VENDING MACHINE SYSTEM

PROJECT REPORT

MADE BY :-

1)LIT2020012 AAYUSH KULKARNI

2)LIT2020041 ABHIJEET ANAND

3)LIT2020042 SHUBHAM KUMAR

4)LIT2020047 VIVEK RANJAN

OUR PROJECT :-

VENDING MACHINE SYSTEM

Our Vending Machine System's main goal was to digitalise the buying of a drink and beverage. This system maintains a list of drinks available in it providing necessary information about the drinks such as its name, its price and the calorie intake in each drink. The system has two modes, mainly consumer and maintenance mode.

1) In Consumer mode the consumer who wants to buy a drink can add the sufficient amount of money and buy drinks. The denominations of the money added are 5 Rupees, 10 Rupees and 50 Rupees. While buying drinks the cost and the calorie intake of the drinks along with the drinks name are shown for the consumer's sake. The system can also refund the credits if the consumer wants it.
2) In Maintenance mode the person adds new drinks, Restock the system if stock is emptied, edit the information of the currently available drinks and can print the status of the

machine which displays the stock left in the machine and prints the money earned by selling the drinks.

MAIN GOAL :-

The main goal of the system is to correctly sell beverages to the people without any mismanagement of money. The process is done by:-

1) The consumer selects consumer mode and adds a sufficient amount of money to buy beverages. He then selects the available beverage and checksout.
2) Refunds the money to the consumer if wanted.
3) In the maintenance mode the system provides the option to add new drinks.
4) The system can display the money earned by it after various purchases by the consumers.

BENEFITS OF OOM:-

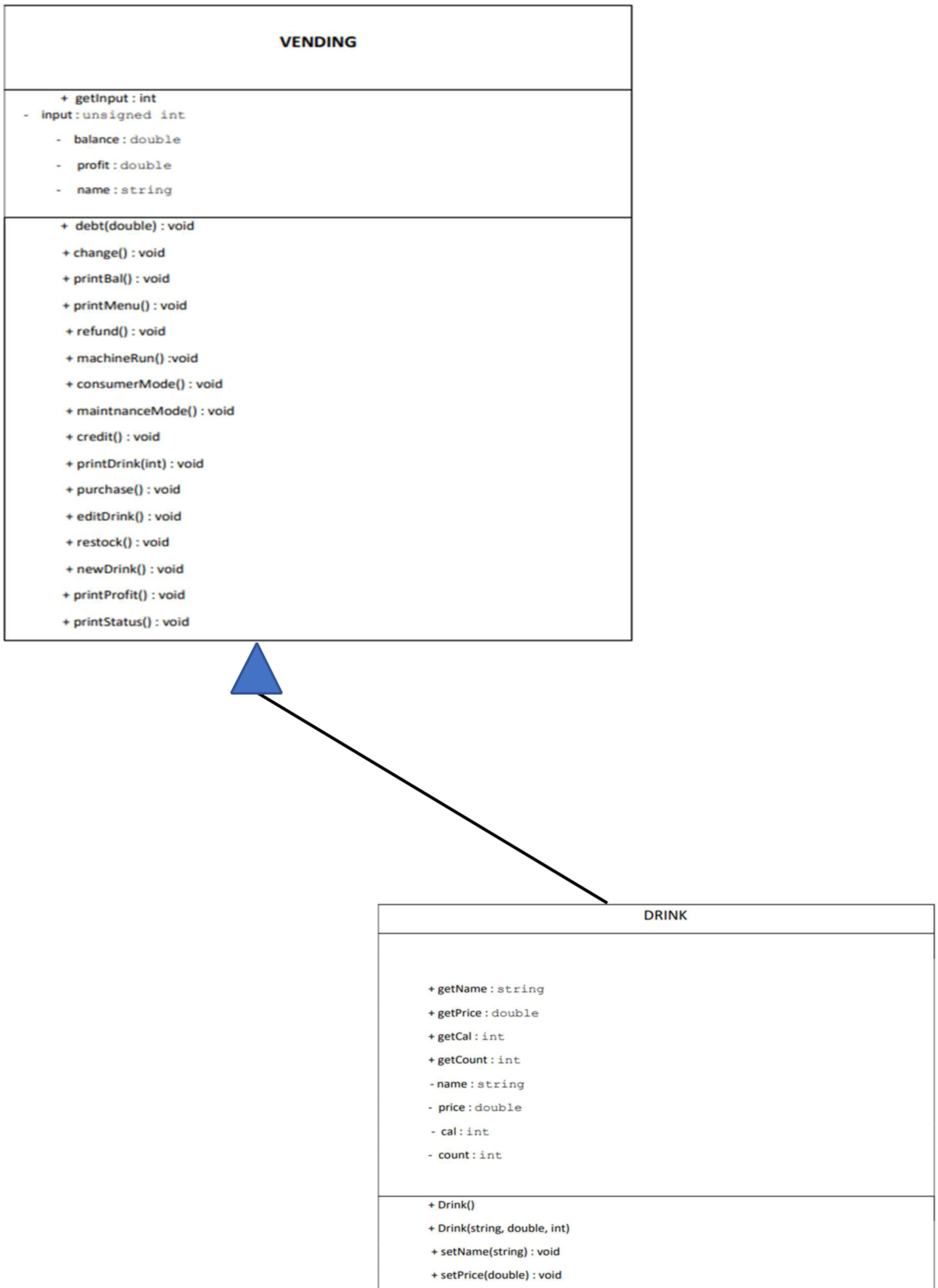
Object oriented contributes to the solution of many problems associated with the development quality of software products. The new technology promises greater programmer productivity, better quality of software and lesser maintenance cost. The principal advantages are:

1) Through inheritance, we can eliminate redundant code and extend the use of existing classes.
2) We can build programs from the standard working modules that communicate with one another rather than having to start writing the code from scratch. This leads to saving of development time and higher productivity.
3) The principle of data hiding helps the programmer to build secure programs that cannot be invaded by code in other parts of the program •It is possible to map objects in the problem domain to those in the program.

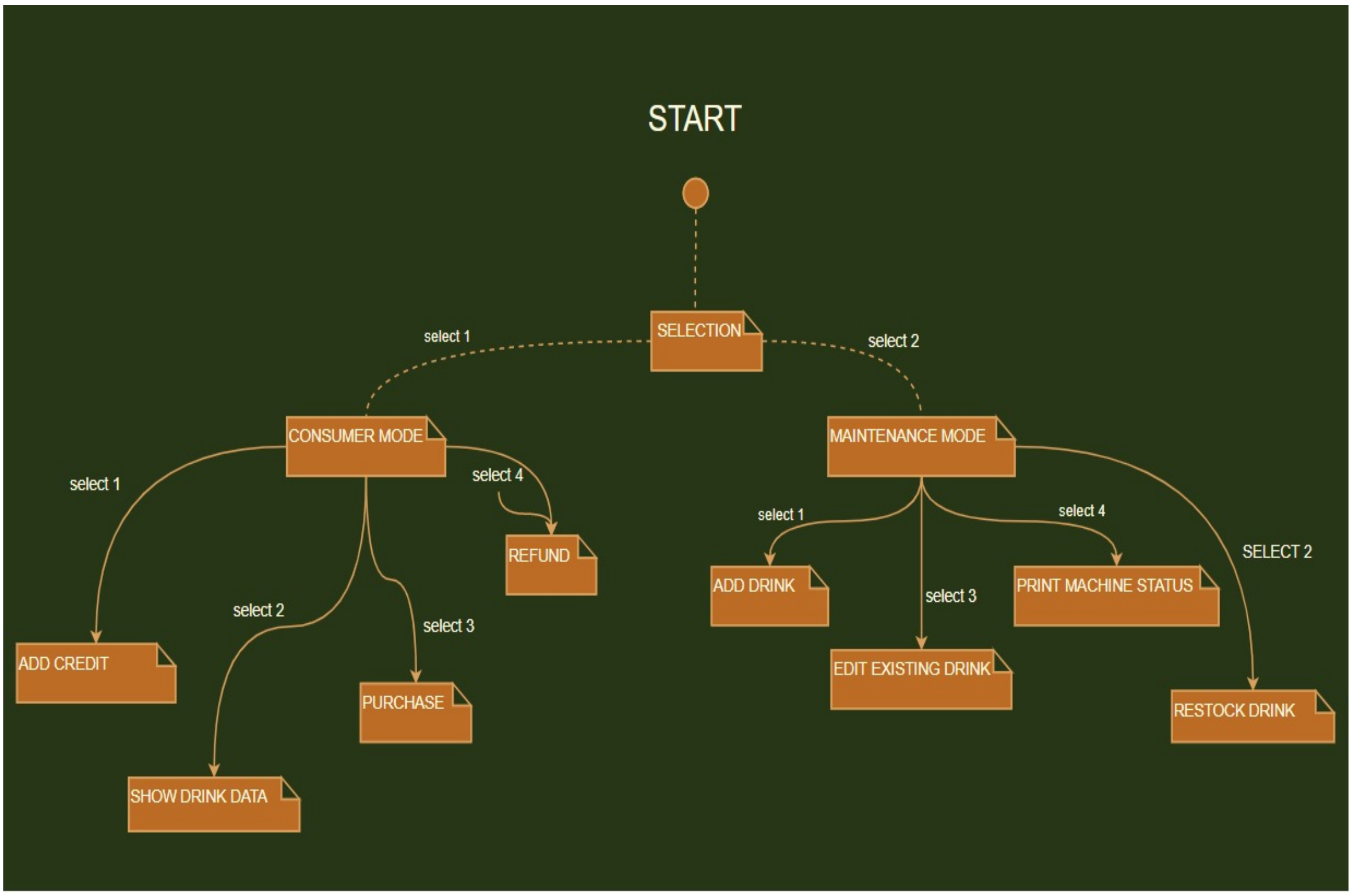
4) It is easy to partition the work in a project based on objects.
5) Object-oriented systems can be easily upgraded from small to large systems.
6) Message passing techniques for communication between objects make the interface descriptions with external systems much simpler.Software complexity can be easily managed

