A Micro Project Report

on

Problem Solving using C Language

Submitted by
Chegu. Pushpa Rupa Sri(23471A05DK)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET

(AUTONOMOUS)

Accredited by NAAC with A+ Grade and NBA under Tier-1

NIRF rank in the band of 201-300 and is an ISO 9001:2015 certified Approved by AICTE, New Delhi, Permanently affiliated to JNTU Kakinada, Approved by AICTE, Accredited by NBA and accredited 'A+' grade by NAAC Narasaraopet-522601, Palnadu(Dt.), Andhra Pradesh, India

2024-2025

NARASARAOPETA ENGINEERING COLLEGE: NARASARAOPET

(AUTONOMOUS)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that Syed. Chegu. Pushpa RupaSri, Roll No: 23471A05DK, a Second Year Student of the Department of Computer Science and Engineering, has completed the Micro Project Satisfactorily in "Problem Solving using C Language" for the Academic Year 2024-2025..

Project Co-Ordinator	HEAD OF THE DEPARTMENT
Dr. Rama Krishna. Eluri, M.Tech., Ph.D. Ph.D.	**Dr. S. N. Tirumala Rao,** M.Tech.,
Asst. Professor	**Professor**

INDEX

Super Market Billing System

AIM:

Develop a Project for Super Market Billing System.

```
// Develop a Project for Super Market Billing System
#include <stdio.h>
#include <string.h>
#define MAX_ITEMS 500
#define MAX_TRANSACTIONS 1000
struct Item {
int code;
char name[30];
float price;
int quantity;
};
struct Transaction {
struct Item items[MAX_ITEMS];
int itemCount;
float totalAmount;
};
struct Item stock[MAX_ITEMS] = {
{101, "Milk", 2.50, 20}, {102, "Bread", 1.75, 15}, {103, "Eggs", 3.00, 50},
{104, "Butter", 4.00, 10}, {105, "Cheese", 5.50, 8}, {106, "Lays", 20.00, 30},
{107, "Books", 50.00, 5}, {108, "Soaps", 20.00, 10}, {109, "Paste", 25.00, 10},

{110, "cold drinks", 10.02, 10}

};

int stockCount = 10;

struct Transaction transactions[MAX_TRANSACTIONS];

int transactionCount = 0;

void displayMenu() {

printf("\n=== Supermarket Billing System ===\n");

printf("1. Add Items to Cart\n2. Display Stock\n3. Find Product in Cart\n");

printf("4. Generate Bill\n5. View Transactions\n6. Replenish Stock\n7.
Exit\nEnterchoice:");

}

void displayStock() {

printf("\n=== Stock ===\nCode\tName\tPrice\tQuantity\n");

for (int i = 0; i < stockCount; i++) {

printf("%d\t%s\t%.2f\t%d\n", stock[i].code, stock[i].name,
stock[i].price,stock[i].quantity);

}

}

int findItem(int code) {

for (int i = 0; i < stockCount; i++) {

if (stock[i].code == code) return i;

}

return -1;

}

int addToCart(struct Item cart[], int cartCount) {

int numItems, code, quantity, stockIndex;
```

```
printf("Enter number of items: ");
scanf("%d", &numItems);
for (int i = 0; i < numItems; i++) {
printf("\nEnter Item Code: ");
scanf("%d", &code);
stockIndex = findItem(code);
if (stockIndex == -1) {
printf("Item not found.\n");
continue;
}
printf("Enter Quantity: ");
scanf("%d", &quantity);
if (quantity > stock[stockIndex].quantity) {
printf("Not enough stock. Available: %d\n", stock[stockIndex].quantity);
continue;
}
cart[cartCount] = stock[stockIndex];
cart[cartCount].quantity = quantity;
stock[stockIndex].quantity -= quantity;
cartCount++;
printf("Item added to cart.\n");
}
return cartCount;
}
void findProductInCart(struct Item cart[], int cartCount) {
int code, found = 0;
```

```
printf("Enter item code to search in cart: ");
scanf("%d", &code);
for (int i = 0; i < cartCount; i++) {
if (cart[i].code == code) {
printf("\nProduct found in cart: %s, Price: %.2f, Quantity: %d\n",cart[i].name,
cart[i].price, cart[i].quantity);
found = 1;
break;
}
}
if (!found) printf("Item not found in cart.\n");
}
void generateBill(struct Item cart[], int cartCount) {
if (cartCount == 0) {
printf("Cart is empty.\n");
return;
}
float total = 0;
struct Transaction transaction = {.itemCount = cartCount};
printf("\n=== BILL ===\nCode\tName\tPrice\tQty\tTotal\n");
for (int i = 0; i < cartCount; i++) {
float itemTotal = cart[i].price * cart[i].quantity;
total += itemTotal;
printf("%d\t%s\t%7.2f\t%d\t%.2f\n", cart[i].code, cart[i].name, cart[i].price,
cart[i].quantity, itemTotal);
transaction.items[i] = cart[i];
}
```

```
transaction.totalAmount = total;
printf("------------------------------------\nTotal: \t%.2f\n", total);
transactions[transactionCount++] = transaction;
FILE *file = fopen("transactions.txt", "a");
if (file) {
fprintf(file, "\n=== Transaction %d ===\n", transactionCount);
for (int i = 0; i < cartCount; i++) {
fprintf(file, "%d\t%s\t%.2f\t%d\t%.2f\n", cart[i].code, cart[i].name, cart[i].price,
cart[i].quantity, cart[i].price * cart[i].quantity);
}
fprintf(file, "Total: \t\t%.2f\n", total);
fclose(file);
}
cartCount = 0;
}
void viewTransactions() {
if (transactionCount == 0) {
printf("No transactions found.\n");
return;
}
for (int i = 0; i < transactionCount; i++) {
printf("\n=== Transaction %d ===\n", i + 1);
for (int j = 0; j < transactions[i].itemCount; j++) {
struct Item item = transactions[i].items[j];
printf("%d\t%s\t%7.2f\t\t%d\t%.2f\n", item.code, item.name, item.price,
item.quantity,
```

```c
item.price * item.quantity);
}
printf("Total: \t \t%.2f\n", transactions[i].totalAmount);
}
}
void replenishStock() {
int code, quantity, stockIndex;
printf("Enter item code to replenish: ");
scanf("%d", &code);
stockIndex = findItem(code);
if (stockIndex != -1) {
printf("Enter quantity to add: ");
scanf("%d", &quantity);
stock[stockIndex].quantity += quantity;
printf("Stock updated: %s, New Quantity: %d\n",
stock[stockIndex].name,stock[stockIndex].quantity);
}
else {
if (stockCount < MAX_ITEMS) {
printf("Item not found. Adding a new item.\n");
stock[stockCount].code = code;
printf("Enter Item Name: ");
scanf("%s", stock[stockCount].name);
printf("Enter Item Price: ");
scanf("%f", &stock[stockCount].price);
printf("Enter quantity to add: ");
```

```
scanf("%d", &quantity);
stock[stockCount].quantity = quantity;
stockCount++;
printf("New item added: %s, Quantity: %d\n", stock[stockCount - 1].name,
stock[stockCount - 1].quantity);
}
else {
printf("Stock is full. Cannot add more items.\n");
}
}
}
int main() {
struct Item cart[MAX_ITEMS];
int choice, cartCount = 0;
while (1)
{
displayMenu();
scanf("%d", &choice);
switch (choice)
{
case 1: cartCount = addToCart(cart, cartCount); break;
case 2: displayStock(); break;
case 3: findProductInCart(cart, cartCount); break;
case 4: generateBill(cart, cartCount); break;
case 5: viewTransactions(); break;
case 6: replenishStock(); break;
case 7: printf("Thank you for using the system.\n"); return 0;
default: printf("Invalid choice, try again.\n");
}
}
return 0;
}
```

