VISVESVARAYA TECHNOLOGICAL UNIVERSITY

**JNANA SANGAMA,BELAGAVI – 590018**

**KARNATAKA**

**Assignment Report**
**On**

# “RESTAURANT BILLING AUTOMATE SOFTWARE”

**SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS**

**FOR THE DATA STRUCTURES AND APPLICATIONS (BCS304) COURSE OF III SEMESTER**

Submitted by

**MALLAYYA SHARANAYYA MATH**
1CG22CS061

**Guide:**
**Mr.Asif Ulla Khan** .M. Tech.
Asst. Prof., Dept. of CSE
CIT, Gubbi.

**HOD:**
**Dr. Shantala C P** PhD.,
Head, Dept. of CSE
CIT, Gubbi.

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**Channabasaveshwara Institute of Technology**
(Affiliated to VTU, Belgaum & Approved by AICTE, New Delhi)
(**NAAC Accredited & ISO 9001:2015 Certified Institution**)
NH 206 (B.H. Road), Gubbi, Tumkur – 572 216. Karnataka

**2023-24**

# Rubric – B.E. Mini-Project [BCS304]

| Course outcome | Rubric/Level | Excellent (91-100%) | Good (81-90%) | Average (61-80%) | Moderate (40-60%) | Score |
|---|---|---|---|---|---|---|
| CO1 | Identification of project proposal (05 Marks) | | | | | |
| CO2 | Design and Implementation (10 Marks) | | | | | |
| CO3 | Presentation skill (05 Marks) | | | | | |
| CO4 | Report (05 Marks) | | | | | |
| Total | | | | | | |

## Course outcome:

**CO 1: Identification of project proposal which is relevant to subject of engineering.**

**CO 2: Design and implement proposed project methodology.**

**CO 3: Effective communication skill to assimilate their project work.**

**CO 4: Understanding overall project progress and performance.**

**Student Signature** **Faculty signature**

# ABSTRACT:

**Objective**: The project aims to enhance the operational efficiency of "KEMPAMABA Restaurant" through the implementation of a tailored restaurant billing system using C programming.

**Features:** The system provides a comprehensive solution to challenges faced by manual billing processes in the food industry. It offers a console-based interface allowing users to effortlessly generate invoices, access historical transaction data, and locate specific records. The system utilizes structures to organize customer details and items, along with modular functions for invoicing components, ensuring a systematic and error-resistant approach to billing.

**Functionality**: The program incorporates dynamic input handling, persistent storage of invoices in the "RestaurantBill.dat" file, and precise arithmetic operations for calculating various financial aspects of each transaction.

**Output:** Detailed invoices are generated, showcasing customer information, itemized details, and a comprehensive summary of subtotal, discounts, taxes, and the grand total.

**Benefits:** The implementation mitigates errors associated with manual processes and aligns with the restaurant's goal of providing a modern and efficient invoicing system, ultimately improving accuracy, saving time, and enhancing overall operational efficiency in small-scale restaurant businesses.

**Future Enhancements:** Potential future enhancements may include user authentication, advanced reporting features, and integration with broader business processes.

## CHPTER-1

## Introduction:

In the realm of modern business operations, efficiency and accuracy in financial transactions are crucial elements for success. For small-scale enterprises, such as " Vajra. Restaurant," the need for a streamlined and error-resistant invoicing system is paramount. This C programming project introduces a restaurant billing system designed to alleviate the challenges associated with traditional manual billing processes.

The primary objective of this system is to automate and enhance the invoicing workflow for the restaurant, providing a user-friendly interface for generating invoices, accessing historical transaction data, and searching for specific records. The program leverages fundamental programming concepts, including structures for data organization, modular functions for code clarity, and file handling for persistent storage of transaction records.

The challenges with manual billing processes, such as potential errors in calculations, time-consuming tasks, and difficulties in record management, prompted the development of this automated billing solution. By incorporating dynamic input handling and precise arithmetic operations, the program ensures accurate calculations of totals, discounts, taxes, and grand totals for each invoice.

The introduction of this restaurant billing system aligns with the broader trend of automation in small-scale businesses, aiming to improve operational efficiency, reduce errors, and provide a systematic approach to financial transactions. The subsequent sections of this project delve into the problem statement, implementation details, and outcomes of the system, showcasing how it addresses the specific needs of "Vajra Restaurant."

# CHAPTER-2

## Problem Statement:

Traditional manual billing processes in the restaurant industry are laden with inefficiencies and potential errors, posing significant challenges for establishments like "KEMPAMABA. Restaurant." The reliance on handwritten invoices and manual record-keeping often leads to miscalculations, time-consuming tasks, and difficulties in managing transaction records. As a response to these challenges, the restaurant billing system presented in this C programming project aims to address the following key issues:

Error-Prone Manual Processes: Handwritten invoices are susceptible to calculation errors, leading to discrepancies in the total amounts, discounts, and taxes. The likelihood of human errors increases with the volume of transactions, impacting the accuracy of financial records.

Time-Consuming Operations: The traditional approach to billing is time-consuming, requiring significant manual effort in generating invoices and maintaining transaction records. This inefficiency hampers the overall operational speed of the restaurant and can lead to delays in customer service.

