Gruppe Pizza

Car Wash

Contents

1.0 ITO..... 3
1.1 Business Model Canvas..... **3**
1.2 Risikoanalyse..... **4**
2.0 Software Design..... 5
2.1 Noun&verb lists **5**
2.2 Use Cases..... **6**
2.3 FURPS+..... **11**
2.4 Sequence Diagram..... **12**
2.5 Domain Model..... **14**
2.6 Class Diagram..... **15**
3.0 Software Construction..... 16

Nøglepartnere

- Gruppe Pizza
- Maskine firma der levere og vedligeholder bilvaskeanlæg
- Ejer

Nøgleaktiviteter

- Vaske biler
- Tilknyttet system(er) til bilvaskeanlæg og vedligeholde
- Fysisk lokation til bilvaskeanlæg

Nøgleressourcer

- Kontakt til Gruppe Pizza som udvikler system til vores vask
- Kontakt til bilvaskeanlæg leverandør for opsætning og vedligeholdelse

Skabelse af kunde-værdi

- Lettere, hurtigere og grundigere vask af biler

Kunderelationer

- Kontakt til kunder via. Mail, telefon og sociale medier
- Bilvaskeanlæg
- Lokation, reklamer, muligvis samarbejde med bilsælger eller brændstofs sælger

Distributions-kanaler

- Vaskehalle til vask af biler på diverse lokationer

Kundesegment

- Private kunder der går op i deres bilers udseende
- Bilsælgere
- Tankstationer

Omkostningsstruktur

- Udstyr, indkøb og vedligeholdelse
- Systemudvikling
- Potentiel reklame

Indtægtsstrømme

- Vask af biler
- Mulig kontrakt med bilsælger og, el. tankstationer

Risikomomenter	Sandsynlighed	Konsekvens	Produkt	Præventive tiltag	Ansvarlig	Løsningsforslag	Ansvarlig
Dårlig Q.A	3	8	24	Bedre testning af program og gennemgang af kode. En fra gruppen bliver ansvarlig for test.	Testansvarlige	Ekstra Testansvarlig til rådighed	Gruppen og den ekstra ansvarlige.
Langtidssygdom	1	5	5	Sørg for at have aftaler på plads om hvem der gør hvad.	De personer som har kendskab til hvad personen lavede	Ved hvem der tager over og muligvis bidrage hjemmefra	Gruppen
Korttidssygdom	2	3	6	Sørg for at have aftaler på plads om hvem der gør hvad.	De personer som har kendskab til hvad personen lavede	Ved hvem der tager over og muligvis bidrage hjemmefra	Gruppen
Medlem forlader gruppen	1	7	7	Sørg for alle har det godt og flere kendskab til hinandens arbejde.	De personer som har kendskab til hvad personen lavede	Ved hvem der tager over og eventuelt har en aftale på plads med kunden i tilfælde da projektet vil tage længere tid	Gruppen
Cloud storage problemer	1	10	10	Sikre sig sit data er sikkert, enten ved at have en kopi lokalt, eller ved at bruge flere cloud storage.	Gruppen	I tilfælde af problemer, hvor data ikke er tilgængelig, prøv at rollback versioner.	Gruppen
Tidsfejlvurderinger	3	4	12	Opfølgning af mål og delmål og sikring af overholdelse af disse.	Gruppen	Arbejd ekstra timer på projektet hvis nødvendigt.	Gruppen
Problemer ved installation af system	5	4	20	Test ofte og muligvis have prototype klar før indgivelse af systemet til test hos kunde.	Gruppen	Ekstra tid bruges på at implementer systemet så det virker. Kan også tilknytte et support firma	Gruppen og support firma
Ændringer af produktkrav	1	8	8	Hold kontakten med kunden løbende under processen	Gruppen	Forhandle nye tiltag med kunden.	Gruppen

Nouns

Customer
Wash card
Car
Credit card
Machine
Receipt
~~System~~
Owner
Amount
Type
Price
~~Value~~
Statistic

Verbs

Wash
Buy
Contain
Insert
Deduct
Recharge
Print

Title: **Buy Wash Card**

Primary Actor: Customer

Main Success Scenario:

1. Customer select no to having a WashCard when prompted by the system
2. Customer fill information such username
3. Customer fills card with credit between 200.-1000 credit
4. Customer insert credit card and pays the amount chosen.
5. User and Washcard are created.

Alternative Flow (Extensions):

At all times, if the system fails:

1. Show error message
2. Reset system
3. Send crash data to developers

At all times, if incorrect input is received:

1. show error message.

System checks if username is already used:

1. System prompts customer telling them username is already in use and they have to choose another.

Title: Buy Wash
Primary Actor: Customer
Main Success Scenario:

1. Customer puts wash card into machine and enter username
2. Customer selects “Buy Wash”
3. Customer chooses “Wash Type”
4. System asks if customer wants receipt printed

Alternative Flow (Extensions):
At all times, if the system fails:

1. Show error message
2. Reset system
3. Send crash data to developers

At all times, if incorrect input is received:

1. Show error message.

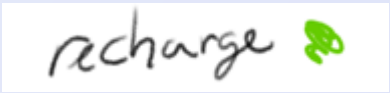
There is insufficient funds on wash card:

1. Prompts customer to choose a cheaper wash or to refill wash card to be able to buy selected wash

Customer accepts or declines receipt:

1. Receipt doesn't print and system goes back to main screen.
2. Receipt prints and system goes back to main screen.

Title: **Recharge Card**



Primary Actor: Customer

Main Success Scenario:

1. Customer puts wash card into machine
2. Customer enters username
3. Customer select to refill “wash card”
4. Customer enters amount they want to refill card with
5. Customer enters credit card info and pays

Alternative Flow (Extensions):

At all times, if the system fails:

1. Show error message
2. Reset system
3. Send crash data to developers

At all times, if incorrect input is received:

1. show error message.

