ISSN 2250-3153

Bluetooth Based Departmental Store Supporting System

Debopam Mukherje, Koustav Ghosh, Debatrayee Roychoudhry, Subhasree Chakraborty

* Supreme Knowledge Foundation Group of Institutions, Department of Electronics and Communication

Abstract- This technical report describes the design and implementation of a Bluetooth based departmental store support system. It has been seen that shop lifting is a common phenomenon in case of the departmental stores and the shopping mall and to prevent it a large number of employees and a great number of CCTVs are installed which costs a lot to the stores or the shopping malls.

In this system a Bluetooth enabled radio device along with a display will be given to every customer when they will enter into the shop. In the shop demo products and shopping items will be placed rather than the all .The customer will choose the required products by observing the demo items and can place order through their respective devices and send the list to the server placed at the counter .Each Bluetooth device is equipped with a unique MAC ID and a bill will be generated against the MAC ID. At the counter the customer will pay the bill and also deposit his\her device.

As a matter of fact several problems may arrive such as a master Bluetooth device can communicate with a maximum of seven devices in a piconet so we can overcome this problem by installing several servers with same MAC ID.

The following methodology and result sections of this report show that the system works as per our expectation.

I. MOTIVATION

As the day passes the appearance of conventional shops become changed to make the life of people easier and smooth. In this context the concepts of shopping malls or multiplexes or departmental stores have come. The meaning of multiplex is many shops in one complex, where a person can buy different things such as cloths, food items, cosmetics and other useful things from one complex no need to go to different shops. But it is seen that, every day, some percentages of different items have stolen from the departmental stores besides having tight security and a number of attendants. Our motivation of doing this project is to minimize stealing of items from the stores. In our project, every customer who is entering in the store has given a Bluetooth radio having a unique address (Mac id). The customer can order anything from the Bluetooth radio. In the store, only some samples (2 or 3) are there, the customers can order their required items by only seeing the samples. As there are some restricted number of samples, stealing is reduced. Not only that this device also reduced the employment cost of the attendants. The device work as follows: when the customer switched on the Bluetooth radio, different categories of items are shown to the customer (such as food, clothes, vegetables etc.). From that list the customers have selected his required category, and then the list of available items with their specification of the selected category has shown to the customer. The customer then ordered items from that list and the order is directly fed to the server. At the server the bill is produced automatically, no need to calculate the bill for individual. This project minimizes manual interference and makes the process smoother. There are various advantages to use Bluetooth connectivity between the server and customer Bluetooth radio. Firstly, it will be very cost-effective. The cost of Bluetooth radio is nearly Rs.800/-, which is affordable. Bluetooth is also friendly with heart patients and the patient who having pacemaker, as it does not interfere with pacemaker. The system is very easy to implement and it does not require continuous maintenance. Finally we can say that the project is very much advantageous in all respect, and very much user friendly.

II. ADVANTAGES OF BLUETOOTH

Bluetooth is a proprietary open wireless technology standard for exchanging data over short distances (using short wavelength radio transmissions in the ISM band from 2400-2480 MHz) from fixed and mobile devices, creating personal area networks (PANs) with high levels of security. Created by telecoms vendor Ericsson in 1994, it was originally conceived as a wireless alternative to RS-232 data cables. It can connect several devices, overcoming problems of synchronization.

1. Wireless

There are many benefits and advantages of using wireless devices. Along with improving safety as a result of eliminating wires you don't need, wireless also offers you plenty of other advantages. When travelling with your laptop or other wireless devices, you'll no longer have to worry about bringing connection cables. Thus we achieve more mobility and flexibility by using wireless technologies.

2. Inexpensive

Bluetooth technology is cheap for companies to implement, which results in lower over-all manufacturing costs. These savings are then passed on to you, the consumer. The end result: Bluetooth devices are relatively inexpensive. The maintenance cost is also low for Bluetooth based systems.

3. Automatic

Bluetooth doesn't require you to think about setting up a connection by pushing too many buttons. When two or more Bluetooth devices enter a range (Up to 30 feet) of one another, one just needs to make the device discoverable and begin the communication with some simple steps, without you having to do anything extra.

4. Standardized protocol – Interoperability

Bluetooth is standardized wireless, meaning that a high level of compatibility among devices is guaranteed. Bluetooth will connect devices to each other, even if they aren't the same model.

5. Low Interference (If Any)

Bluetooth devices avoid interference with other wireless devices by:

a) Using a technique known as spread-spectrum frequency hopping, and

b) Using low power wireless signals

6. Low energy consumption

As a result of Bluetooth using low power signals, the technology requires very little energy and will use less battery or electrical power as a result. This is an excellent benefit for mobile devices, as Bluetooth won't drain the battery.

7. Sharing voice and data

The standard for Bluetooth will allow compatible devices to share data and voice communications. This is great for mobile phones and headsets, as Bluetooth simplifies driving and talking on your cell phone.

8. Instant Personal Area Network (PAN)

Up to seven compatible Bluetooth devices can connect to one another within proximity of up to 30 feet, forming a PAN or piconet. Multiple piconets can be automatically setup for a single room.

9. Upgradeable

The Bluetooth standard is upgradeable. A development group at the Bluetooth Special Interest Group (SIG) has been given the task of working on the new Bluetooth version 2, which offers several new advantages and is backward compatible with the older versions.

