Chapter 1

INTRODUCTION

This project - "auctoBay – An online auction portal", is a platform that allow users at any site to sell and buy products. The sellers set up auctions for their products while the purchaser who bids the highest amount wins the right to purchase the product in an auction.

The actors are - Bidder, Seller, Delivery Department, Visitor, Admin and Customer Care department.

The Seller Agent provides the function of registering goods for an auction to the sellers. This design maximizes the probability that the product auctioned will sell. The second agent is the Purchaser Agent that requires bidding to buy and it suggests a proper bidding price. The third agent is the Facilitator Agent that plays the role of an auctioneer and enables a bidder to look at the other person's auction history while bidding for and buying a product, the facilitator agent/auctioneer refers to the software that acts as platform for these activities. The Delivery Department can view the list of products to be delivered. The customer care department can view the complaints registered by the users. Finally, the admin has the option to check the users details, product details etc.

Chapter 2

ANALYSIS

The first part of the project is an investigation of already existing on-line auction systems around the net. We considered three of the most famous auction web sites: eBay.com, asteinrete.com and onsale.com.

The table below describes the functionality offered to the users by this three big auction systems:

Features	eBay.com	auctoBay
Home Page	Yes	Yes
Registration	Yes	Yes
Login	Yes	Yes
Search	Yes	No
Browse	Yes	No
Item Page	Yes	Yes
Bid	Yes	Yes
Post an auction	Yes	Yes
Customer Support	Yes	Yes
Change Language	Yes	No
Chat	Yes	No
Admin	Yes	Yes

2.1. ACTORS

1. Bidder
2. Seller
3. Delivery Department
4. Visitor
5. Admin
6. Customer Care

2.2. PRECONDITIONS AND POSTCONDITIONS

Precondition is a condition that must be fulfilled before other things can happen or be done.

The preconditions are

i. Creating an account before performing operation such as view item, place a bit etc.
ii. Login before making an payment.
iii. Validate login and display main page in case of invalid credentials.

A postcondition is a condition or predicate that must always be true just after the execution of some section of code or after an operation in a formal specification.

The postconditions are

i. The system must process all the given details such as login credentials, product details, complaints etc. by the actors and store it in the file.
ii. Printing payment status after login and choosing the payment mode.

2.3. ATTRIBUTES AND CLASS NAMES

1. class Bidder
   Attributes: id, name, password, contact, address
   Methods: login(), register(), viewProduct(), complaint()

2. class Seller
   Attributes: id, name, password, contact, address
   Methods: addProduct(), viewProduct()

3. class Visitor
   Methods: viewProduct()

4. class Admin

   Methods: viewProduct(), viewUserDeatils(), viewComplaint()