Title: Check Balance

Primary Actor: Customer

Main Success Scenario:

1. Customer puts wash card into machine
2. Customer enters username
3. Customer select “Check Balance”
4. Balance is displayed and goes back to main menu.

Alternative Flow (Extensions):

At all times, if the system fails:

1. Show error message
2. Reset system
3. Send crash data to developers

At all times, if incorrect input is received:

1. show error message.

Title: **Statistic**

Primary Actor: Owner

Main Success Scenario:

1. Owner enters pin on enter username screen.
2. System shows the statistic over washes bought.

Alternative Flow (Extensions):

At all times, if the system fails:

1. Show error message
2. Reset system
3. Send crash data to developers

At all times, if incorrect input is received:

1. show error message.

FURPS+:

FURPS:

Functional:
Different washes
Has washcard system
- Can refill washcard
- Create washcard

Username for each customer
Owner has a pin-code to view statistic
Can use credit card system
Prints receipts

Usability:
Text based interface
Customer Service number “12345678”
Help function (tells user how to use station)
Saves statistics to file locale
Prints receipts

Reliability:
Simple and small system so frequence of error should be low.
Easy to predict because of simple system.
Information is local saved so easy to recover system.

Performance:
Fast and responsive.
Low resource usage.
Can access internet for credit card info

Supportability:
Maintenance is low
Simple to implement
Can be updated or change prices or washtypes.
Can be updated off station

+:

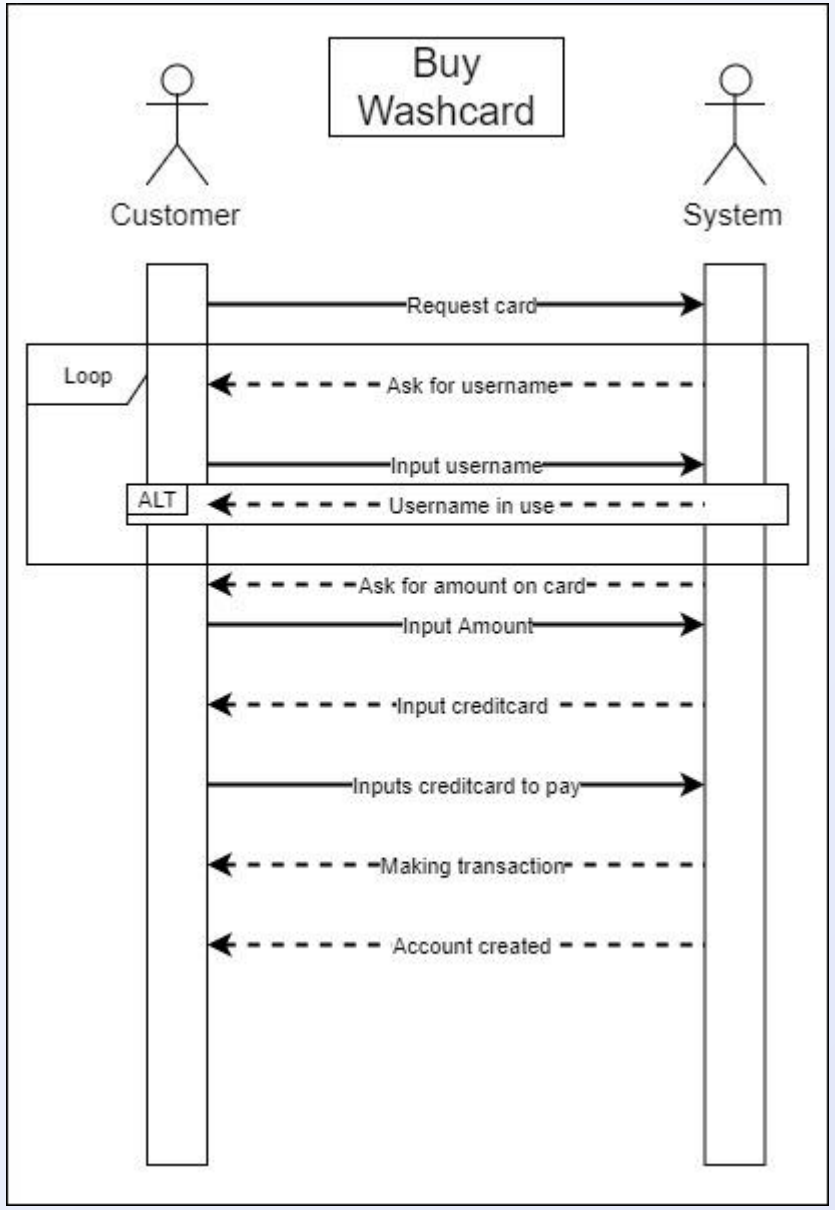
Implementation:
English language
Small computer with year 2000 hardware
Operating system - Windows vista and up