Output :

```
=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
3. Find Product in Cart
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:1
Enter number of items: 1

Enter Item Code: 104
Enter Quantity: 3
Item added to cart.

=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
3. Find Product in Cart
```

```
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:4


=== BILL ===
Code  Name Price  Qty    Total
104    Butter   4.003        12.00
------------------------------------
Total:                        12.00

=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
3. Find Product in Cart
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:5

=== Transaction 1 ===
104    Butter   4.00         3        12.00
Total:                               12.00

=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
```

```
3. Find Product in Cart
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:2

=== Stock ===
Code  Name     Price      Quantity
101    Milk        2.50       20
102    Bread       1.75       15
103    Eggs        3.00       50
104    Butter      4.00       7
105    Cheese      5.50       8
106    Lays       20.00       30
107    Books      50.00       5
108    Soaps      20.00       10
109    Paste      25.00       10
110    cold        10.02      10
      drinks

=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
3. Find Product in Cart
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:6
```

```
Enter item code to replenish: 111
Item not found. Adding a new item.
Enter Item Name: vegetables
Enter Item Price: 40
Enter quantity to add: 60
New item added: vegetables, Quantity: 60

=== Supermarket Billing System ===
1. Add Items to Cart
2. Display Stock
3. Find Product in Cart
4. Generate Bill
5. View Transactions
6. Replenish Stock
7. Exit
Enterchoice:2

=== Stock ===
Code Name        Price    Quantity
101  Milk         2.50      20
102  Bread        1.75      15
103  Eggs         3.00      50
104  Butter       4.00       7
105  Cheese        5.50      8
106  Lays         20.00     30
107  Books        50.00      5
108  Soaps        20.00     10
109  Paste        25.00     10
110  cold drinks  10.02     10
111  vegetables   40.00     60
```