Record Management Challenges: Storing and retrieving historical transaction records can become cumbersome when relying on manual methods. The lack of a systematic approach to record-keeping may result in difficulties in locating specific invoices or analyzing past financial data.

Limited Customization: Handwritten invoices often lack a standardized format, making it challenging to implement customization based on specific customer preferences or business requirements. A lack of standardization can affect the professionalism of the billing process.

Potential for Data Loss: Manual record-keeping may expose the restaurant to the risk of data loss, especially in the case of physical damage to paper invoices or misplacement of important documents. This can lead to a loss of critical business information.

The identified problems highlight the need for an automated billing system tailored to the specific requirements of "KEMPAMABA. Restaurant." The subsequent implementation of the restaurant billing system aims to mitigate these challenges by introducing automation, accuracy, and a systematic approach to invoicing, contributing to enhanced operational efficiency and improved customer service.

## CHAPTER-3

### C code (Can be executed in the online c complied )

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

struct items{
   char item[20];
   float price;
   int qty;
};

struct orders{
   char customer[50];
   char date[50];
   int numOfItems;
   struct items itm[50];
};
void generateBillHeader(char name[50],char date[30]){
   printf("\n\n");
      printf("\t   KEMPAMABA. Restaurant");
      printf("\n\t------------------- ");
      printf("\nDate:%s",date);
      printf("\nInvoice To: %s",name);
      printf("\n");
      printf("--------------------------\n");
      printf("Items\t\t");
      printf("Qty\t\t");
      printf("Total\t\t");
      printf("\n--------------------------");
      printf("\n\n");
}
void generateBillBody(char item[30],int qty, float price){
   printf("%s\t\t",item);
      printf("%d\t\t",qty);
      printf("%.2f\t\t",qty * price);
      printf("\n");
}
```

```
void generateBillFooter(float total){
    printf("\n");
    float dis = 0.1*total;
    float netTotal=total-dis;
    float cgst=0.09*netTotal,grandTotal=netTotal + 2*cgst;//netTotal + cgst +
sgst
    printf("-------------------------\n");
    printf("Sub Total\t\t\t%.2f",total);
    printf("\nDiscount @10%s\t\t\t%.2f","%",dis);
    printf("\n\t\t\t\t-------");
    printf("\nNet Total\t\t\t%.2f",netTotal);
    printf("\nCGST @9%s\t\t\t%.2f","%",cgst);
    printf("\nSGST @9%s\t\t\t%.2f","%",cgst);
    printf("\n-------------------------");
    printf("\nGrand Total\t\t\t%.2f",grandTotal);
    printf("\n-------------------------\n");
}
int main(){

    int opt,n;
    struct orders ord;
    struct orders order;
    char saveBill = 'y',contFlag = 'y';
    char name[50];
    FILE *fp;
    while(contFlag == 'y'){
    system("clear");
    float total = 0;
    int invoiceFound = 0;
    printf("\t============KEMPABA.  RESTAURANT============");
    printf("\n\nPlease select your prefered operation");
    printf("\n\n1.Generate Invoice");
    printf("\n2.Show all Invoices");
    printf("\n3.Search Invoice");
    printf("\n4.Exit");

    printf("\n\nYour choice:\t");
    scanf("%d",&opt);
    fgetc(stdin);
    switch(opt){
        case 1:
        system("clear");
        printf("\nPlease enter the name of the customer:\t");
        fgets(ord.customer,50,stdin);
        ord.customer[strlen(ord.customer)-1] = 0;
        strcpy(ord.date,_DATE_);
        printf("\nPlease enter the number of items:\t");
        scanf("%d",&n);
        ord.numOfItems = n;
        for(int i=0;i<n;i++){
          fgetc(stdin);
          printf("\n\n");
          printf("Please enter the item %d:\t",i+1);
          fgets(ord.itm[i].item,20,stdin);
          ord.itm[i].item[strlen(ord.itm[i].item)-1]=0;
          printf("Please enter the quantity:\t");
          scanf("%d",&ord.itm[i].qty);
          printf("Please enter the unit price:\t");
          scanf("%f",&ord.itm[i].price);
          total += ord.itm[i].qty * ord.itm[i].price;
        }

        generateBillHeader(ord.customer,ord.date);
        for(int i=0;i<ord.numOfItems;i++){
          generateBillBody(ord.itm[i].item,ord.itm[i].qty,ord.itm[i].price);
        }
generateBillFooter(total);

        printf("\nDo you want to save the invoice [y/n]:\t");
        scanf("%s",&saveBill);

        if(saveBill == 'y'){
          fp = fopen("RestaurantBill.dat","a+");
          fwrite(&ord,sizeof(struct orders),1,fp);
          if(fwrite != 0)
          printf("\nSuccessfully saved");
          else
          printf("\nError saving");
          fclose(fp);
        }
        break;

        case 2:
        system("clear");
        fp = fopen("RestaurantBill.dat","r");
        printf("\n  **Your Previous Invoices**\n");
        while(fread(&order,sizeof(struct orders),1,fp)){
          float tot = 0;
          generateBillHeader(order.customer,order.date);
          for(int i=0;i<order.numOfItems;i++){
              generateBillBody(order.itm[i].item,order.itm[i].qty,order.itm[i].price);
              tot+=order.itm[i].qty * order.itm[i].price;
          }
          generateBillFooter(tot);
        }
        fclose(fp);
        break;