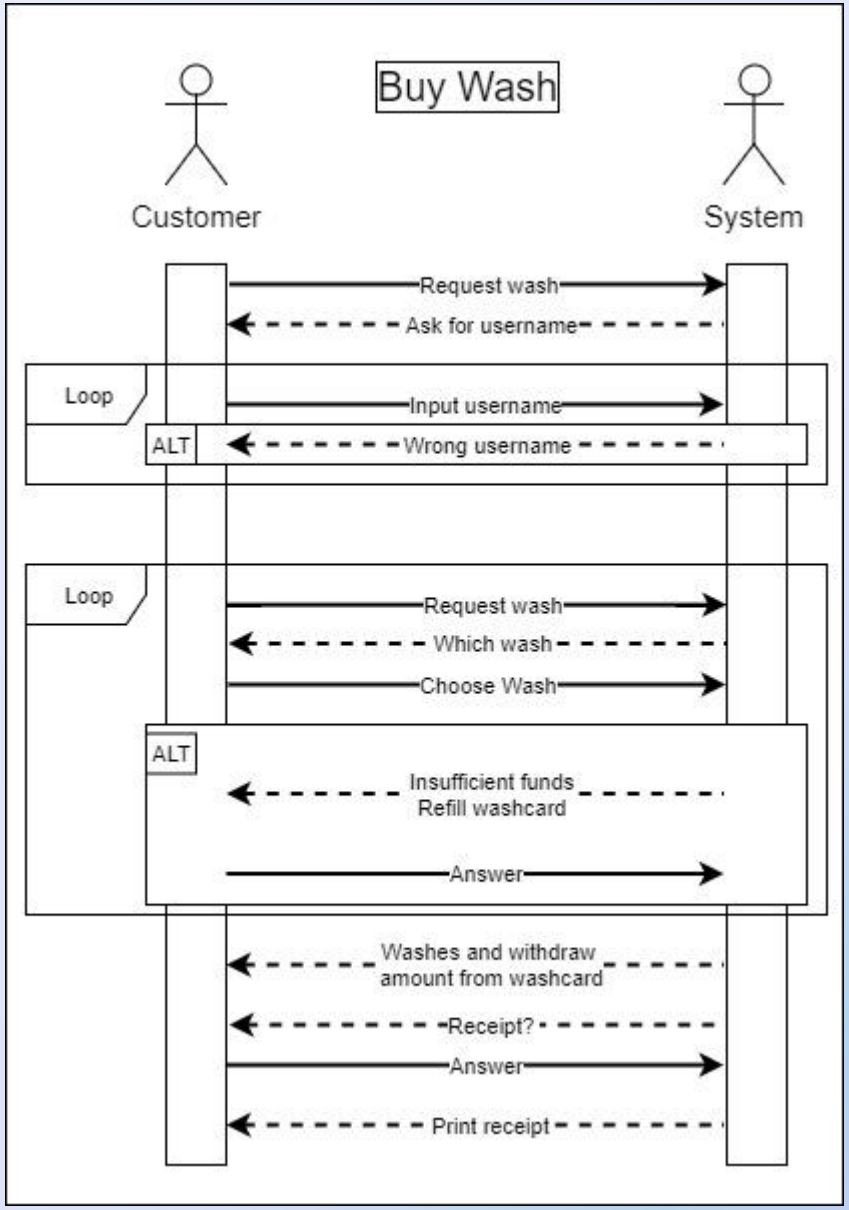
Interface:
Needs to have a internet connection to be able to access credit card system

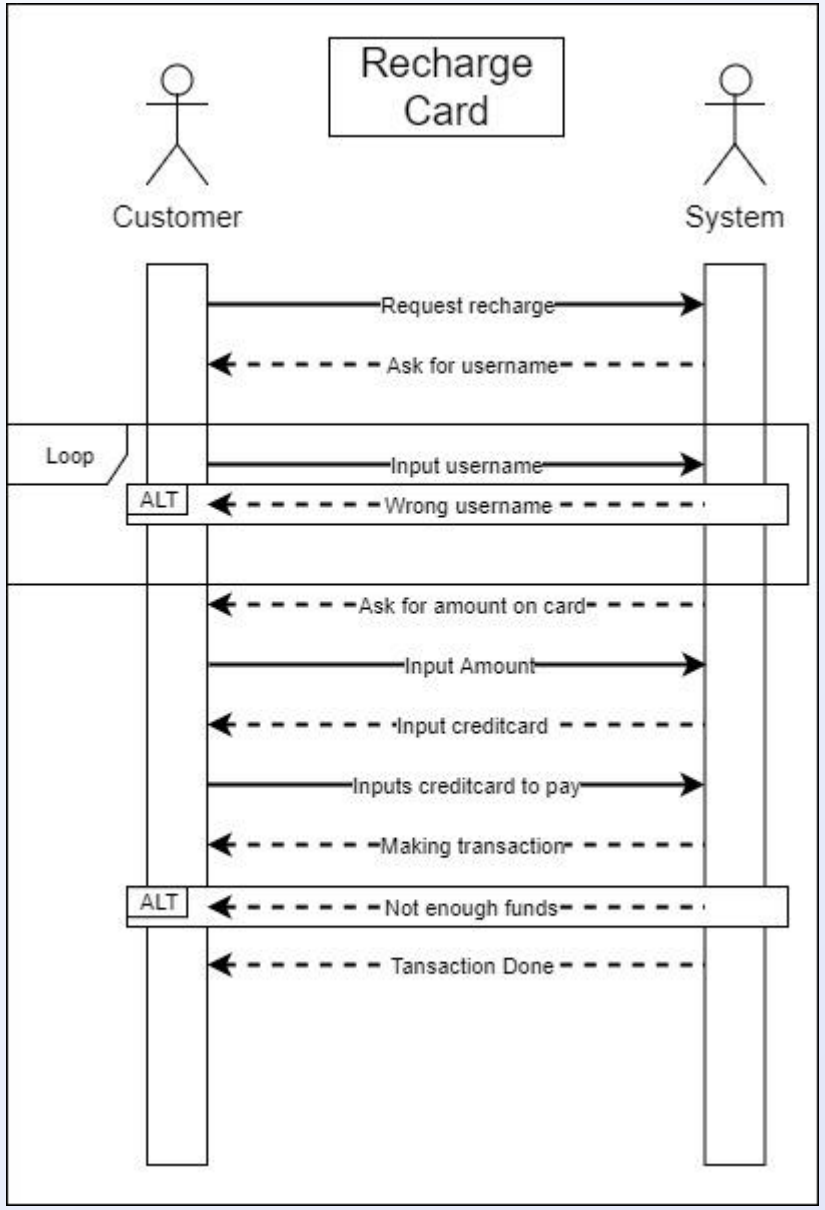
Operation:
Can change prices or wash types directly on system