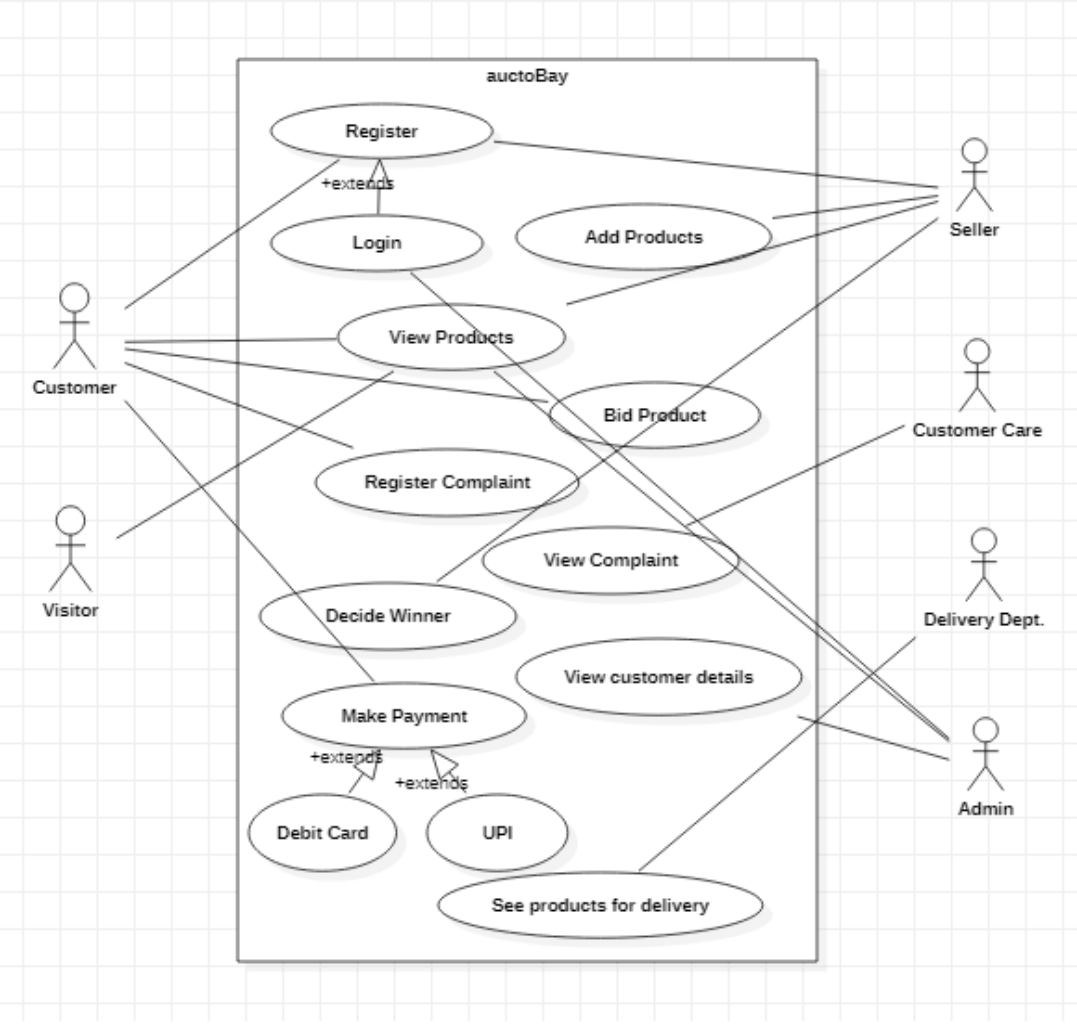
5. class CustomerCare

   Methods: viewComplaint()

6. class Delivery

   Methods: viewProduct()

2.4. USE CASE DIAGRAM



Chapter 3

DESIGN

The design of the project can be well understood with the help of the class diagram.

3.1. CLASSES WITH FINAL ATTRIBUTES AND METHODS

1. class Bidder
   Attributes: id, name, password, contact, address
   Methods: login(), register(), viewProduct(), complaint()

2. class Seller
   Attributes: id, name, password, contact, address
   Methods: addProduct(), viewProduct()

3. class Visitor
   Methods: viewProduct()

4. class Admin
   Methods: viewProduct(), viewUserDeatils(), viewComplaint()

5. class CustomerCare
   Methods: viewComplaint()

6. class Delivery
   Methods: viewProduct()

7. class Register
   Methods: login(), register()