        case 3:
        printf("Enter the name of the customer:\t");
        fgetc(stdin);
        fgets(name,50,stdin);
        name[strlen(name)-1] = 0;
        system("clear");
        fp = fopen("RestaurantBill.dat","r");
        printf("\t**Invoice of %s**",name);
        while(fread(&order,sizeof(struct orders),1,fp)){
          float tot = 0;
          if(!strcmp(order.customer,name)){
          generateBillHeader(order.customer,order.date);
          for(int i=0;i<order.numOfItems;i++){
              generateBillBody(order.itm[i].item,order.itm[i].qty,order.itm[i].price);
              tot+=order.itm[i].qty * order.itm[i].price;
          }
          generateBillFooter(tot);
          invoiceFound = 1;
          }

        }
        if(!invoiceFound){
          printf("Sorry the invoice for %s doesnot exists",name);
        }
        fclose(fp);
        break;

     case 4:
     printf("\n\t\t Bye Bye :)\n\n");
     exit(0);
     break;

     default:
     printf("Sorry invalid option");
     break;
     }
     printf("\nDo you want to perform another operation?[y/n]:\t");
```

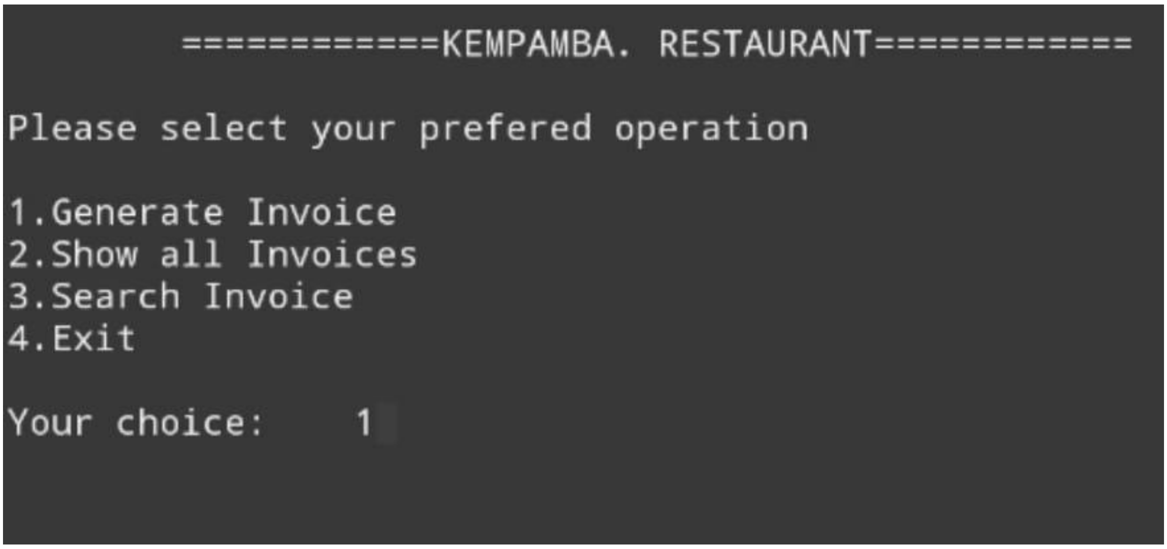
```
    scanf("%s",&contFlag);
    }
    printf("\n\t\t Bye Bye :)\n\n");
    printf("\n\n");