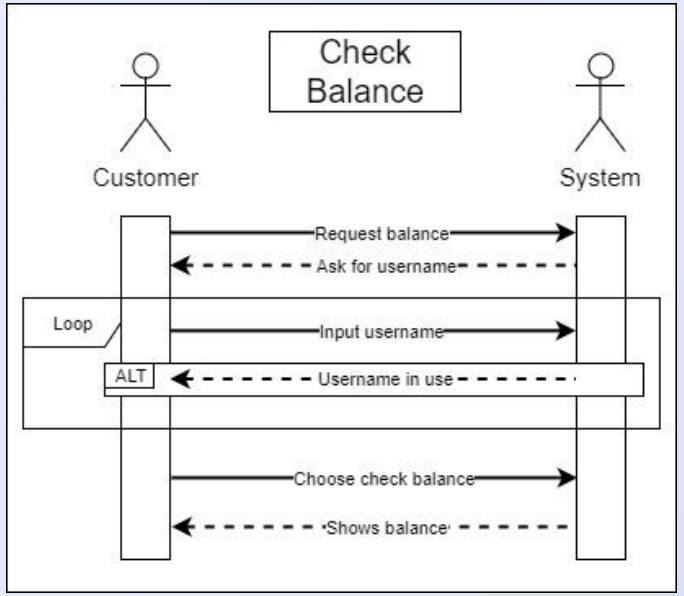
Packaging:
...

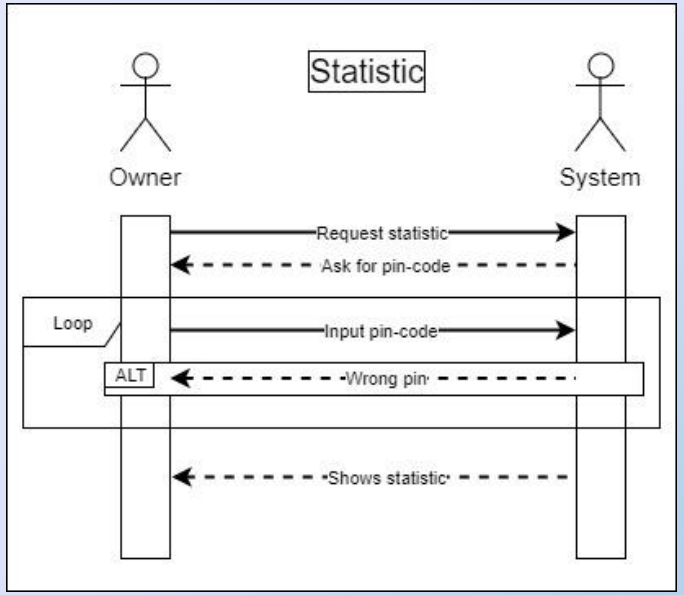
Legal:
Licensing and so forth.


Buy Washcard
Customer
System
Request card
Loop
Ask for username
Input username
ALT
Username in use
Ask for amount on card
Input Amount
Input creditcard
Inputs creditcard to pay
Making transaction
Account created

