Topic: Result System Analysis Report

Team ID: CP013

Team member names:

Shreya Sudhakara Nayak: 4MT21IS042

Sanjana M. :4MT21IS036

Shobitha N.V.: 4MT21IS038

Daxia Vlora Dmello: 4MT21IS011

Shravya Jain: 4MT21IS039

Abstract:

The Result System is a simple yet effective software solution developed using C programming and text files. This report provides insights into the system's features, objectives, implementation details, and user functionalities.

The Result System represents a commendable achievement in academic record management, utilizing the power of C programming and text files to deliver a pragmatic, efficient, and resource-friendly solution. This report provides a comprehensive overview of the system, delving into its creation, objectives, implementation, and user functionalities.

The field of education is evolving rapidly, and with it comes the increasing need for streamlined academic record management. The Result System emerges as a tailored response to this need, offering a minimalist yet robust platform for administrators, teachers, and students to seamlessly navigate the complexities of academic data.

1. Introduction:

1.1 Background:

Managing academic records efficiently is vital for educational institutions of all sizes. The Result System was created to address this need by providing a basic yet functional platform for administrators, teachers, and students to manage academic data.

1.2 Objectives:

- To create a straightforward system for storing and retrieving student and teacher data.

- To enable administrators to add, modify, delete, and search records.

- To facilitate teachers in recording subject marks for students.

- To allow students easy access to their semester results.

2. Implementation Details:

The Result System was implemented using C programming language and text files for data storage. Here are the key components:

- Main Program: The main C program acts as the central controller, handling user interactions and data flow.
- Text Files: Data for students and teachers is stored in separate text files. Each record is stored as a line in the respective text file.
- Data Structures: Structures are used to represent student and teacher records, making it easier to manipulate data.
- Functions: Several functions are implemented to add, modify, delete, and search records, as well as to calculate and display results.

3. System Menus:

Admin Login:

- Admins can add, modify, and delete student and teacher details.
- Records can be searched by ID or USN.
- Detailed reports can be generated and displayed on the console.

Teacher Login:

- Teachers enter student USN and select the semester.
- They input subject marks for each student, which are then saved to the respective text file.

Student Login:

- Students enter their USN and select the semester.
- The system reads their results from the text files and displays them, including marks and grades.

4. Features:

- Basic data entry and management for administrators, teachers, and students.
- Efficient search functionality using ID or USN.
- Recording of subject marks by teachers.
- Quick access to academic results for students.

CODE:

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

// Structure definitions
struct Teacher {
    char id[20];
    char name[50];
    char subjects[100];
};

struct Marks {
    int marks[10];
    float average;
    char grade;
};

struct Student {
    char usn[20];
    char name[50];
    char branch[50];
    int year;
    int semester;
    struct Marks marks;
};

// Function prototypes
void addStudent(struct Student students[], int *numStudents);
void addTeacher(struct Teacher teachers[], int *numTeachers);
void modifyDetails(struct Student students[], int numStudents, struct Teacher
teachers[], int numTeachers);
void displayResult(struct Student students[], int numStudents);
void searchRecord(struct Student students[], int numStudents, struct Teacher
teachers[], int numTeachers);
void displayStudentDetailsByUSN(struct Student students[], int numStudents, const
char *usn);
void displayAllStudentDetails(struct Student students[], int numStudents);
void displayAllTeacherDetails(struct Teacher teachers[], int numTeachers);
void displayTeacherDetailsByID(struct Teacher teachers[], int numTeachers, const
char *id);
```

```
void saveStudentsToFile(struct Student students[], int numStudents, const char
*filename);
void loadStudentsFromFile(struct Student students[], int *numStudents, const char
*filename);
void saveTeachersToFile(struct Teacher teachers[], int numTeachers, const char
*filename);
void loadTeachersFromFile(struct Teacher teachers[], int *numTeachers, const char
*filename);

int main() {
    struct Student students[10]; // Assuming a maximum of 10 students
    struct Teacher teachers[10]; // Assuming a maximum of 10 teachers
    int numStudents = 0;
    int numTeachers = 0;