8. class Payments

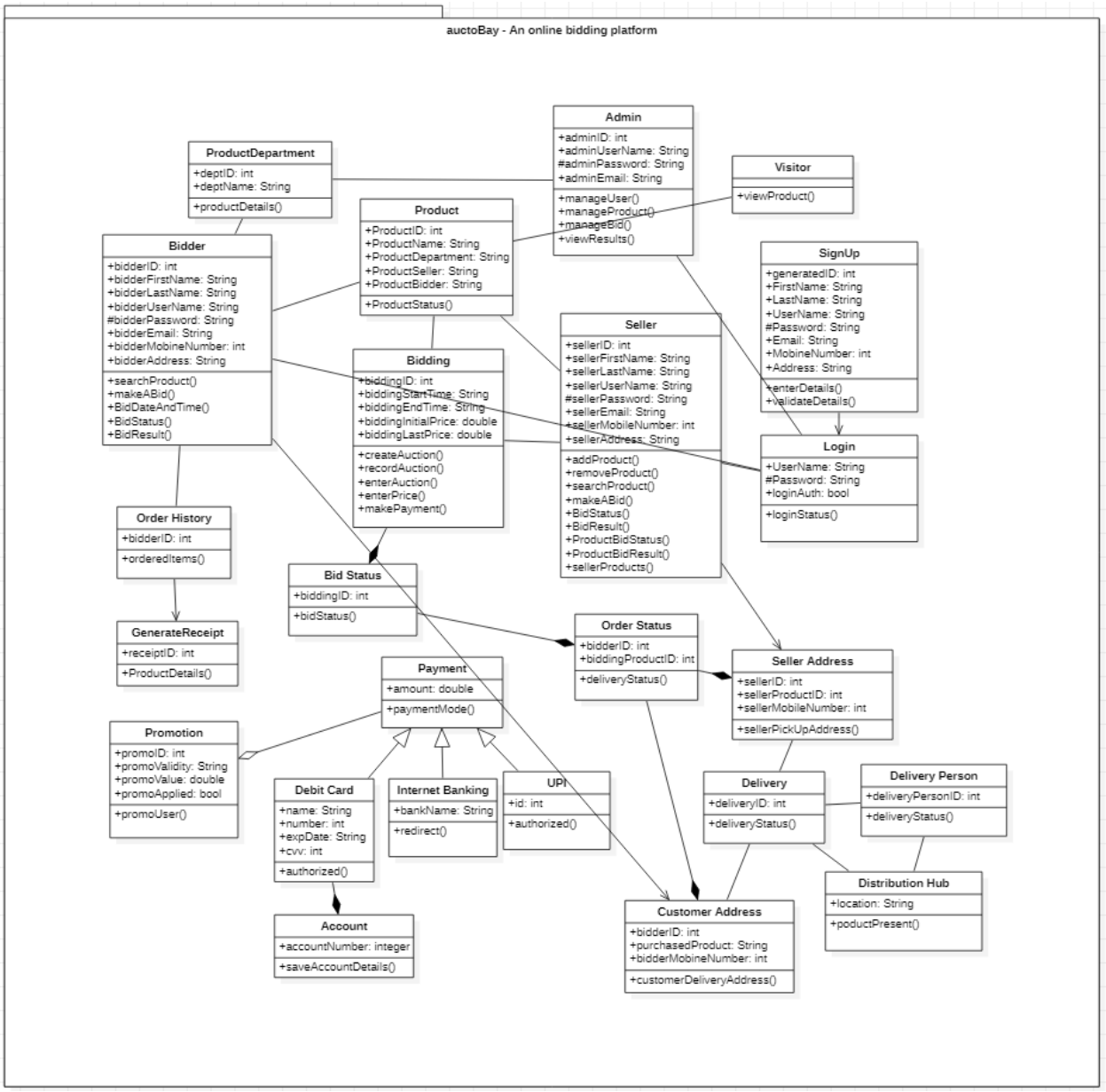
   Methods: paymentMode

3.2. CLASS RELATIONSHIPS

- Classes bidder, payments and admin depend on the login class to validate the authentication.

- Three different classes Debit, Internet banking, UPI extend the login class

3.2. CLASS DIAGRAM



Chapter 4

IMPLEMENTATION

The template of this project was generated using Star UML.