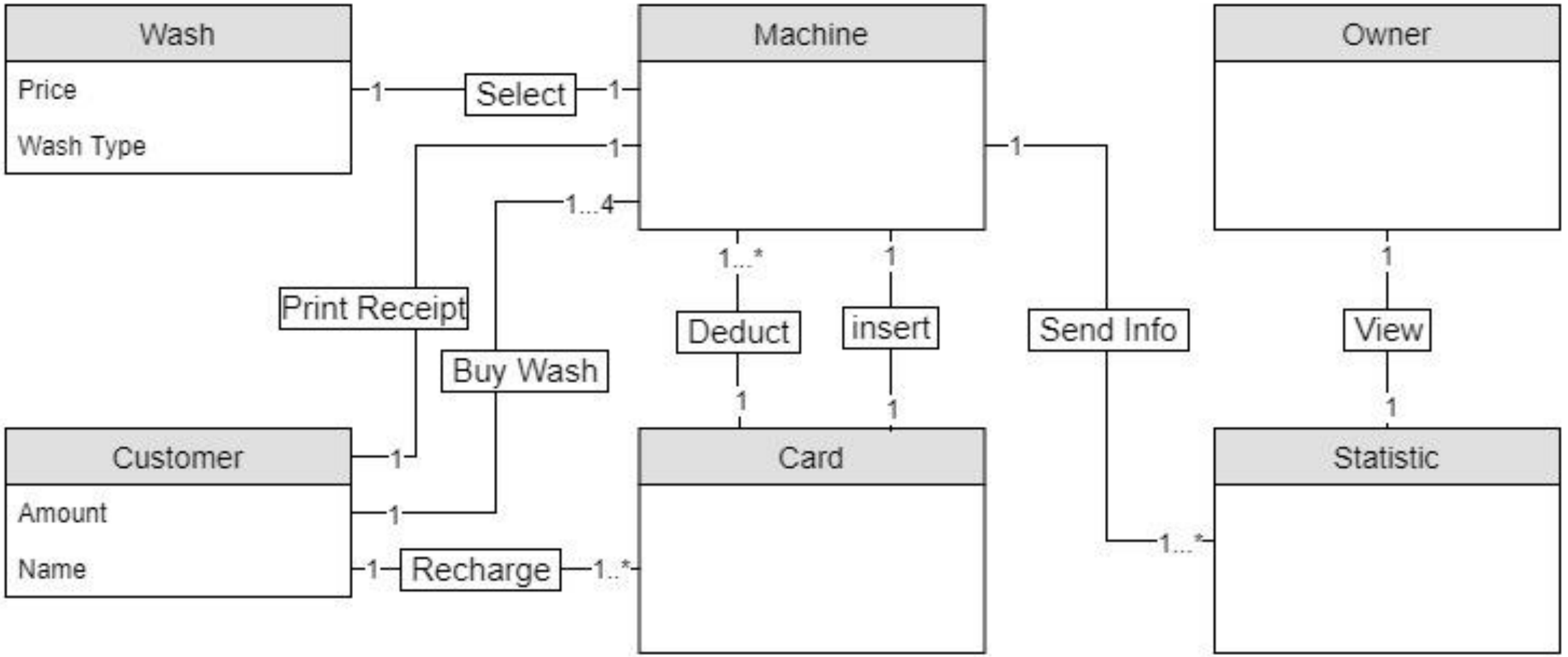

Buy Wash
Customer
System
Request wash
Ask for username
Loop
Input username
ALT
Wrong username
Loop
Request wash
Which wash
Choose Wash
ALT
Insufficient funds
Refill washcard
Answer
Washes and withdraw
amount from washcard
Receipt?
Answer
Print receipt


Recharge Card
Customer
System
Request recharge
Ask for username
Loop
Input username
ALT
Wrong username
Ask for amount on card
Input Amount
Input creditcard
Inputs creditcard to pay
Making transaction
ALT
Not enough funds
Tansaction Done

