**Code Demonstration**

**For**

**Restaurant Management System.**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX\_ITEMS 10

#define MAX\_ORDERS 100

// Define struct for food item

struct FOOD {

int uniqueId;

char name[50];

float price;

} food;

// Define struct for updating food item

struct Update {

char new\_name[50];

float new\_price;

} update;

// Define struct for order

struct Order {

int table\_no;

int itemId;

int quantity;

float price;

int isStudent;

int paymentStatus;

};

// Define struct for login information

struct login {

char fname[50];

char lname[50];

char username[50];

char password[50];

};

// Function prototypes

void insertItem(); // Function to insert a new food item

void displayItem(); // Function to display all food items

void updateItem(); // Function to update a food item

void deleteItem(); // Function to delete a food item

void placeOrder(); // Function for a customer to place an order

void viewOrders(); // Function to view all orders

void payBill(); // Function to mark an order as paid

void registration(); // Function for user registration

int login(char \*role); // Function for user login

// Main function

int main() {

printf("\nRestaurant Management System\n");

int choose = 1;

while (choose) {

printf("\n1. Register\n"); // Option to register

printf("2. Login as Admin\n"); // Option to login as admin

printf("3. Login as Staff\n"); // Option to login as staff

printf("4. Customer\n"); // Option for customer

printf("5. Exit\n"); // Option to exit

printf("\nEnter your choice: ");

int choice;

scanf("%d", &choice);

fflush(stdin);

switch (choice) {

case 1:

registration(); // Call registration function

break;

case 2:

if (login("admin")) { // Call login function for admin

while (1) {

printf("\n\*\* Admin Menu \*\*\n");

printf("1. Insert Item\n"); // Option to insert item

printf("2. Display Item\n"); // Option to display item

printf("3. Update Item\n"); // Option to update item

printf("4. Delete Item\n"); // Option to delete item

printf("5. Exit\n"); // Option to exit

printf("\nEnter your choice: ");

int adminChoice;

scanf("%d", &adminChoice);

fflush(stdin);

if (adminChoice == 5) break;

switch (adminChoice) {

case 1:

insertItem(); // Call insertItem function

break;

case 2:

displayItem(); // Call displayItem function

break;

case 3:

updateItem(); // Call updateItem function

break;

case 4:

deleteItem(); // Call deleteItem function

break;

default:

printf("\nInvalid choice!");

}

}

}

break;

case 3:

if (login("staff")) { // Call login function for staff

while (1) {

printf("\n\*\* Staff Menu \*\*\n");

printf("1. View Orders\n"); // Option to view orders

printf("2. Pay Bill\n"); // Option to pay bill

printf("3. Exit\n"); // Option to exit

printf("\nEnter your choice: ");

int staffChoice;

scanf("%d", &staffChoice);

fflush(stdin);

if (staffChoice == 3) break;

switch (staffChoice) {

case 1:

viewOrders(); // Call viewOrders function

break;

case 2:

payBill(); // Call payBill function

break;

default:

printf("\nInvalid choice!");

}

}

}

break;

case 4:

while (1) {

printf("\n\*\* Customer Menu \*\*\n");

printf("1. Place Order\n"); // Option to place order

printf("2. Exit\n"); // Option to exit

printf("\nEnter your choice: ");

int customerChoice;

scanf("%d", &customerChoice);

fflush(stdin);

if (customerChoice == 2) break;

switch (customerChoice) {

case 1:

placeOrder(); // Call placeOrder function

break;

default:

printf("\nInvalid choice!");

}

}

break;

case 5:

exit(0); // Exit program

break;

default:

printf("\nInvalid choice!");

}

printf("\nDo you want to continue (Type 1 for Yes or 2 for No)? ");

scanf("%d", &choose);

}

return 0;

}

// Function to insert a new food item

void insertItem() {

printf("\n\*\* Insert Item \*\*\n");

FILE \*fd;

fd = fopen("Food\_MenuItem.txt", "a");

if (fd == NULL) {

printf("Error opening file!\n");

exit(1);

}

printf("\nEnter Unique ID: ");

scanf("%d", &food.uniqueId);

printf("Enter food name: ");

getchar();

fgets(food.name, sizeof(food.name), stdin);

food.name[strcspn(food.name, "\n")] = 0;

printf("Enter food price: ");

scanf("%f", &food.price);

fprintf(fd, "%d %s %f\n", food.uniqueId, food.name, food.price);

fclose(fd);

}

// Function to display all food items

void displayItem() {

printf("\n\*\* Display Item \*\*\n");

FILE \*fd;

fd = fopen("Food\_MenuItem.txt", "r");

if (fd == NULL) {

printf("Error opening file!\n");

exit(1);

}

printf("\nUniqueId\tName\t\tPrice\n");

while (fscanf(fd, "%d %s %f", &food.uniqueId, food.name, &food.price) != EOF) {

printf("\n%d\t\t%s\t\t%f\n", food.uniqueId, food.name, food.price);

}

fclose(fd);

