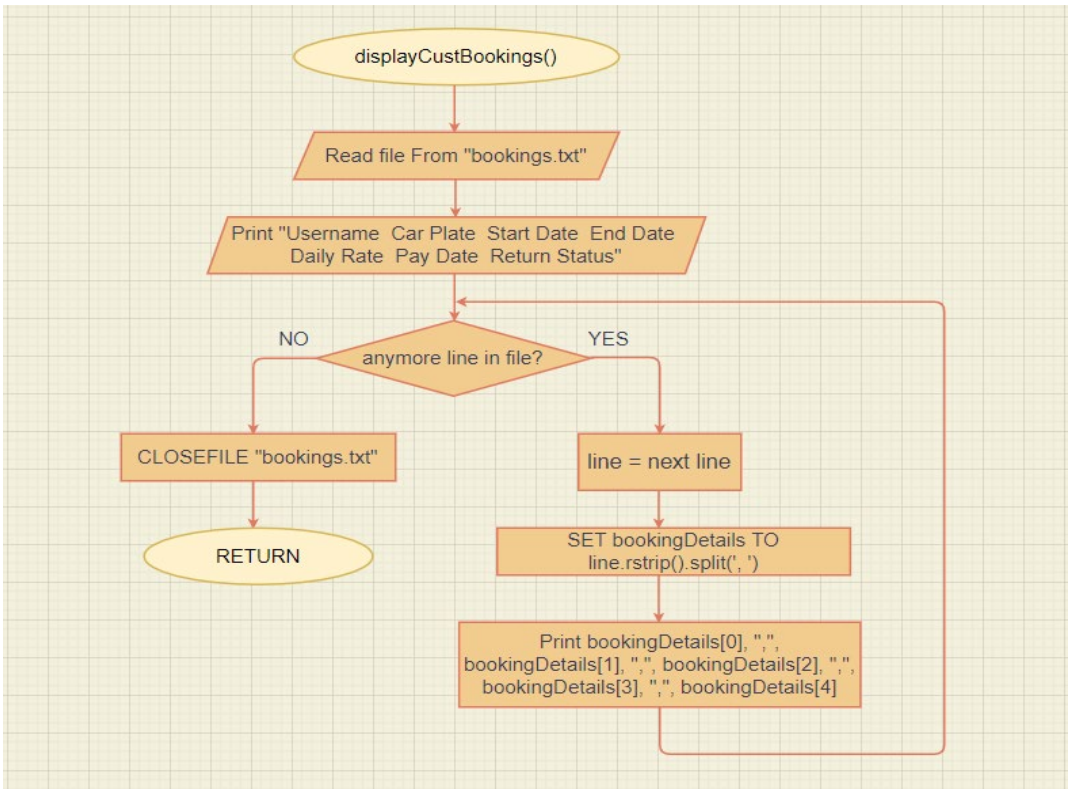

adminView()
Print "Your options are: "
Print "1. Add cars to be rented out."
Print "2. Modify car details."
Print "3. Display all details of cars rented out."
Print "4. Display all details of cars available for rent."
Print "5. Display all details of customer bookings."
Print "6. Display all details of customer payment for a specific time duration."
Print "7. Search specific record of customer booking."
Print "8. Search specific record of customer payment."
Print "9. Return a rented car."
Print "10. Log out."
Read option

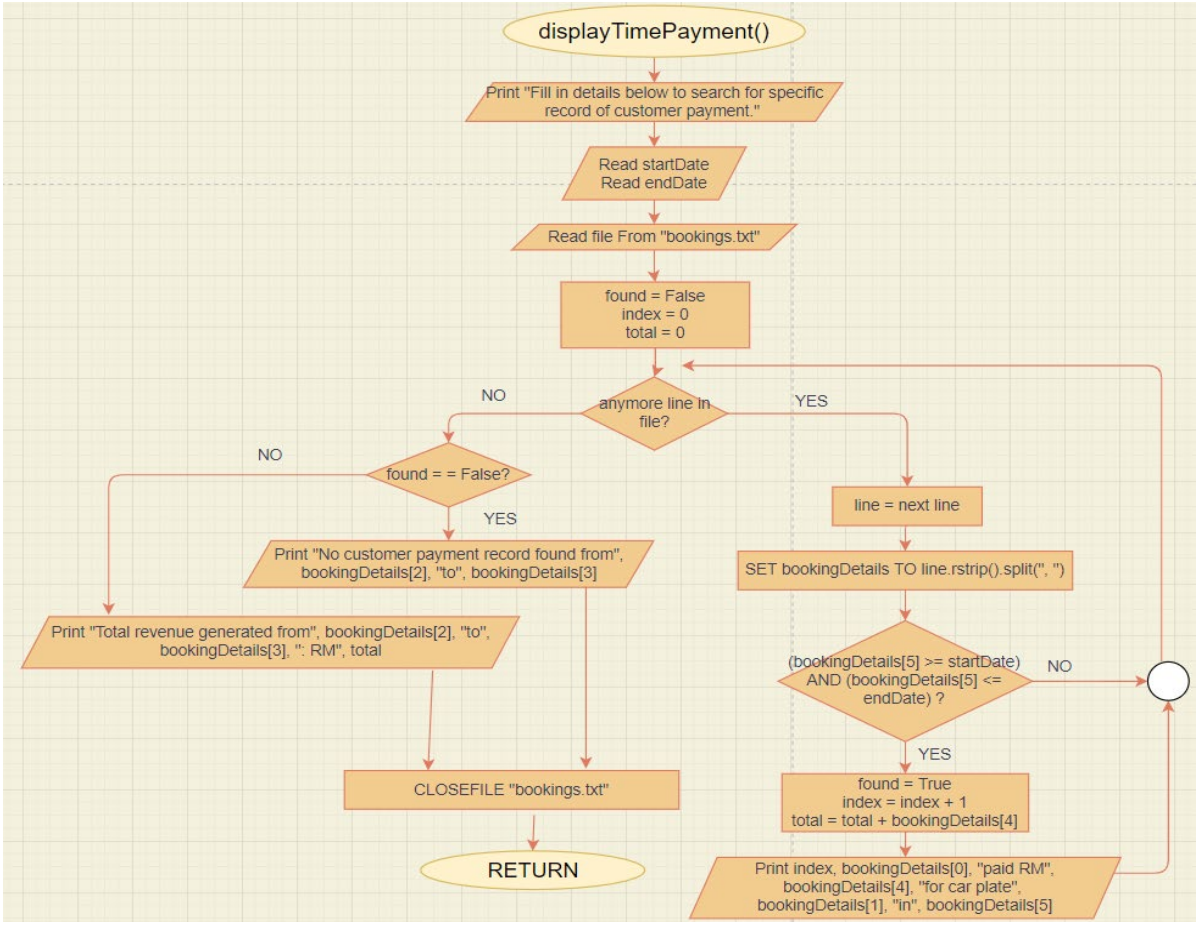

option == 1?
YES
addCars()
NO
option == 2?
YES
modifyCarDetails()
option == 3?
YES
displayCars('unavailable')
NO
option == 4?
YES
displayCars('available')
NO
option == 5?
YES
displayCustBookings()
NO
option == 6?
YES
displayTimePayment()
NO
option == 7?
YES
searchCustBookings()
NO
option == 8?
YES
searchCustPayment()
NO
option == 9?
YES
returnCar()
NO
option == 10?
YES
Print "Successfully logged out."
NO
RETURN
Print "Invalid Input"
RETURN adminView()

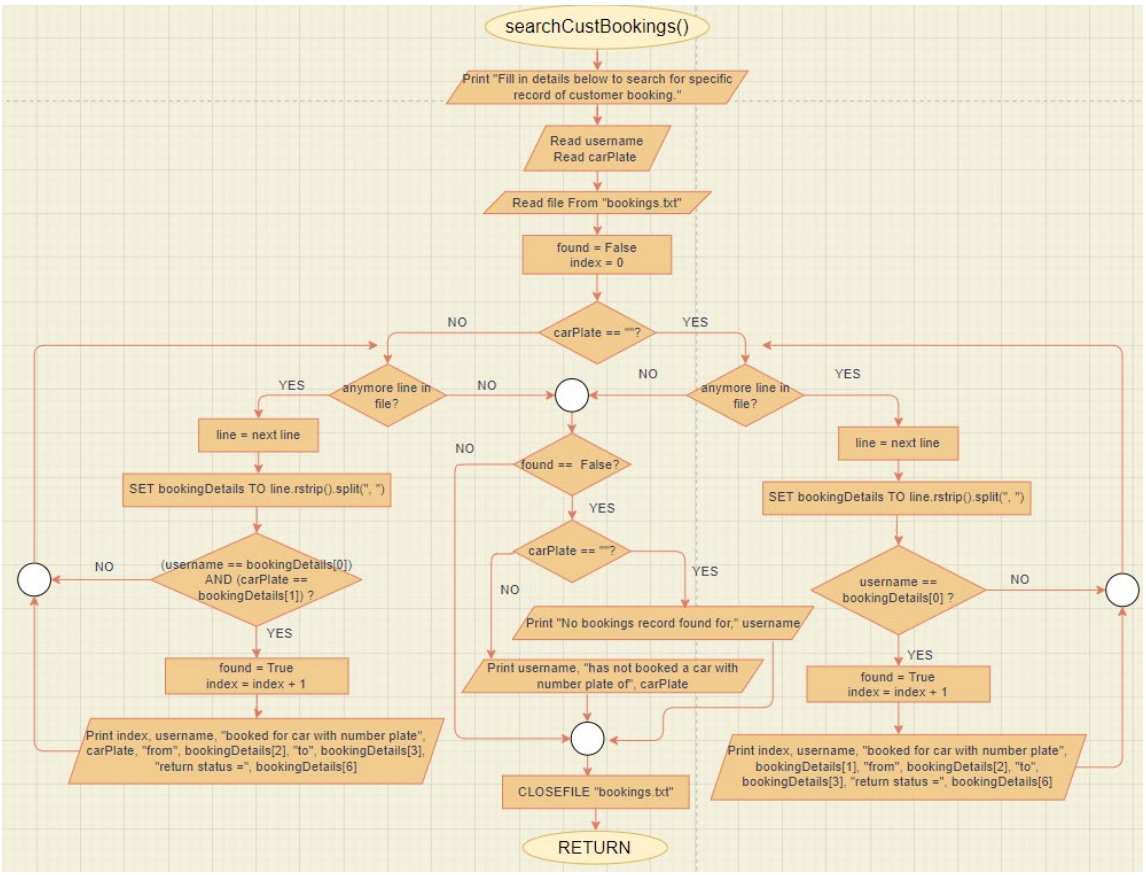

addCars()
Print "Please fill in the information below to add a new car."
Read carPlate
Read carModel
Read seaters
Read price
Read file From "cars.txt"
NO
anymore line in file?
YES
Print "Car model", carModel, "(", carPlate, ")", "with", seaters, "seaters and a daily rental rate of RM", price, "will be added."
Print "Do you wish to add this car? ['0'= No, '1'= Yes]: "
Read answer
YES
answer == 1?
NO
OPENFILE "cars.txt" FOR APPEND
Append carPlate, ",", carModel, ",", seaters, ",", price, "\n" Into "users.txt"
Print "Car with the number plate", carPlate, "succesfully added."
CLOSEFILE "cars.txt" FOR APPEND
CLOSEFILE "cars.txt" FOR READ
Print "No changes made."
CLOSEFILE "cars.txt"
RETURN
line = next line
SET carDetail TO line.rstrip().split(", ")
carDetail[0] == carPlate?
NO
YES
Print "Car with the number plate", carPlate, "already exist."

