```java
import java.time.YearMonth;

public class Person {

    protected String Name;
    protected String Address;
    protected String IC;
    protected String Phone;
    protected int Age;
    public Person(){};

    public Person(String name, String address, String ic, String phone){
        this.Name = name;
        this.Address = address;
        this.IC = ic;
        this.Phone = phone;
        this.Age = 0;
    }

    public void setPersonAge(int age){this.Age = age;}

    public String getPersonName(){return this.Name;}
    public String getPersonIC(){return this.IC;}
    public int getPersonAge(){return this.Age;}

    public int calcAge(){                //declare age
            int age;
            //get first 2 digit of ic
            String yearBorn = this.getPersonIC().substring(0,2);
            int yearIC = Integer.parseInt(yearBorn);
            //get the year born
            if(yearIC >= 0 && yearIC <= 23 ){
                yearIC = 2000 + yearIC;
            }
            else {
                yearIC = 1900 + yearIC;
            }
            //calculate the age by subtracting from current year
            age = YearMonth.now().getYear()-yearIC;
            //returning the age value
            setPersonAge(age);
            return age;

    }
}
```

```
public class Salesperson extends Person {

    private String SPID;
    private double SPcommissionRate;
    private double CommissionEarned;
    private Bill[] SPBill;
    private int numBill;
    private Customer[] servedCustomer;
    private int numCustomer;

    // constructor
    public Salesperson(){};
    public Salesperson(String name,String address,String ic, String phone, String SPID ,
    double commissionRate) {
        super(name, address, ic, phone);
        this.SPID = SPID;
        this.SPcommissionRate = commissionRate;
        this.CommissionEarned = 0;
        this.numBill = 0;
        this.numCustomer = 0;
        this.SPBill = new Bill[100];
        this.servedCustomer = new Customer[100];
    }
    // setter
    public void setSalespersonData(String name,String address,String ic, String phone,
    String SPID , double commissionRate){
        Name = name;
        Address = address;
        IC = ic;
        Phone = phone;
        this.SPID = SPID;
        this.SPcommissionRate = commissionRate;
        this.CommissionEarned = 0;
        this.numBill = 0;
        this.numCustomer = 0;
        this.SPBill = new Bill[100];
        this.servedCustomer = new Customer[100];
    }

    public String getSPID(){return this.SPID;}
    public double getSPcommissionRate(){return this.SPcommissionRate;}


    // processor
       public void addBil(Bill bill){
        if( numBill < SPBill.length){
            SPBill[numBill] = bill;
            numBill++;
        }
    }
        public void addCustomer(Customer customer){
            if(numCustomer < servedCustomer.length){
                servedCustomer[numCustomer] = customer;
                numCustomer++;
            }
        }

        public void calcCommission(double earn){
            CommissionEarned += earn;
        }
    // printer
    public void PrintSalesPersonCommission(){
        System.out.println("+---------------------------------+");
        System.out.println("| Staff ID        : " + SPID);
        System.out.println("| Name            : " + Name);
        System.out.println("|");
        System.out.println("| Commission rate : " + (getSPcommissionRate()*100) + "%") ;
        System.out.println("| Commission      : RM" + CommissionEarned) ;
        System.out.println("| Customer Served : " + numCustomer);
        System.out.println("+---------------------------------+");
    }

    public void PrintSalesPersonInfo(){
                System.out.println("+---------------------------------+");
                System.out.println("| Name            : " + Name);
                System.out.println("| Address         : " + Address);
                System.out.println("| Phone           : " + Phone);
              System.out.println("|");
                System.out.println("| Staff IC        : " + IC);
                System.out.println("| Staff ID        : " + SPID);
                System.out.println("| Commission rate : " + (getSPcommissionRate()*100)
                + "%") ;
                System.out.println("| Commission      : RM" + CommissionEarned) ;
                System.out.println("| Customer Served : " + numCustomer);
                System.out.println("+---------------------------------+");
    }
}
```

```java

public class Customer extends Person {

    private String password;
    private int CustnumPurchasedCars;
    private int numBill;
    private Car[] CustpurchasedCars;
    private Bill[] custBill;

    // constructor
    public Customer(){};

    public Customer(String CustName, String CustAddress, String CustIC, String
    CustPhone, String Password) {
        super(CustName, CustAddress, CustIC, CustPhone);
        this.password = Password;
        this.numBill = 0;
        this.CustnumPurchasedCars = 0;
        this.password = null;
    }

    // mutator
       public void setCustPassword(String Passowrd){
        this.password = Passowrd;
    }
    public void setCustomerData(String CustName, String CustAddress, String CustIC,
    String CustPhone,String Password) {
        this.Name = CustName;
        this.Address = CustAddress;
        this.IC = CustIC;
        this.Phone = CustPhone;
        this.password = Password;
        this.CustpurchasedCars = new Car[100];
        this.custBill = new Bill[100];
        this.numBill = 0;
        this.CustnumPurchasedCars = 0;
        this.Age = calcAge();
    }
    public String getCustPassword(){return this.password;}
    public Bill[] getCustBill(){return custBill;}
    public int getCustNumBill(){return numBill;}
    public Car[] getCustPurchasedCars() {return CustpurchasedCars;}

    // prcoessor
    public void addPurchasedCar(Car car){
        if( CustnumPurchasedCars < CustpurchasedCars.length){
            CustpurchasedCars[CustnumPurchasedCars] = car;
            CustnumPurchasedCars++;
        }
    }

    public void addBil(Bill bill){
        if( numBill < custBill.length){
            custBill[numBill] = bill;
            numBill++;
        }
    }

     public void removeBill(Bill bill,int index){
         for (int i = 0; i < custBill.length; i++) {
            if (custBill[i] == custBill[index]) {
                index = i;
                break;
            }
        }
            for(int i = index;i < custBill.length - 1 ;i++){
                custBill[i] = custBill[i + 1];
        }
        Bill[] newCustBill = new Bill[custBill.length - 1];
        System.arraycopy(custBill, 0, newCustBill, 0, newCustBill.length);

        custBill = newCustBill;

        numBill--;
    }
    // printer
    public String toString(){
        String info = "+---------------------------------+"+
                      "\n| Customer Name           : " + Name.toUpperCase() +
                      "\n| Customer IC             : " + IC +
                      "\n| Customer age            : " + calcAge() +
                      "\n| Customer Address        : " + Address.toUpperCase() +
                      "\n| Customer Phone number   : " + Phone +
                      "\n| Number of Car Purchased : " + CustnumPurchasedCars + " cars";
        return info;
    }
}

```

```
import java.text.DecimalFormat;

public class Bill {

    private String DateGenerated;

    private Salesperson SalesStaff;
    private Car SelectedCar;
    private Customer Customer;
    private int lengthOfLoan;
    private double totalPayment;
    private double monthlyPayment;
    private double paymentLeft;
    private double carDownpayment;

    DecimalFormat decimalFormat = new DecimalFormat("#.##");

    // default constructor
    public Bill(){}

    // normal constructor;
    public Bill(String Date , Salesperson salesStaff, Car selectCar, Customer
    customer,int YearPayment){
        this.DateGenerated = Date;
        this. SalesStaff = new Salesperson();
        this.SelectedCar = new Car();
        this.Customer = new Customer();
        this.lengthOfLoan = YearPayment;

    }

    // setter
    public void setTotalPayment(double totalPayment){
        this.totalPayment = totalPayment;
    }