10. The Technology is here to Stay

Bluetooth is a universal, world-wide, wireless standard. Therefore, you can count on it being around for years to come. As more devices begin to use Bluetooth technology, electronics manufacturers will be increasingly eager to make their products compatible, using Bluetooth. A chain reaction is inevitable, in fact, it has already begun.

III. INTRODUCTION

BLUETOOTH:

The word "Bluetooth" is an anglicised version of the Scandinavian *Blåtand*/*Blåtann*, the epithet of the tenth-century king Harald I of Denmark and parts of Norway who united dissonant Danish tribes into a single kingdom. The idea of this name was proposed by Jim Kardach who developed a system that would allow mobile phones to communicate with computers (at the time he was reading Frans Gunnar Bengtsson's historical novel *The Long Ships* about Vikings and king Harald Bluetooth) The implication is that Bluetooth does the same with communications protocols, uniting them into one universal standard

Bluetooth uses a radio technology called frequency-hopping spread spectrum, which chops up the data being sent and transmits chunks of it on up to 79 bands (1 MHz each; centered from 2402 to 2480 MHz) in the range 2,400–2,483.5 MHz (allowing for guard bands). This range is in the globally unlicensed Industrial, Scientific and Medical (ISM) 2.4 GHz short-range radio frequency band. It usually performs 800 hops per second, with Adaptive Frequency-Hopping (AFH) enabled.

Figure 1: Bluetooth logo

PYTHON:

Python is a widely used general-purpose, high-level programming language. Its design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than would be possible in languages such as C. The language provides constructs intended to enable clear programs on both a small and large scale.

Python supports multiple programming paradigms, including object-oriented, imperative and functional programming styles. It features a dynamic type system and automatic memory management and has a large and comprehensive standard library.

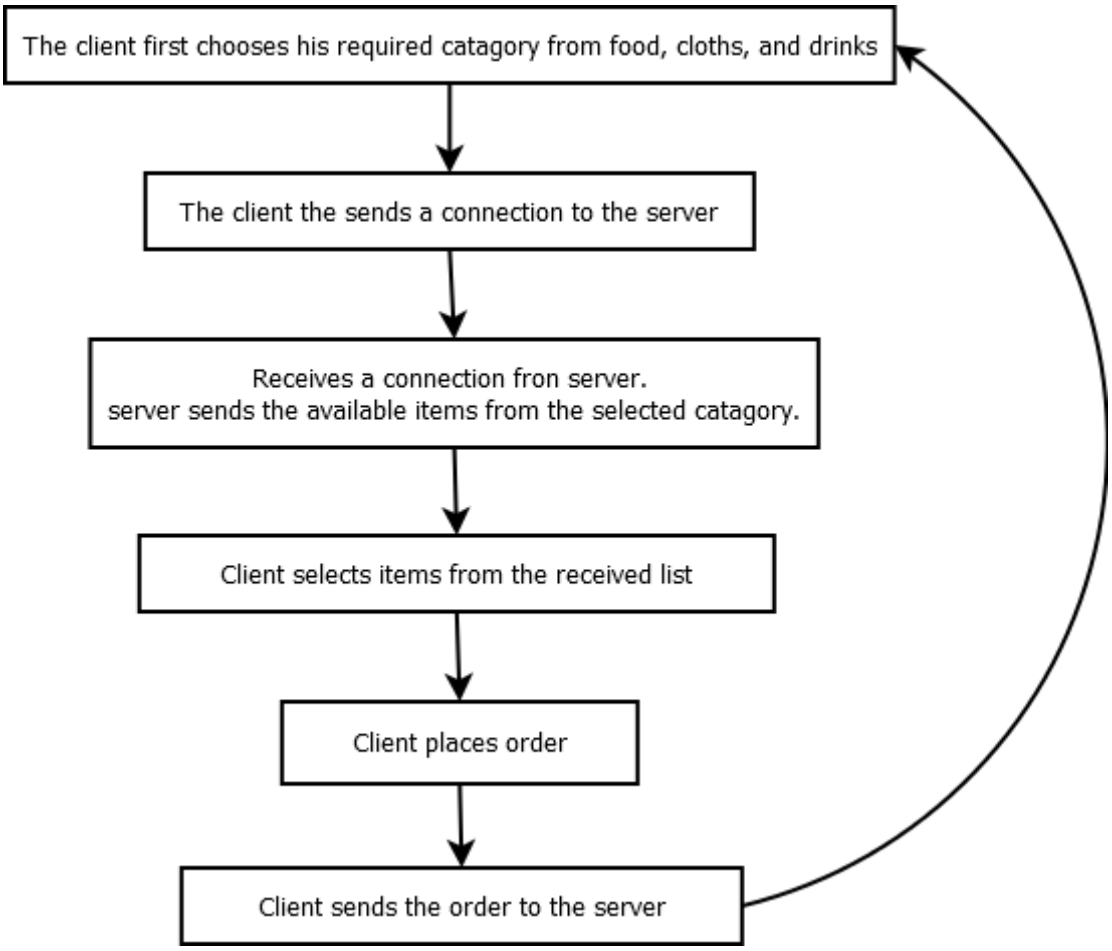
Like other dynamic languages, Python is often used as a scripting language, but is also used in a wide range of non-scripting contexts. Using third-party tools, Python code can be packaged into standalone executable programs. Python interpreters are available for many operating systems.

C Python, the reference implementation of Python, is free and open source software and has a community-based development model, as do nearly all of its alternative implementations. C Python is managed by the non-profit Python Software Foundation.