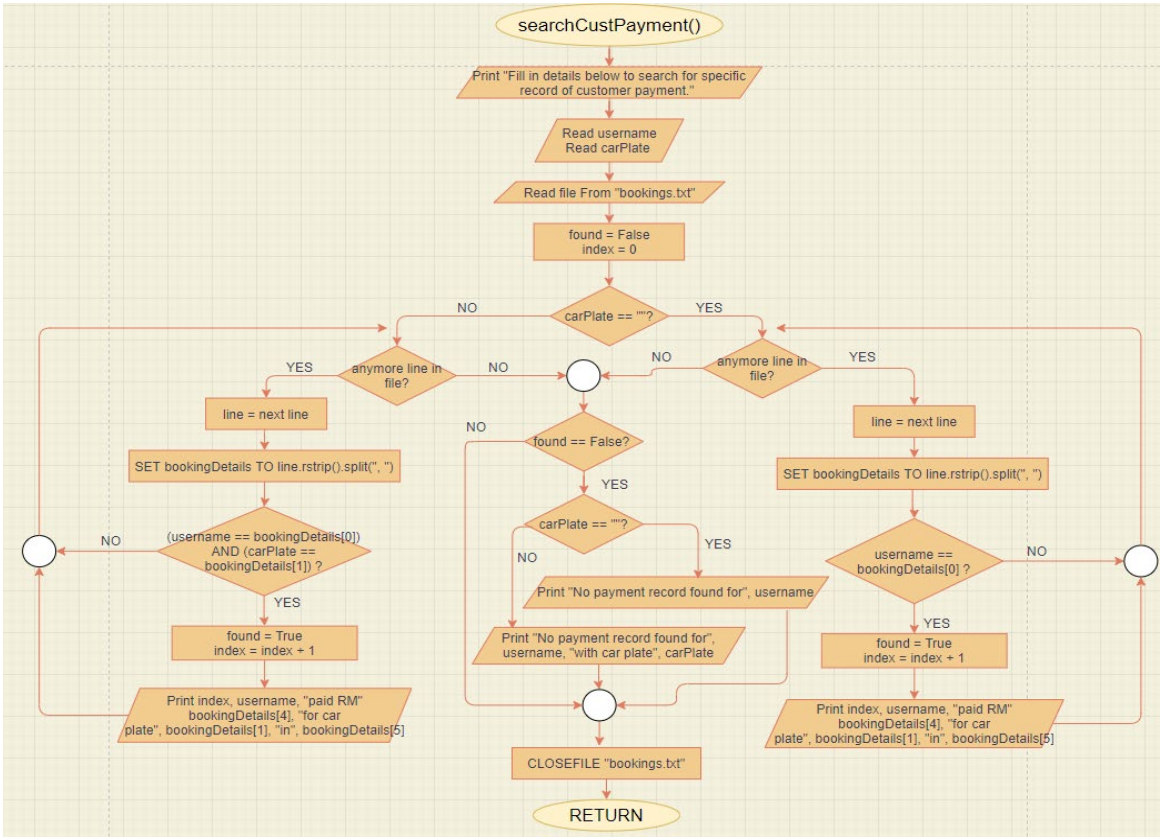

modifyCarDetails()
displayCars('all')
Read file from "cars.txt"
lines = READLINES(file)
Read carModel
Declare sameModelCars[]
found = False
index = 0
NO
anymore line in file?
YES
NO
found == False?
YES
line = next line
CLOSEFILE "cars.txt"
modifyPrice(lines, sameModelCars)
Print "Car model - ", carModel, "does not exist."
CLOSEFILE "cars.txt"
RETURN
index = index + 1
SET carDetails TO line.rstrip().split(', ')
carModel != carDetails[1]?
YES
NO
found = True
Append index Into carDetails
Append carDetails Into sameModelCars

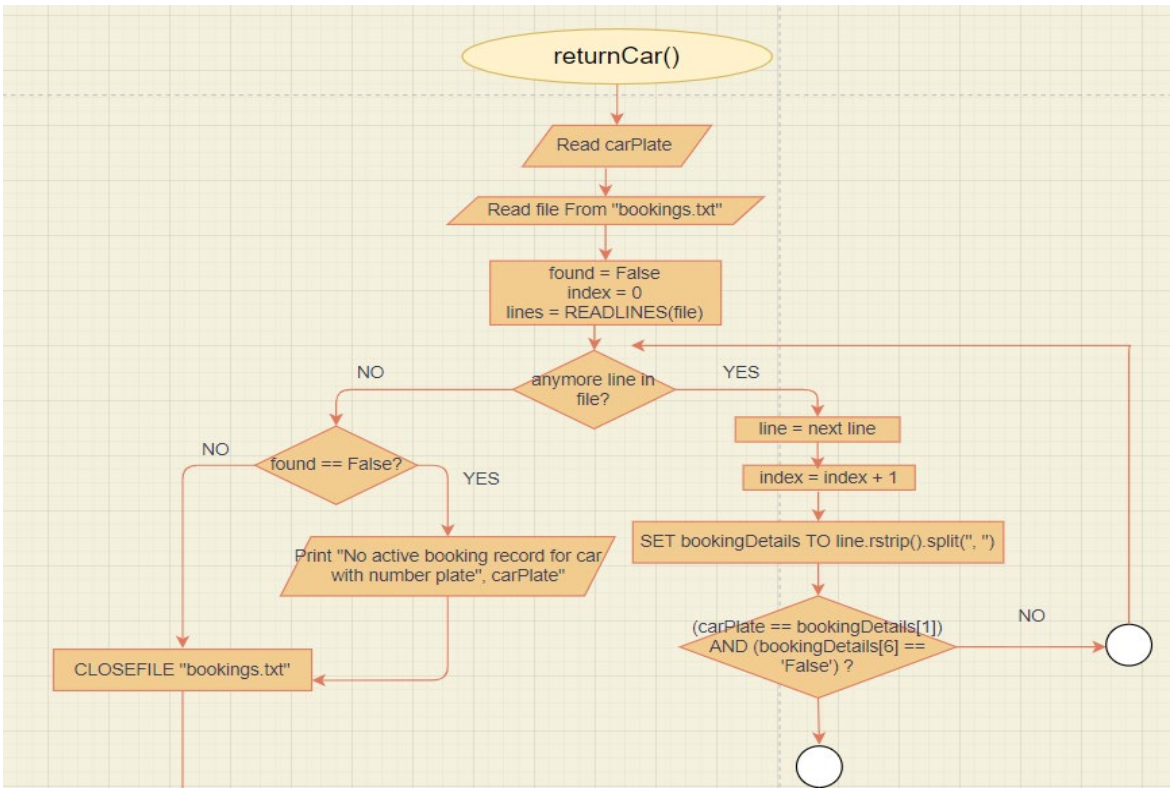

modifyPrice(lines, sameModelCars)
Read new_price
currentPrice = sameModelCars[0][3]
modelName = sameModelCars[0][1]
Print "The daily rental rate for all models of", modelName, "will be modified from RM", currentPrice, "to RM", new_price, "."
Print "Do you agree to this modification? ['0'= No, '1'= Yes]: "
Read answer
answer == 1?
NO
YES
Print "No changes made."
anymore car in sameModelCars?
NO
YES
OPENFILE "cars.txt" FOR WRITE
Write lines INTO "cars.txt"
Print "You have successfully modified the daily rental rate for all models of", modelName, "."
CLOSEFILE "cars.txt"
car = next car
index = car[4]
carPlate = car[0]
lines[index] = carPlate, ",", car[1], ",", car[2], ",", new_price, "\n"
RETURN