    public void setMonthlyPayment(double MP){
        this.monthlyPayment = MP;
    }

    public void setBill(String Date , Salesperson salesperson, Car selectCar, Customer
    customer,int YearPayment){
        this.DateGenerated = Date;
        this. SalesStaff = salesperson;
        this.SelectedCar = selectCar;
        this.Customer = customer;
        this.lengthOfLoan = YearPayment;
        this.paymentLeft = payBill(0);
        this.totalPayment = calcTotalPayment(selectCar);
        this.carDownpayment = calcDownpayment(selectCar);
        this.monthlyPayment = calcMonthlyPayment();

    }

    // processor
    public int calcMonthOfPayment(){
        int month = 0;

        month = lengthOfLoan * 12;

        return month;
    }

    public double calcTotalPayment(Car car){
        double TotalPayment = 0.00;

        TotalPayment = car.calcCarPrice();

        setTotalPayment(TotalPayment);

```

```java
        return TotalPayment;
    }
    public double calcMonthlyPayment(){
        double MPayment = 0.00;

        MPayment = ((paymentLeft - carDownpayment ) / calcMonthOfPayment());

        setMonthlyPayment(MPayment);

        return MPayment;
    }
    public double payBill(double totalToPay){
        paymentLeft = paymentLeft - totalToPay;

        if(paymentLeft < 0){
                    System.out.println("+---------------------------------+");
                    System.out.println("| Balance of RM"+Math.abs(paymentLeft) + " will be returned.");
                    System.out.println("+---------------------------------+");

        }

        return totalPayment;

    }
    public double calcDownpayment(Car car){
        double downpayment = 0.00;

        downpayment = .1 * car.calcCarPrice();

        paymentLeft = paymentLeft-carDownpayment;

        return downpayment;
    }
    // getter
    public Salesperson getsalesPerson(){return this.SalesStaff;}
    public Car getSelectedCar(){return this.SelectedCar;}
    public Customer getCustomer(){return this.Customer;}
    public double getPaymentLeft(){return this.paymentLeft;}

    // printer

    public String SimplifiedBill(){
        String str = "+---------------------------------+" +
                   "\n|               BILL              |" +
                   "\n+---------------------------------+" +
                   "\n+  SALESPERSON NAME  : "+ getsalesPerson().getPersonName() +
                   "\n+  CAR MODEL         : "+ getSelectedCar().getCarModel() +
                   "\n+  CAR BRAND         : "+ getSelectedCar().getCarBrand() +
                   "\n|                                  " +
                   "\n|  PAYMENT LEFT DUE  : RM"+ paymentLeft +
                   "\n+---------------------------------+";
        return str;
    }

    public String toString(){
      String str = "+---------------------------------+" +
                   "\n|               BILL              |" +
                   "\n+---------------------------------+" +
                   "\n+  DATE GENERATED    : "+ DateGenerated +
                   "\n|                                  " +
                   "\n+  SALESPERSON NAME  : "+ getsalesPerson().getPersonName() +
                   "\n+  CUSTOMER NAME     : "+ getCustomer().getPersonName() +
                   "\n|                                  " +
                   "\n+  CAR MODEL         : "+ getSelectedCar().getCarModel() +
                   "\n+  CAR BRAND         : "+ getSelectedCar().getCarBrand() +
                   "\n|                                  " +
                   "\n|  TOTAL PAYMENT     : RM"+ totalPayment   +
                   "\n|                                  " +
                   "\n|  LOAN YEARS TIME   : "+lengthOfLoan+" years" +
                   "\n|  LOAN MONTHLY TIME : "+calcMonthOfPayment()+" month"+
                   "\n|                                  " +
                   "\n|  CAR DOWNPAYMENT   : RM"+carDownpayment+
                   "\n|  PAYMENT PER MONTH : RM"+monthlyPayment+
                   "\n|                                  " +
                   "\n|  PAYMENT LEFT DUE  : RM"+ paymentLeft +
                   "\n+---------------------------------+";
        return str;
    }
}
```

```java
public class Car {

    private String carBrand;
    private String carModel;
    private String carType;
    private double carPrice;

    // constructor
    public Car(){}
    public Car(String carBrand, String carModel, String carType, double carPrice) {
        this.carBrand = carBrand;
        this.carModel = carModel;
        this.carType = carType;
        this.carPrice = carPrice;
    }

    // getter
    public String getCarBrand(){return this.carBrand;}
    public String getCarModel(){return this.carModel;}
    public double getCarPrice(){return this.carPrice;}

    // processor
    public double calcCarPrice(){

        double price = 0.00;

        price = getCarPrice() + (getCarPrice() * .15);

        return price;
    }
    // printer
    @Override
    public String toString(){
        String info = "+---------------------------------+" +
                    "\n| Car Brand         : " + carBrand +
                    "\n| Car Model         : " + carModel +
                    "\n| Car Type          : " + carType +
                    "\n|" +
                    "\n| Car Price         : RM" + calcCarPrice() +
                    "\n+---------------------------------+";
        return info;
    }

    public String WriteString(){
        String write = carBrand + ";" + carModel + ";" + carType + ";" + carPrice;
        return write;
    }

}
```

```java
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class Dealership {

    private Salesperson[] SalesStaff;
    private Car[] CarInventory;
    private Customer[] CustomerData;
    private Bill[] bills;
    private int numSalesStaff;
    private int numCustomer;
    private int numCars;
    private int numBills;
    private Double TotalProfit;
    private Double TotalPaidCommission;
    private double ProfitRate;

    // constructor
    public Dealership(String DealershipName){
        this.SalesStaff = new Salesperson[4];  //4 is max number of staff
        this.CarInventory = new Car[100]; //100 car is limit
        this.CustomerData = new Customer[200]; //200 customer per system
        this.bills = new Bill[500]; //500 bill is limit per system run
        this.numSalesStaff = 0; //start with 0 staff
        this.numCustomer = 0;   //start with 0 customer
        this.numCars = 0; // start w    ith 0 car
        this.numBills = 0;
        this.TotalProfit = 0.00; // start with 0 profit
        this.TotalPaidCommission = 0.00; // start with 0 paid commission
        this.ProfitRate = 0.15; //15% profit rate
    }
    // getter
    public int getNumSalesStaff(){ return this.numSalesStaff;}
    public int getNumCustomer() { return this.numCustomer; }
    public int getNumCars() { return this.numCars; }
    public int getNumbills() { return this.numBills; }
    public double getProfitRate(){return this.ProfitRate;}
    public double getTotalProfit(){return this.TotalProfit;}
    public double getTotalPaidCommission(){return this.TotalPaidCommission;}
    public Salesperson[] getSalesStaff() {return this.SalesStaff;}
    public Car[] getCarInvetory(){return this.CarInventory;}
    public Customer[] getCustomerData(){return this.CustomerData;}

    // adding customer to database
    public void addCustomer(Customer customer){
                if(numCustomer < CustomerData.length){
                CustomerData[numCustomer] = customer;
            numCustomer++;
        }
        else{
            System.out.println("Cannot add more Customer. Limit is reached");
        }
    }
    public void addBill(Bill bill){
        if(numBills < bills.length){
            bills[numBills] = bill;
            numBills++;
        }
                else{
            System.out.println("Cannot add more bill. Limit is reached");
        }
    }
    // hiring staff
    public void hireSaleStaff(Salesperson salesperson){
        if(numSalesStaff < SalesStaff.length){
            SalesStaff[numSalesStaff] = salesperson;
            numSalesStaff++;
        }
        else{
            System.out.println("Cannot add more staff. Limit is reached");
        }
    }
     public void removeCarFromInventory(Car car,int index){