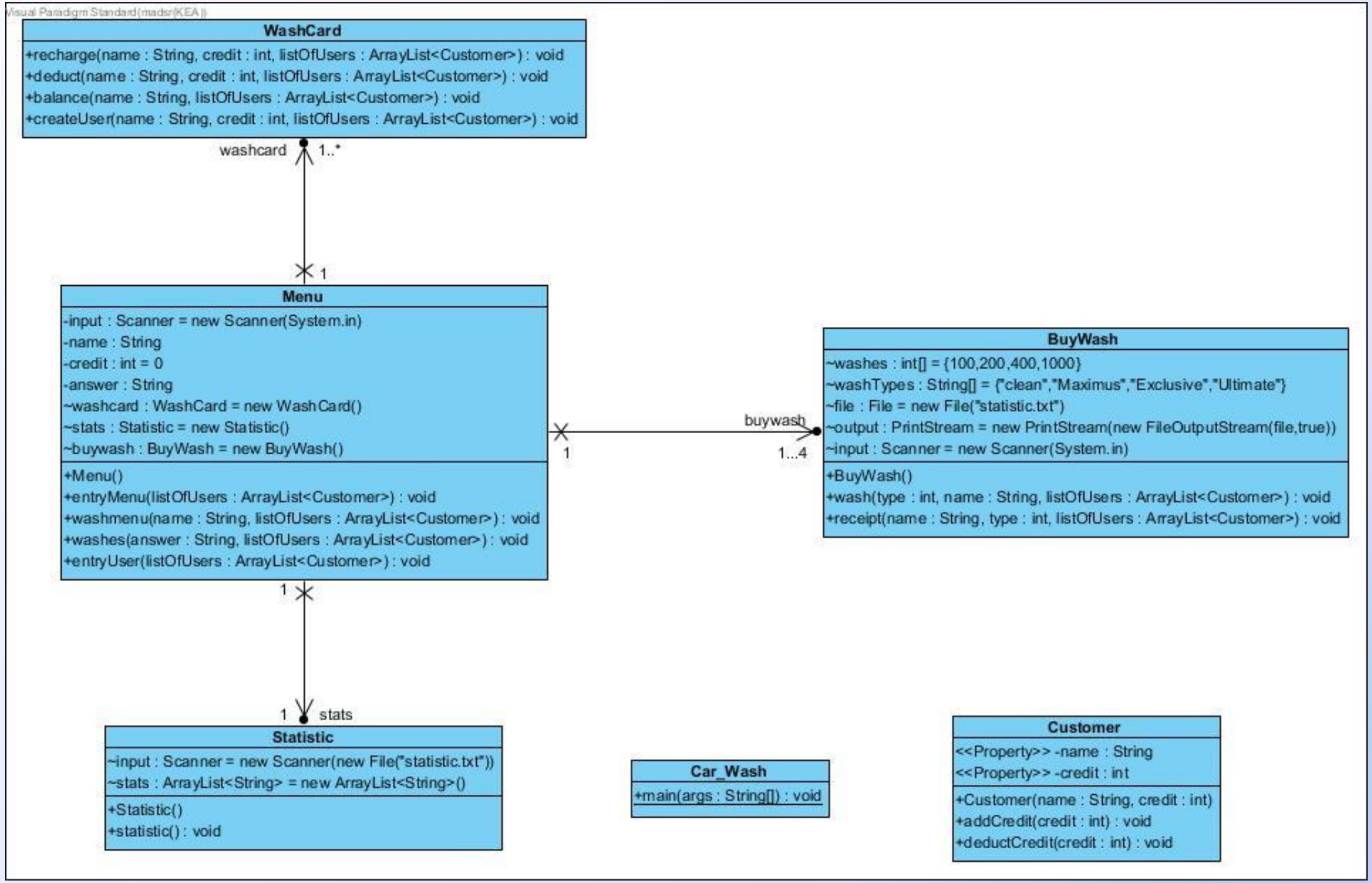

Check Balance
Customer
System
Request balance
Ask for username
Loop
Input username
ALT
Username in use
Choose check balance
Shows balance


Statistic
Owner
System
Request statistic
Ask for pin-code
Loop
Input pin-code
ALT
Wrong pin
Shows statistic

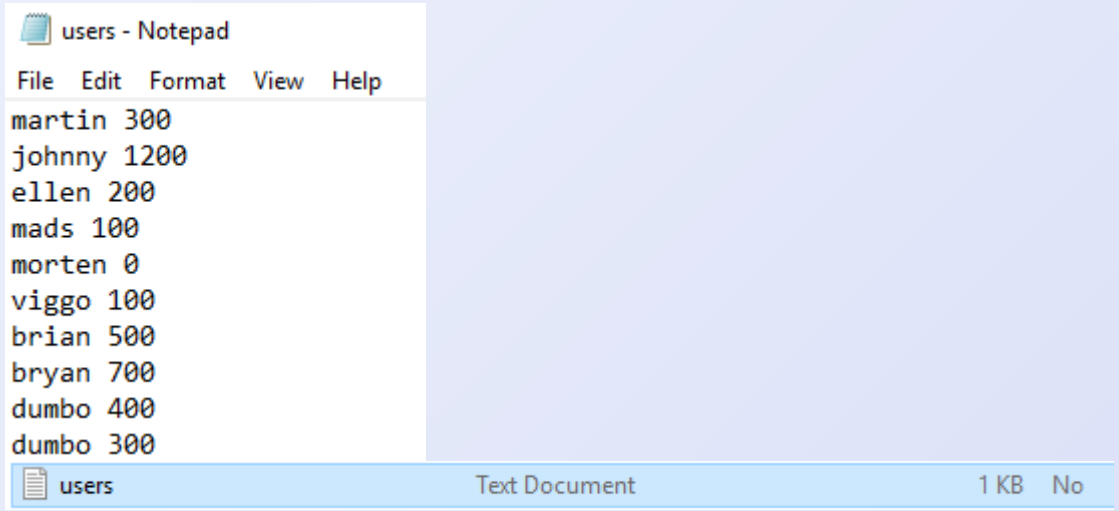
Domain Model



Visual Paradigm Standard (madsr(KEA))



```java
import java.io.*;
import java.util.*;

public class Car_Wash{



public static void main(String[] args)throws FileNotFoundException{


    Scanner input = new Scanner(new File("users.txt"));
    ArrayList<Customer> listOfUsers = new ArrayList<Customer>();

    while(input.hasNext()){
      String foo = input.next();
      int foo1 = input.nextInt();

      listOfUsers.add(new Customer(foo,foo1));
    }



   while (2 != 1){

    PrintStream output = new PrintStream("users.txt");
   for(Customer d : listOfUsers){
      output.println(d.getName() + " " + d.getCredit());
   }


   Menu menu = new Menu();
   menu.entryMenu(listOfUsers);

   }

}

}
```