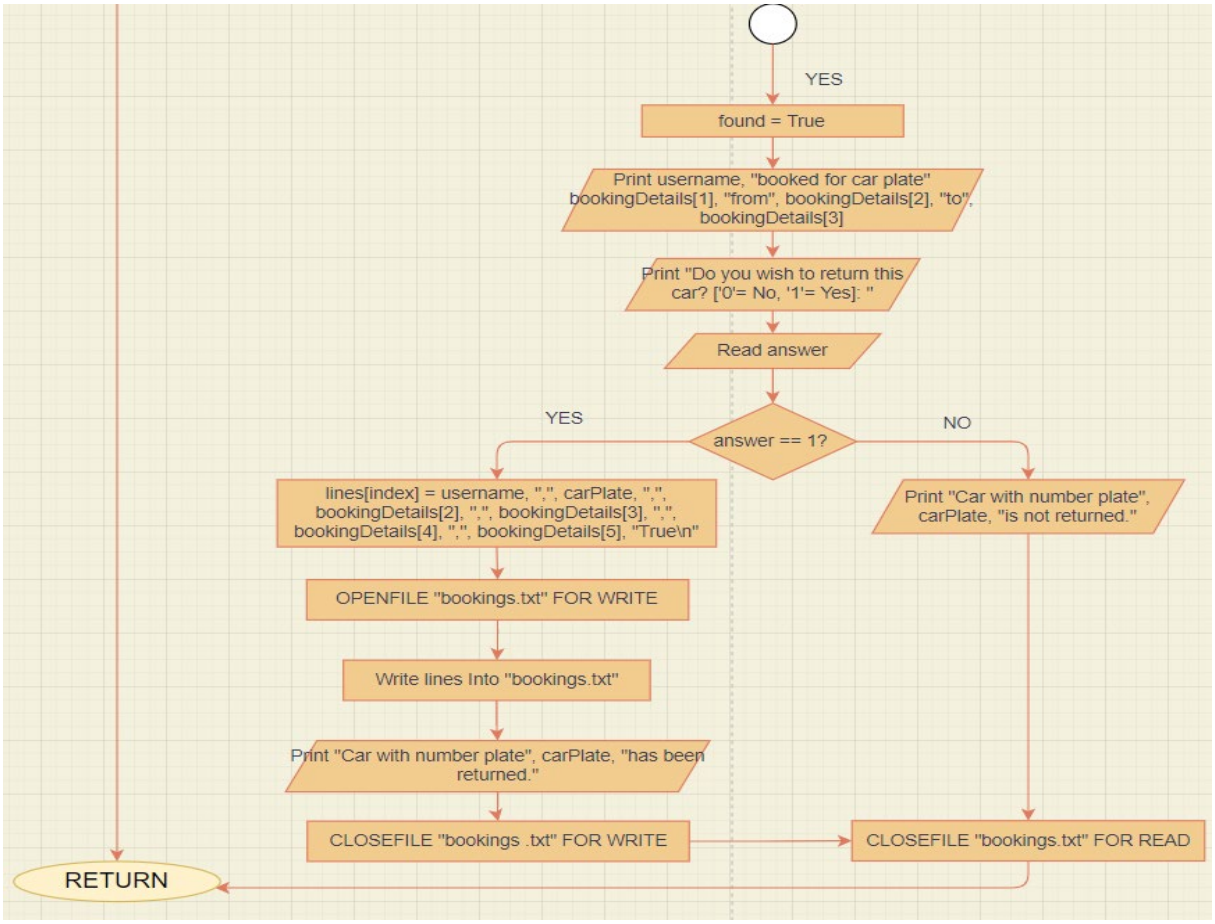

displayCustBookings()
Read file From "bookings.txt"
Print "Username Car Plate Start Date End Date
Daily Rate Pay Date Return Status"
NO
anymore line in file?
YES
CLOSEFILE "bookings.txt"
RETURN
line = next line
SET bookingDetails TO
line.rstrip().split(', ')
Print bookingDetails[0], ",",
bookingDetails[1], ",", bookingDetails[2], ",",
bookingDetails[3], ",", bookingDetails[4]


displayTimePayment()
Print "Fill in details below to search for specific
record of customer payment."
Read startDate
Read endDate
Read file From "bookings.txt"
found = False
index = 0
total = 0
NO
anymore line in file?
YES
NO
found = = False?
YES
line = next line
Print "No customer payment record found from",
bookingDetails[2], "to", bookingDetails[3]
SET bookingDetails TO line.rstrip().split(", ")
Print "Total revenue generated from", bookingDetails[2], "to",
bookingDetails[3], ": RM", total
(bookingDetails[5] >= startDate)
AND (bookingDetails[5] <=
endDate) ?
NO
YES
CLOSEFILE "bookings.txt"
found = True
index = index + 1
total = total + bookingDetails[4]
RETURN
Print index, bookingDetails[0], "paid RM",
bookingDetails[4], "for car plate",
bookingDetails[1], "in", bookingDetails[5]


searchCustBookings()
Print "Fill in details below to search for specific record of customer booking."
Read username
Read carPlate
Read file From "bookings.txt"
found = False
index = 0
carPlate == ""?
NO
YES
anymore line in file?
YES
NO
line = next line
SET bookingDetails TO line.rstrip().split(", ")
(username == bookingDetails[0]) AND (carPlate == bookingDetails[1]) ?
NO
YES
found = True
index = index + 1
Print index, username, "booked for car with number plate", carPlate, "from", bookingDetails[2], "to", bookingDetails[3], "return status =", bookingDetails[6]
found == False?
NO
YES
carPlate == ""?
NO
YES
Print "No bookings record found for," username
Print username, "has not booked a car with number plate of", carPlate
CLOSEFILE "bookings.txt"
RETURN
anymore line in file?
NO
YES
line = next line
SET bookingDetails TO line.rstrip().split(", ")
username == bookingDetails[0] ?
NO
YES
found = True
index = index + 1
Print index, username, "booked for car with number plate", bookingDetails[1], "from", bookingDetails[2], "to", bookingDetails[3], "return status =", bookingDetails[6]


searchCustPayment()
Print "Fill in details below to search for specific record of customer payment."
Read username
Read carPlate
Read file From "bookings.txt"
found = False
index = 0
carPlate == ""?
NO
YES
anymore line in file?
YES
NO
line = next line
SET bookingDetails TO line.rstrip().split(", ")
(username == bookingDetails[0]) AND (carPlate == bookingDetails[1]) ?
NO
YES
found = True
index = index + 1
Print index, username, "paid RM" bookingDetails[4], "for car plate", bookingDetails[1], "in", bookingDetails[5]
found == False?
NO
YES
carPlate == ""?
NO
YES
Print "No payment record found for", username
Print "No payment record found for", username, "with car plate", carPlate
CLOSEFILE "bookings.txt"
RETURN
anymore line in file?
NO
YES
line = next line
SET bookingDetails TO line.rstrip().split(", ")
username == bookingDetails[0] ?
NO
YES
found = True
index = index + 1
Print index, username, "paid RM" bookingDetails[4], "for car plate", bookingDetails[1], "in", bookingDetails[5]


returnCar()
Read carPlate
Read file From "bookings.txt"
found = False
index = 0
lines = READLINES(file)
anymore line in file?
NO
YES
line = next line
index = index + 1
SET bookingDetails TO line.rstrip().split(", ")
(carPlate == bookingDetails[1]) AND (bookingDetails[6] == 'False') ?
NO
found == False?
NO
YES
Print "No active booking record for car with number plate", carPlate"
CLOSEFILE "bookings.txt"


YES
found = True
Print username, "booked for car plate" bookingDetails[1], "from", bookingDetails[2], "to", bookingDetails[3]
Print "Do you wish to return this car? ['0'= No, '1'= Yes]: "
Read answer
YES
answer == 1?
NO
lines[index] = username, ",", carPlate, ",", bookingDetails[2], ",", bookingDetails[3], ",", bookingDetails[4], ",", bookingDetails[5], "True\n"
Print "Car with number plate", carPlate, "is not returned."
OPENFILE "bookings.txt" FOR WRITE
Write lines Into "bookings.txt"
Print "Car with number plate", carPlate, "has been returned."
CLOSEFILE "bookings .txt" FOR WRITE
CLOSEFILE "bookings.txt" FOR READ
RETURN

3.0 Source Code

There are in total of three text files for this program namely, 'users.txt', 'cars.txt', and 'bookings.txt'. Formats of each text file are provided as below:

```
limjiayong, jiayong, customer, lim@apu.com, 011-12345678
leanneooi, apple123, customer, leanne@apu.com, 011-73524538
harrypotter, 123, customer, potter@hogwarts.com, 017-47295923
ronweasley, 123, customer, ron@hogwarts.com, 016-57294729
adminjiayong, admin123, admin, adminjy@gmail.com, 012-54629134
```

Figure 3.0(a): 'users.txt' File

1. Format: username (unique), password, user type, email, phone number.

```
JGK9992, HONDA BR-V, 7, 150.00
SAM5517, HONDA CITY, 5, 150.00
SMB6338, HONDA CIVIC, 5, 180.00
WC6666W, BMW MINI 3DOOR, 4, 265.00
ALM7294, PERODUA MYVI, 5, 105.00
WUA1366, PERODUA ALZA, 7, 140.00
WHY8823, PERODUA BEZZA, 5, 110.00
WAW8888, TOYOTA RUSH, 7, 160.00
WOW8888, TOYOTA RUSH, 7, 160.00
VAT1388, BMW 330E, 5, 300.00
```

Figure 3.0(b): 'cars.txt' File

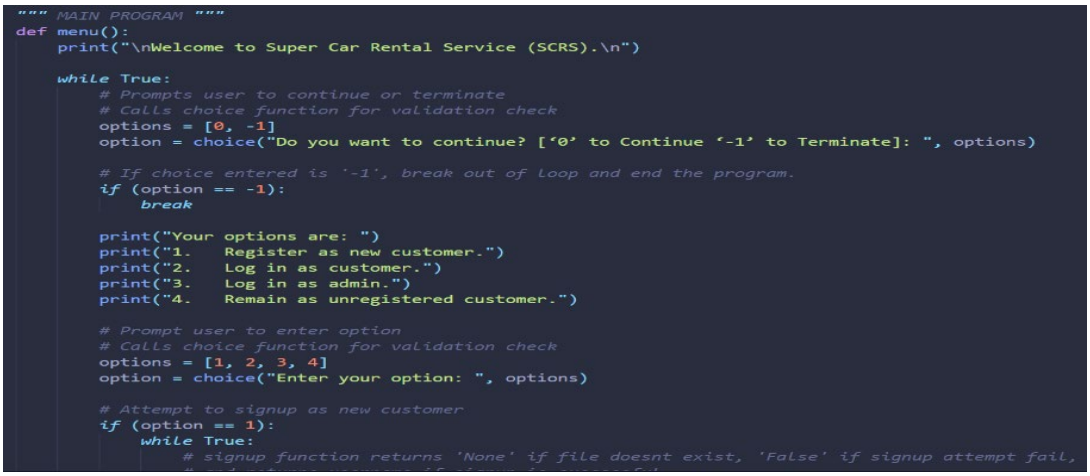
2. Format: car plate (unique), model name, seaters, price (RM per day)

```
limjiayong, SMB6338, 2021-05-25, 2021-05-30, 900.00, 2021-04-20, True
harrypotter, ALM7294, 2021-05-25, 2021-06-02, 840.00, 2021-05-24, True
leanneooi, WC6666W, 2021-05-25, 2021-05-25, 530.00, 2021-05-25, False
limjiayong, WOW8888, 2021-05-31, 2021-06-06, 960.00, 2021-05-24, True
limjiayong, SAM5517, 2021-06-04, 2021-06-08, 600.00, 2021-06-03, False
```

