Student management system

Project Title: Student management system [Team – 1]

Submitted by:

1. B. ANUSHA-24KB1A0531
2. B. PAVANI-24KB1A0540
3. B. JAHNAVI-24KB1A0545
4. B. AMRUTHA-24KB1A0559

**Course**: [Data Structures]

**College**: [N.B.K.R Institute of Science And Technology]

**Branch**: CSE – C

**Year & Sem**: 1y – 2sem

**Acknowledgement:**

We would like to express our heartfelt gratitude to all those who supported us throughout the development of this group project. Our sincere thanks to our faculty members for their continuous guidance and motivation .We are also grateful to our families and friends for their unwavering support and encouragement. Special thanks to the authors of reference books and online resources which helped us gain better insights. Team collaboration played a significant role in overcoming challenges during the project. Without everyone's cooperation and contribution, completing this project would not have been possible.

**Abstract:**

The Student Management System is a console-based group project developed in C programming language .It helps efficiently manage student data with functionalities such as adding, searching, updating, and deleting records. Structures are used to maintain organized data for each student. The project allowed our team to strengthen concepts such as arrays, loops, functions, and modular programming. A user-friendly menu interface was designed for better interaction. Although simple, this project reflects the core fundamentals of software application development.

**Introduction:**

The Student Management System project was developed to provide an organized and efficient way to manage student data. Our group collaborated to design a menu-driven C program that supports various operations such as adding, displaying, updating, and deleting records. We utilized basic programming constructs to maintain simplicity and functionality. Through this project, we aimed to improve our teamwork skills, problem-solving ability, and understanding of structured programming. It simulates a real-world application where data organization is essential.

**Objective:**

The objective of our group project is to develop an efficient Student Management System using the C programming language. We aimed to design a program that can manage student details in an organized manner while practicing modular programming. Through this collaborative effort, we sought to enhance our skills in handling user inputs, processing data, and presenting results. The project also focused on developing our abilities to divide tasks, coordinate as a team, and meet deadlines. Ultimately, our goal was to reinforce theoretical knowledge through practical implementation.

**System Requirements**:

Software Requirements:

- C Language Compiler (GCC/Turbo C/Code::Blocks/DevC++)

- Operating System: Windows 7/8/10, Linux, or macOS

Hardware Requirements:

- Processor: Intel i3/i5/i7 or equivalent

- RAM: 2 GB minimum (4 GB recommended)

- Hard Disk: 200 MB free space

- Input Devices: Keyboard

- Output Devices: Monitor

**Project Description**:

**Problem Statement**:

Managing student data manually can lead to errors and inefficiency, especially when handling large numbers of records.

**Proposed Solution:**

We proposed the development of a Student Management System using C that could store, retrieve, update and delete records efficiently through a menu-driven console application.

**Key Features:**

- Add new student records

- Search for students by roll number

- Update student information

- Delete student records

- Display all student records in a tabular format

- User-friendly and simple interface

**Algorithm:**

1. Start the program.

2. Display menu options to the user.

3. Accept user choice.

4. Perform operations based on the selected choice:

- Add Student

- Display Students

- Search Student

- Update Student

- Delete Student

- Exit

5. Loop back to the main menu until 'Exit' is chosen.

6. End the program

**Testing and Validation**:

Our group conducted thorough testing after each module was developed. We tested functions like adding , searching, updating, and deleting student records independently. Edge cases such as invalid roll numbers and full database conditions were also tested. We ensured the system handled incorrect inputs gracefully without crashing. Integration testing verified that modules worked seamlessly together. The program was validated by simulating various real-world user interactions. As a team, we reviewed and refined the code based on the testing results.

**Limitations:**

Despite successful development, our Student Management System has some limitations. The number of students is limited to 100 due to static array allocation. Data is not persistent; it gets lost once the program closes because file handling was not implemented. User input validation can be further improved. Additionally, there is no sorting of records or filtering based on specific criteria like marks or year. These limitations point toward areas for further enhancement.

**Future Enhancements**:

For future improvements, we plan to incorporate file handling to ensure data persistence. Adding functionalities such as sorting students by name or marks, and filtering based on academic year would make the system more robust. We aim to improve input validation and error handling to make the application more user-friendly. A login system for authorized access can be added for security. Developing a GUI-based version of the system would further enhance user experience and professionalism.

**Conclusion**:

The Student Management System project provided our group with valuable hands-on experience in Cprogramming. Through effective collaboration, we were able to design, develop, test, and validate a fully functional system. It reinforced our understanding of structures, arrays, loops, and modular programming practices. Despite minor limitations, the project achieved its goals successfully. Working as a team helped us learn the importance of communication, task division, and version control. Overall, this project laid a strong foundation for tackling more complex software projects in the future.

**References:**

-Algorithms and Data Structures: The Basic Toolbox by Kurt Mehlhorn and Peter Sanders

-C Data Structures and Algorithms by Alfred V. Aho, Jeffrey D. Ullman, and John E. Hopcroft

-Problem Solving with Algorithms and Data Structures" by Brad Miller and David Ranum