    return 0;
}
```

## CHAPTER-4

## RESULTS/SCREENSHOTS

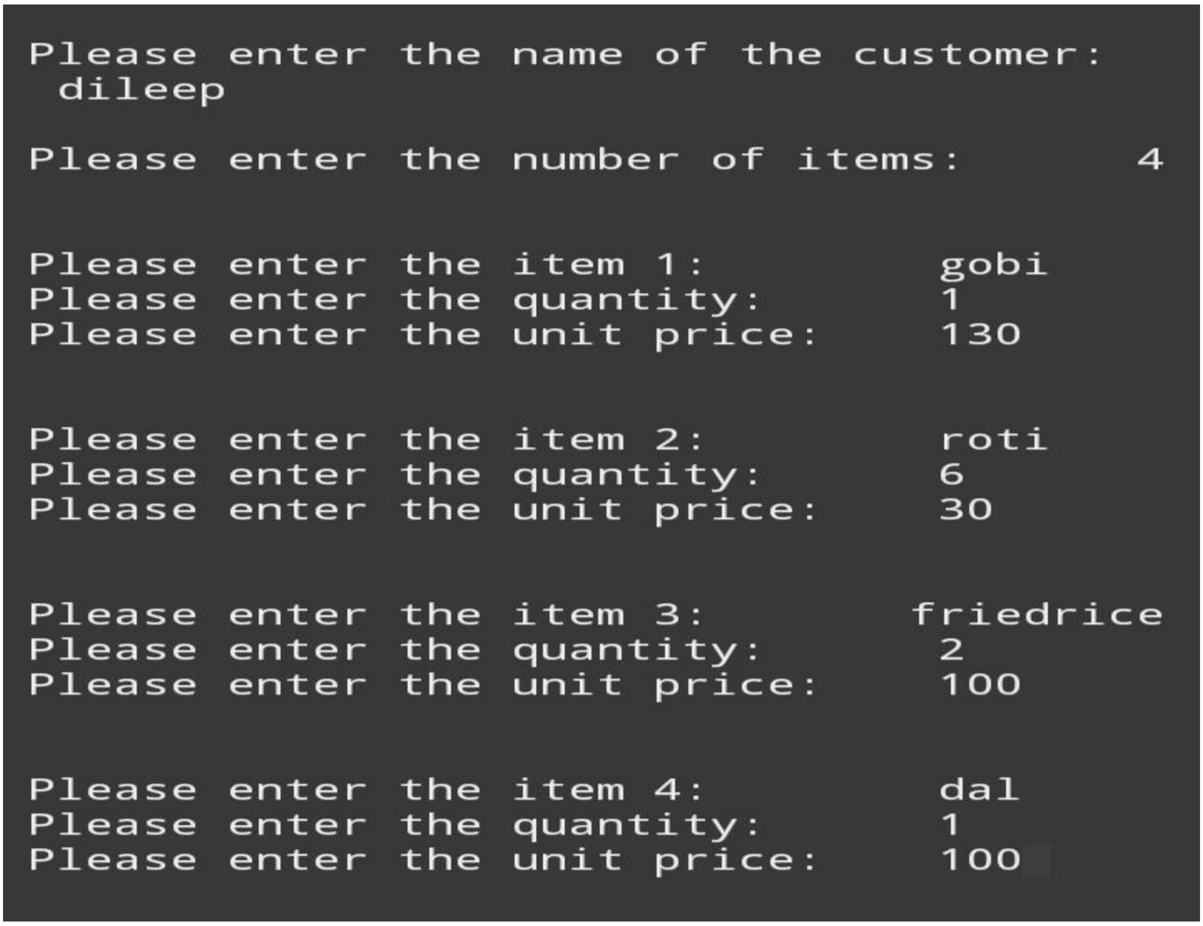
This will show a menu where you can find 4 operation

We will choice 1 to generate invoice



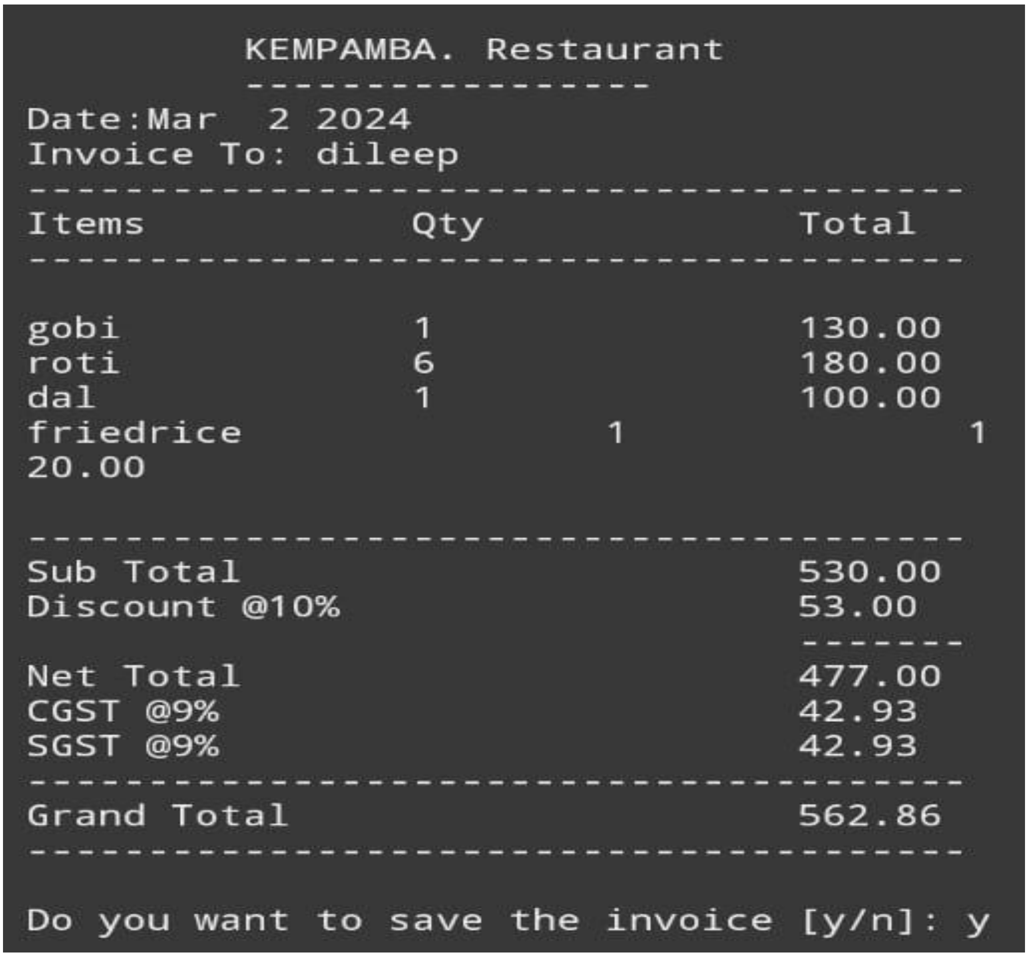
```
        ============KEMPAMBA. RESTAURANT============