         for (int i = 0; i < CarInventory.length; i++) {
            if (CarInventory[i] == CarInventory[index]) {
                index = i;
                break;
            }
        }
            for(int i = index;i < CarInventory.length - 1 ;i++){
                CarInventory[i] = CarInventory[i + 1];
        }
        Car[] newCarArray = new Car[CarInventory.length - 1];
        System.arraycopy(CarInventory, 0, newCarArray, 0, newCarArray.length);
        CarInventory = newCarArray;

        numCars--;
    }
    public void addCarToInventory(Car car){
                if(numCars < CarInventory.length){
                    CarInventory[numCars] = car;
                    numCars++;
            }
               else{
                 System.out.println("Cannot add more car.Inventory is full.");
                }
    }
    public String GenerateDate(){
        String date = "";
        LocalDate currentDate = LocalDate.now();
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
        date = currentDate.format(formatter);

        return date;
    }
    // calculating profit per car
    public double calcProfit(Car car){
        double profit = 0.00;
        profit = car.calcCarPrice() - car.getCarPrice();
        return profit;
    }
    // calculation commission per car that goes to salesPerson
    public double calcCommission(Car car, Salesperson salesperson){
        double Commission = 0.00;
        Commission = (calcProfit(car) * salesperson.getSPcommissionRate()) ;
        return Commission;
    }
    // calculating profit per car that goes to dealership
    public double calcProfitToDealership(Car car, Salesperson salesperson){
        double profit = 0.00;
        profit = calcProfit(car) - calcCommission(car, salesperson);
        return profit;
    }
    // calculating the total profit earn to Dealership;
    public double calcTotalProfit(Car car, Salesperson salesperson){
        TotalProfit += calcProfitToDealership(car, salesperson);
        return TotalProfit;
    }
    // calculating totalCommission paid to SP;
    public double calcTotalPaidCommission(Car car, Salesperson salesperson){
        TotalPaidCommission += calcCommission(car, salesperson);
        return TotalPaidCommission;
    }
    public void sellCar(int index,Car car, Salesperson salesperson, Customer customer,Bill bills){
        calcCommission(car, salesperson);
        addBill(bills);
        calcProfit(car);
        calcTotalProfit(car,salesperson);
        calcTotalPaidCommission(car, salesperson);
        removeCarFromInventory(car, index);
        salesperson.calcCommission(calcCommission(car, salesperson));
        System.out.println("+----------------------------------+");
         System.out.println("| Car sold to " + customer.getPersonName() + " by " + salesperson.getPersonName() +
         " for RM" + car.calcCarPrice());
         System.out.println("+ Revenue Gained : RM" + calcProfit(car));
         System.out.println("+ Commission to " + salesperson.getPersonName()+"    : RM" + calcCommission(car, salesperson));
         System.out.println("| profit gain    : RM" + (calcProfit(car)-calcCommission(car, salesperson)));
         System.out.println("+----------------------------------+");
         System.out.println(bills.toString());
    }
    public void printStaffCommission(Salesperson salesStaff[]){
        for(int i = 0;i<salesStaff.length;i++){
            salesStaff[i].PrintSalesPersonCommission();
        }
        System.out.println("+----------------------------------+");
         System.out.println("| Total Commission paid to Staff: RM" + getTotalPaidCommission());
        System.out.println("+----------------------------------+");
    }
    public void printStaffInformation(Salesperson salesStaff[]){
        for(int i = 0;i<salesStaff.length;i++){
            salesStaff[i].PrintSalesPersonInfo();
        }
    }
    public void PrintCarList(Car car[]){

        for(int i=0;i<numCars;i++){
                if(car[i] != null){
                    System.out.println("| Car Number : " + (i+1));
                    System.out.println(CarInventory[i]);
                }

    }
        if(numCars == 0 ){
        System.out.println("+----------------------------------+");
                System.out.println("|         UNFORTUNATELY            |");
                System.out.println("|     WE HAVE RAN OUT OF CAR       |");
                System.out.println("|     PLEASE COME AGAIN SOON       |");
                System.out.println("+----------------------------------+");
                }
        }
    public void printMainMenu(){
        System.out.println("+----------------------------------+");
        System.out.println("|  WELCOME TO KAMAL CAR SOLUTION  |");
        System.out.println("+----------------------------------+");
        System.out.println("|          [C] = Customer          |");
        System.out.println("+                                  +");
        System.out.println("|          [A] = Admin             |");
        System.out.println("+                                  +");
        System.out.println("|          [E] = exit              |");
        System.out.println("+----------------------------------+");
    }
    public void printCustomerMenu(){
            System.out.println("+----------------------------------+");
             System.out.println("|               MENU               |");
             System.out.println("+----------------------------------+");
             System.out.println("|    [C] = Choose a car to buy     |");
             System.out.println("|    [I] = Car Invenotry           |");
             System.out.println("|    [B] = BIll sectcion           |");
             System.out.println("|    [U] = User Profile            |");
             System.out.println("|    [L] = Log Out                 |");
             System.out.println("+----------------------------------+");
    }
```

```java
    public void PrintStaffMainMenu(){
                       // start of admin page
        System.out.println("+----------------------------------+");
        System.out.println("|           STAFF PAGE             |");
        System.out.println("+----------------------------------+");
        System.out.println("|     [P] = Total Profit           |");
        System.out.println("|     [I] = Import Data            |");
        System.out.println("+     [S] = Staff Management       +");
        System.out.println("|     [C] = Car Management         |");
        System.out.println("|     [A] = System Data            |");
        System.out.println("|     [M] = Main Menu              |");
        System.out.println("+----------------------------------+");
    }
    public void PrintStaffManagement(){
                       // start of admin page
        System.out.println("+----------------------------------+");
        System.out.println("|     STAFF MANAGEMENT PAGE        |");
        System.out.println("+----------------------------------+");
        System.out.println("|     [C] = staff Commission       |");
        System.out.println("|     [I] = Staff information      |");
        System.out.println("|     [M] = Main Menu              |");
        System.out.println("+----------------------------------+");
    }
    public void PrintCarManagement(){
                       // start of admin page
        System.out.println("+----------------------------------+");
        System.out.println("|       CAR MANAGEMENT PAGE        |");
        System.out.println("+----------------------------------+");
        System.out.println("|     [A] = Add New Car            |");
        System.out.println("|     [R] = Remove Car             |");
        System.out.println("|     [I] = Car Inventory          |");
        System.out.println("|     [M] = Main Menu              |");
        System.out.println("+----------------------------------+");
    }
    public void PrintTotalProfit(){
            System.out.println("+----------------------------------+");
            System.out.println("|          TOTAL PROFIT            |");
            System.out.println("+----------------------------------+");
            System.out.println("| Profit Percentage : " + (getProfitRate()*100) + "%");
            System.out.println("| Total Profit  : RM" + getTotalProfit());