SYSTEM DESIGN :-

1) CLASS DIAGRAM :-


VENDING
+ getInput : int
- input : unsigned int
- balance : double
- profit : double
- name : string
+ debt(double) : void
+ change() : void
+ printBal() : void
+ printMenu() : void
+ refund() : void
+ machineRun() :void
+ consumerMode() : void
+ maintnanceMode() : void
+ credit() : void
+ printDrink(int) : void
+ purchase() : void
+ editDrink() : void
+ restock() : void
+ newDrink() : void
+ printProfit() : void
+ printStatus() : void
DRINK
+ getName : string
+ getPrice : double
+ getCal : int
+ getCount : int
- name : string
- price : double
- cal : int
- count : int
+ Drink()
+ Drink(string, double, int)
+ setName(string) : void
+ setPrice(double) : void

2)Activity Diagram :-



System Structure :-

- We used aggregation for building the system. Vending class aggregates the drink class.

Naming Conventions:

	Naming Conventions
Header Files	<string> <vector> <iostream> <time.h> <ionmanip>
Classes	Drink, Vending
Methods	getName(), getCal(), getPrice(), setName(), setCal(), setPrice(), getCount(), setCount(), minusCount() debt(money), change(), printBal(), printMenu(), getInput(int), refund(), machineRun(), consumerMode(), maintenanceMode(), credit(), printDrink(int), purchase(), editDrink(), restock(), newDrink(), printProfit(), printStatus()
Fields	name, price, cal, count Menu, input, balance, profit, name
Program Files	Drink.h, Vending.h, Drink.cpp, Vending.cpp, main.cpp

