

# **PROJECT REPORT**

## **RESTURANT BILLING SYSTEM**



University of Engineering and Technology  
Department of Computer Engineering

## Submitted by:

- Qurat ul ain (2021-CE-02)
- Haseeba Yasin (2021-CE-54)
- Rabia Khanum (2021-CE-56)
- Sumbal Ijaz (2021-CE-05)

## Submitted to:

Raja Muzammal Munir



## **ABSTRACT**

The main goal of this project is to develop a billing system for a restaurant. This application is designed to administer its users and customers. RBS is a billing system, made for the effective utilization of modern technology in the organization. It is an automated software that can handle a lot of information about the restaurant's menu, price, order history, reservation data. It has the capability to process bills and gather information of billing history. It is designed for the sole purpose of efficiency, speed and accuracy

## Table of Contents

Introduction	-----pg#5
Explanation	-----pg#6
Division	-----pg#7
References	-----pg#30
Applications	-----pg #29
Conclusion	-----pg#29

## **INTRODUCTION**

Restaurant Billing System is a computer based billing system with user friendly interface which automatically manages the billing process of the customer very easily taking only a short period of time. The system can large amount of data and also generates bill for the customer. Billing history, reservation information and staff information can also be obtained with the use of RBS. It is an automated desktop based software which has a simple design and very easy to use also. This project's main focus is on proper management of information regarding the staffs, billing and reservation records. It is also specialized in automatically processing the customer bills and discounts.

The proposed system either does not require paper work or very few paper works are required. All the data is fetched into the computer immediately and various bills can be generated through computers. Since all the data is kept in a database, no data of the organization can be destroyed. Moreover, works become very easy because there is no need to keep data on papers.

### **OBJECTIVE:**

The motive and objective to digitalize the restaurant menu and billing system is to save time, resources, reduce human error to minimum and making the menu system more environmentally friendly

## **METHODOLOGY:**

1. For entering the data of one user, a function is used. Its purpose is to show menu and ask for order. After selecting menu item, it will ask for quantity and then after pressing ok it will show the total cost.
2. After showing the cost it will show another option which is re-order. For re-order we have to press 1 or press any key to exit.
3. There are total 3 items. If the user enters 4 then the system will say if you want to add this item press 1 and press 2 to get back to menu. After pressing 1 it will ask for food name and price and then it will return to the menu and also show the added item.

## **TOOL /SOFTWARE:**

Another advantage of 8086 instruction set is that it is much smaller, and thus easier to learn. Emu8086 has a much easier syntax than any of the major assemblers, but will still generate a program that can be executed on any computer that runs 8086 machine code; a great combination for beginners!

## **FEASIBILITY:**

The feasibility study is carried out to test whether the proposed system is worth being implemented. Feasibility study is a test of system proposed regarding its work ability, its impact on the organization to meet user needs and effective use of resources. It is usually carried out by a small number of people who are familiar with the information system techniques, understand the part of the business or organization that will be involved or affected by the project and are skilled in the system analysis and design process.

The key consideration involved in the feasibility study are:

- Technical feasibility
- Economic feasibility
- Operational feasibility
- Schedule feasibility

**CODE:**

**INDEC.ASM:**

INDEC PROC

PUSH BX

PUSH CX

PUSH DX

@BEGIN:

MOV AH, 2

MOV DL, ''

INT 21H

XOR BX, BX

XOR CX, CX

MOV AH, 1

INT 21H

CMP AL, '-'

JE @MINUS

CMP AL, '+'

JE @PLUS

JMP @REPEAT2

@MINUS:

MOV CX, 1

@PLUS:

INT 21H

@REPEAT2:



CMP AL, '0'

JNGE @NOT\_DIGIT

CMP AL, '9'

JNLE @NOT\_DIGIT

AND AX, 000FH

PUSH AX

MOV AX,10

MUL BX

POP BX

ADD BX, AX

MOV AH, 1

INT 21H

CMP AL, 0DH

JNE @REPEAT2

MOV AX, BX

OR CX, CX

JE @EXIT

NEG AX

@EXIT:

POP DX

POP CX

POP BX

RET

@NOT\_DIGIT:

MOV AH, 2

MOV DL, 0DH

INT 21H

MOV DL, 0AH

INT 21H

JMP @BEGIN

INDEC ENDP

```

01  INDEC PROC
02
03  PUSH BX
04  PUSH CX
05  PUSH DX
06  @BEGIN:
07
08  MOV AH, 2
09  MOV DL, ' '
10  INT 21H
11
12  XOR BX, BX
13  XOR CX, CX
14
15  MOV AH, 1
16  INT 21H
17
18  CMP AL, '-'
19  JE @MINUS
20
21  CMP AL, '+'
22  JE @PLUS
23  JMP @REPEAT2
24
25  @MINUS:
26  MOV CX, 1
27  @PLUS:
28  INT 21H
29
30  @REPEAT2:
31
32  CMP AL, '0'
33  JNGE @NOT_DIGIT
34  CMP AL, '9'
35  JNLE @NOT_DIGIT
36
37  AND AX, 000FH
38  PUSH AX
39
40  MOV AX, 10
41  MUL BX
42  POP BX
43  ADD BX, AX
44
45  MOV AH, 1
46  INT 21H
47  CMP AL, 0DH
48  JNE @REPEAT2
49
50  MOV AX, BX

```

```

27 @PLUS:
28 INT 21H
29
30 @REPEAT2:
31
32 CMP AL, '0'
33 JNGE @NOT_DIGIT
34 CMP AL, '9'
35 JNLE @NOT_DIGIT
36
37 AND AX, 000FH
38 PUSH AX
39
40 MOV AX, 10
41 MUL BX
42 POP BX
43 ADD BX, AX
44
45 MOV AH, 1
46 INT 21H
47 CMP AL, 0DH
48 JNE @REPEAT2
49
50 MOV AX, BX
51 OR CX, CX
52
53 JE @EXIT
54 NEG AX
55
56 @EXIT:
57
58 POP DX
59 POP CX
60 POP BX
61 RET
62
63 @NOT_DIGIT:
64
65 MOV AH, 2
66 MOV DL, 0DH
67 INT 21H
68 MOV DL, 0AH
69 INT 21H
70 JMP @BEGIN
71 INDEC ENDP

```

## OUTDEC.ASM:

OUTDEC PROC

PUSH AX

PUSH BX

PUSH CX

PUSH DX

OR AX, AX

JGE @END\_IF1

PUSH AX

MOV DL, ''

MOV AH, 2

INT 21H

POP AX

NEG AX

@END\_IF1:

XOR CX, CX

MOV BX, 10D

@REPEAT1:

XOR DX, DX

DIV BX

PUSH DX

INC CX

OR AX, AX

JNE @REPEAT1

MOV AH, 2

@PRINT\_LOOP:

POP DX

OR DL, 30H

INT 21H

LOOP @PRINT\_LOOP

POP DX

POP CX

POP BX

POP AX

RET

OUTDEC ENDP

```

01 OUTDEC PROC
02
03 PUSH AX
04 PUSH BX
05 PUSH CX
06 PUSH DX
07
08 OR AX, AX
09 JGE @END_IF1
10
11 PUSH AX
12 MOV DL, ' '
13 MOV AH, 2
14 INT 21H
15 POP AX
16 NEG AX
17
18 @END_IF1:
19
20 XOR CX, CX
21 MOV BX, 10D
22
23 @REPEAT1:
24
25 XOR DX, DX
26 DIV BX
27 PUSH DX
28 INC CX
29
30
31 OR AX, AX
32 JNE @REPEAT1
33
34 MOV AH, 2
35
36 @PRINT_LOOP:
37
38 POP DX
39 OR DL, 30H
40 INT 21H
41 LOOP @PRINT_LOOP
42
43 POP DX
44 POP CX
45 POP BX
46 POP AX
47
48 RET
49 OUTDEC ENDP

```

## RESTURANT.ASM:

.model small

.stack 100h

.data

m0 dw "!!!!!!Welcome in our project!!!!!!\$"

m1 dw 10,13,10,13, "Which menu do you want ??please select:\$"

m2 dw 10,13,10,13, "1.Rice 100/- 2.Vegetable 50/- 3.Soup 20/- \$"

m3 dw 10,13,10,13, "Select the menu number:\$"

m8 dw 10,13,10,13, "SORRY!!!There is no more than 3 item,if u want,u can add one\$"

m9 dw 10,13,10,13, "Enter Food name:\$"

m10 dw 10,13,10,13,"        Price:\$"

m4 dw 10,13,10,13, "To add press 1 or press 2 to get back menu :\$"

m5 dw 10,13,10,13, "Enter quantity:\$"

m6 dw 10,13,10,13, "Total price: \$"

m7 dw 10,13,10,13, "     \*\*THANK YOU\*\*\$"

m11 dw "4.\$"

m12 dw "/-\$"

