

# Customer Relationship Management

**Mini Project**



**Name :DEON GEORGE**

**Roll NO. :28**

**Course Name:** C programming

**Date :** 18-07-2024

# INTRODUCTION

## Project overview:

## Problem statement:

Create a system to track customer interactions, manage leads, and maintain a database of customer information for marketing and support purposes.

## Objective:

The objective of this project is to develop a simple, menu-driven project management system in C for adding , viewing , managing leads and interacting among customers .

# System Requirements

Minimum Requirements for C Programming Code to Run:

### Hardware Requirement:

* + A computer with at least 4GB RAM
  + 500MB of free disk space

### Software Requirement:

* + **Operating System:** Windows/Linux/MacOS
  + **Compiler:** GCC or any C compiler
  + **IDE**: Code: Blocks, Dev-C++, or any C IDE

# Design and Development

### Program Logic:

* **Data Structure**:
  1. Define a Customer structure with fields for name, email, phone, and lead status.
* **Main Function**:
  1. Initialize an array of Customer structures.
  2. Keep track of the number of customers (numCustomers).
* **Menu-Driven System**:
  1. Display options:
     + **1. Add Customer**:
       - Prompt for details and add to the array.
     + **2. View Customer Details**:
       - Input email, search array, and display info.
     + **3. Exit**:
       - End the program.
* **Add Customer Function**:
  1. Check capacity.
  2. Prompt for details, create a new customer, and add to the array.
* **View Customer Details Function**:
  1. Search for customer by email.
  2. Display details if found; show error if not.

### Pseudocode:

1. Initialize an empty array of Customer structures (customers)

2. Initialize numCustomers to 0

3. Display menu options:

- 1. Add Customer

- 2. View Customer Details

- 3. Exit

4. Repeat until user chooses to exit:

a. Read user's choice

b. If choice is 1:

i. Prompt for customer details (name, email, phone, lead status)

ii. Create a new Customer structure with entered details

iii. Add the new customer to the customers array

iv. Increment numCustomers

c. If choice is 2:

i. Prompt for customer email

ii. Search customers array for a match based on email

iii. If found, display customer details (name, email, phone, lead status)

iv. If not found, show an error message

d. If choice is 3:

i. Exit the program

5. End

# Testing and Results

### Test cases:

  **Adding a New Customer**:

* **Test Case 1: Valid Customer Details**
  + Input: Name: “DEON GEORGE,” Email: “deon@example.com,” Phone: “1234567890,” Lead Status: 1
  + Expected Output: Customer added successfully!
* **Test Case 2: Maximum Number of Customers Reached**
  + Input: Attempt to add more than 100 customers
  + Expected Output: Error message: “Maximum number of customers reached.”

 **Viewing Customer Details**:

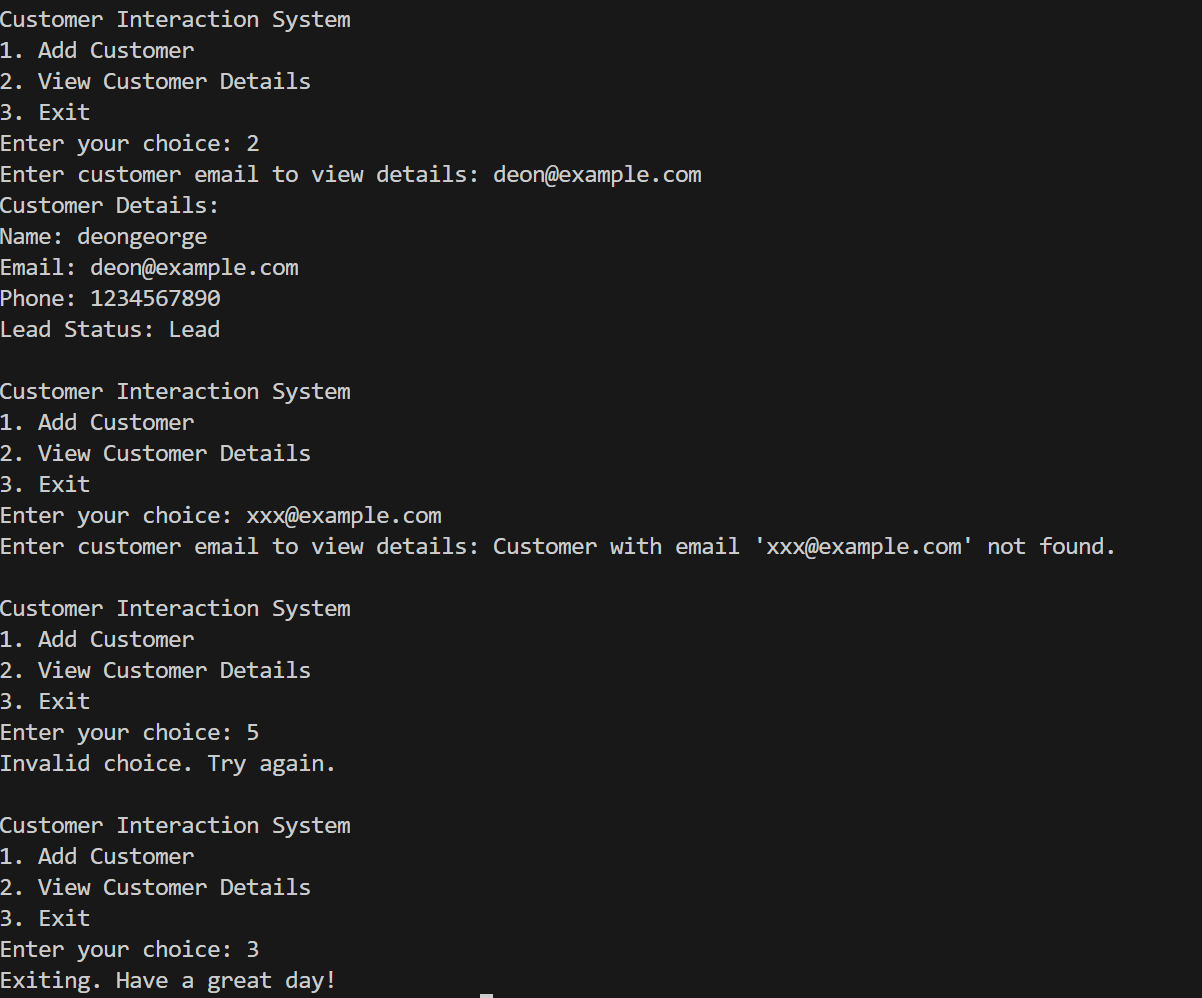
* **Test Case 3: Existing Customer**
  + Input: Enter email: “deon@example.com”
  + Expected Output: Display customer details: Name: “deon george,” Email: “deon@example.com,” Phone: “1234567890,” Lead Status: Lead
* **Test Case 4: Non-Existing Customer**
  + Input: Enter email: “nonexistent@example.com”
  + Expected Output: Error message: “Customer with email ‘nonexistent@example.com’ not found.”