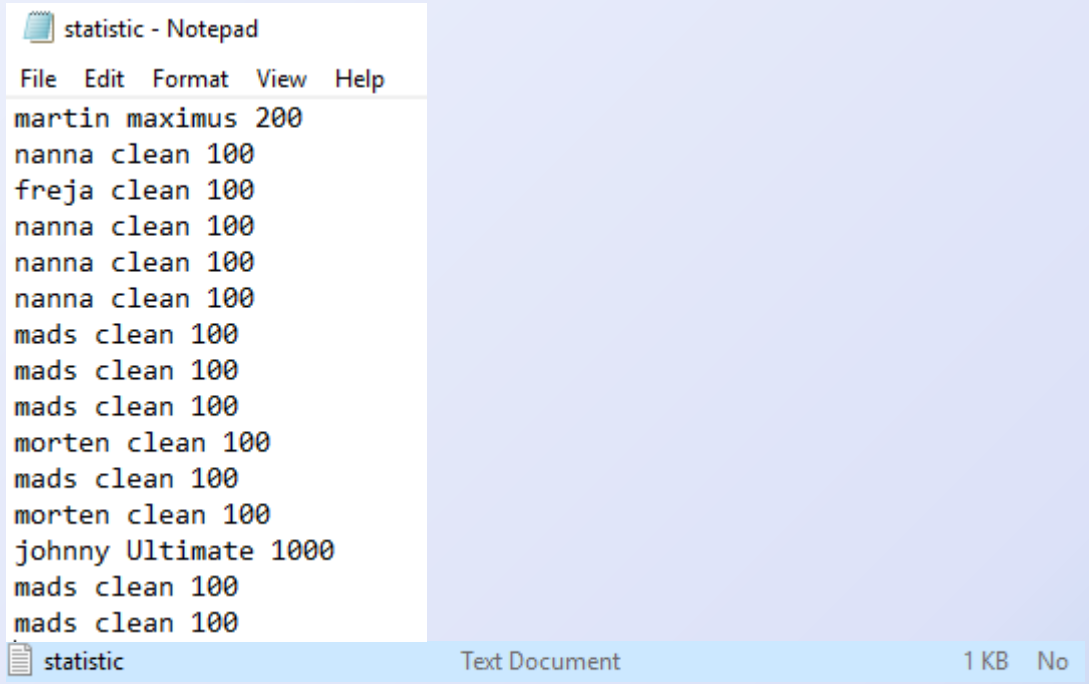


```java
import java.util.*;
import java.io.*;

public class Menu{

   public Menu()throws FileNotFoundException{
   }

   //attri
   private Scanner input = new Scanner(System.in);
   private String name;
   private int credit = 0;
   private String answer;

   //objects
   WashCard washcard = new WashCard();
   Statistic stats = new Statistic();
   BuyWash buywash = new BuyWash();

   //methods
   public void entryMenu(ArrayList<Customer> listOfUsers)throws FileNotFoundException, InterruptedException{
      System.out.println("Do you have a washcard?  [yes/no?]      For statistics input Employe pincode [****]");
      answer = input.nextLine();

      if(answer.equals("no") || answer.equals("NO")){
         entryUser(listOfUsers);
      }
      else if(answer.equals("yes") || answer.equals("YES")){
         System.out.println("Insert Washcard... [Type Username]");
         name = input.nextLine();
         washmenu(name,listOfUsers);
      }else if(answer.equals("1234")){
         System.out.println("Welcome Owner this is the overall statistic of the Carwash");
         stats.statistic();
      }
   }

   public void washmenu(String name,ArrayList<Customer> listOfUsers)throws FileNotFoundException, InterruptedException{
      int i = 0;

      for(Customer d : listOfUsers){
         if(d.getName() != null && d.getName().contains(name)){
            i=1;
         }
      }

      if(i==1){
         System.out.println("Welcome. Choose on of the following:");
         System.out.println("");
         System.out.println("[Buy Wash]       [Check Balance]     [Refill Card]     [Exit]");

         answer = input.nextLine();
         washes(answer,listOfUsers);

      }else {System.out.println("Wrong user name");
         i=0;
      }
   }
```