Please select your prefered operation

1.Generate Invoice
2.Show all Invoices
3.Search Invoice
4.Exit

Your choice:    1
```

It will ask the name of a customer , number of item ,quantity and price



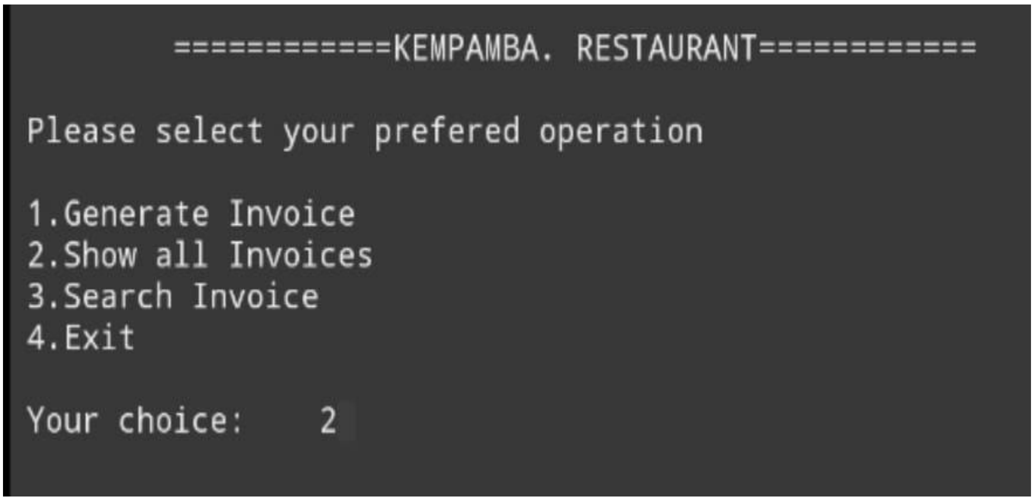
```
Please enter the name of the customer:
 dileep

Please enter the number of items:      4


Please enter the item 1:        gobi
Please enter the quantity:      1
Please enter the unit price:    130


Please enter the item 2:        roti
Please enter the quantity:      6
Please enter the unit price:    30


Please enter the item 3:       friedrice
Please enter the quantity:      2
Please enter the unit price:    100


Please enter the item 4:        dal
Please enter the quantity:      1
Please enter the unit price:    100
```

It will generate invoice

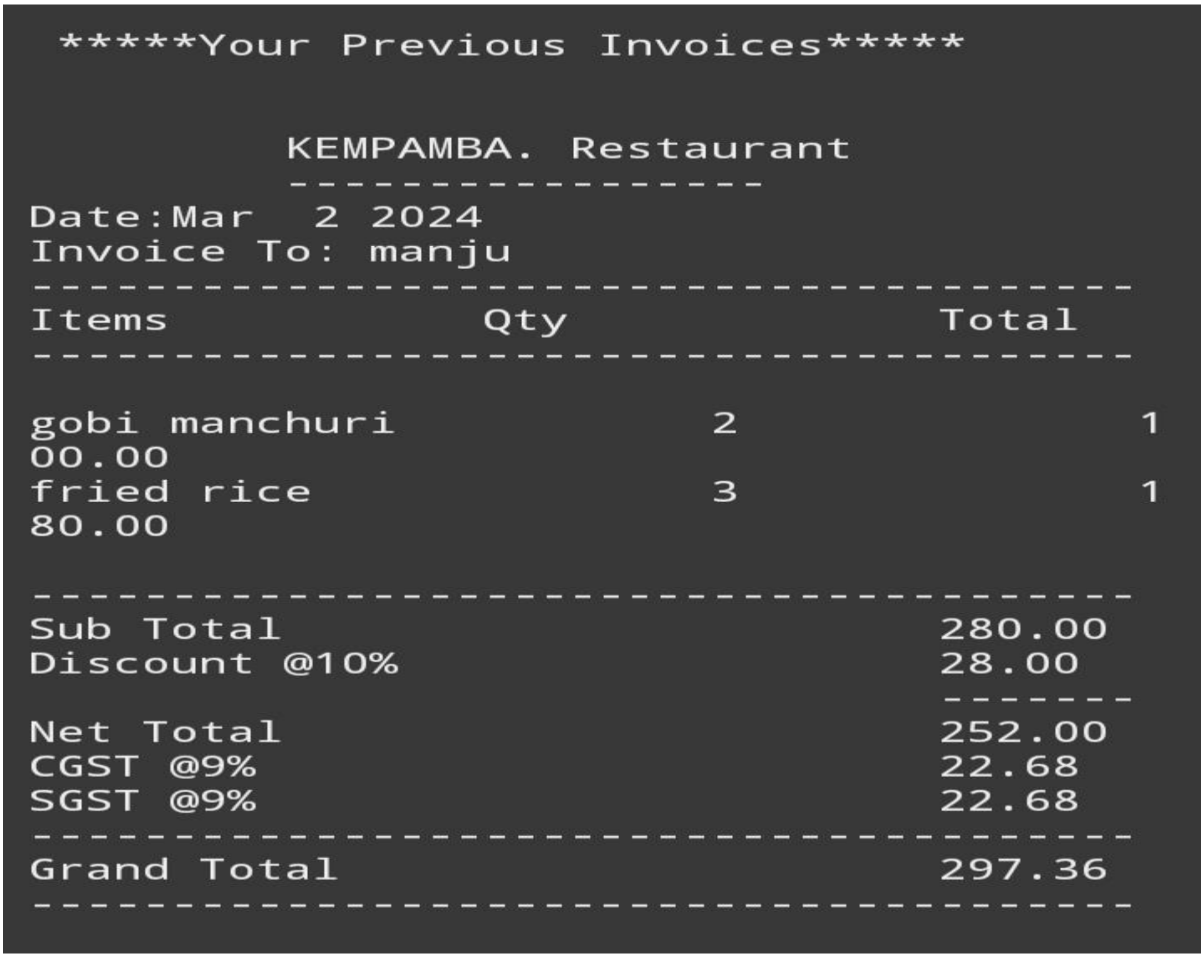