Figure 3.0(c): 'bookings.txt' File

3. Format: customer username, car plate, start date, end date, price, pay date, return status

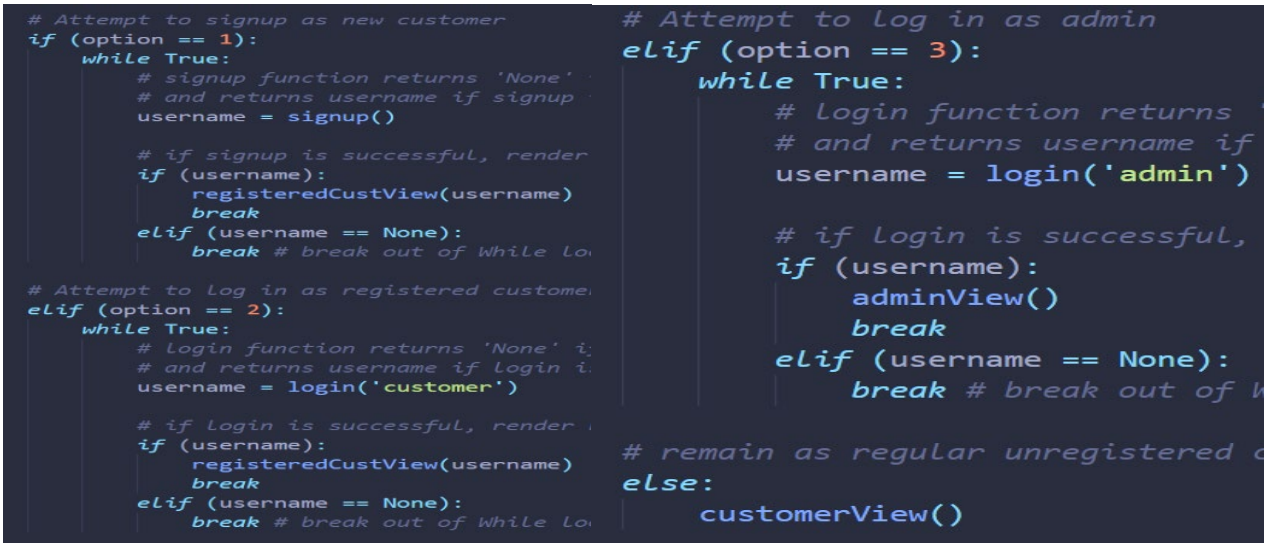
3.1 Main Menu Functions



```
""" MAIN PROGRAM """
def menu():
    print("\nWelcome to Super Car Rental Service (SCRS).\n")

    while True:
        # Prompts user to continue or terminate
        # Calls choice function for validation check
        options = [0, -1]
        option = choice("Do you want to continue? ['0' to Continue '-1' to Terminate]: ", options)

        # If choice entered is '-1', break out of loop and end the program.
        if (option == -1):
            break

        print("Your options are: ")
        print("1.   Register as new customer.")
        print("2.   Log in as customer.")
        print("3.   Log in as admin.")
        print("4.   Remain as unregistered customer.")

        # Prompt user to enter option
        # Calls choice function for validation check
        options = [1, 2, 3, 4]
        option = choice("Enter your option: ", options)

        # Attempt to signup as new customer
        if (option == 1):
            while True:
                # signup function returns 'None' if file doesnt exist, 'False' if signup attempt fail,
```

Figure 3.1(a): Menu Function

The entire Python program is driven by a main function known as 'menu()' *(Figure 3.1a)*, which will be called right at the bottom of the source code. This function consists of an infinite loop and only terminates when user explicitly requests to. Initially, users will be given four options, namely register as new customer, log in as customer, log in as admin, and remain as unregistered customer. Menu function will read the choice given by users and call appropriate functions accordingly (*Figure 3.1b)*.



```
# Attempt to signup as new customer
if (option == 1):
    while True:
        # signup function returns 'None'
        # and returns username if signup
        username = signup()

        # if signup is successful, render
        if (username):
            registeredCustView(username)
            break
        elif (username == None):
            break # break out of While lo

# Attempt to log in as registered custome
elif (option == 2):
    while True:
        # login function returns 'None' i
        # and returns username if login i
        username = login('customer')

        # if login is successful, render
        if (username):
            registeredCustView(username)
            break
        elif (username == None):
            break # break out of While lo

# Attempt to log in as admin
elif (option == 3):
    while True:
        # login function returns
        # and returns username if
        username = login('admin')

        # if login is successful,
        if (username):
            adminView()
            break
        elif (username == None):
            break # break out of

# remain as regular unregistered
else:
    customerView()
```

Figure 3.1(b): Menu Function

There is yet another infinite loop in each option whose purpose is to call the same function again if sign up or log in attempt fails. This may happen if users try to sign up with a taken username, or in the case of login function, wrong credentials have been provided.

```python
def signup():
    print("Please fill in the information below to signup.")
    username = input("Enter Username: ").strip()
    email = get_email("Enter Email: ")
    phoneNo = input("Enter Phone Number [e.g. 011-12345678]: ").strip()
    password = input("Enter Password: ").strip()
    confirmation = input("Enter Password Confirmation: ").strip()

    # Check if passwords match
    if (password != confirmation):
        print("Passwords must match. Please register again.\n")
        input("Press any key to continue...")
        return False

    try:
        # open 'users.txt' and check if username already exist
        with open ('users.txt', 'a+') as file:
            # bring file handler back to position 0 ('a+' will bring handler to end of file)
            file.seek(0)
            for line in file:
                # split each line in file into different values seperated by comma
                user, _, userType, *_ = line.rstrip().split(', ')
                if (user == username) and (userType == 'customer'):
                    print("Sorry, username has been taken. Please register with another username.\n")
                    input("Press any key to continue...")
                    return False

            # if nothing goes wrong, append user credentials to "users.txt" file
            file.write(f"{username}, {password}, customer, {email}, {phoneNo}\n")
            print(f"Registeration successful, welcome {username}.")
    except:
        print("Error occured in 'users.txt' file.\n")
        return

    input("Press any key to continue...")
    return username
```

Figure 3.1(c): Signup Function

Signup function *(Figure 3.1c)* takes necessary inputs from users. Inputs given are then validated before appending new information to 'users.txt' file. This is done to prevent duplication of username in the system, and if there is an attempt to sign up with a taken username, users will be prompted to sign up with a different username.

```python
# login function returns if file does not exist
# returns False if credentials provided are incorrect
# returns username of current user if login is successful
def login(userType):
    print(f"Please fill in your credentials. Note: You are currently trying to login as {userType}.")
    name = input("Enter Username: ")
    password = input("Enter Password: ")

    try:
        with open('users.txt', 'r') as file:
            for line in file:
                # split each line in file into different values seperated by comma
                user, passw, userType, *_ = line.rstrip().split(', ')
                if (user == name) and (passw == password) and (userType == userType):
                    print(f"Welcome back, {user}.\n")
                    input("Press any key to continue...")
                    return user

    except FileNotFoundError:
        print("Could not open 'users.txt' file.\n")
        input("Press any key to continue...")
        return

    print("Invalid username and (or) password.\n")
    input("Press any key to continue...")
    return False
```

Figure 3.1(d): Login Function

Login function request for username and password from users. Credentials are then compared against information in 'users.txt' file. If credential matches, user will be logged in to the system. Conversely, he/she will be directed back to reenter his/her credentials if credentials do not match.

Helper Functions

For the sake of convenience, helper functions such as get_int, get_float, get_date, get_email, choice, and cars are created to be called whenever they are needed.

```
# Helper function - Function that repr
def choice(message, options):
    while True:

        # Prompt user to enter choice
        choice = get_int(message)
        if (choice in options):
            print()
            return choice

        # if choice is not in given l
        print("Invalid Input.")
```

Figure 3.1(e): Choice Function

For instance, Choice function accepts a string 'message' and a list 'options' as parameters. The function then prompts user with 'message' and only accepts specific values which resides in the 'options' list. This helps to validate user input.

```
def cars(availability):
    try:
        with open('bookings.txt', 'r') as bookingsFile:
            availableCars = [] # master list 1
            unavailableCars = [] # master list 2
            bookedCars = []

            for line in bookingsFile:
                # split each line in file into different values seperate
                _, carPlate, *_, status = line.rstrip().split(', ')

                # If car is not returned (status of 'False'), append thi
                if (status == 'False'):
                    bookedCars.append(carPlate)
```

```
            try:
                with open('cars.txt', 'r') as carsFile:
                    if (availability == 'all'):
                        cars = [] # master list
                        for line in carsFile:
                            # read each line and store information in a
                            carDetail = line.rstrip().split(', ')
                            cars.append(carDetail)
                        # returns master list containing all cars
                        return cars

                    # if availability is not 'all', do the following:
                    for row in carsFile:
                        # split each line in file into different values
                        carPlate, *values = row.rstrip().split(', ')

                        if (carPlate not in bookedCars):
                            availableCars.append([carPlate, *values])
                        else:
                            unavailableCars.append([carPlate, *values])
```

Figure 3.1(f): Cars Function

Cars is a function which accepts a string parameter of either 'available', 'unavailable', or 'all', and returns a master list of available cars, unavailable cars, or all cars, depending on the value of string parameter. The function first reads 'bookings.txt' file and append all car plates of booked cars into bookedCars list. It then opens 'cars.txt' file and checks the value of the string parameter passed in. If 'all' cars are requested, it appends every car to a master list known as cars and returns this master list to the caller. Otherwise, it appends cars that are not in bookedCars to availableCars, and the rest to unavailableCars. Master list availableCars will be returned to the caller if string 'available' is passed in a parameter, and vice versa for unavailable cars.

3.2 Registered Customer View Functions

```
"""FUNCTIONS RELATED TO REGISTERED CUSTOMER VIEW"""
# The interface which registered customers get to access
def registeredCustView(username):

    print()
    print("Your options are: ")
    print("1.   Modify personal details.")
    print("2.   View Personal Rental History.")
    print("3.   View Detail of Cars to be Rented Out.")
    print("4.   Select and Book a car for a specific duration.")
    print("5.   Log out.")

    # Validation check
    options = list(range(1, 6))  # list of 1 to 5
    option = choice("Enter your option: ", options)

    if (option == 1):
        modifyCustDetails(username)

    elif (option == 2):
        displayRentalHistory(username)
```

```
elif (option == 3):
    displayCars('available')

elif (option == 4):
    bookCar(username)

else:  # option == 5
    return

return registeredCustView(username)
```

Figure 3.2(a): Registered Customer View Function

RegisteredCustView function (*Figure 3.2a*) is an interface designed for registered customers. It displays an options list consisting of 5 varying options and each option will call the next function related to it.

```
# Option 1 - Allow customer to modify personal details.
def modifyCustDetails(username):
    try:
        # Opening the users.txt to read and obtain the deta
        with open('users.txt', 'r') as file:
            lines = file.readlines()

            for index, line in enumerate(lines):
                # read each line and store information in a
                userDetails = line.rstrip().split(', ')
                # continue if current line in file is not t
                if (userDetails[0] != username):
                    continue

                # display customer information
                print(f"These are your current details:")
                print(f"Email: {userDetails[3]}")
                print(f"Phone Number: {userDetails[4]}\n")
```

```
print("Your options are: ")
print("1. Change Password.")
print("2. Change Email.")
print("3. Change Phone Number.")
print("4. Exit.")
# Validation check
options = list(range(1, 5))  # List of 1 to 4
option = choice("Enter your option: ", options)
```

```
                if (option == 1):
                    modifyPassword(userDetails, lines, index)

                elif (option == 2):
                    modifyEmail(userDetails, lines, index)

                elif (option == 3):
                    modifyPhoneNo(userDetails, lines, index)

                else:  # option == 4
                    return

                input("Press any key to continue...")
                return modifyCustDetails(username)

    except FileNotFoundError:
        print("Could not open 'users.txt' file.\n")
        return
```