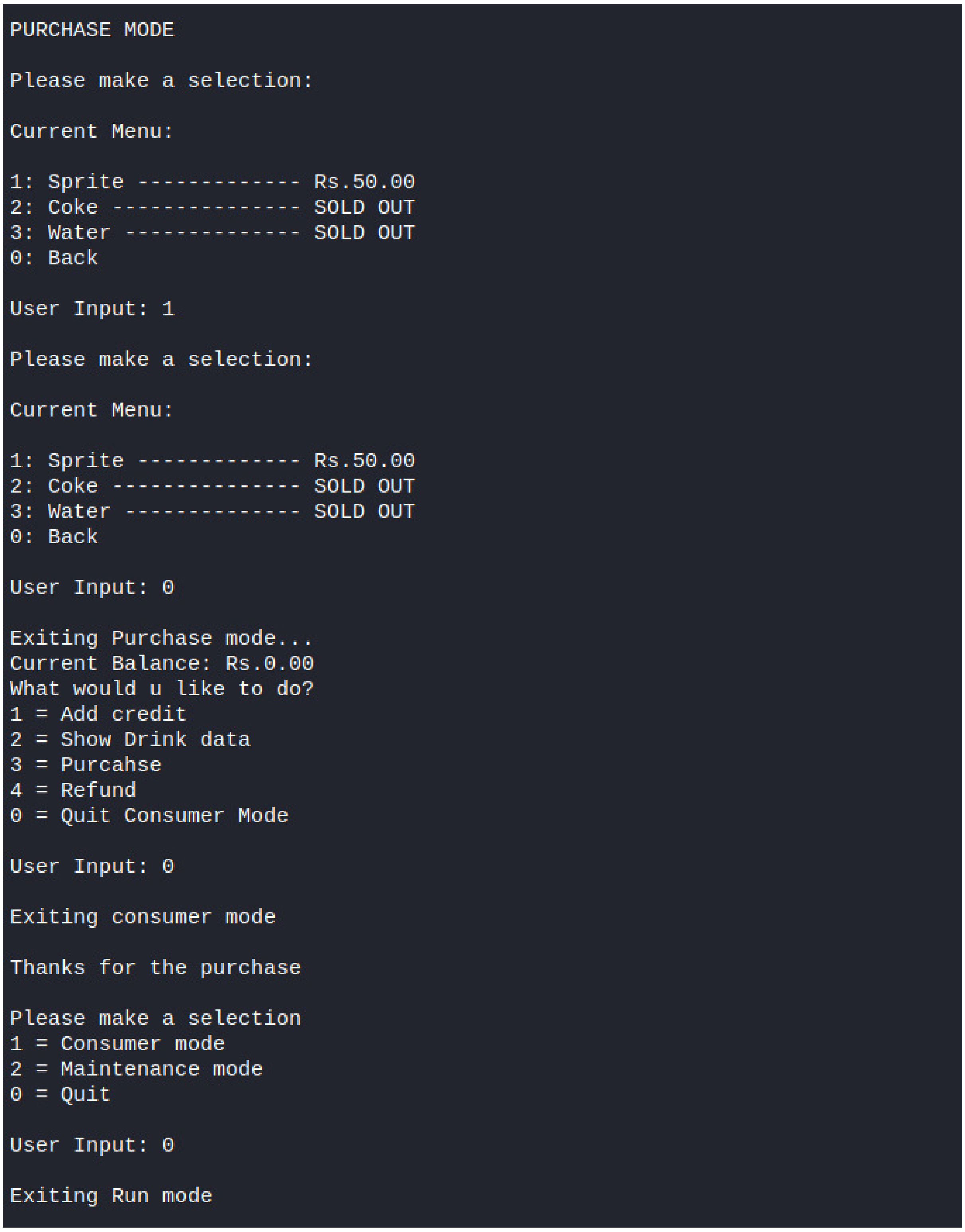
Conclusion :-

With this system the goals were :-

1) To digitalize the purchase of beverages.
2) To reduce the time taken in manual purchase.
3) To make sure to give a refund if needed.
4) To make sure if a new drink comes in the market the system can occupy the new drink in it.
5) To display the money earned by the system after various purchases.

