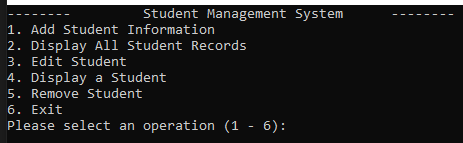
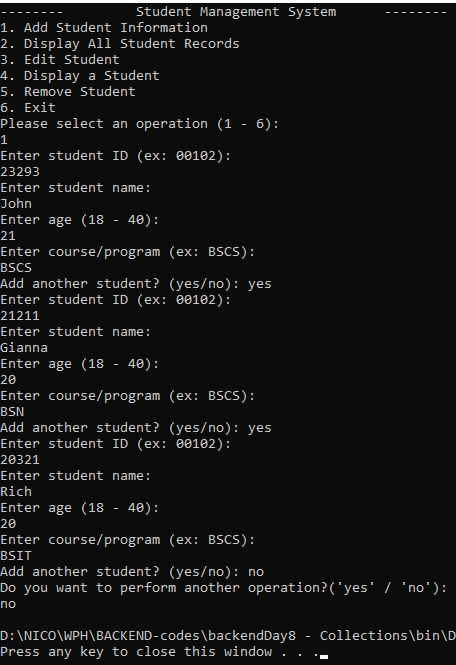