}

// Function to update a food item

void updateItem() {

printf("\n\*\* Update Item \*\*\n");

int update\_id, choice;

int isUpdated = 0;

printf("Enter the ID you want to update:\n");

scanf("%d", &update\_id);

getchar();

FILE \*fd;

FILE \*fd1;

fd = fopen("Food\_MenuItem.txt", "r");

fd1 = fopen("Food\_menuItem.tmp", "w");

if (fd == NULL || fd1 == NULL) {

printf("Error opening file!\n");

exit(1);

}

while (fscanf(fd, "%d %s %f", &food.uniqueId, food.name, &food.price) != EOF) {

if (food.uniqueId == update\_id) {

printf("\nWhat do you want to update: ");

printf("\n1. Name");

printf("\n2. Price");

printf("\n3. Exit");

printf("\nSelect an option: ");

scanf("%d", &choice);

if (choice == 1) {

printf("\nEnter new name: ");

scanf("%s", update.new\_name);

fprintf(fd1, "%d %s %f\n", food.uniqueId, update.new\_name, food.price);

isUpdated = 1;

} else if (choice == 2) {

printf("\nEnter new price: ");

scanf("%f", &update.new\_price);

fprintf(fd1, "%d %s %f\n", food.uniqueId, food.name, update.new\_price);

isUpdated = 1;

} else {

fprintf(fd1, "%d %s %f\n", food.uniqueId, food.name, food.price);

fclose(fd);

fclose(fd1);

remove("Food\_menuItem.tmp");

return;

}

} else {

fprintf(fd1, "%d %s %f\n", food.uniqueId, food.name, food.price);

}

}

fclose(fd);

fclose(fd1);

remove("Food\_MenuItem.txt");

rename("Food\_menuItem.tmp", "Food\_MenuItem.txt");

if (isUpdated) {

printf("Update is successfully done.\n");

} else {

printf("Item with ID %d not found.\n", update\_id);

}

}

// Function to delete a food item

void deleteItem() {

printf("\n\*\* Delete Item \*\*\n");

int delete\_id;

int isDeleted = 0;

printf("\nEnter the id you want to delete: ");

scanf("%d", &delete\_id);

FILE \*fd;

FILE \*fd1;

fd = fopen("Food\_MenuItem.txt", "r");

fd1 = fopen("Food\_menuItem.tmp", "w");

if (fd == NULL || fd1 == NULL) {

printf("Error opening file!\n");

exit(1);

}

while (fscanf(fd, "%d %s %f", &food.uniqueId, food.name, &food.price) != EOF) {

if (food.uniqueId != delete\_id) {

fprintf(fd1, "%d %s %f\n", food.uniqueId, food.name, food.price);

} else {

isDeleted = 1;

}

}

fclose(fd);

fclose(fd1);

remove("Food\_MenuItem.txt");

rename("Food\_menuItem.tmp", "Food\_MenuItem.txt");

if (isDeleted) {

printf("Delete is successfully done.\n");

} else {

printf("Item with ID %d not found.\n", delete\_id);

}

}

// Function for a customer to place an order

void placeOrder() {

FILE \*fp\_menu;

FILE \*fp\_orders;

int itemId, quantity;

float subtotal = 0.0f, discount = 0.0f;

printf("\n\*\* Placing Order \*\*\n");

fp\_menu = fopen("Food\_MenuItem.txt", "r");

if (fp\_menu == NULL) {

printf("Error opening menu file!\n");

return;

}

struct FOOD food;

printf("\nMenu:\n");

printf("ID\tName\t\tPrice\n");

while (fscanf(fp\_menu, "%d %s %f", &food.uniqueId, food.name, &food.price) != EOF) {

printf("%d\t%s\t\t%.2f\n", food.uniqueId, food.name, food.price);

}

fclose(fp\_menu);

int isStudent;

printf("\nAre you a student (1 for yes, 0 for no)? ");

scanf("%d", &isStudent);

fp\_orders = fopen("Orders.txt", "a");

if (fp\_orders == NULL) {

printf("Error opening orders file!\n");

return;

}

int numItems = 0;

struct Order orders[MAX\_ITEMS];

int table\_no;

char date[20];

printf("\nEnter Todays Date,Table no, Item IDs and quantities (0 to finish):\n");

printf("\nDate: ");

scanf("%s", &date);

printf("\nTable no: ");

scanf("%d", &table\_no);

while (numItems < MAX\_ITEMS) {

printf("Item ID: ");

scanf("%d", &itemId);

if (itemId == 0) {

break;

}

fp\_menu = fopen("Food\_MenuItem.txt", "r");

if (fp\_menu == NULL) {

printf("Error opening menu file!\n");

return;

}

int found = 0;

while (fscanf(fp\_menu, "%d %s %f", &food.uniqueId, food.name, &food.price) != EOF) {

if (food.uniqueId == itemId) {

found = 1;

break;

}

}

fclose(fp\_menu);

if (!found) {

printf("Item ID not found in the menu. Please enter a valid ID.\n");

continue;

}

printf("Quantity: ");

scanf("%d", &quantity);

if (quantity <= 0) {

printf("Quantity must be greater than zero. Please enter a valid quantity.\n");

continue;