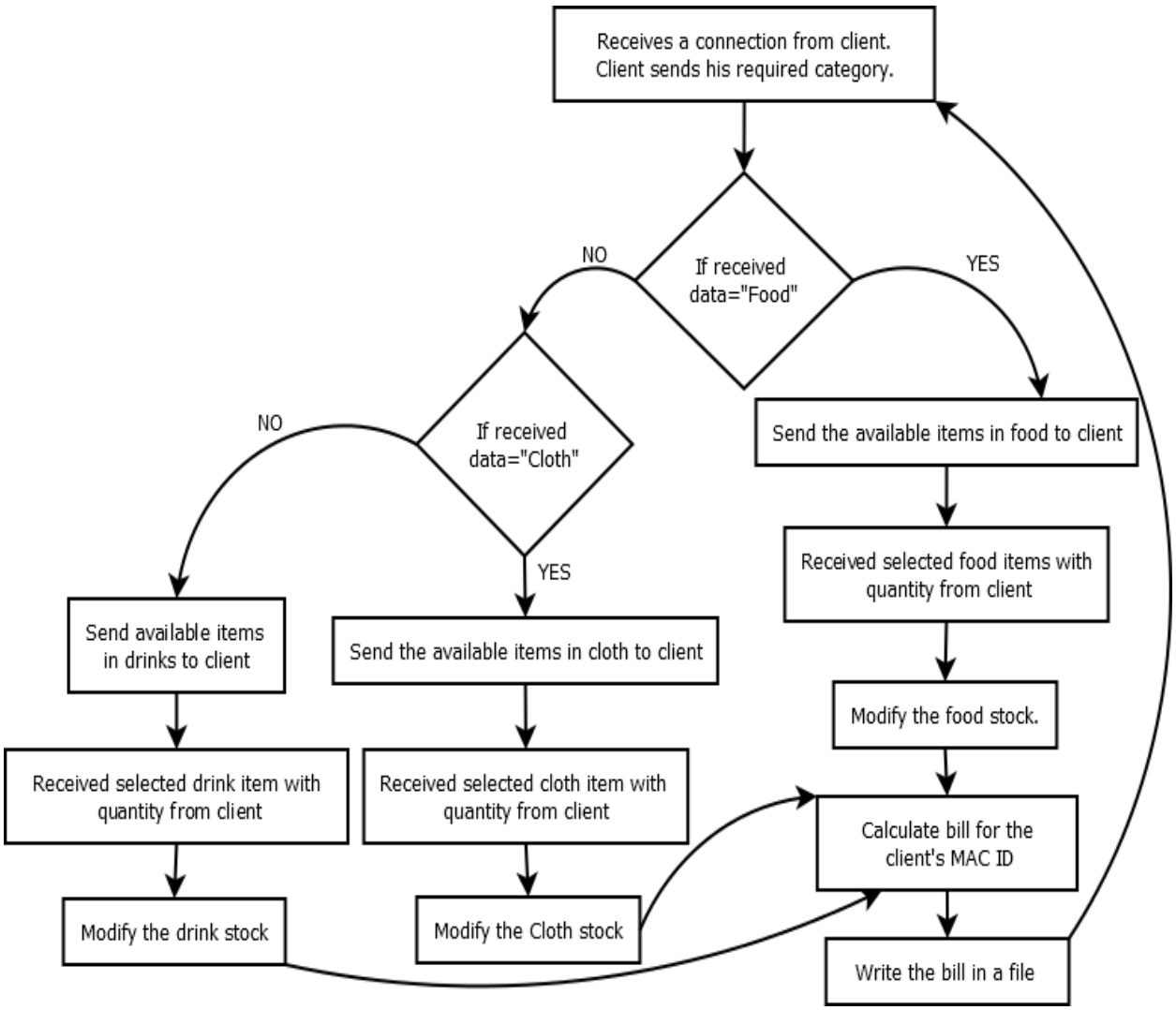
Some features of "Python" are enlisted below:

- Python is a multi-paradigm programming language: object-oriented programming and structured programming are fully supported, and there are a number of language features which support functional programming and aspect-oriented programming (including by programming and by magic methods). Many other paradigms are supported using extensions, including design by contract and logic programming.
- Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. An important feature of Python

IV. METHODOLOGY



Block diagram of Client programme



Block diagram of Server programme

V. STEP ALGORITHM

1. The client chooses a category of items like food, clothes, and drinks.
2. If the client chooses from either of the categories, the server receives a connection from the client.
3. Now, a list of items appears in the client device within the chosen category.
4. On pressing the button 'PLACE ORDER', a window appears where order for the required items to be purchased is placed.
5. After placing orders, the client presses 'OK' button.
6. If the chosen items exist in the list and the total number of items does not exceed the available quantity, then purchased items appear in the server.
7. The server then enters the MAC ID of the client and prints the bill by pressing the 'PRINT BILL' button.
8. On pressing the 'PRINT BILL' button, the bill of the purchased items is generated in the form of .html file.
9. On pressing the 'OK' button, the client returns to the main menu.
10. The same process is followed if client wants to place further orders.
11. Before closing time of the mall, a program is run which empties the dictionary where purchased items are stored.

VI. DATA STRUCTURE

The most basic data structure in Python is the **sequence**. Each element of a sequence is assigned a number - its position, or index. The first index is zero, the second index is one, and so forth.

Python has six built-in types of sequences, but the most common ones are lists and tuples which we have used in our project

Python Lists:

The list is a most versatile data type available in Python, which can be written as a list of comma-separated values (items) between square brackets. Good thing about a list that items in a list need not all have the same type.