Figure 3.2(b): Modify Customer Details Function

Firstly, modifyCustDetails function (*Figure 3.2b*) accepts username as parameter. After locating the user from 'users.txt' file, the function displays all current details in the file. Then, it displays an options list enabling customers to decide on their next action and each option in the list will call the next related function.

```
# Allow user to modify password
def modifyPassword(userDetails, lines, index):
    current_password = input("Enter your current password: ")
    password = userDetails[1]

    # Check if the the current password match with the password in users.txt
    if (current_password != password):
        print("Invalid password.")
        return modifyPassword(userDetails, lines, index)

    new_password = input("Enter your new password: ")
    confirm_password = input("Enter your new password to confirm: ")
    # Check if the new password and the confirmation password matches
    if (new_password != confirm_password):
        print("Passwords must match.")
        input("Press any key to continue...")
        return modifyPassword(userDetails, lines, index)

    # If all validations are passed, update user.txt file to reflect changes
    updateUserFile('password', userDetails, lines, index, new_password)
```

Figure 3.2(c): Modify Password Function

The modifyPassword function is called by option 1 in the modifyCustDetails function. This function accepts a list of 'userDetails', 'lines', and an integer 'index' as parameters, and will request for the customer's current password to carry out validation check on the identity of the customer. After proper validations, the function will obtain a new password and a confirmation password from the customer. If password matches, updateUserFile function will be called to proceed with file modifications. modifyEmail and modifyPhoneNo function works very similarly to modifyPassword function.

```
# logic for updating users.txt file
# key is a string value of either 'password', 'email', or 'phone number',
# userDetails is a list storing all information of current user,
# lines stores the entire content of 'users.txt' file line by line,
# index is the row number of 'users.txt' that should be modified
# information is the new_information to be updated in the file
def updateUserFile(key, userDetails, lines, index, information):

    username = userDetails[0]
    if (key == 'password'):
        currentInfo = userDetails[1]
        lines[index] = (f"{username}, {information}, {userDetails[2]}, {userDetails[3]}, {userDetails[4]}\n")
    elif (key == 'email'):
        currentInfo = userDetails[3]
        lines[index] = (f"{username}, {userDetails[1]}, {userDetails[2]}, {information}, {userDetails[4]}\n")
    else:  # key = phone number
        currentInfo = userDetails[4]
        lines[index] = (f"{username}, {userDetails[1]}, {userDetails[2]}, {userDetails[3]}, {information}\n")
```

```
    # Double check with user
    print(f"{username}, your {key} will be modified from {currentInfo} to {information}.")
    options = [0, 1]
    answer = choice("Do you agree to this modification? ['0'= No, '1'= Yes]: ", options)

    # If user agrees, the modification will take place
    if (answer == 1):
        try:
            # overwriting old file
            with open('users.txt', 'w') as writeFile:
                writeFile.writelines(lines)
                print(f"{username}, you have successfully modified your {key}.")
        except:
            print("Error occured in 'users.txt' file.\n")

    else:
        print(f"No changes made.")

    return
```

Figure 3.2(d): Update User File Function

Function shown in *Figure 3.2d*, updateUserFile(key, userDetails, lines, index, information), is called after passing validation checks in modifyPassword, modifyEmail, and modifyPhoneNo functions. This function overwrites existing information in 'users.txt' file according to the 'key' which represents a string value of 'password', 'email', or 'phone number'. Based on the key, information to be modified will be located in 'users.txt' file with the help of row number known as 'index'. If the customer agrees to the modification, the modification process will take place.

```
# Option 2 - Allow customer to view personal rental history
def displayRentalHistory(user):
    found = False
    try:
        # Opening the bookings.txt file to read
        with open('bookings.txt', 'r') as file:
            print("Username        Car Plate     Start Date      End Date     Daily Rate (RM)     Pay Date    Return Status")
            for line in file:
                # split each line in file into different values seperated by comma
                bookingDetails = line.strip().split(', ')
                if (user == bookingDetails[0]):
                    # Showing the customer's rental history
                    found = True
                    print(f"{bookingDetails[0]}        {bookingDetails[1]}      {bookingDetails[2]}     {bookingDetails[3]}     {bookingDetails[4]}
            if not found:
                print("No personal rental history found.")
    except:
        print("Could not open 'bookings.txt' file.\n")

    input("Press any key to continue...")
    return
```

Figure 3.2(e): Display Rental History Function

DisplayRentalHistory(user) function utilizes customer's username to search for matching information in 'bookings.txt' file and display them.

```
# Option 3 - Display all details of cars available for rent
def displayCars(availability):
    vehicles = cars(availability)
    print(f"{availability.capitalize()} cars for rent: ")
    print("Number Plate     Model Name    Number of Seats    Rental Price (RM)")

    # display information for each vehicle
    for vehicle in vehicles:
        print(f"{vehicle[0]}\t\t{vehicle[1]}\t\t{vehicle[2]}\t\t{vehicle[3]}")

    input("\nPress any key to continue...")
    return
```

Figure 3.2(f): Display Cars Function

Next, the displayCars(availability) function operates by calling 'Cars' helper function, which returns a master list of all available cars for rent. Each vehicle in the 'vehicles' master list will then be displayed.

```python
def bookCar(username):
    # display all available cars to admin before asking which car to modify
    displayCars('available')
    print()
    carNo = input("Enter the number plate of the car that you wish to book: ")

    try:
        # Opening cars.txt to read
        with open('cars.txt', 'r') as file:
            found = False
            for line in file:
                # split each line in file into different values seperated by comma
                carPlate, *_, price = line.strip().split(', ')
                # Checking if car plate entered by customer matches with one of the car plates in text f
                if (carNo != carPlate):
                    continue

                found = True
                # Making sure that rental start date is within a week from today and must not be in the p
                validStartDate = False
                while (validStartDate == False):
                    firstDate = get_date("Enter the start date of your rental. Must be after today and wi
                    today = date.today()
                    numOfDaysFromToday = firstDate - today
                    if numOfDaysFromToday.days <= 7 and numOfDaysFromToday.days > 0:
                        validStartDate = True
                    else:
                        print("Please make sure the starting date is after today and within 7 days from t
                        continue

                    lastDate = get_date("Enter the ending date of your rental  [e.g. 2021-06-03]: ")
                    numberOfDays = (lastDate - firstDate).days
                    # Checking if the starting date is before end date
                    if (numberOfDays <= 0):
                        print("Start date has to be before end date.")
                        continue

                    bookingPrice = numberOfDays * float(price)
                    payDate = date.today()
                    return_status = "False"
                    # Confirming with the customer on his/her booking
                    print(f"{username}, you will be booking {carNo} from {firstDate} to {lastDate} with t
                    options = [0, 1]
                    answer = choice("Do you agree to this booking? ['0'= No, '1'= Yes]: ", options)
                    # If user agrees, the booking will take place
                    if (answer == 1):
                        try:
                            # Opening bookings.txt to append
                            with open('bookings.txt', 'a') as file:
                                file.write(f"{username}, {carNo}, {firstDate}, {lastDate}, {bookingPrice:
                                print(f"Booking successful, thank you {username}.")
                        except:
                            print("Error occured in 'bookings.txt' file.\n")
                            break
```

Figure 3.2(g): Book Car Function

Lastly, bookCar(username) function first displays all available cars for rental. Customer will then be prompted to insert car plate he/she would like to rent. Subsequently, the function locates the desired car in ‘cars.txt’ file, and requests customer for start and end date of rental. Next, the total price will be calculated, and customer will be prompted to decide whether to proceed with the booking. If the decision is to proceed, booking details will be appended to ‘bookings.txt’ file.

3.3 Admin View Functions

```
def adminView():

    print()
    print("Your options are: ")
    print("1.   Add cars to be rented out.")
    print("2.   Modify car details.")
    print("3.   Display all details of cars rented out.")
    print("4.   Display all details of cars available for rent.")
    print("5.   Display all details of customer bookings.")
    print("6.   Display all details of customer payment for a specific time duration.")
    print("7.   Search specific record of customer booking.")
    print("8.   Search specific record of customer payment.")
    print("9.   Return a rented car.")
    print("10.  Log out.")

    # Validation check
    options = list(range(1, 11)) # list of 1 to 10
    option = choice("Enter your option: ", options)

    if (option == 1):
        addCars()

    elif (option == 2):
        modifyCarDetails()

    elif (option == 3):
        displayCars('unavailable')

    elif (option == 4):
        displayCars('available')

    elif (option == 5):
        displayCustBookings()
```

```
    # Display all details of customer payment for a specific time duration
    elif (option == 6):
        displayTimePayment()

    # Search specific record of customer booking
    elif (option == 7):
        searchCustBookings()

    # Search specific record of customer payment
    elif (option == 8):
        searchCustPayment()

    # Return a rented car
    elif (option == 9):
        returnCar()

    else: # option == 10
        print("Successfully logged out.")
        return

    input("Press any key to continue...")
    return adminView()
```

Figure 3.3 (a): Admin View Function

Admin view function (*Figure 3.3a*) is the interface which admins get to access. Admins will choose an option and the function will call related functions separately based on admin's choice.

```
def addCars():
    print("Please fill in the information below to add a new car.")
    carPlate = input("Enter Car Number Plate: ").upper()
    carModel = input("Enter Car Model: ").upper()
    seaters = get_int("How many person can this car be seated? ")
    price = get_float("Enter Rental Rate (per day basis): RM")

    try:
        # Validation check: check if this car plate already exist
        with open('cars.txt', 'a+') as file:
            # bring file handler back to position 0 ('a+' will bring handler to end of file)
            file.seek(0)
            for line in file:
                # split each line in file into different values seperated by comma
                carNo, *_ = line.rstrip().split(', ')
                if (carNo == carPlate):
                    print(f"\nCar with the number plate {carPlate} already exist.")
                    return

            # Displaying final confirmation and prompting admin to continue or cancel
            print(f"""Car model {carModel} ({carPlate}) with {seaters} seaters and a daily rental rate of
                    RM{price:.2f} will be added.""")
            options = [0, 1]
            answer = choice("Do you wish to add this car? ['0'= No, '1'= Yes]: ", options)

            if (answer == 1):
                # append to 'cars.txt' after passing validation check above
                file.write(f"{carPlate}, {carModel}, {seaters}, {price:.2f}\n")
                print(f"Car with the number plate {carPlate} succesfully added.\n")
            else:
                print("No changes made.")

    except:
        print("Error occured in 'cars.txt' file.")

    return
```