```
         KEMPAMBA. Restaurant
         -----------------
Date:Mar  2 2024
Invoice To: dileep
---------------------------------------
Items           Qty             Total
---------------------------------------

gobi            1               130.00
roti            6               180.00
dal             1               100.00
friedrice               1               1
20.00

---------------------------------------
Sub Total                       530.00
Discount @10%                   53.00
                                -------
Net Total                       477.00
CGST @9%                        42.93
SGST @9%                        42.93
---------------------------------------
Grand Total                     562.86
---------------------------------------

Do you want to save the invoice [y/n]: y
```

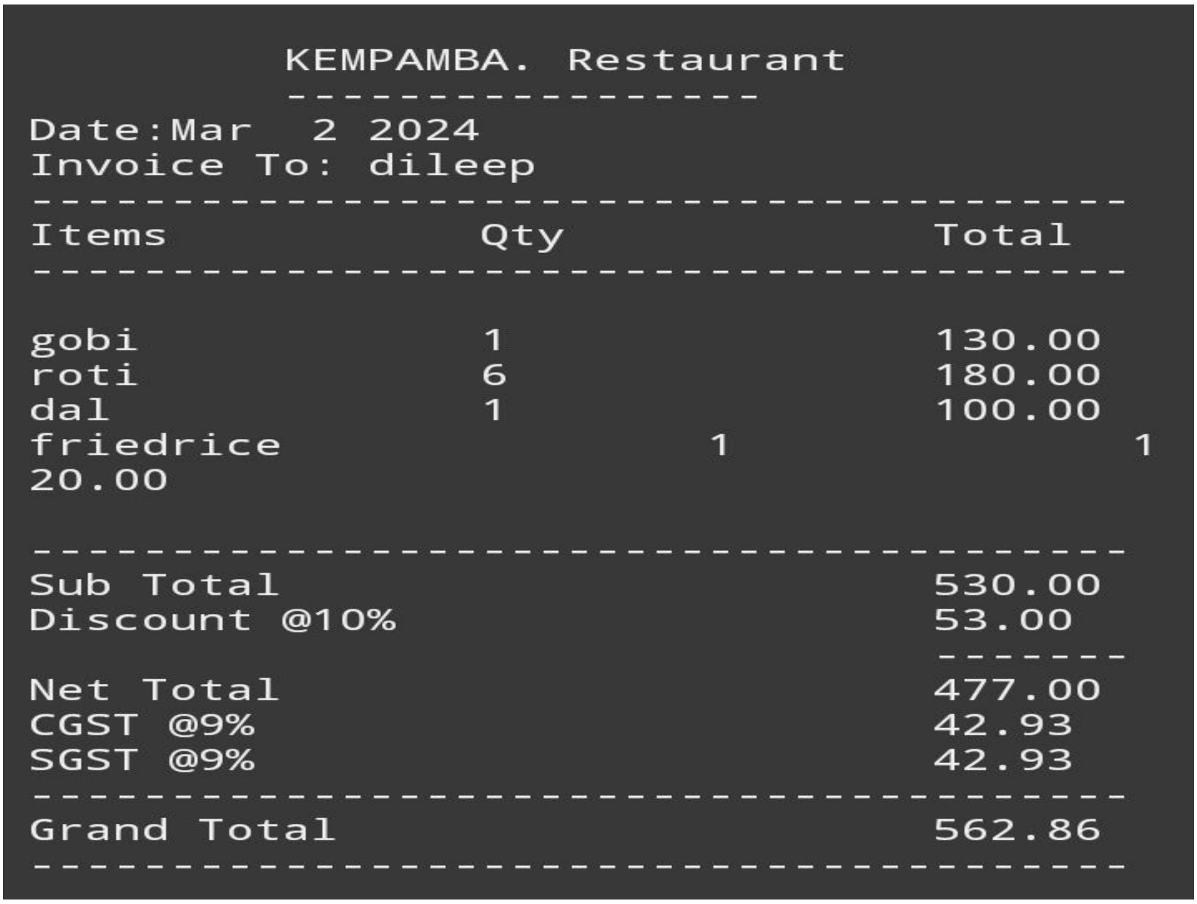
We will give choice 2 to show all invoice