m13 dw 10,13,10,13, " Re-odrer : Press <1>\$",

m14 dw 10,13,10,13, " Exit : Press Any key\$"

q dw 0

r dw 0

v db 0

s dw 0

rprice dw 100

vprice dw 50

sprice dw 20

nprice dw 0



```
var1 db 100 dup('$')
```

```
.code
```

```
main proc
```

```
    mov ax,@data
```

```
    mov ds,ax
```

```
    mov ah,9
```

```
    lea dx,m0
```

```
    int 21h
```

```
start:
```

```
    cmp v,0
```

```
    jg start1
```

```
    mov ah,9
```

```
    lea dx,m1
```

```
    int 21h
```

```
menu:
```

mov ah,9

Lea dx,m2

int 21h

mov ah,9

Lea dx,m3

int 21h

mov ah,1

int 21h

cmp al,31h

je rice\_

cmp al,32h

je veg\_

cmp al,33h

je soup\_

menuadd:

inc v

mov ah,9

Lea dx,m8

int 21h

mov ah,9

Lea dx,m4

int 21h

mov ah,1

int 21h

cmp al,32h

je menu

mov ah,9

Lea dx,m9

int 21h

mov si,offset var1

```
l1:  
mov ah,1  
int 21h  
cmp al,13  
je print  
mov [si],al  
inc si  
jmp l1
```

```
print:
```

```
call price
```

```
start1:
```

```
mov ah,9  
Lea dx,m2  
int 21h
```

```
mov ah,9
```

Lea dx,m11

int 21h

mov dx,offset var1

mov ah,9

int 21h

mov ah,2

mov dl,' '

int 21h

xor ax,ax

mov ax,nprice

call outdec

mov ah,9

Lea dx,m12

int 21h

mov ah,9

Lea dx,m3

int 21h

mov ah,1

int 21h

cmp al,31h

je rice\_

cmp al,32h

je veg\_

cmp al,33h

je soup\_

newmenu\_:

mov ah,9

Lea dx,m5

int 21h

xor ax,ax

call indec

mul nprice

mov bx,ax

jmp totalprice

veg\_:

mov ah,9

Lea dx,m5

int 21h

xor ax,ax

call indec

mul vprice

mov bx,ax

jmp totalprice

rice\_:

mov ah,9

Lea dx,m5

int 21h

xor ax,ax

call indec

mul rprice

mov bx,ax

jmp totalprice



soup\_:

mov ah,9

Lea dx,m5

int 21h

xor ax,ax

call indec

mul sprice

mov bx,ax

jmp totalprice

price:

mov ah,9

Lea dx,m10

int 21h

```
mov ax,0
mov bx,0
mov cx,0
mov dx,0
```

```
input:
```

```
    and ax,000Fh
```

```
    push ax
```

```
    mov ax,10
```

```
    mul bx
```

```
    mov bx,ax
```

```
    pop ax
```

```
    add bx,ax
```

```
    mov ah,1
```

```
    int 21h
```

```
    cmp al,0Dh
```

```
    jne input
```

```
add nprice,bx
```

ret

totalprice:

mov ah,9

Lea dx,m6

int 21h

xor ax,ax

mov ax,bx

call outdec

mov ah,9

Lea dx,m13

int 21h

mov ah,9

Lea dx,m14

int 21h

mov ah,1

int 21h

cmp al,31h

je start

mov ah,9

Lea dx,m7

int 21h

mov ah,4ch

int 21h

main endp

include indec.asm

include outdec.asm

end main

```

0001 |.model small
0002 .stack 100h
0003 .data
0004
0005     m0 dw "                ?????Welcome in our project??????"
0006     m1 dw 10,13,10,13, "Which menu do you want ??please select:$"
0007     m2 dw 10,13,10,13, "1.Rice 100/- 2.Vegetable 50/- 3.Soup 20/- $"
0008     m3 dw 10,13,10,13, "Select the menu number:$"
0009     m8 dw 10,13,10,13, "SORRY!!!There is no more than 3 item,if u want,u can add one$"
0010     m9 dw 10,13,10,13, "Enter Food name:$"
0011     m10 dw 10,13,10,13, "                Price:$"
0012     m4 dw 10,13,10,13, "To add press 1 or press 2 to get back menu :$"
0013     m5 dw 10,13,10,13, "Enter quantity:$"
0014     m6 dw 10,13,10,13, "Total price: $"
0015     m7 dw 10,13,10,13, "                *****THANK YOU*****$"
0016     m11 dw "4.$"
0017     m12 dw "/-$"
0018     m13 dw 10,13,10,13, " Re-odrer : Press <1>$"
0019     m14 dw 10,13,10,13, " Exit : Press Any key$"
0020     q dw 0
0021     r dw 0
0022     v db 0
0023     s dw 0
0024     rprice dw 100
0025     vprice dw 50
0026     sprice dw 20
0027     nprice dw 0
0028
0029     var1 db 100 dup('$')
0030
0031 .code
0032     main proc
0033
0034         mov ax,@data
0035         mov ds,ax
0036
0037         mov ah,9
0038         lea dx,m0
0039         int 21h
0040
0041     start:
0042         cmp v,0
0043         jg start1
0044
0045         mov ah,9
0046         lea dx,m1
0047         int 21h
0048
0049     menu:
0050

