

C Programming Mini Project Documentation

1.Title Page:

- Project Title : FEEDBACK MANAGEMENT SYSTEM
- Student Name : Hema Leelavathi N
- Register Number : 921724102060
- Department : Computer Science and Engineering
- College Name : Sethu Institute of Technology
- Course : Programming in C
- Guide / Trainer Name : Mr.Pugazhmara
- Submission Date : 25-02-2026

2. Abstract

The Feedback Management System is a menu-driven C program designed to collect, store, view, and count user feedback. The problem it solves is managing feedback manually without proper storage. This program provides a simple digital solution using file handling to permanently store feedback in a text file (feedback.txt).

The system works by displaying a menu with options to add feedback, view stored feedback, count total feedback entries, or exit. It uses structures to organize data, functions for modular programming, and file operations to save and retrieve information.

3. Introduction

• What Problem You Are Solving

In many small institutions or systems, feedback is collected manually and not stored properly. This makes tracking and managing feedback difficult. This project solves the problem by creating a simple computerized feedback storage system.

• Why This Project is Useful

- Stores feedback permanently using file handling
- Reduces manual record keeping
- Easy to use menu-driven interface
- Helps understand real-time application of C concepts

• Where It Can Be Used

- Colleges for student feedback
- Small businesses for customer feedback
- Academic mini-project submissions
- Programming lab practice

4. Objectives

- To understand structured programming concepts
- To use loops and conditional statements (menu system)

- To work with structures and functions
- To implement file handling in C
- To improve logical thinking and problem-solving skills

5. Tools & Technology

- Programming Language: C
- Compiler: GCC / Turbo C / CodeBlocks
- Platform: Windows / Linux

6. System Requirements

• Hardware

- Basic computer
- Minimum 4GB RAM

• Software

- C Compiler (GCC / Turbo C / CodeBlocks)
- Any IDE or Text Editor (VS Code / CodeBlocks / Notepad++)