```
        ============KEMPAMBA. RESTAURANT============

Please select your prefered operation

1.Generate Invoice
2.Show all Invoices
3.Search Invoice
4.Exit

Your choice:    2
```

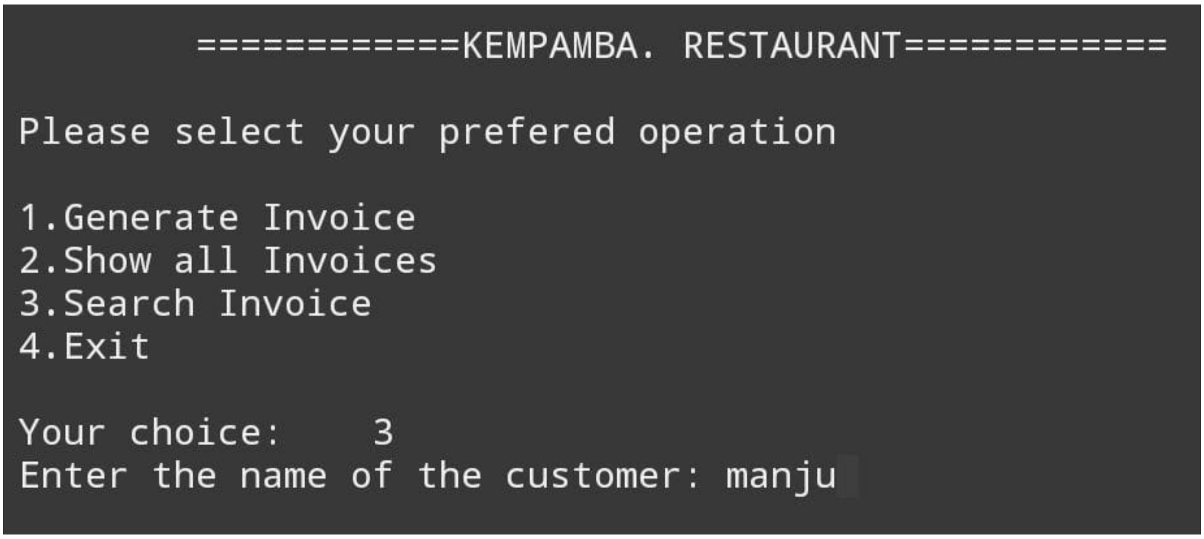


```
 *****Your Previous Invoices*****


          KEMPAMBA. Restaurant
          -----------------
Date:Mar  2 2024
Invoice To: manju
---------------------------------------
Items           Qty             Total
---------------------------------------

gobi manchuri            2              1
00.00
fried rice               3              1
80.00