            System.out.println("+----------------------------------+");
    }
    public void printIOmenu(){
        System.out.println("+----------------------------------+");
        System.out.println("|              IMPORT              |");
        System.out.println("+----------------------------------+");
        System.out.println("|           [I] IMPORT             |");
        System.out.println("|           [B] BACK TO MENU       |");
        System.out.println("+----------------------------------+");
    }
    public void PrintSystemAnalytics(){
        System.out.println("+----------------------------------+");
        System.out.println("|        SYSTEM ANALYTICS          |");
        System.out.println("+----------------------------------+");
        System.out.println("|  NUMBER OF CUSTOMER            : " + numCustomer);
        System.out.println("|  NUMBER OF STAFF               : " + numSalesStaff );
        System.out.println("|  NUMBER OF CAR IN INVENTORY    : " + numCars);
        System.out.println("|  EMPTY SPACE LEFT FOR CARS     : " + (100 - numCars));
        System.out.println("|  NUMBER OF CAR SOLD            : " + numBills);
        System.out.println("|  NMBER OF BILLS GENERATED      : " + numBills);
        System.out.println("+----------------------------------+");
    }
}
```

```java
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class Dealership {

    private Salesperson[] SalesStaff;
    private Car[] CarInventory;
    private Customer[] CustomerData;
    private Bill[] bills;
    private int numSalesStaff;
    private int numCustomer;
    private int numCars;
    private int numBills;
    private Double TotalProfit;
    private Double TotalPaidCommission;
    private double ProfitRate;

    // constructor
    public Dealership(String DealershipName){
        this.SalesStaff = new Salesperson[4];  //4 is max number of staff
        this.CarInventory = new Car[100]; //100 car is limit
        this.CustomerData = new Customer[200]; //200 customer per system
        this.bills = new Bill[500]; //500 bill is limit per system run
        this.numSalesStaff = 0; //start with 0 staff
        this.numCustomer = 0;   //start with 0 customer
        this.numCars = 0; // start w    ith 0 car
        this.numBills = 0;
        this.TotalProfit = 0.00; // start with 0 profit
        this.TotalPaidCommission = 0.00; // start with 0 paid commission
        this.ProfitRate = 0.15; //15% profit rate
    }
    // getter
    public int getNumSalesStaff(){ return this.numSalesStaff;}
    public int getNumCustomer() { return this.numCustomer; }
    public int getNumCars() { return this.numCars; }
    public int getNumbills() { return this.numBills; }
    public double getProfitRate(){return this.ProfitRate;}
    public double getTotalProfit(){return this.TotalProfit;}
    public double getTotalPaidCommission(){return this.TotalPaidCommission;}
    public Salesperson[] getSalesStaff() {return this.SalesStaff;}
    public Car[] getCarInvetory(){return this.CarInventory;}
    public Customer[] getCustomerData(){return this.CustomerData;}

    // adding customer to database
    public void addCustomer(Customer customer){
                if(numCustomer < CustomerData.length){
                CustomerData[numCustomer] = customer;
            numCustomer++;
        }
        else{
            System.out.println("Cannot add more Customer. Limit is reached");
        }
    }
    public void addBill(Bill bill){
        if(numBills < bills.length){
            bills[numBills] = bill;
            numBills++;
        }
                else{
            System.out.println("Cannot add more bill. Limit is reached");
        }
    }
    // hiring staff
    public void hireSaleStaff(Salesperson salesperson){
        if(numSalesStaff < SalesStaff.length){
            SalesStaff[numSalesStaff] = salesperson;
            numSalesStaff++;
        }
        else{
            System.out.println("Cannot add more staff. Limit is reached");
        }
    }
     public void removeCarFromInventory(Car car,int index){
         for (int i = 0; i < CarInventory.length; i++) {
            if (CarInventory[i] == CarInventory[index]) {
                index = i;
                break;
            }
        }
            for(int i = index;i < CarInventory.length - 1 ;i++){
                CarInventory[i] = CarInventory[i + 1];
        }
        Car[] newCarArray = new Car[CarInventory.length - 1];
        System.arraycopy(CarInventory, 0, newCarArray, 0, newCarArray.length);
        CarInventory = newCarArray;

        numCars--;
    }
    public void addCarToInventory(Car car){
                if(numCars < CarInventory.length){
                    CarInventory[numCars] = car;
                    numCars++;
            }
               else{
                 System.out.println("Cannot add more car.Inventory is full.");
                }
    }
    public String GenerateDate(){
        String date = "";
        LocalDate currentDate = LocalDate.now();
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyy-MM-dd");
        date = currentDate.format(formatter);

        return date;
    }
    // calculating profit per car
    public double calcProfit(Car car){
        double profit = 0.00;
        profit = car.calcCarPrice() - car.getCarPrice();
        return profit;
    }
    // calculation commission per car that goes to salesPerson
    public double calcCommission(Car car, Salesperson salesperson){
        double Commission = 0.00;
        Commission = (calcProfit(car) * salesperson.getSPcommissionRate()) ;
        return Commission;
    }
    // calculating profit per car that goes to dealership
    public double calcProfitToDealership(Car car, Salesperson salesperson){
        double profit = 0.00;
        profit = calcProfit(car) - calcCommission(car, salesperson);
        return profit;
    }
    // calculating the total profit earn to Dealership;
    public double calcTotalProfit(Car car, Salesperson salesperson){
        TotalProfit += calcProfitToDealership(car, salesperson);
        return TotalProfit;
    }
    // calculating totalCommission paid to SP;
    public double calcTotalPaidCommission(Car car, Salesperson salesperson){
        TotalPaidCommission += calcCommission(car, salesperson);
        return TotalPaidCommission;
    }
    public void sellCar(int index,Car car, Salesperson salesperson, Customer customer,Bill bills){
        calcCommission(car, salesperson);
        addBill(bills);
        calcProfit(car);
        calcTotalProfit(car,salesperson);
        calcTotalPaidCommission(car, salesperson);
        removeCarFromInventory(car, index);
        salesperson.calcCommission(calcCommission(car, salesperson));
        System.out.println("+----------------------------------+");
         System.out.println("| Car sold to " + customer.getPersonName() + " by " + salesperson.getPersonName() +
         " for RM" + car.calcCarPrice());
         System.out.println("+ Revenue Gained : RM" + calcProfit(car));
         System.out.println("+ Commission to " + salesperson.getPersonName()+"    : RM" + calcCommission(car, salesperson));
         System.out.println("| profit gain    : RM" + (calcProfit(car)-calcCommission(car, salesperson)));
         System.out.println("+----------------------------------+");
         System.out.println(bills.toString());
    }
    public void printStaffCommission(Salesperson salesStaff[]){
        for(int i = 0;i<salesStaff.length;i++){
            salesStaff[i].PrintSalesPersonCommission();
        }
        System.out.println("+----------------------------------+");
         System.out.println("| Total Commission paid to Staff: RM" + getTotalPaidCommission());
        System.out.println("+----------------------------------+");
    }
    public void printStaffInformation(Salesperson salesStaff[]){
        for(int i = 0;i<salesStaff.length;i++){
            salesStaff[i].PrintSalesPersonInfo();
        }
    }
    public void PrintCarList(Car car[]){

        for(int i=0;i<numCars;i++){
                if(car[i] != null){
                    System.out.println("| Car Number : " + (i+1));
                    System.out.println(CarInventory[i]);
                }
    }
        if(numCars == 0 ){
        System.out.println("+----------------------------------+");