Creating a list is as simple as putting different comma-separated values between square brackets.

Item= ['ParleG', 5,100]

Here mixed data types are the members of the list named item, where 'ParleG' is a string and 5 and 100 are the int values holding position 0, 1, 2 in the list.

We can access the different elements by the following code

Item [0] = 'ParleG'

Item [1] =5

A list can be updated by append method

Item.append ('food')

Now Item= ['ParleG', 5, 100,'food']

Python Tuple:

A tuple is a sequence of immutable Python objects. Tuples are sequences, just like lists. The only difference is that tuples can't be changed ie. tuples are immutable and tuples use parentheses and lists use square brackets.

Creating a tuple is as simple as putting different comma-separated values and a tuple ,a list can also be the members of the tuple.

tup=(12,14,9,(12,123),[1,2,3])

tup [0] =12

Python Dictionary:

A dictionary is mutable and is another container type that can store any number of Python objects, including other container types. Dictionaries consist of pairs (called items) of keys and their corresponding values.

Python dictionaries are also known as associative arrays or hash tables. The general syntax of a dictionary is as follows:

food={'Item1':['ParleG',5,100],'Item2':['Lays',10,100],'Item3':['Cadbery Silk',70,100] }

Each key is separated from its value by a colon (:), the items are separated by commas, and the whole thing is enclosed in curly braces. An empty dictionary without any items is written with just two curly braces, like this: {}.Here key values are Item1, Item 2 and Item 3.Keys are unique within a dictionary while values may not be. The values of a dictionary can be of any type, but the keys must be of an immutable data type such as strings, numbers, or tuples.

To access dictionary elements, we can use the familiar square brackets along with the key to obtain its value. Following is a simple example:

food ['Item1'] = ['ParleG',5,100]

To access dictionary elements, we can use the familiar square brackets along with the key to obtain its value. Following is a simple example:

food['Item4']= ['Cadbury Cracker',70,100]

Now the modifying dictionary will be

food={'Item1':['ParleG',5,100],'Item2':['Lays',10,100],'Item3':['CadburySilk',70,100],
'Item4':['Cadbury Cracker',70,100] }

Properties of dictionary keys

Dictionary values have no restrictions. They can be any arbitrary Python object, either standard objects or user-defined objects. However, same is not true for the keys.

There are two important points to remember about dictionary keys:

(a) More than one entry per key not allowed. Which means no duplicate key is allowed. When duplicate keys encountered during assignment, the last assignment wins.

(b) Keys must be immutable. Which means you can use strings, numbers, or tuples as dictionary keys but something like ['key'] is not allowed.

In this project we have used three types of sequences as lists, tuples, and dictionary. When the client machine sends request to server to see the items in the shop then the server send the item list through a dictionary.

food={'Item1':['ParleG',5,100],'Item2':['Lays',10,100],'Item3':['Cadbery Silk',70,100] }

cloth={'Item1':['Tshirt',500,100],'Item2':['Trousers',1000,100],'Item3':['Jeans',2000,100] }

drinks={'Item1':['Pepsi',30,100],'Item2':['Limca',30,100],'Item3':['7up',30,100] }

We have used dictionary as it works like a container and we can access the keys to get the values assigned against the key value.

Dictionary_name.keys () returns all the key values present in that dictionary

We have put the values as form of list against the key as Item1':['ParleG',5,100] ,where Item1[0] product name , Item1[1] unit price of the product and Item[2] quantity available.

We can use tuples instead of list but each time when a customer buy some product the quantity is decremented as tuple elements cannot be updated we have used list as it can be modified

VII. RESULTS AND EXPLANATION

Figure 2: Screen shot of the starting screen

- Entering in the store when the customer first switches on the Bluetooth radio the above window is displayed in the Bluetooth radio screen. From this category of items customer chooses "food item", which is send to the server.

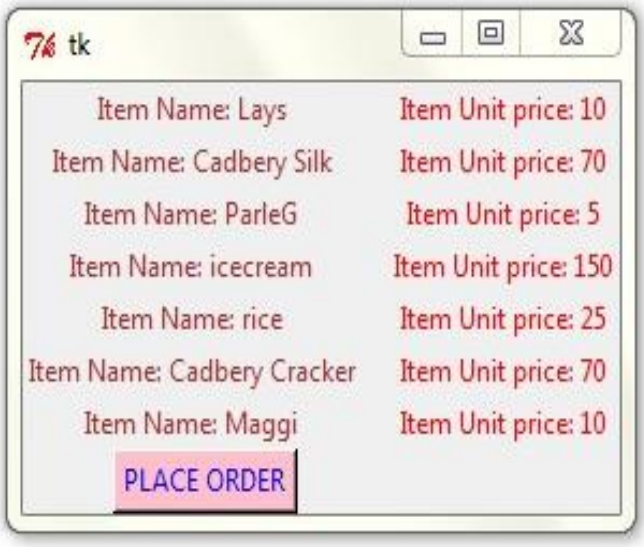


Figure 3: Screen shot of display list

- Server accepts the connection from customer. Server sends the available food items to the customer with their unit price. Now the customer can order his required food items by clicking in the "PLACE ORDER" button.