    // Load existing data from files (if available)
    loadStudentsFromFile(students, &numStudents, "students.txt");
    loadTeachersFromFile(teachers, &numTeachers, "teachers.txt");

    int choice;
    do {
        printf("\nMenu:\n");
        printf("1. Admin\n");
        printf("2. Teacher\n");
        printf("3. Student\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                // Admin menu
                printf("\nAdmin Menu:\n");
                printf("1. Add Student\n");
                printf("2. Add Teacher\n");
                printf("3. Modify/Delete Details\n");
                printf("4.Display Student Details based on USN\n");
                printf("5.Display Teacher Details based on ID\n");
                printf("6.Display All Teacher Details\n");
                printf("7.Display All Student Details\n");
                printf("8. Back to Main Menu\n");
                printf("Enter your choice: ");
                scanf("%d", &choice);

                switch (choice) {
                    case 1:
                        addStudent(students, &numStudents);
                        break;
                    case 2:
```

```
                addTeacher(teachers, &numTeachers);
                break;
            case 3:
                modifyDetails(students, numStudents, teachers, numTeachers);
                break;
            case 4:
                {
                    char usnToSearch[20];
                    printf("Enter USN of the student to search: ");
                    scanf("%s", usnToSearch);
                    displayStudentDetailsByUSN(students, numStudents,
usnToSearch);
                }
                break;
            case 5:
                {
                    char idToSearch[20];
                    printf("Enter ID of the teacher to search: ");
                    scanf("%s", idToSearch);
                    displayTeacherDetailsByID(teachers, numTeachers, idToSearch);
                }
                break;
            case 6:
                displayAllTeacherDetails(teachers, numTeachers);
                break;
            case 7:
                displayAllStudentDetails(students, numStudents);
                break;
            case 8:
                break;
            default:
                printf("Invalid choice.\n");
        }
        break;
    case 2:
        // Teacher menu
        printf("\nTeacher Menu:\n");
        printf("1. Search Student by USN\n");
        printf("2. Search Teacher by ID\n");
        printf("3. Display All Student Details\n");
        printf("4. Display All Teacher Details\n");
        printf("5. Back to Main Menu\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                {
                    char usn[20];
                    printf("Enter USN of the student to search: ");
```

```
                    scanf("%s", usn);
                    displayStudentDetailsByUSN(students, numStudents, usn);
                }
                break;
            case 2:
                {
                    char id[20];
                    printf("Enter ID of the teacher to search: ");
                    scanf("%s", id);
                    displayTeacherDetailsByID(teachers, numTeachers, id);
                }
                break;
            case 3:
                displayAllStudentDetails(students, numStudents);
                break;
            case 4:
                displayAllTeacherDetails(teachers, numTeachers);
                break;
            case 5:
                break;
            default:
                printf("Invalid choice.\n");
        }
        break;
    case 3:
        // Student menu
        printf("\nStudent Menu:\n");
        displayResult(students, numStudents);
        break;
    case 4:
        printf("Exiting...\n");
        break;
    default:
        printf("Invalid choice. Please enter a valid option.\n");
  }
} while (choice != 4);

// Save data to files before exiting
saveStudentsToFile(students, numStudents, "students.txt");
saveTeachersToFile(teachers, numTeachers, "teachers.txt");

return 0;
}

// Function to add student details
void addStudent(struct Student students[], int *numStudents) {
    printf("Enter USN: ");
    scanf("%s", students[*numStudents].usn);
    printf("Enter Name: ");
    scanf(" %[^\n]", students[*numStudents].name);
```

```
    printf("Enter Branch: ");
    scanf(" %[^\n]", students[*numStudents].branch);
    printf("Enter Year: ");
    scanf("%d", &students[*numStudents].year);
    printf("Enter Semester: ");
    scanf("%d", &students[*numStudents].semester);