                System.out.println("|         UNFORTUNATELY            |");
                System.out.println("|     WE HAVE RAN OUT OF CAR       |");
                System.out.println("|     PLEASE COME AGAIN SOON       |");
                System.out.println("+----------------------------------+");
                }
        }
    public void printMainMenu(){
        System.out.println("+----------------------------------+");
        System.out.println("|  WELCOME TO KAMAL CAR SOLUTION  |");
        System.out.println("+----------------------------------+");
        System.out.println("|          [C] = Customer          |");
        System.out.println("+                                  +");
        System.out.println("|          [A] = Admin             |");
        System.out.println("+                                  +");
        System.out.println("|          [E] = exit              |");
        System.out.println("+----------------------------------+");
    }
    public void printCustomerMenu(){
            System.out.println("+----------------------------------+");
             System.out.println("|               MENU               |");
             System.out.println("+----------------------------------+");
             System.out.println("|    [C] = Choose a car to buy     |");
             System.out.println("|    [I] = Car Invenotry           |");
             System.out.println("|    [B] = BIll sectcion           |");
             System.out.println("|    [U] = User Profile            |");
             System.out.println("|    [L] = Log Out                 |");
             System.out.println("+----------------------------------+");
    }
    public void PrintStaffMainMenu(){
```

```
                       // start of admin page
        System.out.println("+---------------------------------+");
        System.out.println("|           STAFF PAGE            |");
        System.out.println("+---------------------------------+");
        System.out.println("|     [P] = Total Profit          |");
        System.out.println("|     [I] = Import Data           |");
        System.out.println("+     [S] = Staff Management      +");
        System.out.println("|     [C] = Car Management        |");
        System.out.println("|     [A] = System Data           |");
        System.out.println("|     [M] = Main Menu             |");
        System.out.println("+---------------------------------+");
    }
    public void PrintStaffManagement(){
                       // start of admin page
        System.out.println("+---------------------------------+");
        System.out.println("|     STAFF MANAGEMENT PAGE       |");
        System.out.println("+---------------------------------+");
        System.out.println("|     [C] = staff Commission      |");
        System.out.println("|     [I] = Staff information     |");
        System.out.println("|     [M] = Main Menu             |");
        System.out.println("+---------------------------------+");
    }
    public void PrintCarManagement(){
                       // start of admin page
        System.out.println("+---------------------------------+");
        System.out.println("|       CAR MANAGEMENT PAGE       |");
        System.out.println("+---------------------------------+");
        System.out.println("|     [A] = Add New Car           |");
        System.out.println("|     [R] = Remove Car            |");
        System.out.println("|     [I] = Car Inventory         |");
        System.out.println("|     [M] = Main Menu             |");
        System.out.println("+---------------------------------+");
    }
    public void PrintTotalProfit(){
            System.out.println("+---------------------------------+");
            System.out.println("|          TOTAL PROFIT           |");
            System.out.println("+---------------------------------+");
            System.out.println("| Profit Percentage : " + (getProfitRate()*100) + "%");
            System.out.println("| Total Profit  : RM" + getTotalProfit());
            System.out.println("+---------------------------------+");
    }
    public void printIOmenu(){
        System.out.println("+---------------------------------+");
        System.out.println("|              IMPORT             |");
        System.out.println("+---------------------------------+");
        System.out.println("|           [I] IMPORT            |");
        System.out.println("|           [B] BACK TO MENU      |");
        System.out.println("+---------------------------------+");
    }
    public void PrintSystemAnalytics(){
        System.out.println("+---------------------------------+");
        System.out.println("|        SYSTEM ANALYTICS         |");
        System.out.println("+---------------------------------+");
        System.out.println("|  NUMBER OF CUSTOMER           : " + numCustomer);
        System.out.println("|  NUMBER OF STAFF              : " + numSalesStaff );
        System.out.println("|  NUMBER OF CAR IN INVENTORY   : " + numCars);
        System.out.println("|  EMPTY SPACE LEFT FOR CARS    : " + (100 - numCars));
        System.out.println("|  NUMBER OF CAR SOLD           : " + numBills);
        System.out.println("|  NMBER OF BILLS GENERATED     : " + numBills);
        System.out.println("+---------------------------------+");
    }
}
import java.util.*;
import java.io.*;

public class CarDealershipApp {

    private static final String CAR_FILE = "Car.txt";
    public static int Size = 100;
```

```java
    public static void main(String[] args) throws Exception{
        // Create a new Dealership
        Dealership dealership = new Dealership("KAMAL CAR SOLUTION SDN BHD");

        Customer[] customer = new Customer[Size];
        Salesperson[] SalesStaff = new Salesperson[3];
        Car[] cars = new Car[Size];

        // add salestaff
        SalesStaff[0] = new Salesperson("SITI BINTI HALIM","NO 30, JALAN RIMAU 16, TAMAN RIMAU,43650, BANDAR BARU BANGI","650525035450","011-2345 6789","SP01", 0.1);
        SalesStaff[1] = new Salesperson("AHMAD BIN MUHAMMAD","NO 25, JALAN EMPAYAN 4,TAMAN EMPAYAN,43650, BANDAR BARU BANGI","650525035450","011-2345 6789","SP02", 0.15);
        SalesStaff[2] = new Salesperson("ABU BIN KASIM","NO 1, JALAN EMPAYAN 20, TAMAN EMPAYAN,43650, BANDAR BARU BANGI","650525035450","012-4363 5764","SP03", 0.12);

        for(int i = 0; i < SalesStaff.length; i++){
            dealership.hireSaleStaff(SalesStaff[i]);
        }
        // get customer data
        Scanner input = new Scanner(System.in);
        String optionTemp;
        char option;

        readFile(cars, dealership);
        do{

        dealership.printMainMenu();

        System.out.print("| Choose one to continue : ");
         optionTemp = input.next();
        option = optionTemp.toUpperCase().charAt(0);

        if(option == 'C'){
            Scanner customerin = new Scanner(System.in);
            String optionCust = "E";
            while(optionCust.toUpperCase().charAt(0) != 'M'){
             System.out.println("+----------------------------------+");
             System.out.println("|              LOGIN               |");
             System.out.println("+----------------------------------+");
             System.out.println("|    [L] = Login to account        |");
             System.out.println("|    [R] = Register an account     |");
             System.out.println("|    [M] = Back to menu            |");
             System.out.println("+----------------------------------+");
            System.out.print("| Choose : ");
             optionCust = customerin.next();
             if(optionCust.toUpperCase().charAt(0) == 'R'){
                RegisterCustomerProcess(customer, dealership);
                }
             else if(optionCust.toUpperCase().charAt(0) == 'L'){
                int attempt = 3;
                boolean success = false;
                Scanner customerLogin = new Scanner(System.in);
                if(dealership.getNumCustomer() == 0){
                System.out.println("+----------------------------------+");
                System.out.println("|    ERROR, NO CUSTOMER DATA       |");
                System.out.println("|    PLEASE CREATE AN ACCOUNT      |");
                System.out.println("+----------------------------------+");
                break;
