Daffodil International University

Project Report

Restaurant Management System

Only for course Teacher						
		Needs Improvement	Developing	Sufficient	Above Average	Total Mark
Allocate mark & Percentage		25%	50%	75%	100%	25
Understanding	3					
Analysis	4					
Implementation	8					
Report Writing	10					
					Total obtained mark	
Comments						

Semester: Fall 2024 **Group No:- (1)**

Name: Imtiaz Ibna Kamal	**ID: 232-35-680**
Name: Jubaidul Islam Rimon	**ID: 232-35-154**
Name: Salim Sadman	**ID: 232-35-129**
Name: Akash Roy	**ID: 232-35-141**
Name: Sabbir Alam Alif	**ID: 232-35-160**

Batch: 41th **Section: E1** **Course Code: SE 133**

Course Name: Software Development Capstone Project

Course Teacher Name: Most. Munira Tabassum **Designation: Lecturer**

Submission Date: 30/Nov/2024

Code Demonstration

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

#define MAX_DISHES 100
#define MAX_SALES 100
#define NAME_LENGTH 50
#define CODE_LENGTH 10
#define DATE_LENGTH 11

// Admin Credentials
#define ADMIN_EMAIL "admin.com"
#define ADMIN_PASSWORD "admin123"

// Structure Definitions
typedef struct
{
    char name[NAME_LENGTH];
    char foodCode[CODE_LENGTH];
    float price;
    int stock;
} Dish;

typedef struct
{
    char dishName[NAME_LENGTH];
    char foodCode[CODE_LENGTH];
    int quantity;
    float totalPrice;
    char date[DATE_LENGTH];
} Sale;

// Global Variables
Dish dishes[MAX_DISHES];
```

```
Sale sales[MAX_SALES];
int dishCount = 0;
int saleCount = 0;

// Function Prototypes
void adminMenu();
int login(const char *email, const char *password);
void addDish();
void updateDish();
void deleteDish();
void viewStock();
void processOrder();
void viewTotalSales();
void loadData();
void saveData();
int findDishByCode(const char *code);
void clearScreen();
void printLine();

int login(const char *email, const char *password)
{
    char inputEmail[50], inputPassword[50];
    printLine();

    // Display Restaurant Name
    printf("       Daffodil Green Garden       \n");
    printLine();

    printf("          LOGIN          \n");
    printLine();

    printf("Enter Email: ");
    scanf("%s", inputEmail);
    printf("Enter Password: ");
    scanf("%s", inputPassword);

    if (strcmp(inputEmail, email) == 0 && strcmp(inputPassword,
password) == 0)
    {
        printf("Login successful!\n");
        clearScreen();

        return 1; // Login successful
    }
    else
    {
        printf("Invalid email or password. Please try again.\n");
        return 0; // Login failed
    }
}


void adminMenu()
{
    int choice;
    do
    {

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

        printf("1. Add Dish\n");
        printf("2. Update Dish\n");
        printf("3. Delete Dish\n");
        printf("4. View Stock\n");
        printf("5. Process Order\n");
        printf("6. View Total Sales\n");
        printf("0. Logout\n");

        printLine();
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice)
        {
            case 1:
                addDish();
                break;
            case 2:
                updateDish();
                break;
            case 3:
                deleteDish();
                break;
            case 4:
                viewStock();
                break;
            case 5:
                processOrder();
                break;
            case 6:
                viewTotalSales();
                break;
            case 0:
                printf("\nLogging out...\n");
                break;
            default:
                printf("\nInvalid choice. Please try again.\n");
        }

        // Adding a line after each operation for better 
separation
        if (choice != 0)
        {
            printLine();
        }

    } while (choice != 0);
}


void addDish()
{

    viewStock();
    printLine();
    printf("          Add Dish          \n");
    printLine();

    if (dishCount >= MAX_DISHES)
    {
        printf("\nInventory is full. Cannot add more
dishes.\n\n");
        return;
    }

    Dish newDish;
    printf("\nEnter the name of the new dish: ");
    scanf(" %[^\n]", newDish.name);

    printf("Enter the food code for the new dish: ");
    scanf("%s", newDish.foodCode);

    printf("Enter the price for '%s' (Food Code: %s): ",
newDish.name, newDish.foodCode);
    scanf("%f", &newDish.price);

    printf("Enter the stock quantity for '%s' (Food Code: %s): ",
newDish.name, newDish.foodCode);
    scanf("%d", &newDish.stock);

    dishes[dishCount++] = newDish;
    printf("\nDish '%s' (Food Code: %s) added successfully!\n",
newDish.name, newDish.foodCode);
}


void updateDish()
{

    viewStock();
    printLine();
    printf("          Update Dish          \n");
    printLine();

    char foodCode[CODE_LENGTH];
    printf("\nEnter the food code of the dish to update: ");
    scanf("%s", foodCode);

    int index = findDishByCode(foodCode);
    if (index == -1)
    {
        printf("\nDish not found. Please check the food code and 
try again.\n");
        return;
    }