Figure 4: Screen shot of client device

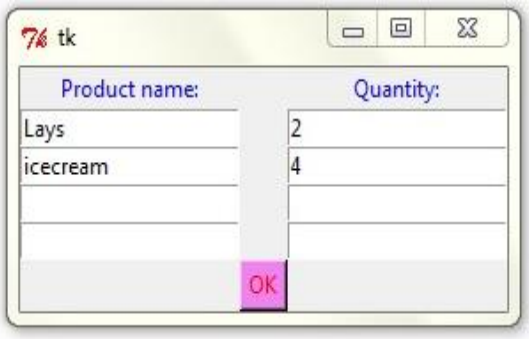


Figure 5: Screen shot of client device during shopping

- After pressing PLACE ORDER button the window (a) will appear in the screen of Bluetooth radio. Then the customer order required things and at last he will press the "OK" button, it will look like window (b). Here the customer has ordered 2 packets of Lays and 4 ice creams.

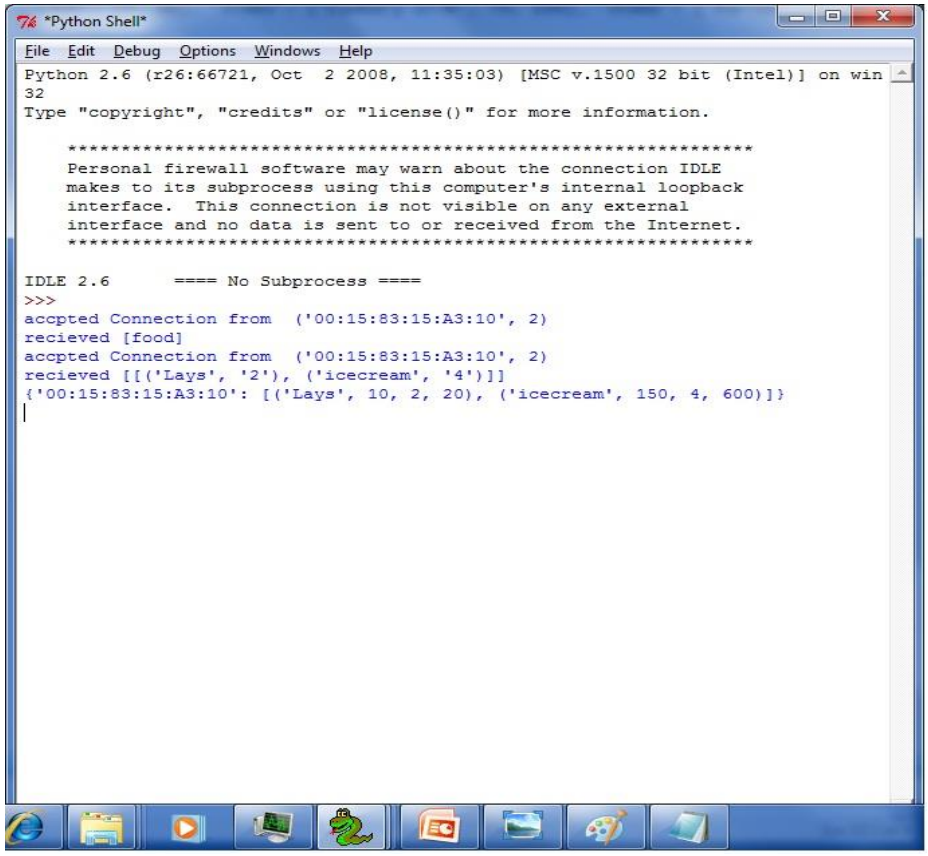


Figure 6: Screen shot of client device

- Server accepts connection from client device. The order from the customer is sent to the server. According to the order server produces bill and modify the stock.

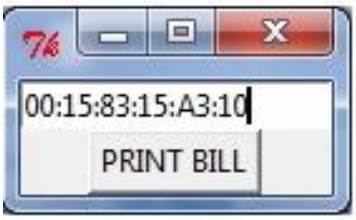


Figure 7: Screen shot of print device

- For printing bill, server put the customer's MAC ID in the entry box and after clicking the button "PRINT BILL", the bill for a specific customer (MAC ID) will save in html format. The bill is shown below-