    // Simulate marks input
    for (int i = 0; i < 5; ++i) {
        printf("Enter Marks for Subject %d: ", i + 1);
        scanf("%d", &students[*numStudents].marks.marks[i]);
    }

    // Calculate average and grade
    int totalMarks = 0;
    for (int i = 0; i < 5; ++i) {
        totalMarks += students[*numStudents].marks.marks[i];
    }
    students[*numStudents].marks.average = (float)totalMarks / 5;
    if (students[*numStudents].marks.average >= 90) {
        students[*numStudents].marks.grade = 'A';
    } else if (students[*numStudents].marks.average >= 80) {
        students[*numStudents].marks.grade = 'B';
    } else if (students[*numStudents].marks.average >= 70) {
        students[*numStudents].marks.grade = 'C';
    } else if (students[*numStudents].marks.average >= 60) {
        students[*numStudents].marks.grade = 'D';
    } else {
        students[*numStudents].marks.grade = 'F';
    }

    (*numStudents)++;
}

// Function to add teacher details
void addTeacher(struct Teacher teachers[], int *numTeachers) {
    printf("Enter ID: ");
    scanf("%s", teachers[*numTeachers].id);
    printf("Enter Name: ");
    scanf(" %[^\n]", teachers[*numTeachers].name);
    printf("Enter Subjects: ");
    scanf(" %[^\n]", teachers[*numTeachers].subjects);

    (*numTeachers)++;
}

// Function to modify/delete student details
void modifyDetails(struct Student students[], int numStudents, struct Teacher
teachers[], int numTeachers) {
    char usn[20];
```

```
    printf("Enter USN of the student to modify: ");
    scanf("%s", usn);

    for (int i = 0; i < numStudents; ++i) {
        if (strcmp(students[i].usn, usn) == 0) {
            // Found the student, modify details
            printf("Enter new Name: ");
            scanf(" %[^\n]", students[i].name);
            printf("Enter new Year: ");
            scanf("%d", &students[i].year);
            // ... other fields
            printf("Student details modified successfully.\n");
            return;
        }
    }

    printf("Student with USN %s not found.\n", usn);
}

// Function to display student result
void displayResult(struct Student students[], int numStudents) {
    char usn[20];
    printf("Enter USN of the student: ");
    scanf("%s", usn);

    for (int i = 0; i < numStudents; ++i) {
        if (strcmp(students[i].usn, usn) == 0) {
            // Found the student, display result
            printf("Result for Student %s:\n", students[i].name);
            printf("USN: %s\n", students[i].usn);
            printf("Name: %s\n", students[i].name);
            printf("Branch: %s\n", students[i].branch);
            printf("Year: %d\n", students[i].year);
            printf("Semester: %d\n", students[i].semester);
            printf("Marks: ");
            for (int j = 0; j < 5; ++j) {
                printf("%d ", students[i].marks.marks[j]);
            }
            printf("\nAverage: %.2f\n", students[i].marks.average);
            printf("Grade: %c\n", students[i].marks.grade);
            return;
        }
    }

    printf("Student with USN %s not found.\n", usn);
}

// Function to display all student details
void displayAllStudentDetails(struct Student students[], int numStudents) {
    printf("Student Details:\n");
```

```
    for (int i = 0; i < numStudents; ++i) {
        printf("Student %d:\n", i + 1);
        printf("USN: %s\n", students[i].usn);
        printf("Name: %s\n", students[i].name);
        printf("Branch: %s\n", students[i].branch);
        printf("Year: %d\n", students[i].year);
        printf("Semester: %d\n", students[i].semester);
        printf("Marks: ");
        for (int j = 0; j < 5; ++j) {
            printf("%d ", students[i].marks.marks[j]);
        }
        printf("\nAverage: %.2f\n", students[i].marks.average);
        printf("Grade: %c\n\n", students[i].marks.grade);
    }
}