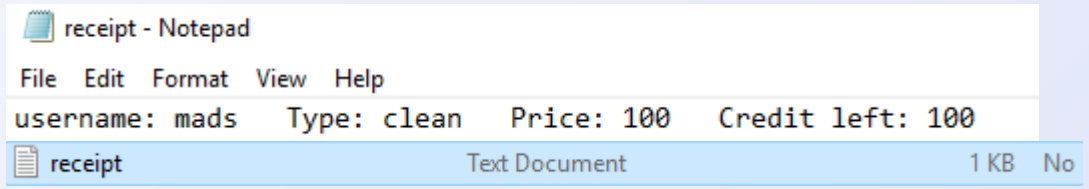
```java
   public void washes(String answer,ArrayList<Customer> listOfUsers)throws FileNotFoundException, InterruptedException{

      if(answer.equals("buy wash") || answer.equals("BUY WASH")){
         System.out.println("[1] clean - 100cred     [2]  Maximus - 200cred     [3] Exclusive - 400cred     [4] Ultimate - 1000cred");
         int type=input.nextInt();
         buywash.wash(type, name, listOfUsers);
      }
      else if(answer.equals("check balance") || answer.equals("CHECK BALANCE")){
         washcard.balance(name, listOfUsers);
         washmenu(name, listOfUsers);
      }
      else if(answer.equals("refill card") || answer.equals("REFILL CARD")){
         System.out.println("how much do you want?");
         credit = input.nextInt();

         System.out.println("Insert credit card");
         Thread.sleep(1000);
         System.out.println(".");
         Thread.sleep(1000);
         System.out.println(".");
         Thread.sleep(1000);
         System.out.println("Thank you");

         washcard.recharge(name, credit, listOfUsers);

      }
      else if(answer.equals("exit") || answer.equals("EXIT")){
         System.exit(0);
      }else{
         System.out.println("Error, Wrong Input");
      }
   }
```

```
92    //ask user for information
93    public void entryUser(ArrayList<Customer> listOfUsers)throws InterruptedException{
94       System.out.println("input your username");
95       name = input.nextLine();
96
97       int i=1;
98
99       for(Customer d : listOfUsers){
100        if(d.getName() != null && d.getName().contains(name)){
101           System.out.println("User already exists");
102           i=0;
103         }
104       }
105
106      if(i==1){
107
108      System.out.println("input amount of credit you want on wash card  [200-1000]");
109
110      credit = input.nextInt();
111
112      System.out.println("Insert credit card");
113      Thread.sleep(1000);
114      System.out.println(".");
115      Thread.sleep(1000);
116      System.out.println(".");
117      Thread.sleep(1000);
118      System.out.println("Thank you");
119
120
121      if(credit < 1001 && credit > 200){
122         washcard.createUser(name, credit, listOfUsers);
123      }else{System.out.println("Wrong credit amount");
124      }
125
126      }
127
128   }
129}
```

```java
import java.io.*;
import java.util.*;
import javax.swing.JOptionPane;

public class BuyWash{

public BuyWash()throws FileNotFoundException{}

   int[] washes = {100,200,400,1000};
   String[] washTypes ={"clean","Maximus","Exclusive","Ultimate"};

   File file = new File("statistic.txt");
   PrintStream output = new PrintStream(new FileOutputStream(file,true));

   Scanner input = new Scanner(System.in);

   public void wash(int type, String name, ArrayList<Customer> listOfUsers)throws FileNotFoundException{

      switch (type) {
         case 1:

            for(Customer d : listOfUsers){
               if(d.getName() != null && d.getName().contains(name)){
                  if(washes[type-1]<=d.getCredit()){
                     d.deductCredit(washes[type-1]);
                     output.println(name+" "+washTypes[type-1]+" "+washes[type-1]);
                     receipt(name, type, listOfUsers);
                  }else if (washes[type-1]>d.getCredit()){
                     System.out.println("not enough funds!");
                  }
               }
            }
            break;


         case 2:

            for(Customer d : listOfUsers){
               if(d.getName() != null && d.getName().contains(name)){
                  if(washes[type-1]<=d.getCredit()){
                     d.deductCredit(washes[type-1]);
                     output.println(name+" "+washTypes[type-1]+" "+washes[type-1]);
                     receipt(name, type, listOfUsers);
                   }else if (washes[type-1]>d.getCredit()){
                     System.out.println("not enough funds!");

                  }
                 }
            }
            break;
```



```
82      public void receipt(String name,int type, ArrayList<Customer> listOfUsers)throws FileNotFoundException{
83      System.out.println("Want Receipt?");
84      String answer = input.next();
85
86      if (answer.equals("yes")){
87      File receipt = new File("receipt.txt");
88      PrintStream receipt_output = new PrintStream(receipt);
89         for(Customer d : listOfUsers){
90            if(d.getName() != null && d.getName().contains(name)){
91              receipt_output.println("username: " + name + "    " + "Type: " + washTypes[type-1] + "    " + "Price: " + washes[type-1] + "    " + "Credit left: " + d.getCredit());
92              JOptionPane.showMessageDialog(null, "username: " + name + "    " + "Type: " + washTypes[type-1] + "    " + "Price: " + washes[type-1] + "    " + "Credit left: " + d.getCredit());
93            }
94         }
95      }else if (answer.equals("no")){
96        System.out.println("Returning to menu");
97      }else{
98      System.out.println("Wrong input");
99      }
100    }
101
102 }
```



```java
import java.io.*;
import java.util.*;

public class Customer{

   String name;
   int credit;

   public Customer(String name, int credit){
      this.name=name;
      this.credit=credit;
   }

   public void setName(String name){
      this.name=name;
   }

   public String getName(){
      return name;
   }

   public void addCredit(int credit){
      this.credit=this.credit + credit;
   }

   public void deductCredit(int credit){
      this.credit=this.credit - credit;
   }

   public int getCredit(){
      return credit;
   }
}
```

```java
import java.io.*;
import java.util.*;

public class WashCard{

   public void recharge(String name, int credit, ArrayList<Customer> listOfUsers){

      for(Customer d : listOfUsers){
        if(d.getName() != null && d.getName().contains(name)){
           d.addCredit(credit);
         }
      }
   }

   public void deduct(String name, int credit, ArrayList<Customer> listOfUsers){

      for(Customer d : listOfUsers){
        if(d.getName() != null && d.getName().contains(name)){
           d.deductCredit(credit);
         }
      }
   }

   public void balance(String name, ArrayList<Customer> listOfUsers){

      for(Customer d : listOfUsers){
        if(d.getName() != null && d.getName().contains(name)){
           System.out.println(d.getCredit());
         }
      }
   }

   public void createUser(String name,int credit, ArrayList<Customer> listOfUsers){

      listOfUsers.add(new Customer(name,credit));

   }

}
```

```java
import java.io.*;
import java.util.*;

public class Statistic{

public Statistic()throws FileNotFoundException{
}

Scanner input = new Scanner(new File("statistic.txt"));
ArrayList<String> stats = new ArrayList<String>();

   public void statistic(){

    int count =0;
    int numall =0;
    int countclean=0;
    int countmaximus=0;

    while(input.hasNext()){

    String foo = input.next();
    stats.add(foo);

      String foo1 = input.next();
      if(foo1.equals("clean")){
        countclean++;
      }else if(foo1.equals("maximus")){
        countmaximus++;
      }

      int num = input.nextInt();
      numall = numall + num;
      count++;
    }

    Set<String> unique = new HashSet<String>(stats);
    for (String key : unique) {
      System.out.println(key + ": " + Collections.frequency(stats, key));
    }
      System.out.println();
      System.out.println("average price:\t"+numall/count);
      System.out.println("Clean:\t\t\t"+countclean);
      System.out.println("Maximus:\t\t\t"+countmaximus);
}


}
```