---------------------------------------
Sub Total                       280.00
Discount @10%                   28.00
                                -------
Net Total                       252.00
CGST @9%                        22.68
SGST @9%                        22.68
---------------------------------------
Grand Total                     297.36
---------------------------------------
```

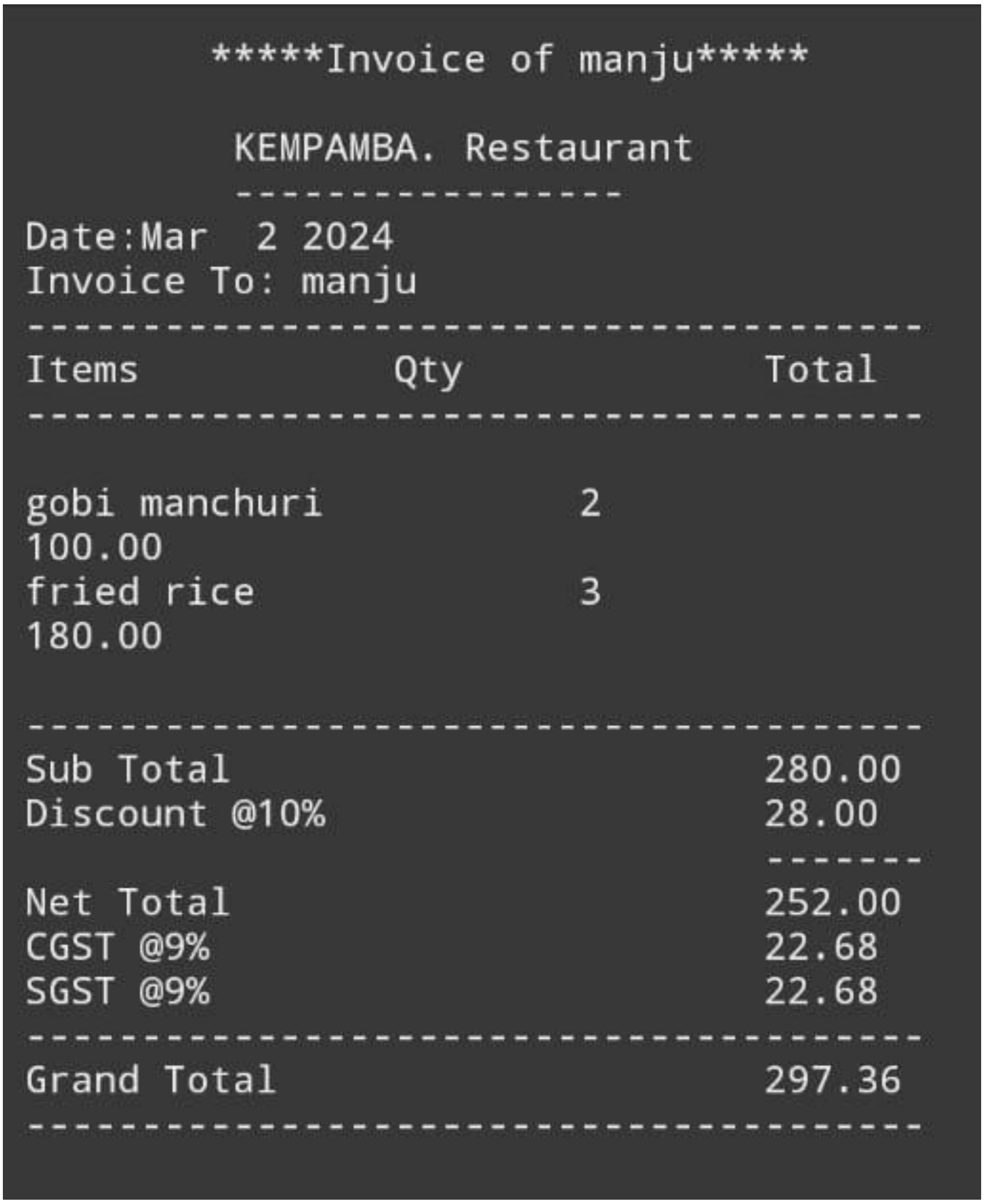


```
          KEMPAMBA. Restaurant
          -----------------
Date:Mar  2 2024
Invoice To: dileep
---------------------------------------
Items           Qty             Total
---------------------------------------

gobi            1               130.00
roti            6               180.00
dal             1               100.00
friedrice                1              1
20.00

---------------------------------------
Sub Total                       530.00
Discount @10%                   53.00
                                -------
Net Total                       477.00
CGST @9%                        42.93
SGST @9%                        42.93
---------------------------------------
Grand Total                     562.86
---------------------------------------
```

We will give choice 3 to search invoice



```
        ============KEMPAMBA. RESTAURANT============

Please select your prefered operation

1.Generate Invoice
2.Show all Invoices
3.Search Invoice
4.Exit

Your choice:    3
Enter the name of the customer: manju
```



```
        *****Invoice of manju*****

         KEMPAMBA. Restaurant
         -----------------
Date:Mar  2 2024
Invoice To: manju
---------------------------------------
Items          Qty              Total
---------------------------------------

gobi manchuri          2
100.00
fried rice             3
180.00

---------------------------------------
Sub Total                       280.00
Discount @10%                   28.00
                                -------
Net Total                       252.00
CGST @9%                        22.68
SGST @9%                        22.68
---------------------------------------
Grand Total                     297.36
---------------------------------------
```

## CHAPTER-5

## Conclusion:

In conclusion, the restaurant billing system presented in this C programming project offers a comprehensive solution to the challenges associated with manual billing processes for "KEM-PAMABA. Restaurant." The implementation successfully introduces automation, accuracy, and a systematic approach to the invoicing workflow, providing tangible benefits to both the restaurant staff and customers.

The utilization of structures for organizing customer details, items, and orders enhances code clarity and facilitates effective data management. Modular functions for bill generation contribute to code readability and maintainability. The integration of file handling ensures persistent storage of transaction records, allowing for easy retrieval and analysis of historical data.

Through a user-friendly console-based interface, the program empowers users to generate invoices, view historical records, and search for specific transactions. This enhances the overall operational efficiency of the restaurant by reducing the time and effort required for manual billing tasks.

The program's calculation and display capabilities further contribute to its effectiveness. Precise arithmetic operations for subtotal, discounts, taxes, and grand total calculations ensure accurate financial representations on the generated invoices. The comprehensive output provides detailed information about items, quantities, and prices, contributing to a professional and customer-friendly billing process.

As an automated billing system, this implementation aligns with modern trends in small-scale businesses, where efficiency and accuracy are paramount. The success of this project underscores the potential for automation to improve operational processes, reduce errors, and enhance the overall customer service experience.

Future enhancements to this billing system could include additional features such as user authentication, advanced reporting functionalities, and integration with broader business processes. However, the current implementation serves as a robust foundation for

"KEMPAMABA. Restaurant" to transition from manual to automated billing, contributing to a more streamlined and efficient business operation.

# REFERENCE

- https://github.com/Adv-learning/billing_system/blob/main/restaurantBilling.c
- https://chat.openai.com/
- https://www.geeksforgeeks.org/c-projects/