// Function to display all teacher details
void displayAllTeacherDetails(struct Teacher teachers[], int numTeachers) {
    printf("Teacher Details:\n");
    for (int i = 0; i < numTeachers; ++i) {
        printf("Teacher %d:\n", i + 1);
        printf("ID: %s\n", teachers[i].id);
        printf("Name: %s\n", teachers[i].name);
        printf("Subjects: %s\n\n", teachers[i].subjects);
    }
}

// Function to display student details based on USN
void displayStudentDetailsByUSN(struct Student students[], int numStudents, const
char *usn) {
    for (int i = 0; i < numStudents; ++i) {
        if (strcmp(students[i].usn, usn) == 0) {
            printf("Student Details:\n");
            printf("USN: %s\n", students[i].usn);
            printf("Name: %s\n", students[i].name);
            printf("Branch: %s\n", students[i].branch);
            printf("Year: %d\n", students[i].year);
            printf("Semester: %d\n", students[i].semester);
            printf("Marks: ");
            for (int j = 0; j < 5; ++j) {
                printf("%d ", students[i].marks.marks[j]);
            }
            printf("\nAverage: %.2f\n", students[i].marks.average);
            printf("Grade: %c\n", students[i].marks.grade);
            return;
        }
    }
    printf("Student with USN %s not found.\n", usn);
}
```

```
// Function to display teacher details based on ID
void displayTeacherDetailsByID(struct Teacher teachers[], int numTeachers, const
char *id) {
    for (int i = 0; i < numTeachers; ++i) {
        if (strcmp(teachers[i].id, id) == 0) {
            printf("Teacher Details:\n");
            printf("ID: %s\n", teachers[i].id);
            printf("Name: %s\n", teachers[i].name);
            printf("Subjects: %s\n", teachers[i].subjects);
            return;
        }
    }
    printf("Teacher with ID %s not found.\n", id);
}

// Function to save student data to a file
void saveStudentsToFile(struct Student students[], int numStudents, const char
*filename) {
    FILE *file = fopen(filename, "w");
    if (file == NULL) {
        printf("Error opening file for writing: %s\n", filename);
        return;
    }

    for (int i = 0; i < numStudents; ++i) {
        fprintf(file, "%s;%s;%s;%d;%d;", students[i].usn, students[i].name,
students[i].branch, students[i].year, students[i].semester);
        for (int j = 0; j < 5; ++j) {
            fprintf(file, "%d;", students[i].marks.marks[j]);
        }
        fprintf(file, "%.2f;%c\n", students[i].marks.average, students[i].marks.grade);
    }

    fclose(file);
}

// Function to load student data from a file
void loadStudentsFromFile(struct Student students[], int *numStudents, const char
*filename) {
    FILE *file = fopen(filename, "r");
    if (file == NULL) {
        printf("File not found: %s\n", filename);
        return;
    }

    *numStudents = 0;
    char line[256];

    while (fgets(line, sizeof(line), file) != NULL && *numStudents < 10) {
        struct Student *student = &students[*numStudents];
```

```
        sscanf(line, "%19[^;];%49[^;];%49[^;];%d;%d;%d;%d;%d;%d;%d;%d;%f;%c",
               student->usn, student->name, student->branch, &student->year, &student->semester,
               &student->marks.marks[0], &student->marks.marks[1], &student->marks.marks[2],
               &student->marks.marks[3], &student->marks.marks[4],
               &student->marks.average, &student->marks.grade);
        (*numStudents)++;
    }

    fclose(file);
}

// Function to save teacher data to a file
void saveTeachersToFile(struct Teacher teachers[], int numTeachers, const char *filename) {
    FILE *file = fopen(filename, "w");
    if (file == NULL) {
        printf("Error opening file for writing: %s\n", filename);
        return;
    }

    for (int i = 0; i < numTeachers; ++i) {
        fprintf(file, "%s;%s;%s\n", teachers[i].id, teachers[i].name, teachers[i].subjects);
    }

    fclose(file);
}

// Function to load teacher data from a file
void loadTeachersFromFile(struct Teacher teachers[], int *numTeachers, const char *filename) {
    FILE *file = fopen(filename, "r");
    if (file == NULL) {
        printf("File not found: %s\n", filename);
        return;
    }