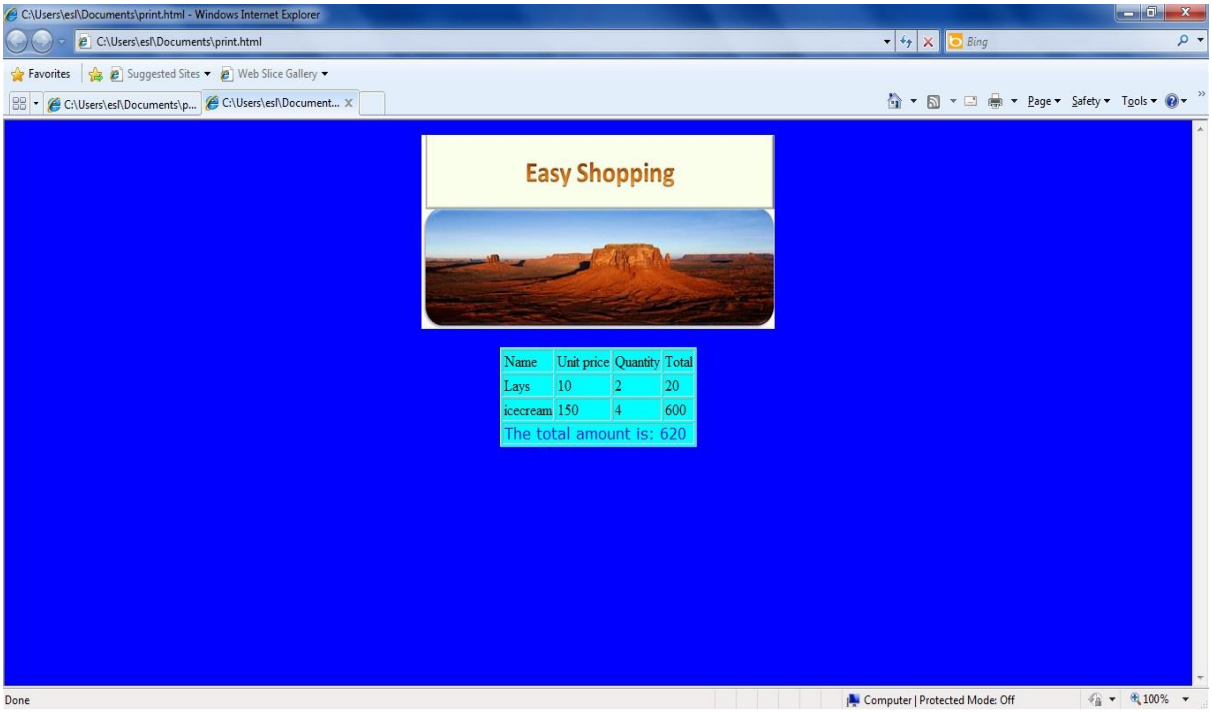


Figure 8: Screen shot of printed bill

- This is the bill for a specific customer.
- The food stock also modified according to the purchased quantity.

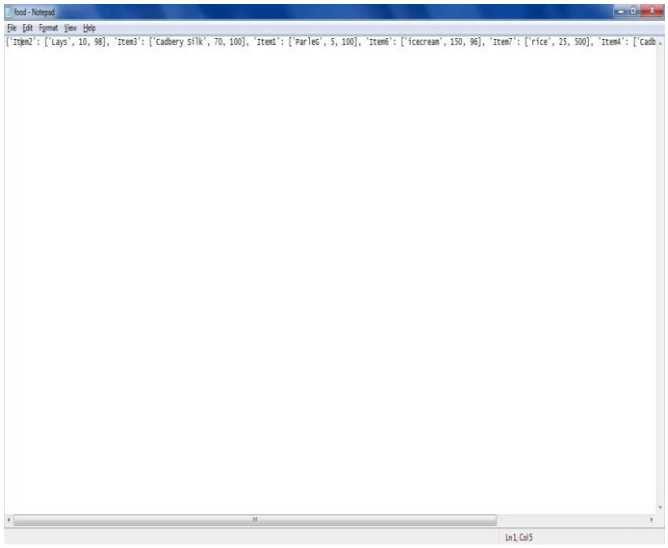


Figure 9: Screen shot before purchasing

- This the food stock before purchasing.

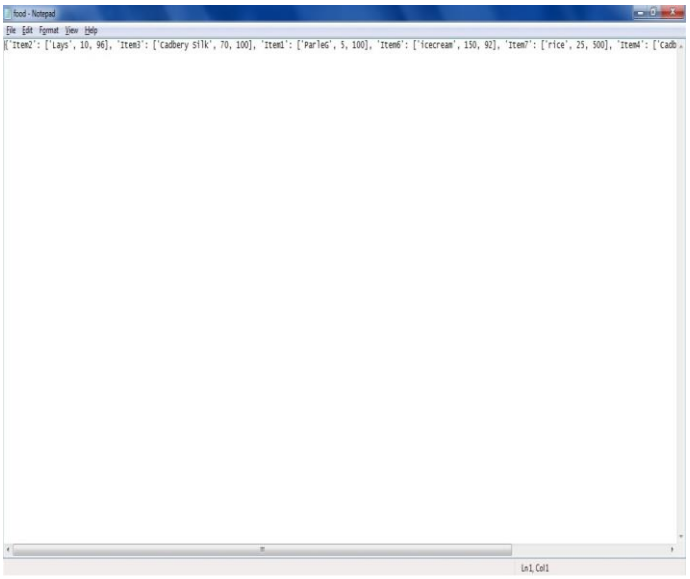


Figure 9: Screen shot before purchasing

- This is the food stock after purchasing where the quantity of Lays and ice cream is reduced according to the purchased quantity.