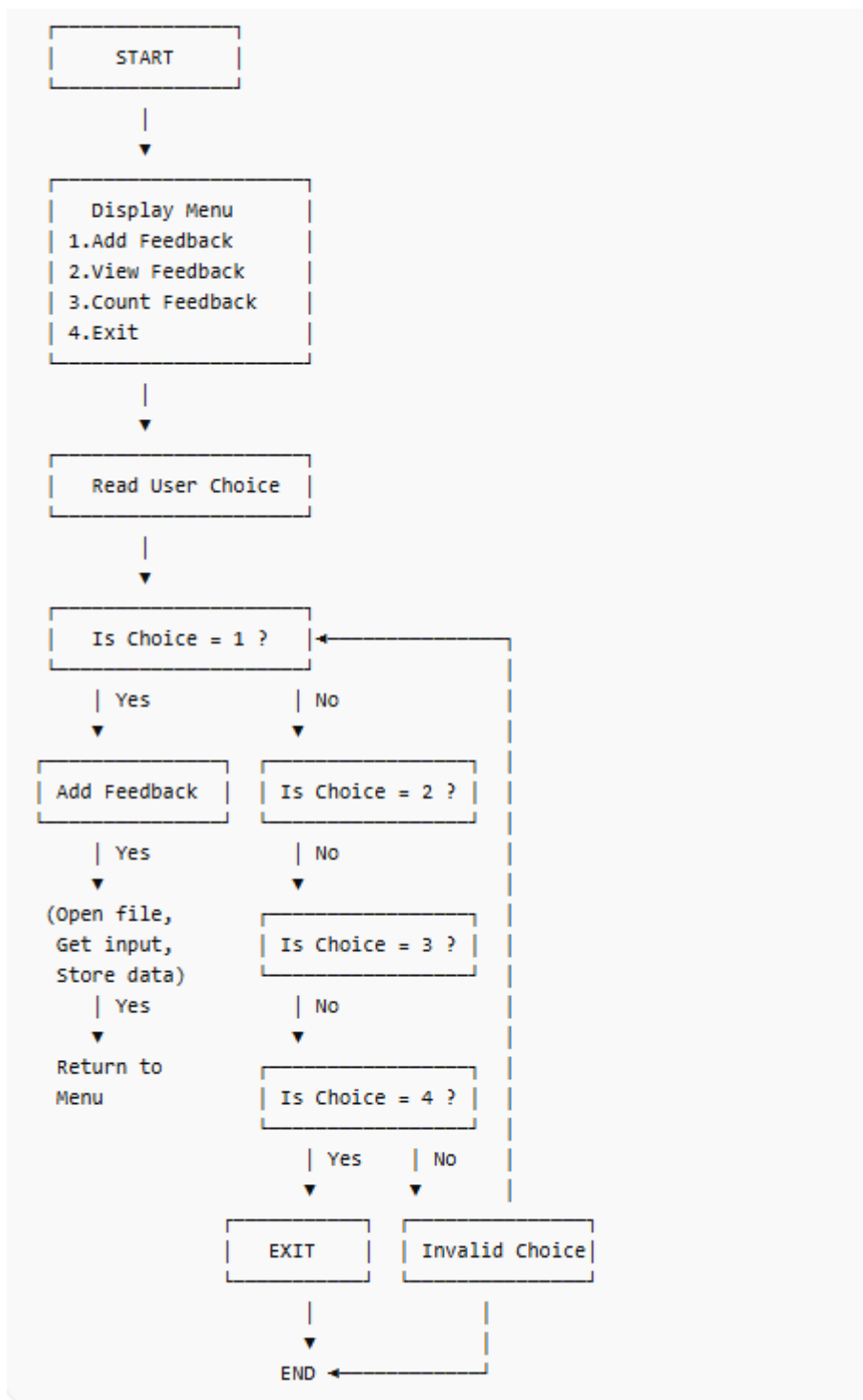
7. Methodology / Algorithm

Algorithm Steps:

1. Start the program
2. Display the main menu
3. Read user choice
4. If choice = 1
 - Open file in append mode
 - Read name, message, and rating
 - Store data in file
5. If choice = 2
 - Open file in read mode
 - Display all feedback
6. If choice = 3
 - Open file

- Count total feedback entries
- 7. If choice = 4
 - Exit program
- 8. Repeat menu until user exits
- 9. End program

8. Flowchart



9. Program Code

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

#define FILE_NAME "feedback.txt"

struct Feedback {
    char name[50];
    char message[200];
    int rating;
};

void addFeedback() {
    FILE *fp;
    struct Feedback f;

    fp = fopen(FILE_NAME, "a");
    if (fp == NULL) {
        printf("Error opening file!\n");
        return;
    }

    printf("Enter your name: ");
    getchar();
    fgets(f.name, sizeof(f.name), stdin);

    printf("Enter your feedback message: ");
    fgets(f.message, sizeof(f.message), stdin);

    printf("Enter rating (1-5): ");
    scanf("%d", &f.rating);

    fprintf(fp, "Name: %sMessage: %sRating: %d\n-----\n",
        f.name, f.message, f.rating);

    fclose(fp);

    printf("Feedback submitted successfully!\n");
}
```

```

void viewFeedback() {
    FILE *fp;
    char ch;

    fp = fopen(FILE_NAME, "r");
    if (fp == NULL) {
        printf("No feedback available.\n");
        return;
    }

    printf("\n----- All Feedback ----- \n");

    while ((ch = fgetc(fp)) != EOF) {
        putchar(ch);
    }

    fclose(fp);
}

void countFeedback() {
    FILE *fp;
    char line[300];
    int count = 0;

    fp = fopen(FILE_NAME, "r");
    if (fp == NULL) {
        printf("No feedback available.\n");
        return;
    }

    while (fgets(line, sizeof(line), fp)) {
        if (strstr(line, "Name:") != NULL) {
            count++;
        }
    }

    fclose(fp);

    printf("Total Feedback Entries: %d\n", count);
}

int main() {

```

```

int choice;

while (1) {
    printf("\n===== Feedback System =====\n");
    printf("1. Add Feedback\n");
    printf("2. View Feedback\n");
    printf("3. Count Feedback\n");
    printf("4. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1:
            addFeedback();
            break;
        case 2:
            viewFeedback();
            break;
        case 3:
            countFeedback();
            break;
        case 4:
            printf("Exiting program...\n");
            exit(0);
        default:
            printf("Invalid choice! Try again.\n");
    }
}

return 0;
}

```

10. Sample Input / Output

Sample Run:

```

===== Feedback System =====
1. Add Feedback
2. View Feedback
3. Count Feedback
4. Exit

```

Enter your choice: 1

Enter your name: Hema

Enter your feedback message: Good program

Enter rating (1-5): 5

Feedback submitted successfully!

===== Feedback System =====

1. Add Feedback
2. View Feedback
3. Count Feedback
4. Exit

Enter your choice: 3

Total Feedback Entries: 1

11. Result

The program was compiled and executed successfully. It allowed users to add, view, and count feedback entries correctly using file handling.

12. Applications

- Learning C programming
- Academic mini project
- Beginner-level software development
- Feedback management in small systems

13. Conclusion

The Feedback System project demonstrates important C programming concepts such as structures, file handling, loops, conditional statements, and functions. It provides practical knowledge of storing and retrieving data using files.

14. Future Enhancements

- Add delete and edit options

- Add search functionality
- Validate rating input (1-5 only)
- Improve user interface
- Store data using database instead of text file

15. References

- The C Programming Language – Kernighan & Ritchie
- C Programming textbooks
- Online tutorials (GeeksforGeeks, TutorialsPoint)
- Class notes