Figure 3.3(b): Add Cars Function

If admin selects option 1, addCars function will be called (*Figure 3.3b*). This function takes in car plate, car model, number of seaters, and rental rate per day as input from admin. After validating input to ensure no duplication of data in text file, car details will be displayed, and admin will be prompted for confirmation. Upon confirmation, new car along with its details will be appended into 'cars.txt' file.

```
# Option 2 - Allow admin to modify car details
def modifyCarDetails():
    # display all cars to admin before asking which car to modify
    displayCars('all')
    try:
        # Opening cars.txt to read and obtain the details
        with open('cars.txt', 'r') as file:
            lines = file.readlines()
            carModel = input("Enter the car model you wish to modify: ").upper().strip()
            sameModelCars = [] # master list
            found = False

            for index, line in enumerate(lines):
                # split each line in file into different values seperated by comma
                carDetails = line.rstrip().split(', ')
                # continue looping if current car model in line is not what we are searching for
                if (carModel != carDetails[1]):
                    continue

                found = True
                # append index to keep track of which row to be modified later on
                carDetails.append(index)
                sameModelCars.append(carDetails)

            if not found:
                print(f"Car model - {carModel} does not exist.")
            else:
                modifyPrice(lines, sameModelCars)

    except:
        print("Could not open 'cars.txt' file.\n")

    input("Press any key to continue...")
    return
```

Figure 3.3(c): Modify Car Details Function

The modifyCarDetails function will prompt admin to enter the car model he/she wishes to modify. The car model provided will then be checked against car models residing in 'cars.txt' file. All cars with that specific model will be appended to sameModelCars list. Next, the function will call modifyPrice function, passing in 'lines' and 'sameModelCars' as argument. Conversely, if no such car model exists in 'cars.txt' file, an error message will be displayed.

```python
# Modify rental rate per day
def modifyPrice(lines, sameModelCars):
    new_price = get_float("Enter the new daily rental rate (RM): ")
    currentPrice = float(sameModelCars[0][3])
    modelName = sameModelCars[0][1]

    # Displaying final confirmation and prompting admin to continue or cancel
    print(f"The daily rental rate for all models of {modelName} will be modified from RM{currentPrice:.2f} to RM{new_price:.2f}.")
    options = [0, 1]
    answer = choice("Do you agree to this modification? ['0'= No, '1'= Yes]: ", options)

    # If user agrees, the modification will take place
    if (answer == 1):
        for car in sameModelCars:
            index = car[4]
            carPlate = car[0]
            lines[index] = (f"{carPlate}, {car[1]}, {car[2]}, {new_price:.2f}\n")

        try:
            # overwriting old file
            with open('cars.txt', 'w') as writeFile:
                writeFile.writelines(lines)
                print(f"You have successfully modified the daily rental rate for all models of {modelName}.")
                print(f"{len(sameModelCars)} record(s) affected.")
        except:
            print("Error occured in 'cars.txt' file.\n")

    else:
        print(f"No changes made.")

    input("Press any key to continue...")
    return
```

Figure 3.3(d): Modify Price Function

The modifyPrice(lines, sameModelCars) function will read in new daily rental rate for the car model. The details for both current price and model name will be obtained from sameModelCars master list. Subsequently, admin will be prompted to confirm the modification process. Upon confirmation, 'cars.txt' file will be overwritten with new price.

```python
# Option 5 - Display all details of customer bookings
def displayCustBookings():
    try:
        # Opening the bookings.txt to read
        with open('bookings.txt', 'r') as file:
            print("Username    Car Plate    Start Date    End Date    Daily Rate    Pay Date    Return Status")

            for line in file:
                # split each line in file into different values seperated by comma
                bookingDetails = line.rstrip().split(', ')
                print(""" f"{bookingDetails[0]}        {bookingDetails[1]}      {bookingDetails[2]}    {bookingDetails[3]}     {bookingDetails[4]}
                      {bookingDetails[5]}     {bookingDetails[6]}" """)
    except:
        print("Error opening bookings.txt.")

    input("Press any key to continue...")
    return
```

Figure 3.3(e): Display Customer Bookings Function

Next, displayCustBookings function opens 'bookings.txt' file and display all customer booking details.

```python
# Option 6 - Display all details of customer payment for a specific time duration
def displayTimePayment():
    print("Fill in details below to search for specific record of customer payment.")
    startDate = get_date("Enter Start Date [e.g. 2021-05-31]: ")
    endDate = get_date("Enter End Date [e.g. 2021-05-31]: ")
    print()

    try:
        with open('bookings.txt', 'r') as file:
            found = False
            index = 0
            total = 0
            for line in file:
                # split each line in file into different values seperated by comma
                username, carPlate, *_, price, payDate, _ = line.strip("[]\n").split(', ')
                payDate = date.fromisoformat(payDate)
                if (payDate >= startDate) and (payDate <= endDate):
                    found = True
                    index += 1
                    total += float(price)
                    # displaying details of customer payment
                    print(f"{index}. {username} paid RM {price} for car plate - {carPlate} in {payDate}.")

        # print message if no record is found
        if not found:
            print(f"No customer payment record found from {startDate} to {endDate}.")
        else:
            print(f"Total revenue generated from {startDate} to {endDate}: RM{total:.2f}]\n")

    except:
        print("Could not open 'bookings.txt' file.")

    return
```

Figure 3.3(f): Display Time Payment Function

Display time payment function (*Figure 3.3f*) accepts two inputs from admin, namely start date and end date. The function then opens ‘bookings.txt’ file and extract information from the file. Customer’s payment date stored in the text file will be compared against the two dates provided by admin. If the payment date is between the dates provided by admin, the record will be displayed. Total revenue generated will be displayed as well.

```python
def searchCustBookings():
    print("Fill in details below to search for specific record of customer booking.")
    username = input("Enter customer's username: ")
    carPlate = input(f"Enter car plate booked by {username} [Leave this field empty if you wis
    print()

    try:
        with open('bookings.txt', 'r') as file:
            found = False
            index = 0

            # if car plate is not provided, print all booking information associated with this
            if (carPlate == ''):
                for line in file:
                    # split each line in file into different values seperated by comma
                    name, carNo, startDate, endDate, *_, status = line.rstrip().split(', ')

                    if (username == name):
                        found = True
                        index += 1

                        # displaying details of customer booking
                        print("{index}. {username} booked for car with number plate - {carNo} 

            # if car plate is provided, print booking information matching username and car pl
            else:
                for line in file:
                    # split each line in file into different values seperated by comma
                    name, carNo, startDate, endDate, *_, status = line.strip("[]\n").split(', 

                    if (username == name) and (carPlate == carNo):
                        found = True
                        index += 1

                        # displaying details of customer booking
                        print(f"{index}. {username} booked for car plate - {carNo} from {start

        # print message if no record is found
        if not found:
            if (carPlate == ''):
                print(f"No bookings record found for {username}.")
            else:
                print(f"{username} has not booked a car with number plate of {carPlate}.")
```

Figure 3.3(g): Search Customer Bookings Function

On the other hand, searchCustBookings function (*Figure 3.3g)* enables admin to search for customer bookings record by providing customer's username and optionally the car plate booked by them. Next, 'bookings.txt' file will be opened in read mode and if car plate is provided, the function checks for record matching the username and the car plate; if none is provided, the function checks for details matching the username. Matching records will be displayed to admin accordingly.

Likewise, for searchCustPayment function, it enables admin to look up for specific records of customer payment. The overall logic is similar to searchCustBookings function, with the only distinction in the message displayed.

```python
def returnCar():
    carPlate = input(f"Enter car plate to return: ").upper()
    print()
    try:
        with open('bookings.txt', 'r') as file:
            found = False
            lines = file.readlines()

            for index, line in enumerate(lines):
                username, carNo, startDate, endDate, price, payDate, status = line.rstrip().split(', ')
                if (carPlate == carNo) and (status == 'False'):
                    found = True
                    # displaying booking details and prompting admin to continue or cancel
                    print(f"{username} booked for car plate - {carNo} from {startDate} to {endDate}.")
                    options = [0, 1]
                    answer = choice("Do you wish to return this car? ['0'= No, '1'= Yes]: ", options)
                    # if admin agrees, proceed with returning the car
                    if (answer == 1):
                        lines[index] = f"{username}, {carNo}, {startDate}, {endDate}, {price}, {payDate}, True\n"
                        try:
                            # overwriting old file
                            with open('bookings.txt', 'w') as writeFile:
                                writeFile.writelines(lines)
                                print(f"Car with number plate - {carNo} has been returned.")
                        except:
                            print("Error occured in 'bookings.txt' file.\n")
                    else:
                        print(f"Car with number plate - {carNo} is not returned.")

                    # once record is found, break out of loop
                    break
            if not found:
                print(f"No active booking record for car with number plate - {carPlate}.")
    except:
        print("Could not open 'bookings.txt' file.")
    return
```

Figure 3.3(l): Return Car Function

Finally, returnCar function (*Figure 3.3l*) allows admin to return a specific car rented by customers. Admin will be prompted to enter car number plate, and the function checks 'bookings.txt' file for that specific car with current return status of false, or in other words, a car that has not been returned yet. If there is no matching car records, error message will be displayed, otherwise admin will be asked to confirm the return of that car, and upon confirmation, records will be updated accordingly in 'bookings.txt' file.

3.4 Regular Customer View

```python
"""UNREGISTERED CUSTOMER VIEW"""
# The interface which unregistered customers get to access
def customerView():
    # Show all available cars for rent
    displayCars('available')
    return
```

Figure 3.4: Customer View Function

CustomerView function (*Figure 3.4*) operates by calling displayCars function to display all available cars.