                }
                else{
                System.out.println("+----------------------------------+");
                System.out.println("|              LOGIN               |");
                System.out.println("+----------------------------------+");
                }

                while(attempt != 0 && success == false){

```

```
            System.out.print("| IC NUMBER [without -] : ");
            String IC = customerLogin.nextLine();
            System.out.print("| PASSWORD :");
            String pass = customerLogin.nextLine();

            int indexofCustomer = 0;
    for(int j = 0 ; j<dealership.getNumCustomer();j++){

    if(customer[j].getCustPassword().equals(pass) && customer[j].getPersonIC().equals(IC)){
        String optionCust2 = "Meee";
            success = true;
            System.out.println("+---------------------------------+");
            System.out.println("|             SUCCESS             |");
            System.out.println("+---------------------------------+");
            indexofCustomer = j;
             do{
        Scanner customerMenu = new Scanner(System.in);
        dealership.printCustomerMenu();
       System.out.print("| Choose : ");
    optionCust2 = customerMenu.nextLine();
    if(optionCust2.toUpperCase().charAt(0) == 'C'){
    if(dealership.getNumCars() == 0){
    System.out.println("+---------------------------------+");
        System.out.println("|         UNFORTUNATELY           |");
        System.out.println("|     WE HAVE RAN OUT OF CAR      |");
        System.out.println("|     PLEASE COME AGAIN SOON      |");
        System.out.println("+---------------------------------+");
}
else{
System.out.println("+---------------------------------+");
System.out.println("|           Available Car         |");
System.out.println("+---------------------------------+");

dealership.PrintCarList(cars);
 while(true){
System.out.println("+---------------------------------+");
System.out.println("|            Choose a car         |");
System.out.println("+---------------------------------+");
System.out.print("|  car number : ");
int carChoice = customerin.nextInt();

if(carChoice >=1 && carChoice <= cars.length){
    Car selectedCar = cars[ carChoice - 1];

    //code index to randomly assign a salesperson to a customer
    int startIndex = 1;
    int endIndex = dealership.getNumSalesStaff();
    Random random = new Random();
    int randomIndex = random.nextInt(endIndex - startIndex ) + startIndex;

    System.out.println("+---------------------------------+");
    System.out.println("| Mr." + SalesStaff[randomIndex].getPersonName() + "( ID: " + SalesStaff[randomIndex].getSPID() + " ) " + "will handle\n| your purchase transaction.");
    System.out.println("+---------------------------------+");
    System.out.println("| do you wish to proceed and buy the following vehicle :");
    System.out.println(selectedCar);
    System.out.print("| Answer Y/N : ");
    String answer = customerin.next();
    SalesStaff[randomIndex].addCustomer(customer[indexofCustomer]);
    if(answer.toUpperCase().charAt(0) == 'Y'){
        System.out.println("+---------------------------------+");
        System.out.println("|       Choose Loan Length        |");
        System.out.println("+---------------------------------+");
        System.out.println("|   [6] = 6 Year Loan             |");
        System.out.println("|   [9] = 9 Year Loan             |");
        System.out.println("+---------------------------------+");
```

```
                    System.out.print("| Answer : ");
                    String paymentAnswer = customerin.next();
                    int loanLength = 0;

                    Bill bills = new Bill();
                    String date = dealership.GenerateDate();

                    if(paymentAnswer.charAt(0) == '6'){
                    loanLength = 6;
                    bills.calcTotalPayment(selectedCar);
                    bills.setBill(date, SalesStaff[randomIndex], selectedCar,
                    customer[indexofCustomer],loanLength);
                    bills.payBill(bills.calcDownpayment(selectedCar));
                    customer[indexofCustomer].addPurchasedCar(cars[carChoice - 1]);
                    SalesStaff[randomIndex].addBil(bills);
                    customer[indexofCustomer].addBil(bills);
                    deleteCarData(cars, dealership, (carChoice-1));
                    dealership.sellCar(carChoice-1,cars[carChoice-1],
                    SalesStaff[randomIndex], customer[indexofCustomer],bills);

                    System.out.println(dealership.getNumbills());
                    break;
                }
                    else if(paymentAnswer.charAt(0) == '9'){
                    loanLength = 9;
                    bills.calcTotalPayment(selectedCar);
                    bills.setBill(date, SalesStaff[randomIndex], selectedCar,
                    customer[indexofCustomer],loanLength);
                    bills.payBill(bills.calcDownpayment(selectedCar));
                    customer[indexofCustomer].addPurchasedCar(cars[carChoice - 1]);
                    SalesStaff[randomIndex].addBil(bills);
                    customer[indexofCustomer].addBil(bills);
                    deleteCarData(cars, dealership, (carChoice-1));
                    dealership.sellCar(carChoice-1,cars[carChoice-1],
                    SalesStaff[randomIndex], customer[indexofCustomer],bills);
                    System.out.println(dealership.getNumbills());
                    break;
                    }

                }
                else{
                    System.out.println("+---------------------------------+");
                    System.out.println("|       Choose another car ?      |");
                    System.out.println("+---------------------------------+");
                    System.out.print("| Answer Y/N : ");
                    answer = customerin.next();
                    if(answer.toUpperCase().charAt(0) == 'Y'){
                        dealership.PrintCarList(cars);
                    }
                    else{
                        System.out.println("+---------------------------------+");
                        System.out.println("|    Thank you for using us !     |");
                        System.out.println("+---------------------------------+");
                        break;
                    }

                }

            }
            else {
                System.out.println("|  Invalid car choice.");
            }
        }

            }
            // opt user to choose which car to buyE
                }
                else if(optionCust2.toUpperCase().charAt(0) == 'I'){
                // display all available car list
```

```
System.out.println("+---------------------------------+");
System.out.println("|          Available Car          |");
System.out.println("+---------------------------------+");

dealership.PrintCarList(cars);
}
else if(optionCust2.toUpperCase().charAt(0) == 'B'){

Scanner optionbillCust = new Scanner(System.in);

int repeat5 = 0;
do{
 System.out.println("+---------------------------------+");
 System.out.println("|            BILL MENU            |");
 System.out.println("+---------------------------------+");
 System.out.println("|    [P] = Pay Bill               |");
 System.out.println("|    [L] = List of Bill           |");
 System.out.println("|    [B] = Back to Menu           |");
 System.out.println("+---------------------------------+");
System.out.print("| Choose : ");
String optionBill = optionbillCust.next();
Bill[] customerbill = customer[indexofCustomer].getCustBill();
if(optionBill.toUpperCase().charAt(0) == 'P'){
        int chosenBill = 0;
        double amountToPay = 0;
    if(customer[indexofCustomer].getCustNumBill() == 0) {
            System.out.println("| NO BILL AVAILABLE");
    }
    else {
        for(int i =0;i<customer[indexofCustomer].getCustNumBill(); i++){


        Bill bill = customerbill[i];
        System.out.println("| BILL NUMBER " + (i+1));
        System.out.println(bill.SimplifiedBill());

    }
        System.out.print("| Choose number of bills : ");
        chosenBill = optionbillCust.nextInt();
        int indexofBill = chosenBill - 1;
        if(indexofBill > 99 || indexofBill < 0){
