Employee Shift Planner in C

Abstract

The Employee Shift Planner is a console-based application developed in C that allows administrators to assign and manage employee shifts (Morning, Evening, Night) on a daily and weekly basis. This application stores schedules in a file for persistence and enables users to view and manage shift details efficiently.

Introduction

Employee scheduling is crucial for organizations operating in multiple shifts. Manual scheduling can lead to errors and inefficiencies. This project automates the process by allowing dynamic shift assignment and record maintenance using C programming. The application is lightweight and suitable for small to mid-size teams.

Objective

- To create a simple and effective shift planner.
- To assign Morning, Evening, or Night shifts to employees.
- To store shift schedules in a file.
- To display daily and weekly schedules.
- To enhance productivity by minimizing scheduling conflicts.

Methodology

The application is developed using the C programming language. The console interface collects employee and shift data, which is then stored in a text file. Basic file operations (fopen, fwrite, fread) are used for persistent storage. The user can select options from a menu to add or display shift schedules.

Project Description

Key functionalities include:

1. Assign Shifts - Admin enters employee name and selects a shift type.

Employee Shift Planner in C

2. Display Daily Schedule - Shows all employees and their shifts for a selected date.

3. Display Weekly Schedule - Shows shifts for each day of the week.

4. Store/Load Schedule - Data is stored in a file (schedule.txt) to retain schedules across sessions.

Algorithm

1. Start
2. Display menu:
   a. Assign Shift
   b. Display Daily Schedule
   c. Display Weekly Schedule
   d. Exit
3. If Assign Shift:
   - Input: Employee name, Date, Shift type
   - Store record in file
4. If Display Daily:
   - Input: Date
   - Read file and filter records
5. If Display Weekly:
   - Read all records and group by date
6. Exit on command

Limitations

- No GUI interface.
- Limited data validation.
- No concurrency handling.
- Manual entry required for each shift.

Advantages

Employee Shift Planner in C

- Simple and easy to use.
- File-based data storage ensures persistence.
- Saves time compared to manual scheduling.

Disadvantages

- Lacks advanced features like conflict checking or automatic scheduling.
- Not suitable for large enterprises with complex scheduling needs.
- Text file storage can be inefficient for large data.

Conclusion

The Employee Shift Planner in C provides a basic yet functional solution for managing employee shifts. It demonstrates the application of file handling, structures, and menu-driven programming in C. Future improvements could include GUI integration, database support, and shift conflict detection.

Sample Code (C)

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

typedef struct {
    char name[50];
    char date[11];
    char shift[10];
} Shift;

void assignShift() {
    Shift s;
    FILE *fp = fopen("schedule.txt", "a");
```

```
    printf("Enter Employee Name: ");
    scanf("%s", s.name);
    printf("Enter Date (YYYY-MM-DD): ");
    scanf("%s", s.date);
    printf("Enter Shift (Morning/Evening/Night): ");
    scanf("%s", s.shift);
    fwrite(&s, sizeof(s), 1, fp);
    fclose(fp);
    printf("Shift Assigned Successfully.\n");
}

void displayDailySchedule() {
    Shift s;
    char date[11];
    FILE *fp = fopen("schedule.txt", "r");
    printf("Enter Date (YYYY-MM-DD): ");
    scanf("%s", date);
    printf("Schedule for %s:\n", date);
    while (fread(&s, sizeof(s), 1, fp)) {
        if (strcmp(s.date, date) == 0)
            printf("%s - %s\n", s.name, s.shift);
    }
    fclose(fp);
}

void displayWeeklySchedule() {
    Shift s;
    FILE *fp = fopen("schedule.txt", "r");
    printf("Weekly Schedule:\n");
    while (fread(&s, sizeof(s), 1, fp)) {
        printf("%s - %s - %s\n", s.date, s.name, s.shift);
    }
```

Employee Shift Planner in C

```
    fclose(fp);
}

int main() {
    int choice;
    while (1) {
        printf("\n--- Employee Shift Planner ---\n");
        printf("1. Assign Shift\n");
        printf("2. Display Daily Schedule\n");
        printf("3. Display Weekly Schedule\n");
        printf("4. Exit\n");
        printf("Enter choice: ");
        scanf("%d", &choice);
        switch (choice) {
            case 1: assignShift(); break;
            case 2: displayDailySchedule(); break;
            case 3: displayWeeklySchedule(); break;
            case 4: exit(0);
            default: printf("Invalid choice.\n");
        }
    }
    return 0;
}
```