 **Exiting the Program**:

* **Test Case 5: Choose Exit Option**
  + Input: Choose option 3
  + Expected Output: Program exits gracefully.

 **Invalid Choices**:

* **Test Case 6: Invalid Menu Choice**
  + Input: Choose an option other than 1, 2, or 3
  + Expected Output: Error message: “Invalid choice. Try again.”

**Output** :

# Discussion of results

The code efficiently allows dynamic additions and viewing the added data with clear user interactions, ensuring operational limits for customers .

Conclusion

### Summary of the Project:

### The project implements a task management system using C, featuring task addition, assignment, completion marking, and customer relationship management. It uses structures for tasks and projects, with limits on task and team member counts, providing a menu-driven interface for user interaction and project details display.

### Future Enhancements:

### Potential enhancements include file I/O for data persistence, task priority management, and user authentication for secure project access.

# References

[Stack Overflow - Where Developers Learn, Share, & Build Careers](https://stackoverflow.com/)

[GeeksforGeeks | A computer science portal for geeks](https://www.geeksforgeeks.org/)

[Microsoft Copilot in Bing](https://www.bing.com/chat?form=NTPCHB)

# Appendices

### Source code:

#include <stdio.h>

#include <string.h>

// Define Task and Project structures

typedef struct {

    char name[50];

    char assignedTo[50];

    int completed;

} Task;

typedef struct {

    char name[50];

    Task tasks[10];

    int taskCount;

} Project;

Project project;

char teamMembers[10][50];

int teamMemberCount = 0;

// Function to add a task to the project

void addTask() {

    if (project.taskCount >= 10) {

        printf("Task limit reached.\n");

        return;

    }

    // Get task details from the user

    printf("Enter task name: ");

    scanf(" %[^\n]s", project.tasks[project.taskCount].name);

    printf("Enter team member to assign task: ");

    scanf(" %[^\n]s", project.tasks[project.taskCount].assignedTo);

    project.tasks[project.taskCount].completed = 0;

    project.taskCount++;

    printf("Task added successfully.\n");

}

// Function to add a team member to the project

void addTeamMember() {

    if (teamMemberCount >= 10) {

        printf("Team member limit reached.\n");

        return;

    }

    // Get team member name from the user

    printf("Enter team member name: ");

    scanf(" %[^\n]s", teamMembers[teamMemberCount]);

    teamMemberCount++;

    printf("Team member added successfully.\n");

}

// Function to view details of the project

void viewDetails() {

    for (int i = 0; i < project.taskCount; i++) {

        printf("Task %d: %s\n", i + 1, project.tasks[i].name);

        printf("Assigned to: %s\n", project.tasks[i].assignedTo);

        printf("Status: %s\n", project.tasks[i].completed ? "Completed" : "Incomplete");

    }

    if (teamMemberCount != 0) {

        printf("Additional Team Members:\n");

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

            printf("%s\n", teamMembers[i]);

        }

    }

}

// Function to view only the task names

void viewTasks() {

    for (int i = 0; i < project.taskCount; i++) {

        printf("Task %d: %s\n", i + 1, project.tasks[i].name);

    }

}

// Function to mark a task as complete

void markTaskAsComplete() {

    viewTasks();

    int taskNumber;

    printf("Enter task number to mark as complete: ");

    scanf("%d", &taskNumber);

    if (taskNumber < 1 || taskNumber > project.taskCount) {

        printf("Invalid task number.\n");

        return;

    }

    project.tasks[taskNumber - 1].completed = 1;

    printf("Task marked as complete.\n");

}

int main() {

    // Initialize the project

    printf("Enter project name: ");

    scanf(" %[^\n]s", project.name);

    project.taskCount = 0;

    int choice;

    do {

        // Print the menu options

        printf("\nMenu:\n");

        printf("1. Add Task\n");

        printf("2. Add Team Member\n");

        printf("3. View Details\n");

        printf("4. Mark Task as Complete\n");

        printf("5. Exit\n");

        printf("Enter your choice: ");

        scanf("%d", &choice);

        // Handle the user's menu choice

        switch (choice) {

            case 1:

                addTask();

                break;

            case 2:

                addTeamMember();

                break;

            case 3:

                viewDetails();

                break;

            case 4:

                markTaskAsComplete();

                break;

            case 5:

                printf("EXITING...\n");

                break;

            default:

                printf("invalid choice, please try again.\n");

        }

    } while (choice != 5);

    return 0;

}