            System.out.println("| ERROR : Bill Not Found.");
            System.out.println("| Going back to menu.");
            break;
        }
        Bill bill = customerbill[indexofBill];

        if(bill != null){
        System.out.println("+---------------------------------+");
        System.out.println("|          BILL NUMBER "+chosenBill+"
        |");
        System.out.println("|         HAS BEEN CHOSEN         |");
        System.out.println("+---------------------------------+");
        System.out.println(bill);
        System.out.println("| How much would you like to pay ?");
        System.out.print("| RM");
        amountToPay = optionbillCust.nextDouble();
        bill.payBill(amountToPay);
        System.out.println("+---------------------------------+");
        System.out.println("|        PAYMENT SUCCESSFUL       |");
        System.out.println("+---------------------------------+");


        if(bill.getPaymentLeft() == 0 || bill.getPaymentLeft() < 0){
            customer[indexofCustomer].removeBill(bill, indexofBill);
        System.out.println("+---------------------------------+");
        System.out.println("|         BILL FULLY PAID         |");
        System.out.println("|         TERMINATING BILL        |");
        System.out.println("+---------------------------------+");
```

```
                }
                }
                else{
                    System.out.println("| ERROR, NO BILL FOUND");
                }
            }

        }
       else if(optionBill.toUpperCase().charAt(0) == 'L'){
            for(int i = 0; i < customer[indexofCustomer].getCustNumBill(); i++){
                Bill bill = customerbill[i];
                System.out.println(bill.SimplifiedBill());
            }
            if(customer[indexofCustomer].getCustNumBill() == 0) {
                    System.out.println("| NO BILL AVAILABLE");
            }
        }
        else if(optionBill.toUpperCase().charAt(0) == 'B'){
            repeat5 = 1;
        }
        }while(repeat5 == 0);

        }
        else if(optionCust2.toUpperCase().charAt(0) == 'U'){
            System.out.println(customer[indexofCustomer]);
        }
        else if(optionCust2.toUpperCase().charAt(0) == 'L'){
            // memang kosong sebab taknak buat apa apa sebenarnya
        }
        else{
            System.out.println("| ERROR : Wrong Code Entered");
        }



        }while(optionCust2.toUpperCase().charAt(0) != 'L');

            }
        else{
            System.out.println("+---------------------------------+");
            System.out.println("|              ERROR              |");
            System.out.println("|     YOU HAVE "+(attempt-1)+" ATTEMPT LEFT     |");
            System.out.println("+---------------------------------+");
                attempt--;
        }
        }

    }
     }
    }
}
    else if(option == 'A'){
        Scanner admin = new Scanner(System.in);
        int attemptleft = 3;
        boolean success = false;
        System.out.println("+---------------------------------+");
        System.out.println("|           ADMIN PAGE            |");
        System.out.println("+---------------------------------+");
        System.out.println("|          PLEASE LOGIN           |");
        System.out.println("|        YOU HAVE 3 ATTEMPT       |");
        System.out.println("+---------------------------------+");
        while(attemptleft != 0 && success == false){
            System.out.print("| Username : ");
            String username = admin.nextLine();
            System.out.print("| Password : ");
            String Password = admin.nextLine();
        if(username.equals("staffKamalSolution") && Password.equals("admin123")){
            success = true;
        int repeat2 = 0;
```

```java
while(repeat2 != 1){
dealership.PrintStaffMainMenu();
System.out.print("| Choose one to continue : ");
String option2 = input.next();

if(option2.toUpperCase().charAt(0) == 'P'){
    dealership.PrintTotalProfit();
}
else if(option2.toUpperCase().charAt(0) == 'I'){

    Scanner adminIO = new Scanner(System.in);

    String optionIO = "C";
    while(optionIO.toUpperCase().charAt(0) != 'B'){
    dealership.printIOmenu();
    System.out.print("| Choose : ");
    optionIO = adminIO.next();

    if(optionIO.toUpperCase().charAt(0) == 'I'){
            importCar(cars, dealership,"ImportCar.txt");
    }else{

    }
}
}
else if(option2.toUpperCase().charAt(0) == 'C'){

    Scanner Manage = new Scanner(System.in);
    String carOption = "G";

    while(carOption.toUpperCase().charAt(0) != 'M'){
    dealership.PrintCarManagement();
    System.out.print("| Choose : ");
    carOption = Manage.nextLine();
     if(carOption.toUpperCase().charAt(0) == 'A'){
    addNewCar(cars, dealership);
    }

    else if(carOption.toUpperCase().charAt(0) == 'R'){
        RemoveCar(cars, dealership);
    }
    else if(carOption.toUpperCase().charAt(0) == 'I'){
    System.out.println("+----------------------------------+");
    System.out.println("|            CAR STOCK             |");
    System.out.println("+----------------------------------+");
    dealership.PrintCarList(cars);
}
else {

}
}


}
else if(option2.toUpperCase().charAt(0) == 'S'){
Scanner Manage = new Scanner(System.in);
String staffOption = "G";
while(staffOption.toUpperCase().charAt(0) != 'M'){

dealership.PrintStaffManagement();
System.out.print("| Choose : ");
staffOption = Manage.nextLine();

    if(staffOption.toUpperCase().charAt(0) == 'C'){
        dealership.printStaffCommission(SalesStaff);
    }
    else if(staffOption.toUpperCase().charAt(0) == 'I'){
        dealership.printStaffInformation(SalesStaff);
    }
```

```
            else{
            }
        }

        }
        else if(option2.toUpperCase().charAt(0) == 'A'){
            dealership.PrintSystemAnalytics();
        }
        else if(option2.toUpperCase().charAt(0) == 'M'){
            repeat2++;
        }
        }
    }
        else {
            attemptleft--;
            System.out.println("+----------------------------------+");
            System.out.println("|             ERROR              |");
            System.out.println("|    YOU HAVE "+attemptleft+" ATTEMPT LEFT     |");
            System.out.println("+-------------------------------------+");
        }

    }
    }
    else if(option == 'E'){
        System.out.println("+---------------------------------+");
        System.out.println("|    THE SYSTEM WILL EXIT NOW     |");
        System.out.println("+---------------------------------+");

    }
    }while(option != 'E');


}
public static void RegisterCustomerProcess(Customer[] customer,Dealership dealership){
            int j = 0;
            while(j < 1){
                Scanner CustomerRegist = new Scanner(System.in);

                customer[dealership.getNumCustomer()] = new Customer();

                System.out.println("+---------------------------------+");
                System.out.println("|      Fill you information       |");
                System.out.println("+---------------------------------+");
                  System.out.print("| Full Name      : ");
                String name = CustomerRegist.nextLine();
                  System.out.print("| NRIC [no - ]   : ");
                String ic = CustomerRegist.nextLine();
                  System.out.print("| Set a Password : ");
                String password = CustomerRegist.nextLine();
                  System.out.print("| Home Address   : ");
                String address = CustomerRegist.nextLine();
                  System.out.print("| Phone Number   : ");
                String phone = CustomerRegist.nextLine();

                name = name.toUpperCase();
                address = address.toUpperCase();
                    customer[dealership.getNumCustomer()].setCustomerData(name, address, ic, phone, password);

                System.out.print(customer[dealership.getNumCustomer()]);
                System.out.println("\n+---------------------------------+");
                System.out.println("|   is the information correct ?  |");
                System.out.println("+    [Y] = Yes   ||   [N] = NO    +");
                System.out.println("+---------------------------------+");
                  System.out.print("| Answer : ");
                String answer = CustomerRegist.nextLine();