VIII. APPENDIX

Code for Client:

```
from Tkinter import *
import bluetooth
f=0
```

```
def ord_final():
    global root,a,a1,a2,a3,a4,a5,b1,b2,b3,b4,b5,b,root2
    temp=[]

    if (a.get()!=""):
        temp.append((a.get(),b.get()))
    if (a1.get()!=""):
        temp.append((a1.get(),b1.get()))
    if (a2.get()!=""):
        temp.append((a2.get(),b2.get()))
    if (a3.get()!=""):
        temp.append((a3.get(),b3.get()))
    bd_addr="00:15:83:3D:0A:57"
    port=2
    sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
    sock.connect((bd_addr,port))
    sock.send(str(temp))
    sock.close()
    root11.destroy()
    root2.destroy()
    main()

def order():
    global root,a,a1,a2,a3,a4,a5,b1,b2,b3,b4,b5,b,f,root11
    f=1
    root11=Tk()
    Label(root11,text="Product
name:",fg="blue").grid(row=0,column=0)

Label(root11,text="Quantity:",fg="blue").grid(row=0,column=2)
    a=Entry(root11)
    a.grid(row=1,column=0)
    b=Entry(root11)
    b.grid(row=1,column=2)

    a1=Entry(root11)
    a1.grid(row=2,column=0)
    b1=Entry(root11)
    b1.grid(row=2,column=2)

    a2=Entry(root11)
    a2.grid(row=3,column=0)
    b2=Entry(root11)
    b2.grid(row=3,column=2)
    a3=Entry(root11)
    a3.grid(row=4,column=0)
    b3=Entry(root11)
    b3.grid(row=4,column=2)
Button(root11,text="OK",fg="red",bg="violet",command=ord_fi
nal).grid(row=5,column=1)
root11.mainloop()

def food():
    global root1,stock,root,root2
    root1.destroy()
    root2=Tk()
    bd_addr="00:15:83:3D:0A:57"
    port=2
    sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
    sock.connect((bd_addr,port))
    sock.send("food")
    sock.close()

    server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
    port =2
    server_sock.bind(("",port))
    server_sock.listen(1)
    client_sock, address=server_sock.accept()
    print "accpted Connection from ", address
    mac=address[0]
    data =client_sock.recv(1024)
    print "recieved [%s]" %data
    food=eval(data)
    client_sock.close()
    server_sock.close()
    r=0
    for i in food.keys():
        Label(root2,text="Item Name: "+food[i][0]+"
",fg="brown").grid(row=r,column=0)
        Label(root2,text="Item Unit price:
"+str(food[i][1]),fg="red").grid(row=r,column=1)
        r=r+3
    Button(root2,text="PLACE
ORDER",command=order,bg="pink",fg="blue")

.grid(row=25,column=3)
    root2.mainloop()

def clothes():
    global root1,stock,root,f,root2
    root1.destroy()
    root2=Tk()
    bd_addr="00:15:83:3D:0A:57"
    port=2
    sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
    sock.connect((bd_addr,port))
    sock.send("cloth")
    sock.close()

    server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
    port =2
    server_sock.bind(("",port))
    server_sock.listen(1)
    client_sock, address=server_sock.accept()
    print "accpted Connection from ", address
    mac=address[0]
    data =client_sock.recv(1024)
    print "recieved [%s]" %data
    cloth=eval(data)
    client_sock.close()
    server_sock.close()
    r=0
    for i in cloth.keys():
        Label(root2,text="Item Name: "+cloth[i][0]+"
",fg="brown").grid(row=r,column=0)
        Label(root2,text="Item Unit price:
"+str(cloth[i][1]),fg="red").grid(row=r,column=1)
        r=r+3
```

```
   Button(root2,text="PLACE
ORDER",command=order,bg="pink",fg="blue")

.grid(row=25,column=0)
   root2.mainloop()


def drinks():
   global root1,stock,root,f,root2
   root1.destroy()
   root2=Tk()
   bd_addr="00:15:83:3D:0A:57"
   port=2
   sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
   sock.connect((bd_addr,port))
   sock.send("drinks")
   sock.close()
   server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
   port =2
   server_sock.bind(("",port))
   server_sock.listen(1)
   client_sock, address=server_sock.accept()
   print "accpted Connection from ", address
   mac=address[0]
   data =client_sock.recv(1024)
   print "recieved [%s]" %data
   drinks=eval(data)
   client_sock.close()
   server_sock.close()
   r=0
   for i in drinks.keys():
      Label(root2,text="Item Name: "+drinks[i][0]+"
",fg="brown").grid(row=r,column=0)
      Label(root2,text="Item Unit price: "+str
(drinks[i][1]),fg="red").grid(row=r,column=1)
      r=r+3
   Button(root2,text="PLACE
ORDER",command=order,bg="pink",fg="blue")

.grid(row=25,column=0)
   root.mainloop()
def main():
   global root1,root
   root1=Tk()

Label(root1,text="Welcome",fg="blue").grid(row=0,column=1)
   Button(root1,text="Food
item",command=food).grid(row=1,column=0)

Button(root1,text="clothes",command=clothes).grid(row=1,colu
mn=1)

Button(root1,text="Drinks",command=drinks).grid(row=1,colum
n=2)
   root1.mainloop()
main()
```