```

```

031 .code
032     main proc
033
034         mov ax,@data
035         mov ds,ax
036
037         mov ah,9
038         lea dx,m0
039         int 21h
040
041     start:
042     cmp v,0
043     jg start1
044
045     mov ah,9
046     lea dx,m1
047     int 21h
048
049     menu:
050
051     mov ah,9
052     lea dx,m2
053     int 21h
054
055     mov ah,9
056     lea dx,m3
057     int 21h
058
059     mov ah,1
060     int 21h
061
062     cmp al,31h
063     je rice_
064     cmp al,32h
065     je veg_
066     cmp al,33h
067     je soup_
068
069
070
071     menuadd:
072     inc v
073
074     mov ah,9
075     lea dx,m8
076     int 21h
077
078     mov ah,9
079     lea dx,m4
080     int 21h

```

```

078      mov ah,9
079      lea dx,m4
080      int 21h
081
082      mov ah,1
083      int 21h
084      cmp al,32h
085      je menu
086
087      mov ah,9
088      lea dx,m9
089      int 21h
090
091      mov si,offset var1
092
093      l1:
094      mov ah,1
095      int 21h
096      cmp al,13
097      je print
098      mov [si],al
099      inc si
100      jmp l1
101
102      print:
103
104      call price
105
106      start1:
107
108      mov ah,9
109      lea dx,m2
110      int 21h
111
112      mov ah,9
113      lea dx,m11
114      int 21h
115
116      mov dx,offset var1
117      mov ah,9
118      int 21h
119
120      mov ah,2
121      mov dl,', '
122      int 21h
123
124      xor ax,ax
125      mov ax,nprice
126      call outdec
127

```

```

125     mov ax,nprice
126     call outdec
127
128     mov ah,9
129     lea dx,m12
130     int 21h
131
132
133
134     mov ah,9
135     lea dx,m3
136     int 21h
137
138     mov ah,1
139     int 21h
140
141     cmp al,31h
142     je rice_
143     cmp al,32h
144     je veg_
145     cmp al,33h
146     je soup_
147
148     newmenu_:
149
150         mov ah,9
151         lea dx,m5
152         int 21h
153
154         xor ax,ax
155
156         call indec
157
158         mul nprice
159
160         mov bx,ax
161
162         jmp totalprice
163
164
165     veg_:
166
167         mov ah,9
168         lea dx,m5
169         int 21h
170
171         xor ax,ax
172
173         call indec
174

```



```

171         xor ax,ax
172
173         call indec
174
175         mul vprice
176
177         mov bx,ax
178
179         jmp totalprice
180
181     rice_:
182
183         mov ah,9
184         lea dx,m5
185         int 21h
186
187         xor ax,ax
188
189         call indec
190
191         mul rprice
192
193         mov bx,ax
194
195         jmp totalprice
196
197     soup_:
198         mov ah,9
199         lea dx,m5
200         int 21h
201
202         xor ax,ax
203
204         call indec
205
206         mul sprice
207
208         mov bx,ax
209
210         jmp totalprice
211
212     price:
213
214         mov ah,9
215         lea dx,m10
216         int 21h
217
218         mov ax,0
219         mov bx,0
220         mov cx,0