4.0 Sample Input Output

4.1 Main Menu

To register as new customer:

```
Welcome to Super Car Rental Service (SCRS).

Do you want to continue? ['0' to Continue '-1' to Terminate]: 0

Your options are:
1.      Register as new customer.
2.      Log in as customer.
3.      Log in as admin.
4.      Remain as unregistered customer.
Enter your option: 1

Please fill in the information below to signup.
Enter Username: Esther
Enter Email: estheryip@gmail.com
Enter Phone Number [e.g. 011-12345678]: 019-29332011
Enter Password: estheryip
Enter Password Confirmation: estheryip
Registeration successful, welcome Esther.
Press any key to continue...
```

users - Notepad

File Edit Format View Help

```
limjiayong, jiayong, customer, lim@apu.com, 011-12345678
leanneooi, apple123, customer, leanne@apu.com, 011-73524538
harrypotter, 123, customer, potter@hogwarts.com, 017-47295923
ronweasley, 123, customer, ron@hogwarts.com, 016-57294729
adminjiayong, admin123, admin, adminjy@gmail.com, 012-54629134
Amanda, Aamanda, customer, amandasoong@gmail.com, 011-02993822
linda, lindalin, customer, linda@gmail.com, 016-29921001
Esther, estheryip, customer, estheryip@gmail.com, 019-29332011
```

Figure 4.1 (a): Registration Successful and New Information in 'users.txt' File

To log in as customer:

```
Welcome to Super Car Rental Service (SCRS).

Do you want to continue? ['0' to Continue '-1' to Terminate]: 0

Your options are:
1.      Register as new customer.
2.      Log in as customer.
3.      Log in as admin.
4.      Remain as unregistered customer.
Enter your option: 2

Please fill in your credentials. Note: You are currently trying to login as customer.
Enter Username: Esther
Enter Password: estheryip
Welcome back, Esther.

Press any key to continue...
```

Figure 4.1(b): Logging in as Customer

4.2 Registered Customer

To modify personal details:

```
Your options are:
1.    Modify personal details.
2.    View Personal Rental History.
3.    View Detail of Cars to be Rented Out.
4.    Select and Book a car for a specific duration.
5.    Log out.
Enter your option: 1

These are your current details:
Email: estheryip@gmail.com
Phone Number: 019-29332011

Your options are:
1. Change Password.
2. Change Email.
3. Change Phone Number.
4. Exit.
Enter your option:
```

Figure 4.2(a): Option 1- Modify Personal Details

To modify password:

```
Your options are:
1. Change Password.
2. Change Email.
3. Change Phone Number.
4. Exit.
Enter your option: 1

Enter your current password: estheryip
Enter your new password: estheryip11
Enter your new password to confirm: estheryip11
Esther, your password will be modified from estheryip to estheryip11.
Do you agree to this modification? ['0'= No, '1'= Yes]: 1

Esther, you have successfully modified your password.
Press any key to continue...
```

users - Notepad

File Edit Format View Help

```
limjiayong, jiayong, customer, lim@apu.com, 011-12345678
leanneooi, apple123, customer, leanne@apu.com, 011-73524538
harrypotter, 123, customer, potter@hogwarts.com, 017-47295923
ronweasley, 123, customer, ron@hogwarts.com, 016-57294729
adminjiayong, admin123, admin, adminjy@gmail.com, 012-54629134
Amanda, Aamanda, customer, amandasoong@gmail.com, 011-02993822
linda, lindalin, customer, linda@gmail.com, 016-29921001
Esther, estheryip11, customer, estheryip@gmail.com, 019-29332011
```

Figure 4.2(b): Option 1.1- Modify Password and Modification in 'users.txt' File

To modify Email:

```
Your options are:
1. Change Password.
2. Change Email.
3. Change Phone Number.
4. Exit.
Enter your option: 2

Enter your new email address: estheryip22@gmail.com
Enter your new new email address to confirm: estheryip22@gmail.com
Esther, your email will be modified from estheryip@gmail.com to estheryip22@gmail.com.
Do you agree to this modification? ['0'= No, '1'= Yes]: 1

Esther, you have successfully modified your email.
Press any key to continue...
```

users - Notepad

File Edit Format View Help

```
limjiayong, jiayong, customer, lim@apu.com, 011-12345678
leanneooi, apple123, customer, leanne@apu.com, 011-73524538
harrypotter, 123, customer, potter@hogwarts.com, 017-47295923
ronweasley, 123, customer, ron@hogwarts.com, 016-57294729
adminjiayong, admin123, admin, adminjy@gmail.com, 012-54629134
Amanda, Aamanda, customer, amandasoong@gmail.com, 011-02993822
linda, lindalin, customer, linda@gmail.com, 016-29921001
Esther, estheryip11, customer, estheryip22@gmail.com, 019-29332011
```

Figure 4.2(c): Option 1.2 – Modify Email and Modification in 'users.txt' File

To modify phone number:

```
Your options are:
1. Change Password.
2. Change Email.
3. Change Phone Number.
4. Exit.
Enter your option: 3

Enter your new phone number: 011-29382019
Enter your new new phone number to confirm: 011-29382019
Esther, your phone number will be modified from 019-29332011 to 011-29382019.
Do you agree to this modification? ['0'= No, '1'= Yes]: 1

Esther, you have successfully modified your phone number.
Press any key to continue...
```

users - Notepad

File Edit Format View Help

```
limjiayong, jiayong, customer, lim@apu.com, 011-12345678
leanneooi, apple123, customer, leanne@apu.com, 011-73524538
harrypotter, 123, customer, potter@hogwarts.com, 017-47295923
ronweasley, 123, customer, ron@hogwarts.com, 016-57294729
adminjiayong, admin123, admin, adminjy@gmail.com, 012-54629134
Amanda, Aamanda, customer, amandasoong@gmail.com, 011-02993822
linda, lindalin, customer, linda@gmail.com, 016-29921001
Esther, estheryip11, customer, estheryip22@gmail.com, 011-29382019
```

Figure 4.2(d): Option 1.3 - Modify Phone Number and Modification in 'user.txt' File

To exit from the modify personal details function:

```
Your options are:
1. Change Password.
2. Change Email.
3. Change Phone Number.
4. Exit.
Enter your option: 4


Your options are:
1.   Modify personal details.
2.   View Personal Rental History.
3.   View Detail of Cars to be Rented Out.
4.   Select and Book a car for a specific duration.
5.   Log out.
Enter your option:
```

Figure 4.2(e): Exiting Function

To view personal rental history:

```
Your options are:
1.   Modify personal details.
2.   View Personal Rental History.
3.   View Detail of Cars to be Rented Out.
4.   Select and Book a car for a specific duration.
5.   Log out.
Enter your option: 2

Username        Car Plate     Start Date      End Date      Daily Rate (RM)      Pay Date     Return Status
limjiayong        SMB6338      2021-05-25     2021-05-30        900.00          2021-04-20       True
limjiayong        WOW8888      2021-05-31     2021-06-06        960.00          2021-05-24       True
limjiayong        SAM5517      2021-06-04     2021-06-08        600.00          2021-06-03       False
Press any key to continue...
```

Figure 4.2(f): Customer's Rental History

To view details of cars to be rented out:

```
Your options are:
1.   Modify personal details.
2.   View Personal Rental History.
3.   View Detail of Cars to be Rented Out.
4.   Select and Book a car for a specific duration.
5.   Log out.
Enter your option: 3

Available cars for rent:
Number Plate      Model Name     Number of Seats     Rental Price (RM)
JGK9992           HONDA BR-V              7                150.00
SMB6338           HONDA CIVIC             5                180.00
ALM7294           PERODUA MYVI            5                105.00
WUA1366           PERODUA ALZA            7                140.00
WHY8823           PERODUA BEZZA           5                110.00
WAW8888           TOYOTA RUSH             7                160.00
WOW8888           TOYOTA RUSH             7                160.00
VAT1388           BMW 330E                5                300.00

Press any key to continue...
```

Figure 4.2(g): Displaying Available Cars

To book a car:

```
Your options are:
1.   Modify personal details.
2.   View Personal Rental History.
3.   View Detail of Cars to be Rented Out.
4.   Select and Book a car for a specific duration.
5.   Log out.
Enter your option: 4

Available cars for rent:
Number Plate      Model Name     Number of Seats     Rental Price (RM)
JGK9992           HONDA BR-V              7                150.00
SMB6338           HONDA CIVIC             5                180.00
ALM7294           PERODUA MYVI            5                105.00
WUA1366           PERODUA ALZA            7                140.00
WHY8823           PERODUA BEZZA           5                110.00
WAW8888           TOYOTA RUSH             7                160.00
WOW8888           TOYOTA RUSH             7                160.00
VAT1388           BMW 330E                5                300.00

Press any key to continue...

Enter the number plate of the car that you wish to book: JGK9992
Enter the start date of your rental. Must be after today and within 7 days from today [e.g. 2021-06-03]: 2021-06-13
Enter the ending date of your rental  [e.g. 2021-06-03]: 2021-06-14
limjiayong, you will be booking JGK9992 from 2021-06-13 to 2021-06-14 with the price of RM150.00.
Do you agree to this booking? ['0'= No, '1'= Yes]: 1

Booking successful, thank you limjiayong.
Press any key to continue...
```

bookings - Notepad

File Edit Format View Help

```
limjiayong, SMB6338, 2021-05-25, 2021-05-30, 900.00, 2021-04-20, True
harrypotter, ALM7294, 2021-05-25, 2021-06-02, 840.00, 2021-05-24, True
leanneooi, WC6666W, 2021-05-25, 2021-05-25, 530.00, 2021-05-25, False
limjiayong, WOW8888, 2021-05-31, 2021-06-06, 960.00, 2021-05-24, True
limjiayong, SAM5517, 2021-06-04, 2021-06-08, 600.00, 2021-06-03, False
limjiayong, JGK9992, 2021-06-13, 2021-06-14, 150.00, 2021-06-12, False
```

Figure 4.2(h): Option 4 - Booking a Car and Modification in 'bookings.txt' File

To log out of account:

```
Your options are:
1.   Modify personal details.
2.   View Personal Rental History.
3.   View Detail of Cars to be Rented Out.
4.   Select and Book a car for a specific duration.
5.   Log out.
Enter your option: 5

Do you want to continue? ['0' to Continue '-1' to Terminate]:
```

Figure 4.2(i): Option 5 - Log out from Registered Customer

4.3 Admin View

To log in as admin:

```
Your options are:
1.      Register as new customer.
2.      Log in as customer.
3.      Log in as admin.
4.      Remain as unregistered customer.
Enter your option: 3

Please fill in your credentials. Note: You are currently trying to login as admin.
Enter Username: adminjiayong
Enter Password: admin123
Welcome back, adminjiayong.

Press any key to continue...
```

Figure 4.3(a): Admin Logging In

To add cars:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 1

Please fill in the information below to add a new car.
Enter Car Number Plate: AKL9980
Enter Car Model: HONDA JAZZ
How many person can this car be seated? 5
Enter Rental Rate (per day basis): RM30
Car model HONDA JAZZ (AKL9980) with 5 seaters and a daily rental rate of RM30.00 will be added.
Do you wish to add this car? ['0'= No, '1'= Yes]: 1

Car with the number plate AKL9980 succesfully added.