4.1. HARDWARE AND SOFTWARE REQUIREMENTS:

Hardware Requirements – PC with Software requirements installed

Software Requirements – INTELLIJ IDE, STARUML

4.2. CODE

```
import sun.font.TrueTypeFont;

import java.awt.*;
import java.io.*;
import java.util.*;

public class Main {

   public static void main(String[] agrs) throws IOException {

      Bidder b = new Bidder();
      Seller s = new Seller();
      Delivery d = new Delivery();
      Visitor v = new Visitor();
      Admin a = new Admin();
      CustomerCare c = new CustomerCare();


      int ch,ch2;
```

```
Scanner input = new Scanner(System.in);
while(true) {
System.out.println("\n1.Bidder\n2.Seller\n3.Delivery\n4.Visitor\n5.Admin\n6.Customer Care\n7.Payment\n8.Exit");
System.out.print("Enter Choice : ");
ch = input.nextInt();

//
   switch (ch) {
      case 1: //bidder
         System.out.println("\n\t1.Login\n\t2.Register\n\t3.Place Bids\n\t4.Write Complaint\n\t5.Exit");
         System.out.print("Enter Choice : ");
         ch2 = input.nextInt();
         switch (ch2) {

            case 1:
               System.out.print("Enter user name : ");
               String uname = input.next();
               System.out.print("Enter password : ");
               String pass = input.next();
               boolean l = b.login(uname, pass);
               break;

            case 2: b.register();
                  break;

            case 3: b.viewProduct();
                  break;

            case 4: b.complaint();
                  break;

            case 5: System.exit(0);
```

```
                break;
            }
            break;

        case 2: //seller
            System.out.println("\n\t1. View Product\n\t2. Add Product\n\t5. Exit");
            System.out.print("Enter Choice : ");
            ch2 = input.nextInt();
             switch (ch2) {
                case 1:
                    s.viewProduct();
                    break;

                case 2: s.addProduct();
                    break;

                case 3: System.exit(0);
                    break;

             }
            break;

        case 3: //delivery
            System.out.println("\n\t1. View Products to be delivered\n\t2.Exit");
            System.out.print("Enter Choice : ");
            ch2 = input.nextInt();
            switch (ch2) {
                case 1:
                    d.viewProduct();
                    break;

                case 2: System.exit(0);
                    break;
```

```
            }
            break;

        case 4: //visitor
            while(true) {
                System.out.println("\n\t1. View Products to be delivered\n\t2. Exit");
                System.out.print("Enter Choice : ");
                ch2 = input.nextInt();
                switch (ch2) {
                    case 1:
                        v.viewProduct();
                        break;

                    case 2:
                        System.exit(0);
                        break;

                }
                break;
            }

        case 5: //Admin
            System.out.println("\n\t1. View User Details\n\t2. View Products\n\t3. View
Complaint\n\t4. Exit");
            System.out.print("Enter Choice : ");
            ch2 = input.nextInt();
            switch (ch2) {

                case 1:
                    a.viewUserDeatils();
                    break;

                case 2: a.viewProduct();
                    break;
```

```
            case 3: a.viewComplaint();
                break;

            case 4: System.exit(0);
                break;
        }
        break;

    case 6: //Customer Care
        System.out.println("\n\t1. View Complaints\n\t2. Exit");
        System.out.print("Enter Choice : ");
        ch2 = input.nextInt();
        switch (ch2) {
            case 1:
                c.viewComplaint();
                break;

            case 2: System.exit(0);
                break;
        }
        break;

    case 7: //Payments
        System.out.print("You need to login to make payments!\n");
        System.out.print("Enter user name : ");
        String uname = input.next();
        System.out.print("Enter password : ");
        String pass = input.next();
        boolean l = b.login(uname, pass);
        if (l!=true){
            System.out.print("Authentication Failed : ");
            break;
        }
```

```
            System.out.println("\nChoose mode of payment:\n\t1. Debit\n\t2.Internet
Banking\n\t3.UPI");
            System.out.print("Enter Choice : ");
            ch2 = input.nextInt();
            switch (ch2) {
               case 1:
                  System.out.println("Redirecting to Debit Card Payments Page! ");
                  break;

               case 2:
                  System.out.println("Redirecting to Internet Banking Page! ");
                  break;

               case 3:
                  System.out.println("Redirecting to UPI Payments Page! ");
                  break;

               case 4: System.exit(0);
                  break;

            }
            break;

         case 8: System.exit(0);

            break;

         default:
            System.out.println("Invalid Choice");

            break;
      }
   }
```

```
    }
}

class Bidder{
    int id;
    String name, password, contact, address;

    Scanner input = new Scanner(System.in);
    //login
    public boolean login(String username, String password) throws FileNotFoundException {
        File file=new File("login.txt");
        Scanner sc=new Scanner(file);
        boolean t=true;
        String s="";
        String x[]=new String[4];
        while(sc.hasNextLine())
        {
            s=sc.nextLine();
            x=s.split("\t");
            if(username.equals(x[0])&&password.equals(x[1]))
            {
                t=true;
                System.out.println("Login Successfull.!!!!");

                break;
            }
            else t=false;
        }
        return t;

    }

    //register
    void register() throws IOException {
```

```
        System.out.print("Enter id: ");
        id = input.nextInt();
        System.out.print("Enter name: ");
        name = input.next();
        System.out.print("Enter password: ");
        password = input.next();
        System.out.print("Enter contact: ");
        contact = input.next();
        System.out.print("Enter address: ");
        address = input.next();

        File file = new File("bidders.txt");
        FileWriter fr = new FileWriter(file, true);
        BufferedWriter br = new BufferedWriter(fr);
        br.write(id+"\t"+name+"\t"+password+"\t"+contact+"\t"+address+"\n");
        br.close();
        fr.close();
        System.out.println("Registration Successfull..!!!!");

        file = new File("login.txt");
        fr = new FileWriter(file, true);
        br = new BufferedWriter(fr);
        br.write(name+"\t"+password+"\n");
        br.close();
        fr.close();
    }

    //view product
    void viewProduct() throws IOException {
        File fi = new File("products.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
        //desktop.close();
```

```
    }

    //write complaint
    void complaint() throws IOException {
        System.out.print("Enter complaint: ");
        String c = input.nextLine();

        File file = new File("complaint.txt");
        FileWriter fr = new FileWriter(file, true);
        BufferedWriter br = new BufferedWriter(fr);
        br.write(c+"\n");
        br.close();
        fr.close();
        System.out.println("Complaint registered!");
    }
}

class Seller{
    Scanner input = new Scanner(System.in);
    int id;
    String name, password, contact, address;
    int pid; String pname;

    void addProduct() throws IOException {
        System.out.print("Enter id: ");
        pid = input.nextInt();

        System.out.print("Enter product name: ");
        pname = input.next();

        File file = new File("products.txt");
        FileWriter fr = new FileWriter(file, true);
        BufferedWriter br = new BufferedWriter(fr);
        //br.write(id+"\t"+name+"\t"+password+"\t"+contact+"\t"+address+"\n");
```

```
        br.write(pid+"\t"+pname+"\n");
        br.close();
        fr.close();
        System.out.println("Registration Successfull..!!!!");
    }

    void viewProduct() throws IOException {
        File fi = new File("products.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
        //desktop.close();
    }

}

class Delivery{

    void viewProduct() throws IOException {
        File fi = new File("products.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);

    }
}
class Visitor{

    void viewProduct() throws IOException {
        File fi = new File("products.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
```

```
    }
}

class Admin{

    void viewUserDeatils() throws IOException {
        File fi = new File("bidders.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
    }

    void viewProduct() throws IOException {
        File fi = new File("products.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
    }

    void viewComplaint() throws IOException {
        File fi = new File("complaint.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
    }

}

class CustomerCare{
    void viewComplaint() throws IOException {
        File fi = new File("complaint.txt");
        Desktop desktop = Desktop.getDesktop();
        if(fi.exists())
            desktop.open(fi);
```

```
    }

}

interface Payments{
    void Payment();
}


class Debit implements Payments{
    void Payment() throws IOException {
        System.out.println("Redirecting to Debit Card Payments Page! ");                    }
}

class InternetBanking implements Payments{
    void Payment() throws IOException {
        System.out.println("Redirecting to Internet Banking Payments Page! ");                    }
}

class UPI implements Payments{
    void Payment() throws IOException {
        System.out.println("Redirecting to UPI Payments Page! ");                    }
}
```