}

orders[numItems].table\_no = table\_no;

orders[numItems].itemId = itemId;

orders[numItems].quantity = quantity;

orders[numItems].price = food.price \* quantity;

orders[numItems].isStudent = isStudent;

orders[numItems].paymentStatus = 0;

subtotal += orders[numItems].price;

numItems++;

}

float totalCost = subtotal;

if (isStudent) {

discount = subtotal \* 0.1;

totalCost -= discount;

}

fprintf(fp\_orders, "Date : %s\n", date);

fprintf(fp\_orders, "Order for Table No: %d\n", table\_no);

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

fprintf(fp\_orders, "Item ID: %d, Quantity: %d, Price: %.2f, Payment Status: %s\n",

orders[i].itemId, orders[i].quantity, orders[i].price, orders[i].paymentStatus ? "Paid" : "Not Paid");

}

fprintf(fp\_orders, "Subtotal: %.2f\n", subtotal);

fprintf(fp\_orders, "Discount: %.2f\n", discount);

fprintf(fp\_orders, "Total Cost: %.2f\n\n", totalCost);

fclose(fp\_orders);

printf("\nOrder Summary :\n");

printf("Date : %s\n", date);

printf("Table No : %d\n", table\_no);

printf("Total Items : %d\n", numItems);

printf("Subtotal : %.2f\n", subtotal);

printf("Discount : %.2f\n", discount);

printf("Total Cost : %.2f\n", totalCost);

}

// Function to mark the order as paid

void payBill() {

int table\_no;

printf("\n\*\* Pay Bill \*\*\n");

printf("Enter the Table No: ");

scanf("%d", &table\_no);

FILE \*fp\_orders = fopen("Orders.txt", "r");

FILE \*fp\_temp = fopen("Orders\_temp.txt", "w");

if (fp\_orders == NULL || fp\_temp == NULL) {

printf("Error opening orders file!\n");

return;

}

char line[256];

int found = 0;

int currentTableNo = -1;

while (fgets(line, sizeof(line), fp\_orders)) {

int temp\_table\_no, temp\_itemId, temp\_quantity;

float temp\_price;

char temp\_status[10];

if (sscanf(line, "Order for Table No: %d", &temp\_table\_no) == 1) {

currentTableNo = temp\_table\_no;

fprintf(fp\_temp, "%s", line);

} else if (sscanf(line, "Item ID: %d, Quantity: %d, Price: %f, Payment Status: %s",

&temp\_itemId, &temp\_quantity, &temp\_price, temp\_status) == 4) {

if (currentTableNo == table\_no) {

found = 1;

fprintf(fp\_temp, "Item ID: %d, Quantity: %d, Price: %.2f, Payment Status: Paid\n",

temp\_itemId, temp\_quantity, temp\_price);

} else {

fprintf(fp\_temp, "%s", line);

}

} else {

fprintf(fp\_temp, "%s", line);

}

}

fclose(fp\_orders);

fclose(fp\_temp);

remove("Orders.txt");

rename("Orders\_temp.txt", "Orders.txt");

if (found) {

printf("Payment for Table No: %d marked as successful.\n", table\_no);

} else {

printf("Order for Table No: %d not found.\n", table\_no);

}

}

// Function to view all orders

void viewOrders() {

FILE \*fp\_orders;

fp\_orders = fopen("Orders.txt", "r");

if (fp\_orders == NULL) {

printf("Error opening orders file!\n");

return;

}

char ch;

printf("\n\*\* All Orders \*\*\n");

while ((ch = fgetc(fp\_orders)) != EOF) {

putchar(ch);

}

fclose(fp\_orders);

}

// Function for user registration

void registration() {

FILE \*log;

log = fopen("login.txt", "a");

if (log == NULL) {

printf("Error opening file!\n");

exit(1);

}

struct login l;

printf("\nEnter First Name: ");

scanf("%s", l.fname);

printf("\nEnter Last Name: ");

scanf("%s", l.lname);

printf("\nEnter Username: ");

scanf("%s", l.username);

printf("\nEnter Password: ");

scanf("%s", l.password);

fprintf(log, "%s %s %s %s\n", l.fname, l.lname, l.username, l.password);

fclose(log);

printf("\nRegistration successful! Now login with Username and Password.\n");

}

// Function for user login

int login(char \*role) {

char username[50], password[50];

FILE \*log;

log = fopen("login.txt", "r");

if (log == NULL) {

printf("Error opening file!\n");

exit(1);

}

struct login l;

printf("\nEnter Username: ");

scanf("%s", username);

printf("\nEnter Password: ");

scanf("%s", password);

while (fscanf(log, "%s %s %s %s", l.fname, l.lname, l.username, l.password) != EOF) {

if (strcmp(username, l.username) == 0 && strcmp(password, l.password) == 0) {

printf("\nLogin successful as %s!\n", role);

fclose(log);

return 1;

}

}

printf("\nInvalid Username or Password. Please try again.\n");

fclose(log);

return 0;

}