    *numTeachers = 0;
    char line[256];

    while (fgets(line, sizeof(line), file) != NULL && *numTeachers < 10) {
        struct Teacher *teacher = &teachers[*numTeachers];
        sscanf(line, "%19[^;];%49[^;];%99[^\n]", teacher->id, teacher->name, teacher->subjects);
        (*numTeachers)++;
    }

    fclose(file);
}
```

Output:

Admin:

```
Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 1

Admin Menu:
1. Add Student
2. Add Teacher
3. Modify/Delete Details
4.Display Student Details based on USN
5.Display Teacher Details based on ID
6.Display All Teacher Details
7.Display All Student Details
8. Back to Main Menu
Enter your choice: 1
Enter USN: 4MT21IS001
Enter Name: Abhilash Shetty
Enter Branch: ISE
Enter Year: 2
Enter Semester: 4
Enter Marks for Subject 1: 50
Enter Marks for Subject 2: 50
Enter Marks for Subject 3: 50
Enter Marks for Subject 4: 50
Enter Marks for Subject 5: 50

Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 1
```

```
Admin Menu:
1. Add Student
2. Add Teacher
3. Modify/Delete Details
4.Display Student Details based on USN
5.Display Teacher Details based on ID
6.Display All Teacher Details
7.Display All Student Details
8. Back to Main Menu
Enter your choice: 1
Enter USN: 4MT21IS002
Enter Name: Adithi
Enter Branch: ISE
Enter Year: 2
Enter Semester: 4
Enter Marks for Subject 1: 50
Enter Marks for Subject 2: 49
Enter Marks for Subject 3: 50
Enter Marks for Subject 4: 49
Enter Marks for Subject 5: 50

Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 1

Admin Menu:
1. Add Student
2. Add Teacher
3. Modify/Delete Details
4.Display Student Details based on USN
5.Display Teacher Details based on ID
6.Display All Teacher Details
7.Display All Student Details
```

```
Enter Marks for Subject 2: 49
Enter Marks for Subject 3: 50
Enter Marks for Subject 4: 49
Enter Marks for Subject 5: 50

Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 1

Admin Menu:
1. Add Student
2. Add Teacher
3. Modify/Delete Details
4.Display Student Details based on USN
5.Display Teacher Details based on ID
6.Display All Teacher Details
7.Display All Student Details
8. Back to Main Menu
Enter your choice: 4
Enter USN of the student to search: 4MT21IS001
Student Details:
USN: 4MT21IS001
Name: Abhilash Shetty
Branch: ISE
Year: 2
Semester: 4
Marks: 50 50 50 50 50
Average: 50.00
Grade: F