Code for Server:

```
import bluetooth
from Tkinter import *

def main():
   global address,data
   while(1):
      global a

server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
      port =2
      server_sock.bind(("",port))
      server_sock.listen(1)
      client_sock, address=server_sock.accept()
      print "accpted Connection from ", address
      mac=address[0]
      data =client_sock.recv(2048)
      print "recieved [%s]" %data
      client_sock.close()
      server_sock.close()

      if (data=="food"):
         bd_addr=mac
         port=2
         sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
         sock.connect((bd_addr,port))
         sock.send(str(food))
         sock.close()


server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
      port =2
      server_sock.bind(("",port))
      server_sock.listen(1)
      client_sock, address=server_sock.accept()
      print "accpted Connection from ", address
      mac=address[0]
      data =client_sock.recv(1024)
      print "recieved [%s]" %data
      final_food=eval(data)
      client_sock.close()
      server_sock.close()
      for j in food.keys():
         for k in final_food:
            if(k[0]==food[j][0]):
               food[j][2]-=int(k[1])

      for i in food.keys():
         for j in final_food:
            if (j[0]==food[i][0]):
               b=(j[0],food[i][1],int(j[1]),food[i][1]*int(j[1]))
               bill[mac].append(b)
      print bill
      f=open("bill.txt","w")
      f.write(str(bill))
      f.close()
      f=open("food.txt","w")
      f.write(str(food))
      f.close()
```

```
    if (data=="cloth"):
        bd_addr=mac
        port=2
        sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
        sock.connect((bd_addr,port))
        sock.send(str(cloth))
        sock.close()


server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
        port =2
        server_sock.bind(("",port))
        server_sock.listen(1)
        client_sock, address=server_sock.accept()
        print "accpted Connection from ", address
        mac=address[0]
        data =client_sock.recv(1024)
        print "recieved [%s]" %data
        final_cloth=eval(data)
        client_sock.close()
        server_sock.close()

        for i in cloth.keys():
            for j in final_cloth:
                if (j[0]==cloth[i][0]):
                    b=(j[0],cloth[i][1],int(j[1]),cloth[i][1]*int(j[1]))
                    bill[mac].append(b)
        print bill
        f=open("bill.txt","w")
        f.write(str(bill))
        f.close()
        f=open("cloth.txt","w")
        f.write(str(food))
        f.close()

    if (data=="drinks"):
        bd_addr=mac
        port=2
        sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
        sock.connect((bd_addr,port))
        sock.send(str(drinks))
        sock.close()


server_sock=bluetooth.BluetoothSocket(bluetooth.RFCOMM)
        port =2
        server_sock.bind(("",port))
        server_sock.listen(1)
        client_sock, address=server_sock.accept()
        print "accpted Connection from ", address
        mac=address[0]
        data =client_sock.recv(1024)
        print "recieved [%s]" %data
        final_drinks=eval(data)
        client_sock.close()
        server_sock.close()

        for i in drinks.keys():
            for j in final_drinks:
                if (j[0]==drinks[i][0]):
                    b=(j[0],drinks[i][1],int(j[1]),drinks[i][1]*int(j[1]))
                    bill[mac].append(b)
        print bill
        f=open("bill.txt","w")
        f.write(str(bill))
        f.close()
        f=open("drinks.txt","w")
        f.write(str(food))
        f.close()
def bill1():
    global root,a
    root=Tk()
    a=Entry(root)
    a.grid(row=0,column=0)
    Button(root,text="PRINT
BILL",command=show_bill).grid(row=1,column=0)
    root.mainloop()
def show_bill():
    global root,a
    print a.get()
    f=open("bill.txt","r")
    d=eval(f.read())
    temp=""
    total=0
    temp='<body bgcolor=blue><center><img src=Desert.jpg
height=200width=400>
            </center><br><table align=center bgcolor=cyan
border=1><tr><td>Name
            </td>'+'<td>Unit
price</td>'+"<td>Quantity</td>"+"<td>Total</td></tr>"
    for i in d[a.get()]:
temp+="<tr><td>"+i[0]+"</td><td>"+str(i[1])+"</td><td>"+str(
i[2])+"</td>
<td>"+str(i[3])+"</td></tr>"
            total+=i[3]
   temp+="<tr><td colspan=4><font color=blue face=verdana
size=3>
             The total amount is: "+str(total)+"</font></td></tr>"
    temp+="</table>"
    g=open("print.html","w")
    g.write(temp)
    g.close()
    main()
food1=open("food.txt","r")
food=eval(food1.read())
cloth1=open("cloth.txt","r")
cloth=eval(cloth1.read())
drinks1=open("drinks.txt","r")
drinks=eval(drinks1.read())
f=open("bill.txt","r")
bill=eval(f.read())
main()
#bill1()
```

IX. FUTURE WORK

1. The project can be improved more and more by including compact security system.
2. Encryption and decryption can be incorporated to the data records to improve the security based on a keyword at the server and client part. By adding this facility we will prevent the problem of hacking.

X. CONCLUSION

After completing the project it can be concluded that using Bluetooth technology makes this system cost effective. It is based on departmental store support system, which in turn reduces the number of employees and prevents item missing from store. This system will provide efficient order and automatic billing system corresponding to the MAC ID generated from Bluetooth devices provided for each customer. Moreover using this system will also be helpful for maintaining records at the server. The main problem or disadvantage associated with Bluetooth is that it cannot be connected to more than seven devices, to overcome this problem we have to use several servers to maintain connections. This technique is easy, inexpensive and efficient to implement and maintain.

ACKNOWLEDGMENT

The opportunity to thank those who have contributed to this report writing is by far most impressive part of this report. Unfortunately the list of expression of thanks, no matter extensive, is always in adequate and this acknowledgement is no exception.

I would like to offer my most sincere and humble thanks to Mr. Avranil Tah for not only providing motivation for the report writing but also for keeping the momentum alive by lending her helping hand whenever and wherever required.

Last but not least, I would like to thank my team members (Debatrayee Roychowdhury, Koustav Ghosh, Subhasree Chakroborty), who were the best to work with as a mutual understanding and unconditional co-operation existed between four of us.

AUTHORS

First Author – Debopam Mukherjee, B.Tech (Department of Electronics and Communication), Supreme Knowledge Foundation Group of Institutions, mail id- dbpm.mukherjee@gmail.com
Second Author – Debatrayee Roychowdhury, B.Tech (Department of Electronics and Communication), Supreme Knowledge Foundation Group of Institutions, mail id- debatrayeeroychowdhury1@gmail.com
Third Author – Koustav Ghosh, B.Tech (Department of Electronics and Communication), Supreme Knowledge Foundation Group of Institutions, mail id- kou.skf@gmail.com
Fourth Author – Subhasree Chakraborty , B.Tech (Department of Electronics and Communication), Supreme Knowledge Foundation Group of Institutions, mail id- contact.subhasreechakraborty@gmail.com