```

```
                 if(answer.toUpperCase().charAt(0) == 'Y'){

                     dealership.addCustomer(customer[dealership.getNumCustomer()]);

                     System.out.println(customer[dealership.getNumCustomer() - 1]);
                     System.out.println(dealership.getNumCustomer());
                     System.out.println(customer.length);
                     j++;
                 }
                 else{
                     System.out.println("| Please fill the the form again.");
                 }
     }
}
 public static void addNewCar(Car[] cars, Dealership dealership) throws Exception {
         System.out.println("+----------------------------------+");
         System.out.println("|          ADD NEW CAR             |");
         System.out.println("+----------------------------------+");
             Scanner adminCar = new Scanner(System.in);
             System.out.println("+----------------------------------+");
             System.out.print("| Car Brand : ");
             String carB = adminCar.nextLine();
             System.out.print("| Car Model : ");
             String carM = adminCar.nextLine();
             System.out.print("| Car Type  : ");
             String carT = adminCar.nextLine();
             System.out.print("| Car Price : RM");
             double carp = adminCar.nextDouble();

             cars[dealership.getNumCars()] = new Car(carB, carM, carT, carp);

              dealership.addCarToInventory(cars[dealership.getNumCars()]);
         try{
             writeFile(cars, dealership);
         }
         catch(Exception ex){}

             System.out.println("+----------------------------------+");
             System.out.println("| Car Added Successfully.");
             System.out.println("+----------------------------------+");

 }
 public static void RemoveCar(Car[] cars, Dealership dealership) throws FileNotFoundException, Exception{
     Scanner adminCar = new Scanner(System.in);
     int error = 1;
     while(error == 1){
     if(dealership.getNumCars() == 0){
     System.out.println("+----------------------------------+");
     System.out.println("|       NO CARS IN INVENTORY      |");
     System.out.println("+----------------------------------+");
         break;
     }

     System.out.println("+----------------------------------+");
     System.out.println("|         REMOVE A CAR            |");
     System.out.println("+----------------------------------+");
     dealership.PrintCarList(cars);
     System.out.println("+----------------------------------+");
     System.out.println("|      Choose car to remove       |");
     System.out.println("+----------------------------------+");
     System.out.print("|  car number : ");
     int indextoRemove = adminCar.nextInt();
     indextoRemove = indextoRemove - 1;

     if(cars[indextoRemove] == null){
     System.out.println("+----------------------------------+");
     System.out.println("|   A     CAR DOES NOT EXIST       |");
     System.out.println("+----------------------------------+");
```

```
      }

      else {
      Car carToRemove = cars[indextoRemove];
      dealership.removeCarFromInventory(carToRemove, indextoRemove);

      System.out.println("+---------------------------------+");
      System.out.println("|     CAR REMOVED SUCCESSFULLY    |");
      System.out.println("+---------------------------------+");
      error = 0;

      deleteCarData(cars, dealership, indextoRemove);
      }

      }

  }
  public static void importCar(Car[] cars, Dealership dealership,String Filename) throws Exception {

      File CarImport = new File(Filename);
      FileReader CarReader = new FileReader(CarImport);
      BufferedReader bufferRead = new BufferedReader(CarReader);


      int carnum = 0;
          String line;
          while((line = bufferRead.readLine()) != null){
              StringTokenizer tokenizer = new StringTokenizer(line,";");

              if(tokenizer.countTokens() == 4){
                  carnum++;
                  String Brand = tokenizer.nextToken();
                  String Model = tokenizer.nextToken();
                  String Type = tokenizer.nextToken();
                  double price = Double.parseDouble(tokenizer.nextToken());

                  cars[dealership.getNumCars()] = new Car(Brand, Model, Type, price);

                  dealership.addCarToInventory(cars[dealership.getNumCars()]);

                  try{
                      writeFile(cars, dealership);
                  }catch(Exception ex){}
              }
          }
                      System.out.println("| "+carnum+" Car(s) Imported.");
                      System.out.println("| Car Imported Successfully.");
                      bufferRead.close();

          clearFile(Filename);
}
public static void writeFile(Car[] cars, Dealership dealership) throws Exception {
      try{
                      // EXPORT CAR
                      File ExportCar = new File(CAR_FILE);
                      FileWriter WriteCar = new FileWriter(ExportCar);
                      PrintWriter PrintCar = new PrintWriter(WriteCar);
                      for(int i =0;i<cars.length;i++){
                          if(cars[i] != null){
                              Car[] carData = dealership.getCarInvetory();
                              PrintCar.println(carData[i].WriteString());
                          }
                      }
                      PrintCar.close();
      }
      catch(Exception ex){}
  }
  public static void readFile(Car[] cars, Dealership dealership) throws Exception {
```

```java
        try{
            File CarImport = new File(CAR_FILE);
            FileReader CarReader = new FileReader(CarImport);
            BufferedReader bufferRead = new BufferedReader(CarReader);

            int carnum = 0;
                        String line;
                        while((line = bufferRead.readLine()) != null){
                            StringTokenizer tokenizer = new StringTokenizer(line,";");

                            if(tokenizer.countTokens() == 4){
                                carnum++;
                                String Brand = tokenizer.nextToken();
                                String Model = tokenizer.nextToken();
                                String Type = tokenizer.nextToken();
                                double price = Double.parseDouble(tokenizer.nextToken());
                                cars[dealership.getNumCars()] = new Car(Brand, Model, Type, price);
                                dealership.addCarToInventory(cars[dealership.getNumCars()]);
                            }
                        }
                        bufferRead.close();
        }
        catch(Exception ex){}
    }
    public static void clearFile(String fileName){

        try {
            // CLEAR OUT IMPORt.TXT
            FileWriter fileWriter = new FileWriter(fileName);
            fileWriter.close();

        } catch (IOException e) {
            System.out.println("An error occurred: " + e.getMessage());
    }

    }
    public static void deleteCarData(Car[] cars, Dealership dealership,int indextoRemove) throws FileNotFoundException, Exception {

        File OriginalFile = new File(CAR_FILE);
        File tempFile = new File("tempt.txt");

            try {

                BufferedReader reader = new BufferedReader(new FileReader(OriginalFile));
                BufferedWriter writer = new BufferedWriter(new FileWriter(tempFile));

                String currentLine;
                int CurrentLineNumber = 1;
                int LineToDelete = (indextoRemove+1);

                while((currentLine = reader.readLine()) != null){
                    if(CurrentLineNumber != LineToDelete){
                        writer.write(currentLine);
                        writer.newLine();
                    }
                    CurrentLineNumber++;
                }

                 writer.close();
                reader.close();

                boolean successful = OriginalFile.delete();
                boolean successful2 = tempFile.renameTo(OriginalFile);
                System.out.println(successful);
                System.out.println(successful2);

            } catch (IOException e) {
                e.printStackTrace();
            }
    }
}

```