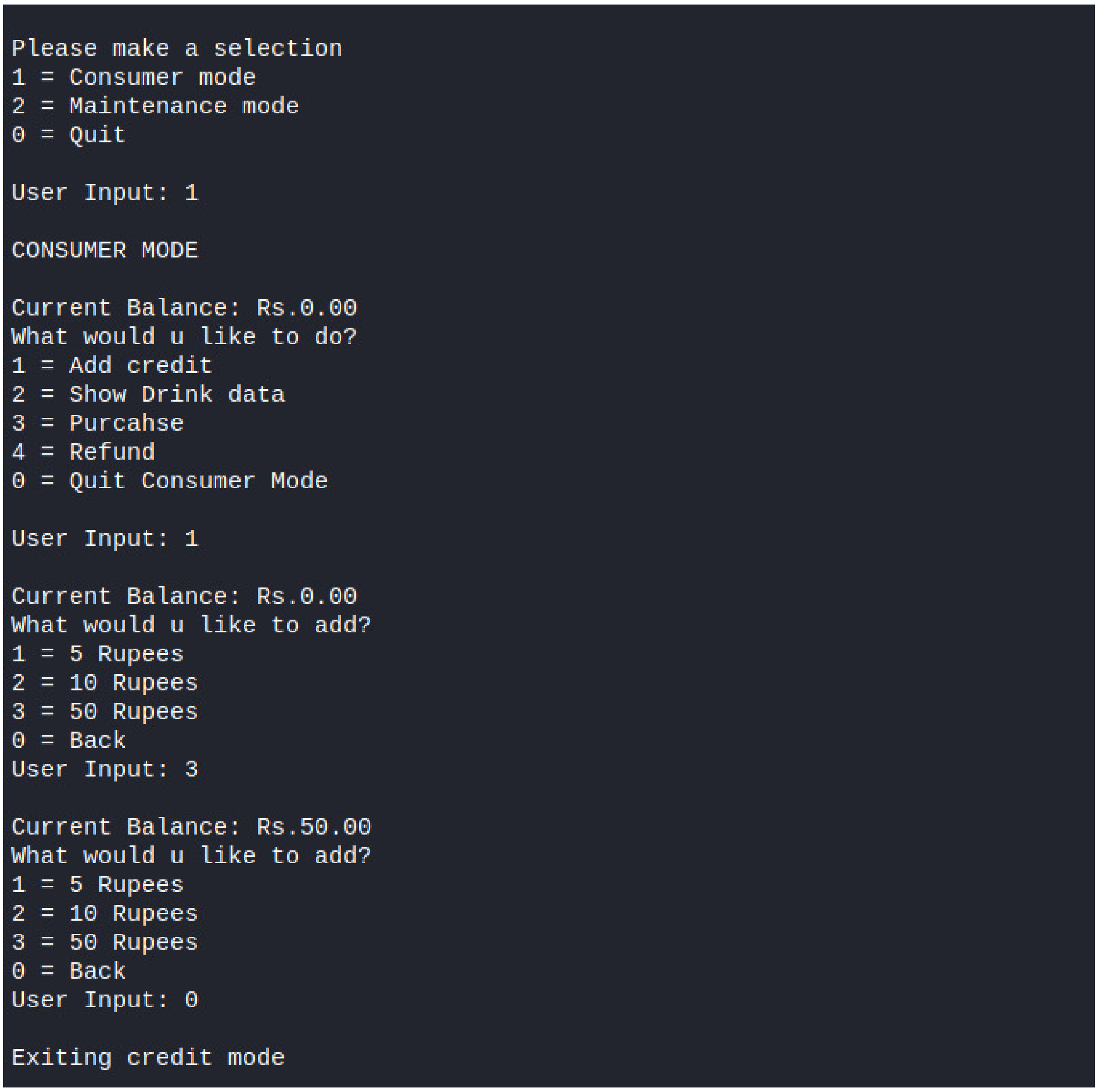
Some Screenshots of the System :-

1) Purchasing a Drink :-



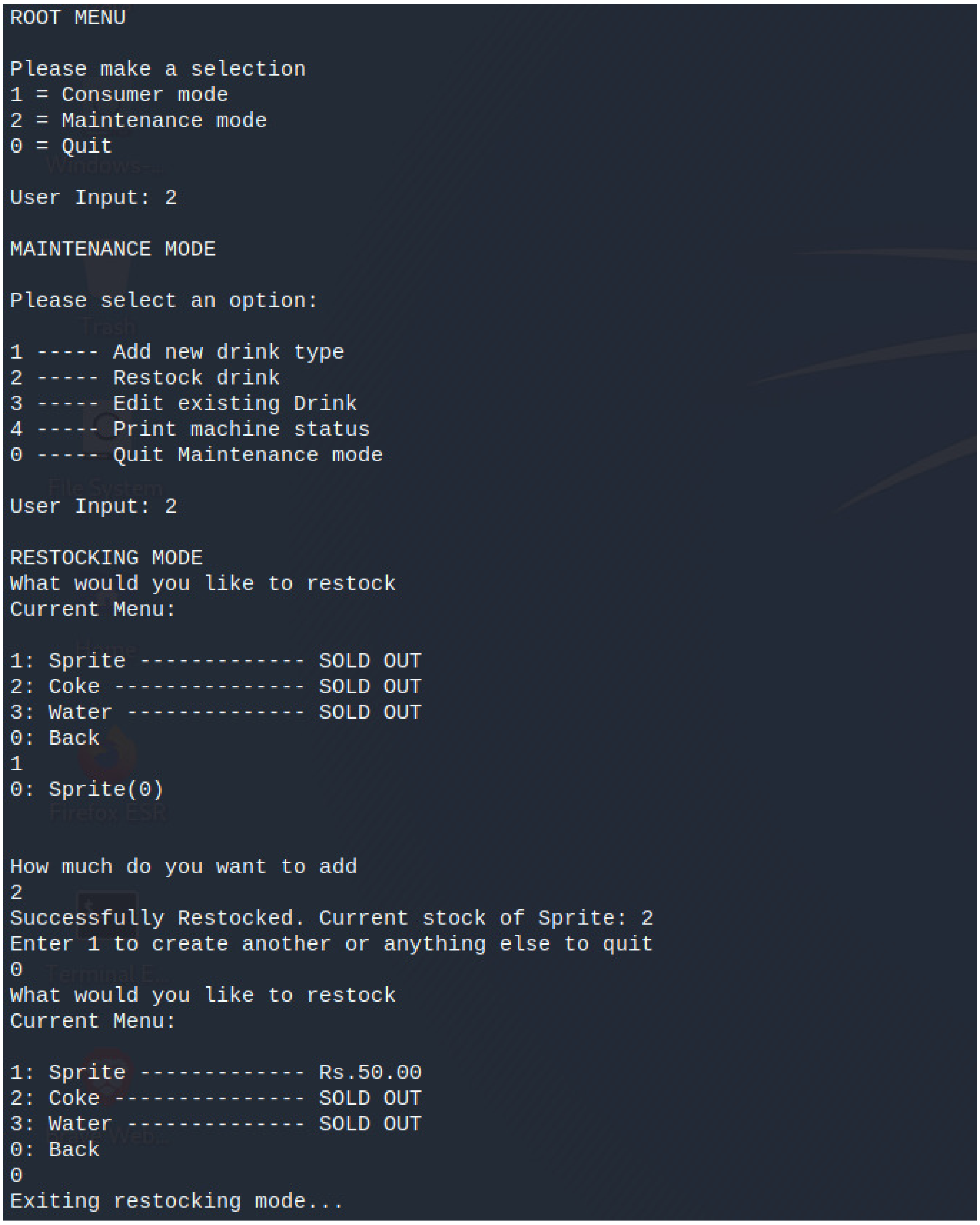
```
PURCHASE MODE

Please make a selection:

Current Menu:

1: Sprite ------------- Rs.50.00
2: Coke --------------- SOLD OUT
3: Water -------------- SOLD OUT
0: Back

User Input: 1

Please make a selection:

Current Menu:

1: Sprite ------------- Rs.50.00
2: Coke --------------- SOLD OUT
3: Water -------------- SOLD OUT
0: Back

User Input: 0

Exiting Purchase mode...
Current Balance: Rs.0.00
What would u like to do?
1 = Add credit
2 = Show Drink data
3 = Purcahse
4 = Refund
0 = Quit Consumer Mode

User Input: 0

Exiting consumer mode

Thanks for the purchase

Please make a selection
1 = Consumer mode
2 = Maintenance mode
0 = Quit

User Input: 0

Exiting Run mode
```

2) Putting Money



```
Please make a selection
1 = Consumer mode
2 = Maintenance mode
0 = Quit

User Input: 1

CONSUMER MODE

Current Balance: Rs.0.00
What would u like to do?
1 = Add credit
2 = Show Drink data
3 = Purcahse
4 = Refund
0 = Quit Consumer Mode

User Input: 1