    printf("\nEnter new price for '%s' (Food Code: %s): ", 
dishes[index].name, dishes[index].foodCode);
    scanf("%f", &dishes[index].price);

    printf("Enter new stock quantity for '%s' (Food Code: %s): ", 
dishes[index].name, dishes[index].foodCode);
    scanf("%d", &dishes[index].stock);

    printf("\nDish '%s' (Food Code: %s) updated successfully!\n", 
dishes[index].name, dishes[index].foodCode);
}

void deleteDish()
{

    viewStock();
    printLine();
    printf("          Delete Dish          \n");
    printLine();

    char foodCode[CODE_LENGTH];
    printf("\nEnter the food code of the dish to delete: ");
    scanf("%s", foodCode);

    int index = findDishByCode(foodCode);
    if (index == -1)
    {
        printf("\nDish not found. Please check the food code and 
try again.\n");
        return;
    }

    // Deleting the dish
    for (int i = index; i < dishCount - 1; i++)
    {
        dishes[i] = dishes[i + 1];
    }
    dishCount--;
    printf("\nDish '%s' (Food Code: %s) deleted successfully!\n",
dishes[index].name, dishes[index].foodCode);
}


void viewStock()
{
    clearScreen();
    printf("       Daffodil Green Garden       \n");
    printLine();
    printf("          Current Stock          \n");
    printLine();

    // Print the headers for the stock table
    printf("%-30s | %-12s | %-10s | %-6s\n", "Dish Name", "Food
Code", "Price", "Stock");
    printf("-----------------------------------------------------
-------\n");

    // Print the details of each dish in the stock
    for (int i = 0; i < dishCount; i++)
    {
        printf("%-30s | %-12s | %-10.2f | %-6d\n",
               dishes[i].name, dishes[i].foodCode,
dishes[i].price, dishes[i].stock);
    }
}


void processOrder()
{
    clearScreen();
    viewStock();
    printLine();
    printf("          Process Order          \n");
    printLine();

    char date[DATE_LENGTH];
    printf("Enter the date of the order (DD/MM/YYYY): ");
    scanf("%s", date);

    char customerName[NAME_LENGTH];
    printf("Enter the customer's name: ");
    scanf(" %[^\n]", customerName);  // To allow spaces in the
name

    // Create a file named based on the customer's name (sanitize
it for file naming)
    char fileName[100];
    snprintf(fileName, sizeof(fileName), "order_%s.txt",
customerName);

    FILE *billFile = fopen(fileName, "w");
    if (billFile == NULL)
    {
        printf("Error creating bill file.\n");
        return;
    }

    float totalBill = 0;
    int moreOrders = 1;
    int orderCount = 1;

    // Print header information to the bill file
    fprintf(billFile, "-------------------------------------------
-------------------\n");
    fprintf(billFile, "                    Daffodil Green Garden
\n");
    fprintf(billFile, "                     Customer: %s\n",
customerName);
    fprintf(billFile, "                      Date: %s\n", date);
    fprintf(billFile, "-------------------------------------------
-------------------\n");
    fprintf(billFile, "%-30s | %-8s | %-8s | %-8s\n", "Dish Name",
"Quantity", "Price", "Total");
```

```
    fprintf(billFile, "----------------------------------------
-------------------\n");

    while (moreOrders)
    {
        char foodCode[CODE_LENGTH];
        int quantity;

        printf("\nOrder #%d\n", orderCount);
        printf("Enter food code to order: ");
        scanf("%s", foodCode);

        int index = findDishByCode(foodCode);
        if (index == -1)
        {
            printf("Dish not found.\n");
            continue;
        }

        printf("Enter quantity: ");
        scanf("%d", &quantity);

        if (quantity > dishes[index].stock)
        {
            printf("Insufficient stock. Only %d available.\n\n",
dishes[index].stock);
            continue;
        }

        // Update stock
        dishes[index].stock -= quantity;

        // Calculate total price for this item
        float totalPrice = quantity * dishes[index].price;
        totalBill += totalPrice;

        // Print order details in the console
        printf("Dish ordered: %s | Quantity: %d | Total Price:
%.2f\n",
               dishes[index].name, quantity, totalPrice);
        // Save order details to the file with left alignment for
dish name
        fprintf(billFile, "%-30s | %-8d | %-8.2f | %-8.2f\n",
                dishes[index].name, quantity,
                dishes[index].price, totalPrice);

        // Add the sale details to the sales array (tracking total
sales)
        Sale newSale;
        strcpy(newSale.dishName, dishes[index].name);
        strcpy(newSale.foodCode, dishes[index].foodCode);
        newSale.quantity = quantity;
        newSale.totalPrice = totalPrice;
        strcpy(newSale.date, date);
        sales[saleCount++] = newSale;  // Update the global sales
array

        // Ask if the admin wants to add more orders
        printf("Do you want to add more orders? (1 for Yes, 0 for
No): ");
        scanf("%d", &moreOrders);

        orderCount++;  // Increment the order count for the next
order
    }

    // Show the total bill before discount
    printf("\nTotal bill before discount: %.2f\n", totalBill);