Press any key to continue...
```

cars - Notepad

File Edit Format View Help

```
JGK9992, HONDA BR-V, 7, 150.00
SAM5517, HONDA CITY, 5, 150.00
SMB6338, HONDA CIVIC, 5, 180.00
WC6666W, BMW MINI 3DOOR, 4, 265.00
ALM7294, PERODUA MYVI, 5, 105.00
WUA1366, PERODUA ALZA, 7, 140.00
WHY8823, PERODUA BEZZA, 5, 110.00
WAW8888, TOYOTA RUSH, 7, 160.00
WOW8888, TOYOTA RUSH, 7, 160.00
VAT1388, BMW 330E, 5, 300.00
AKL9980, HONDA JAZZ, 5, 30.00
```

Figure 4.3(b): Option 1 - Adding New Car and Addition in 'cars.txt' File

To modify car details:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 2

All cars for rent:
Number Plate     Model Name    Number of Seats    Rental Price (RM)
JGK9992          HONDA BR-V             7              150.00
SAM5517          HONDA CITY             5              150.00
SMB6338          HONDA CIVIC            5              180.00
WC6666W          BMW MINI 3DOOR         4              265.00
ALM7294          PERODUA MYVI           5              105.00
WUA1366          PERODUA ALZA           7              140.00
WHY8823          PERODUA BEZZA          5              110.00
WAW8888          TOYOTA RUSH            7              160.00
WOW8888          TOYOTA RUSH            7              160.00
VAT1388          BMW 330E               5              300.00
AKL9980          HONDA JAZZ             5              30.00

Press any key to continue...
Enter the car model you wish to modify: TOYOTA RUSH
Enter the new daily rental rate (RM): 180
The daily rental rate for all models of TOYOTA RUSH will be modified from RM160.00 to RM180.00.
Do you agree to this modification? ['0'= No, '1'= Yes]: 1

You have successfully modified the daily rental rate for all models of TOYOTA RUSH.
2 record(s) affected.
Press any key to continue...
```

cars - Notepad

File Edit Format View Help

```
JGK9992, HONDA BR-V, 7, 150.00
SAM5517, HONDA CITY, 5, 150.00
SMB6338, HONDA CIVIC, 5, 180.00
WC6666W, BMW MINI 3DOOR, 4, 265.00
ALM7294, PERODUA MYVI, 5, 105.00
WUA1366, PERODUA ALZA, 7, 140.00
WHY8823, PERODUA BEZZA, 5, 110.00
WAW8888, TOYOTA RUSH, 7, 180.00
WOW8888, TOYOTA RUSH, 7, 180.00
VAT1388, BMW 330E, 5, 300.00
AKL9980, HONDA JAZZ, 5, 30.00
```

Figure 4.3(c): Option 2- Modify Toyota Rush's Rental Rate and Modification in 'cars.txt' File

To display details of cars rented out:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 3

Unavailable cars for rent:
Number Plate     Model Name    Number of Seats    Rental Price (RM)
JGK9992          HONDA BR-V             7              150.00
SAM5517          HONDA CITY             5              150.00
WC6666W          BMW MINI 3DOOR         4              265.00

Press any key to continue...
```

Figure 4.3(d): Displaying Rented Cars

To display details of cars available for rent:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 4

Available cars for rent:
Number Plate     Model Name     Number of Seats     Rental Price (RM)
SMB6338          HONDA CIVIC              5                180.00
ALM7294          PERODUA MYVI             5                105.00
WUA1366          PERODUA ALZA             7                140.00
WHY8823          PERODUA BEZZA            5                110.00
WAW8888          TOYOTA RUSH              7                180.00
WOW8888          TOYOTA RUSH              7                180.00
VAT1388          BMW 330E                 5                300.00
AKL9980          HONDA JAZZ               5                30.00

Press any key to continue...
```

Figure 4,3(e): Option 4- Display Available Cars

To display details of customer bookings:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 5

Username        Car Plate       Start Date      End Date        Daily Rate      Pay Date        Return Status
limjiayong      SMB6338         2021-05-25      2021-05-30      900.00          2021-04-20      True
harrypotter     ALM7294         2021-05-25      2021-06-02      840.00          2021-05-24      True
leanneooi       WC6666W         2021-05-25      2021-05-25      530.00          2021-05-25      False
limjiayong      WOW8888         2021-05-31      2021-06-06      960.00          2021-05-24      True
limjiayong      SAM5517         2021-06-04      2021-06-08      600.00          2021-06-03      True
limjiayong      JGK9992         2021-06-13      2021-06-14      150.00          2021-06-12      False
LennonLim       SAM5517         2021-06-15      2021-06-17      300.00          2021-06-14      False
Christian       SMB6338         2021-06-15      2021-06-18      540.00          2021-06-14      False
AmandaChong     WOW8888         2021-06-15      2021-06-16      180.00          2021-06-14      False
harrypotter     VAT1388         2021-06-17      2021-06-19      600.00          2021-06-14      False
Press any key to continue...
```

Figure 4.3(f): Option 5- Display Customer Bookings Details

To display all details of customer payment for a specific time duration:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 6

Fill in details below to search for specific record of customer payment.
Enter Start Date [e.g. 2021-05-31]: 2021-06-04
Enter End Date [e.g. 2021-05-31]: 2021-06-12

1. limjiayong paid RM 150.00 for car plate - JGK9992 in 2021-06-12.
Total revenue generated from 2021-06-04 to 2021-06-12: RM150.00

Press any key to continue...
```

Figure 4.3(g): Option 6- Search Customer Payment in a Particular Duration

To search specific customer booking record(s):

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 7

Fill in details below to search for specific record of customer booking.
Enter customer's username: limjiayong
Enter car plate booked by limjiayong [Leave this field empty if you wish to search for all records associated with limjiayong]: SMB6338

1. limjiayong booked for car plate - SMB6338 from 2021-05-25 to 2021-05-30, return status = True.
Press any key to continue...
```

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 7

Fill in details below to search for specific record of customer booking.
Enter customer's username: limjiayong
Enter car plate booked by limjiayong [Leave this field empty if you wish to search for all records associated with limjiayong]:

1. limjiayong booked for car with number plate - SMB6338 from 2021-05-25 to 2021-05-30, return status = True.
2. limjiayong booked for car with number plate - WOW8888 from 2021-05-31 to 2021-06-06, return status = True.
3. limjiayong booked for car with number plate - SAM5517 from 2021-06-04 to 2021-06-08, return status = True.
4. limjiayong booked for car with number plate - JGK9992 from 2021-06-13 to 2021-06-14, return status = False.
Press any key to continue...
```

Figure 4.3(h): Searching for Specific Booking Record(s) with and without Car Plate

To search for specific record(s) on customer payment:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 8

Fill in details below to search for specific record of customer payment.
Enter customer's username: limjiayong
Enter car plate booked by limjiayong [Leave this field empty if you wish to search for all records associated with limjiayong]: SAM5517

1. limjiayong paid RM 600.00.00 for car plate - SAM5517 in 2021-06-03.
Press any key to continue...
```

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 8

Fill in details below to search for specific record of customer payment.
Enter customer's username: limjiayong
Enter car plate booked by limjiayong [Leave this field empty if you wish to search for all records associated with limjiayong]:

1. limjiayong paid RM 900.00.00 for car plate - SMB6338 in 2021-04-20.
2. limjiayong paid RM 960.00.00 for car plate - WOW8888 in 2021-05-24.
3. limjiayong paid RM 600.00.00 for car plate - SAM5517 in 2021-06-03.
4. limjiayong paid RM 150.00.00 for car plate - JGK9992 in 2021-06-12.
Press any key to continue...
```

Figure 4.3(i): Searching for Specific Payment Record(s) with and without Car Plate

To return a car:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 9

Enter car plate to return: SAM5517

limjiayong booked for car plate - SAM5517 from 2021-06-04 to 2021-06-08.
Do you wish to return this car? ['0'= No, '1'= Yes]: 1

Car with number plate - SAM5517 has been returned.
Press any key to continue...
```

bookings - Notepad

File Edit Format View Help

```
limjiayong, SMB6338, 2021-05-25, 2021-05-30, 900.00, 2021-04-20, True
harrypotter, ALM7294, 2021-05-25, 2021-06-02, 840.00, 2021-05-24, True
leanneooi, WC6666W, 2021-05-25, 2021-05-25, 530.00, 2021-05-25, False
limjiayong, WOW8888, 2021-05-31, 2021-06-06, 960.00, 2021-05-24, True
limjiayong, SAM5517, 2021-06-04, 2021-06-08, 600.00, 2021-06-03, True
limjiayong, JGK9992, 2021-06-13, 2021-06-14, 150.00, 2021-06-12, False
```

Figure 4.3(m): Option 9- Return Car and Update in 'bookings.txt' File

To log out of admin account:

```
Your options are:
1.      Add cars to be rented out.
2.      Modify car details.
3.      Display all details of cars rented out.
4.      Display all details of cars available for rent.
5.      Display all details of customer bookings.
6.      Display all details of customer payment for a specific time duration.
7.      Search specific record of customer booking.
8.      Search specific record of customer payment.
9.      Return a rented car.
10.     Log out.
Enter your option: 10

Successfully logged out.
Do you want to continue? ['0' to Continue '-1' to Terminate]:
```

Figure 4.3(o): Option 10- Logging Out from Admin

4.4 Unregistered Customer View

To enter unregistered customer view:

```
Your options are:
1.      Register as new customer.
2.      Log in as customer.
3.      Log in as admin.
4.      Remain as unregistered customer.
Enter your option: 4

Available cars for rent:
Number Plate     Model Name    Number of Seats    Rental Price (RM)
SAM5517          HONDA CITY              5              150.00
SMB6338          HONDA CIVIC             5              180.00
ALM7294          PERODUA MYVI            5              105.00
WUA1366          PERODUA ALZA            7              140.00
WHY8823          PERODUA BEZZA           5              110.00
WAW8888          TOYOTA RUSH             7              180.00
WOW8888          TOYOTA RUSH             7              180.00
VAT1388          BMW 330E                5              300.00
AKL9980          HONDA JAZZ              5              30.00

Press any key to continue...
```

Figure 4.4(a): Remaining as Unregistered Customer

To terminate the program:

```
Do you want to continue? ['0' to Continue '-1' to Terminate]: -1

Thank you and have a nice day!
```

Figure 4.4(b): Terminating the Program

5.0 Conclusion

In conclusion, the Super Car Rental Services (SCRS) Python program has successfully been created with the help of designs such as pseudocodes and flowcharts. The Python program is also equipped with suitable functions and helper functions to meet the company's need of having different features for three various users. Hence, this Python program will definitely give SCRS a strong boost in their business, and it will help them to portray a positive image to customers as a brand that is always determined to improve and move along with time.

References

S., T., 2018. *Why You Need to Automate Your Business Right Now.* [Online]
Available at: https://medium.com/swlh/why-you-need-to-automate-your-business-right-now-9b4039aba484
[Accessed 2 June 2021].