...Program finished with exit code 0
Press ENTER to exit console.
```

Teacher:

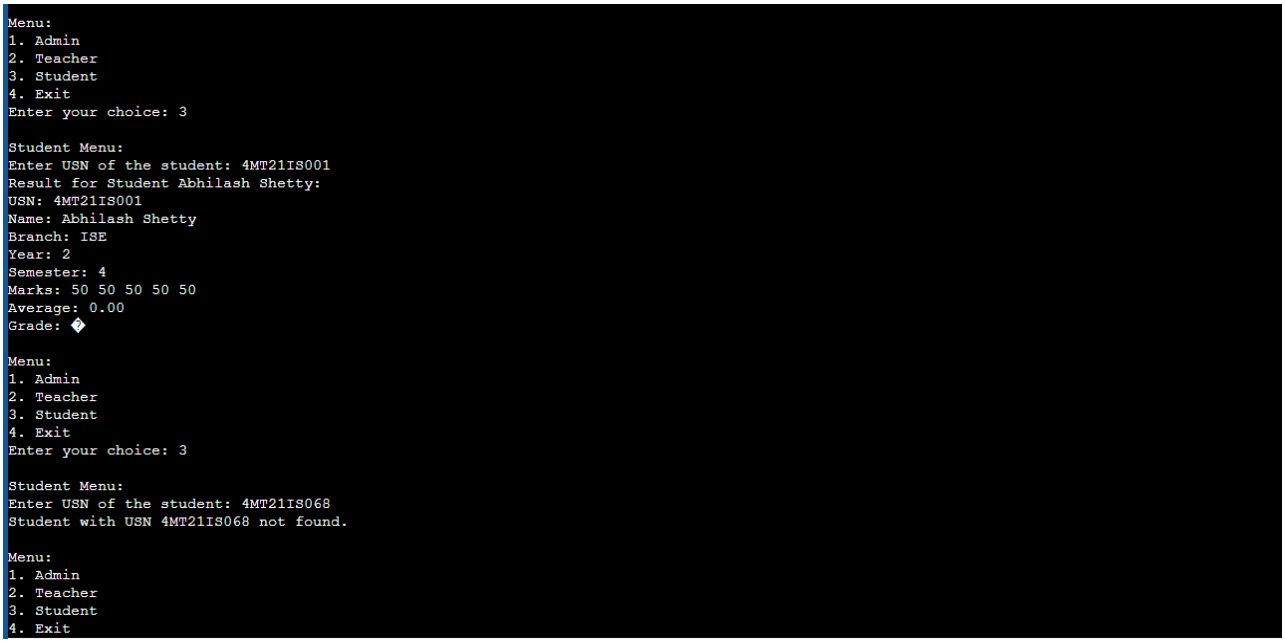
```
Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 2

Teacher Menu:
1. Search Student by USN
2. Search Teacher by ID
3. Display All Student Details
4. Display All Teacher Details
5. Back to Main Menu
Enter your choice: 1
Enter USN of the student to search: 4MT21IS001
Student Details:
USN: 4MT21IS001
Name: Abhilash Shetty
Branch: ISE
Year: 2
Semester: 4
Marks: 50 50 50 50 50
Average: 0.00
Grade: 
```

```
Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 2

Teacher Menu:
1. Search Student by USN
2. Search Teacher by ID
```

```
Teacher Menu:
1. Search Student by USN
2. Search Teacher by ID
3. Display All Student Details
4. Display All Teacher Details
5. Back to Main Menu
Enter your choice: 3
Student Details:
Student 1:
USN:

Name:
Branch:
Year: 0
Semester: 0
Marks: 0 0 -184401028 32685 -184453936
Average: -140917538722569456465519424045056.00
Grade: 

Student 2:
USN: ???
Name:
Branch:
Year: -184443992
Semester: 32685
Marks: -184444712 32685 -652715408 32767 -652715424
Average: 0.00
Grade:

Student 3:
USN: 4MT21IS001
Name: Abhilash Shetty
Branch: ISE
Year: 2
Semester: 4
```

```
Marks: -184444712 32685 -652715408 32767 -652715424
Average: 0.00
Grade:

Student 3:
USN: 4MT21IS001
Name: Abhilash Shetty
Branch: ISE
Year: 2
Semester: 4
Marks: 50 50 50 50 50
Average: 0.00
Grade: 

Student 4:
USN: 4MT21IS002
Name: Adithi
Branch: ISE
Year: 2
Semester: 4
Marks: 50 49 50 49 50
Average: 0.00
Grade: 


Menu:
1. Admin
2. Teacher
3. Student
4. Exit
Enter your choice: 4
Exiting...


...Program finished with exit code 0
Press ENTER to exit console.
```

Student:



5. Conclusion:

The Result System, developed using C programming and text files, provides a simple yet functional solution for educational institutions to manage academic records efficiently. While it may not have the complexity of larger systems, it fulfils its objectives effectively.

6. Future Enhancements:

Future enhancements may include the addition of password protection for user accounts, data encryption for increased security, and the incorporation of more advanced data manipulation features.