    // Ask for discount percentage
    float discountPercentage;
    printf("Enter discount percentage (0 for no discount): ");
    scanf("%f", &discountPercentage);

    // Calculate the discount and apply it to the total bill
    float discountAmount = (discountPercentage / 100) * totalBill;
    float discountedTotalBill = totalBill - discountAmount;
    // Print the discounted total bill
    printf("Discount applied: %.2f\n", discountAmount);
    printf("Total bill after discount: %.2f\n",
discountedTotalBill);
    // Print the bill to the file with discount details
    fprintf(billFile, "-------------------------------------------
-------------------\n");
    fprintf(billFile, "%-30s | %-8s | %-8s | %-8s\n", "Payment
Details", "", "", "");
    fprintf(billFile, "-------------------------------------------
-------------------\n");
    fprintf(billFile, "%-30s   %-8s   %-8s   %-8.2f\n",    "Total
Bill", "", "", totalBill);
    fprintf(billFile, "%-30s   %-8s   %-8s   %-8.2f\n",
"Discount", "", "", discountAmount);
    fprintf(billFile, "%-30s   %-8s   %-8s   %-8.2f\n", "Total
After Discount", "", "", discountedTotalBill);

    fprintf(billFile, "-------------------------------------------
-------------------\n");
    fprintf(billFile, "                 Thank you for your order!
\n");
    fprintf(billFile, "-------------------------------------------
-------------------\n");

    fclose(billFile); // Close the bill file

    // Notify the admin that the bill file has been created
    printf("\nThe total bill with discount has been saved to
'%s'.\n", fileName);
}

void viewTotalSales()
{
    clearScreen();
    printLine();
    printf("          Total Sales          \n");
    printLine();

    float totalSales = 0;
    char currentDate[DATE_LENGTH] = "";  // To track the current
date being printed
```

```
    for (int i = 0; i < saleCount; i++)
    {
        // Check if the date has changed
        if (strcmp(sales[i].date, currentDate) != 0)
        {
            // If it's a new date, print the date header
            if (strlen(currentDate) > 0)  // To avoid printing an
extra line at the start
            {
                printf("\n");
            }

            // Print the date header and update the current date
            printf("----------------------------------------------
----------------\n");
            printf("                   Sales for Date: %s\n",
sales[i].date);
            printf("----------------------------------------------
----------------\n");
            printf("%-30s | %-8s | %-8s | %-8s\n", "Dish Name",
"Quantity", "Price", "Total");
            printf("----------------------------------------------
----------------\n");

            // Update currentDate to the current sale's date
            strcpy(currentDate, sales[i].date);
        }

        // Print each sale's details
        printf("%-30s | %-8d | %-8.2f | %-8.2f\n",
               sales[i].dishName, sales[i].quantity,
               sales[i].totalPrice / sales[i].quantity,
sales[i].totalPrice);

        // Accumulate the total sales amount
        totalSales += sales[i].totalPrice;
    }
    // Print the total sales at the end
    printf("\n----------------------------------------------------
----------\n");
```

```
    printf("%-30s | %-8s | %-8s | %-8.2f\n", "Total Sales", "",
"", totalSales);
    printf("-------------------------------------------------------
--------\n");
}

void loadData()
{
    FILE *file = fopen("dishes.txt", "r");
    if (file != NULL)
    {
        while (fscanf(file, "%49[^,],%9[^,],%f,%d\n",
dishes[dishCount].name,
                      dishes[dishCount].foodCode,
&dishes[dishCount].price, &dishes[dishCount].stock) == 4)
        {
            dishCount++;
        }
        fclose(file);
    }

    file = fopen("sales.txt", "r");
    if (file != NULL)
    {
        while (fscanf(file, "%49[^,],%9[^,],%d,%f,%10s\n",
sales[saleCount].dishName,
                      sales[saleCount].foodCode,
&sales[saleCount].quantity, &sales[saleCount].totalPrice,
sales[saleCount].date) == 5)
        {
            saleCount++;
        }
        fclose(file);
    }
}
void saveData()
{
    FILE *file = fopen("dishes.txt", "w");
    for (int i = 0; i < dishCount; i++)
    {
        fprintf(file, "%s,%s,%.2f,%d\n", dishes[i].name,
dishes[i].foodCode, dishes[i].price, dishes[i].stock);
    }
    fclose(file);

    file = fopen("sales.txt", "w");
    for (int i = 0; i < saleCount; i++)
    {
        fprintf(file, "%s,%s,%d,%.2f,%s\n", sales[i].dishName,
sales[i].foodCode, sales[i].quantity, sales[i].totalPrice,
sales[i].date);
    }
    fclose(file);
}
int findDishByCode(const char *code)
{
    for (int i = 0; i < dishCount; i++)
    {
        if (strcmp(dishes[i].foodCode, code) == 0)
        {
            return i;
        }
    }
    return -1;
}
void clearScreen()
{
    system("clear");
}
void printLine()
{
    printf("============================================\n");

}
int main()
{
    loadData();
    if (login(ADMIN_EMAIL, ADMIN_PASSWORD))
    {
        adminMenu();
    }
    saveData();
    return 0;
}
```