```

```

220         mov cx,0
221         mov dx,0
222
223     input:
224         and ax,000Fh
225         push ax
226         mov ax,10
227         mul bx
228         mov bx,ax
229         pop ax
230         add bx,ax
231
232         mov ah,1
233         int 21h
234
235         cmp al,0Dh
236         jne input
237
238     add nprice,bx
239     ret
240
241
242
243     totalprice:
244
245         mov ah,9
246         lea dx,m6
247         int 21h
248
249         xor ax,ax
250
251
252         mov ax,bx
253         call outdec
254
255         mov ah,9
256         lea dx,m13
257         int 21h
258
259         mov ah,9
260         lea dx,m14
261         int 21h
262
263         mov ah,1
264         int 21h
265
266         cmp al,31h
267         je start

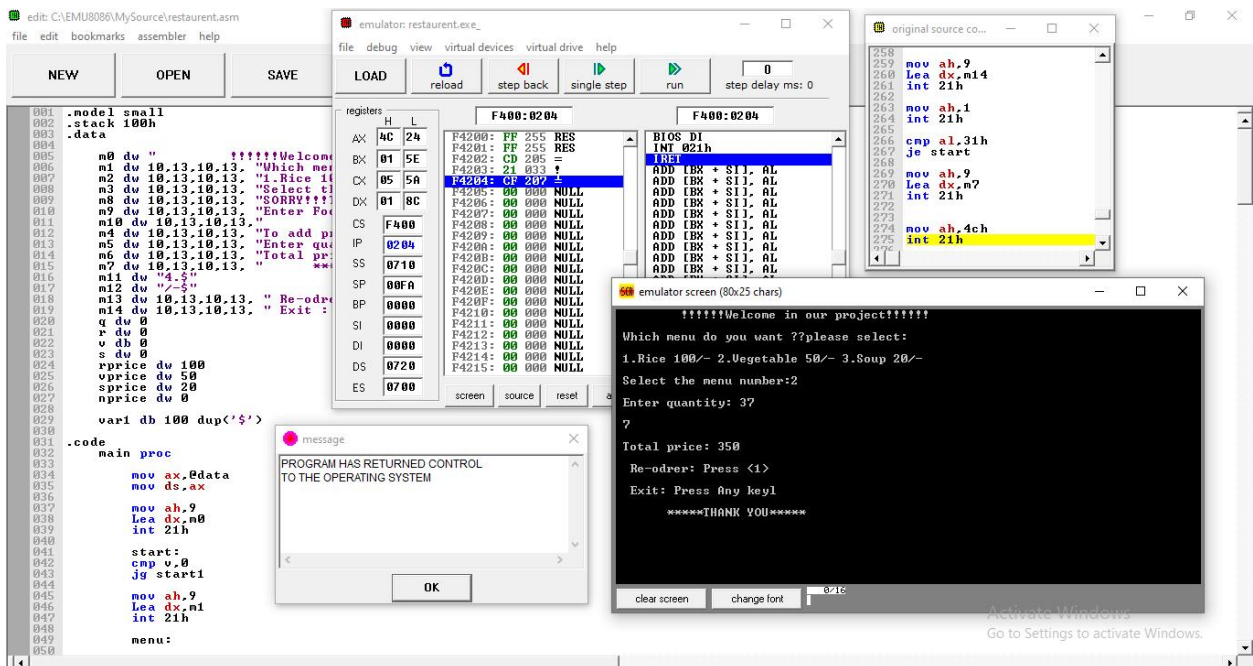
```

```

234
235         cmp al,0Dh
236         jne input
237
238         add nprice,bx
239         ret
240
241
242
243     totalprice:
244
245         mov ah,9
246         lea dx,m6
247         int 21h
248
249         xor ax,ax
250
251
252         mov ax,bx
253         call outdec
254
255         mov ah,9
256         lea dx,m13
257         int 21h
258
259         mov ah,9
260         lea dx,m14
261         int 21h
262
263         mov ah,1
264         int 21h
265
266         cmp al,31h
267         je start
268
269         mov ah,9
270         lea dx,m7
271         int 21h
272
273
274         mov ah,4ch
275         int 21h
276
277     main endp
278     include indec.asm
279     include outdec.asm
280     end main
281
282

```

**FINAL OUTPUT:**



## CONCLUSION:

The documentation includes all necessary information on the structure and the coding of the program created for Restaurant Billing system. Creating the program was an overwhelming task that required a lot of analyzing, research work and personal skills. Creating this report has been a great experience and numerous facts have been learned since the

required tasks were very challenging. Tasks such as creating a system to a restaurant, needed research work as well as personal skills. Creating proper design and smooth flow of operation was a very tiring task that consumed a lot of time

## **Applications**

1. Improved accuracy.
2. It saves you time.
3. It allows customer to track menu items easily.

## **References**

<https://youtu.be/juk4lp80KLQ>

<https://youtu.be/zM1GEM8DALY>

<https://www.nibizsoft.com/26-restaurant-billing-project-assembly-language-programming-x86-emu8086/>