Chapter 4

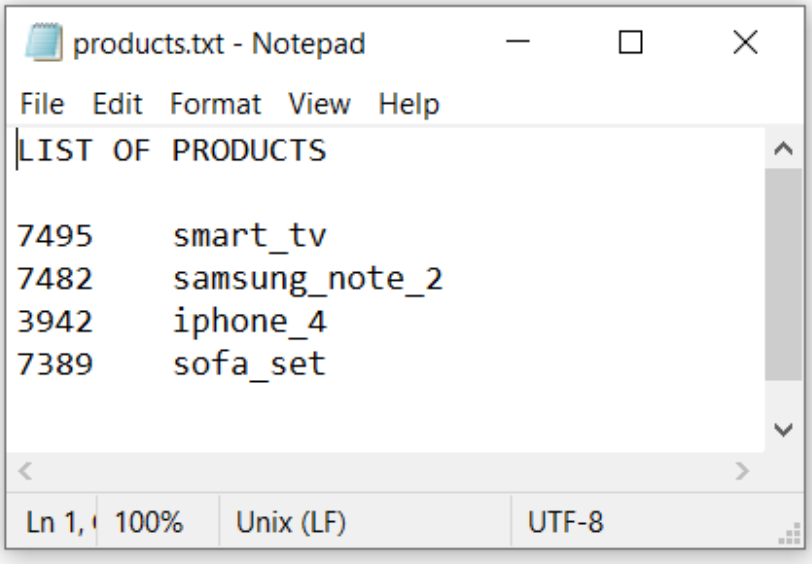
OUTPUT

1. BIDDER

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 1

    1.Login
    2.Register
    3.Place Bids
    4.Write Complaint
    5.Exit
Enter Choice : 1
Enter user name : kartik
Enter password : 123
Login Successfull.!!!!
```

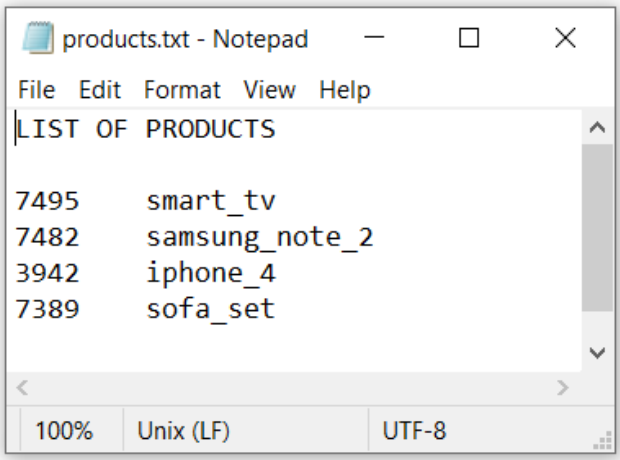
2. SELLER

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 2

    1. View Product
    2. Add Product
    5. Exit
Enter Choice : 1
```