Current Balance: Rs.0.00
What would u like to add?
1 = 5 Rupees
2 = 10 Rupees
3 = 50 Rupees
0 = Back
User Input: 3

Current Balance: Rs.50.00
What would u like to add?
1 = 5 Rupees
2 = 10 Rupees
3 = 50 Rupees
0 = Back
User Input: 0

Exiting credit mode
```

3) Maintenance Mode (Restocking a Drink)



```
ROOT MENU

Please make a selection
1 = Consumer mode
2 = Maintenance mode
0 = Quit

User Input: 2

MAINTENANCE MODE

Please select an option:

1 ----- Add new drink type
2 ----- Restock drink
3 ----- Edit existing Drink
4 ----- Print machine status
0 ----- Quit Maintenance mode

User Input: 2

RESTOCKING MODE
What would you like to restock
Current Menu:

1: Sprite ------------- SOLD OUT
2: Coke --------------- SOLD OUT
3: Water -------------- SOLD OUT
0: Back
1
0: Sprite(0)


How much do you want to add
2
Successfully Restocked. Current stock of Sprite: 2
Enter 1 to create another or anything else to quit
0
What would you like to restock
Current Menu:

1: Sprite ------------- Rs.50.00
2: Coke --------------- SOLD OUT
3: Water -------------- SOLD OUT
0: Back
0
Exiting restocking mode...
```