3. DELIVERY

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 3

    1. View Products to be delivered
    2.Exit
Enter Choice : 1
```

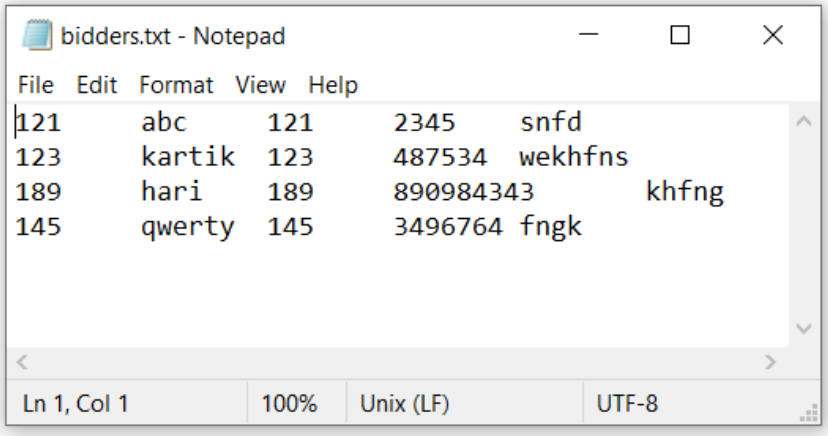


4. VISITOR

```
Enter Choice : 4

    1. View Products to be delivered
    2. Exit
```

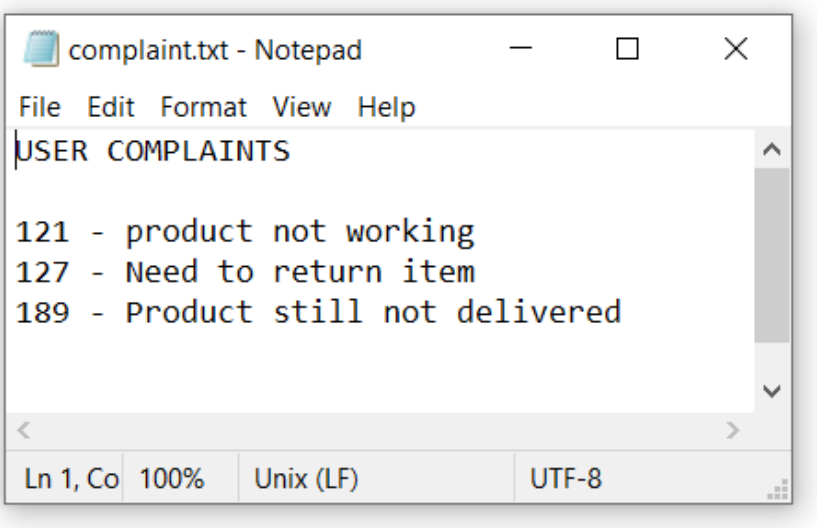
5. ADMIN

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 5

    1. View User Details
    2. View Products
    3. View Complaint
    4. Exit
Enter Choice : 1
```



6. CUSTOMER CARE

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 6

    1. View Complaints
    2. Exit
Enter Choice : 1
```



7. PAYMENT

```
1.Bidder
2.Seller
3.Delivery
4.Visitor
5.Admin
6.Customer Care
7.Payment
8.Exit
Enter Choice : 7
You need to login to make payments!
Enter user name : kartik
Enter password : 123
Login Successfull.!!!!

Choose mode of payment:
    1. Debit
    2.Internet Banking
    3.UPI
Enter Choice : 3
Redirecting to UPI Payments Page!
```

Chapter 6

REFERENCES

(i) Ali Bahrami, “Object Oriented Systems Development”, Tata McGraw-Hill, 2008

(ii) Martin Fowler, “UML Distilled”, Third Edition, PHI/Pearson Education, 2011 Edition.