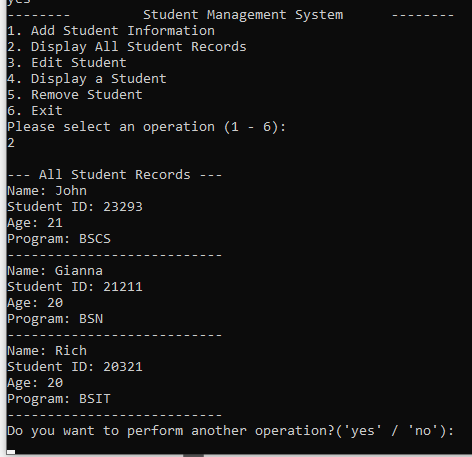
**OUTPUTS**

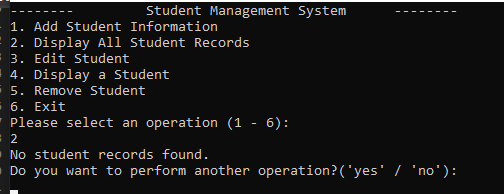
****

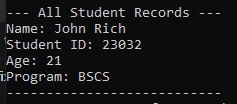
**ADD STUDENT**

****

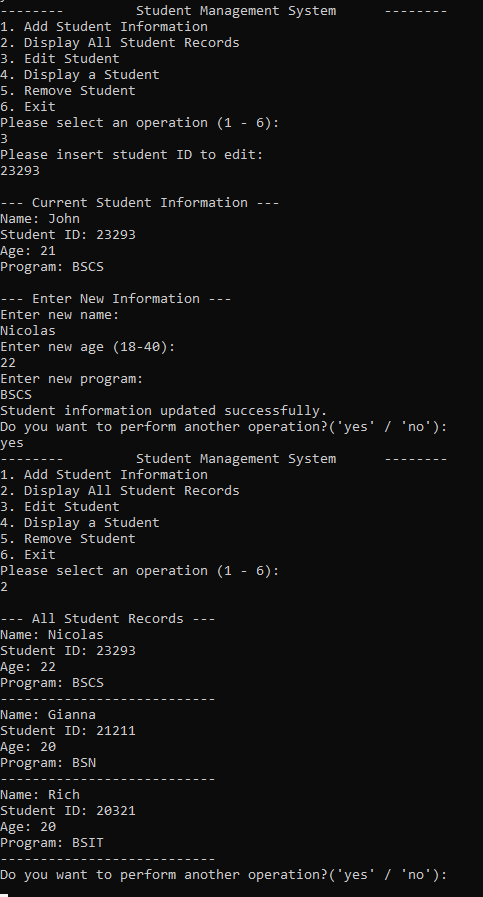
**DISPLAY ALL STUDENTS**

****

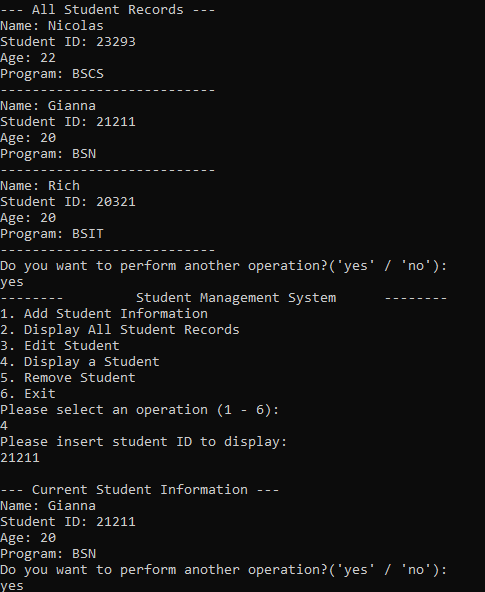
****

****

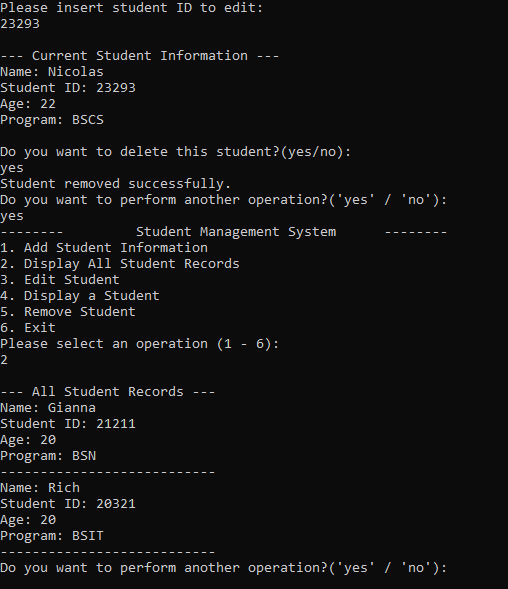
**EDIT STUDENT**

****

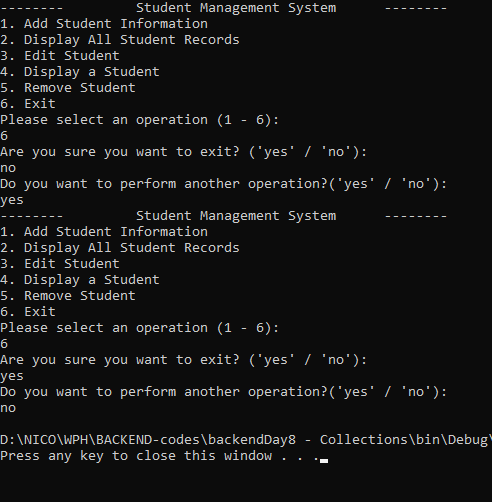
**DISPLAY A STUDENT**

****

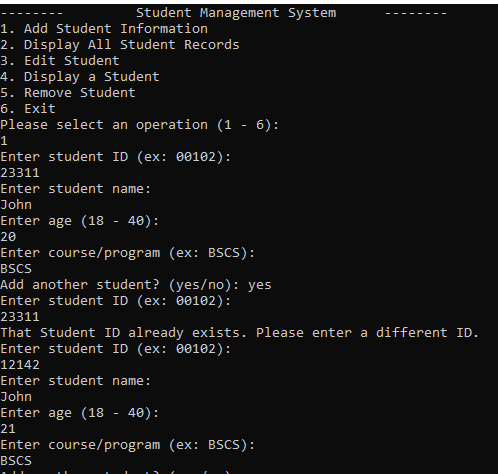
**DELETE A STUDENT**

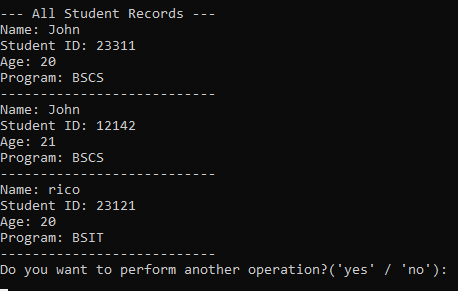
****

**EXIT SYSTEM**

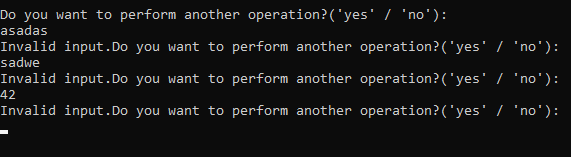
****

**STUDENT VALIDATION – NO SAME STUDENT ID**

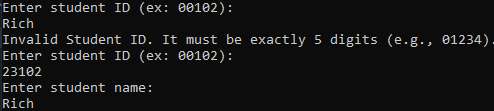
****

****

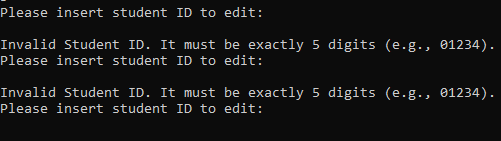
**INVALID INPUTS**

****

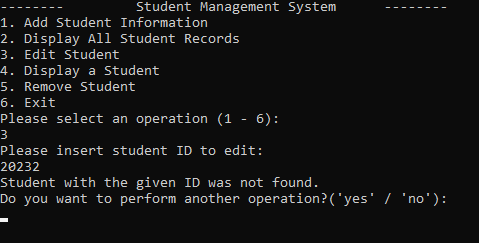
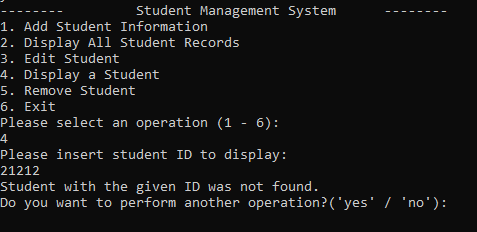
**Names in NUMBER REQUIRED areas**

****

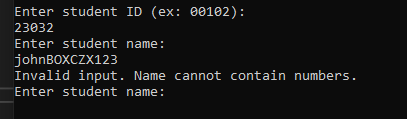
**EMPTY INPUTS**

****

**WRONG ID’s**

****

**NAMES WITH NUMBERS**

****

**REFLECTION**

While working on my Student Management System, I started by creating a Student class to store details like name, student ID, age, and program. I then made a separate class called ManageStudent where I placed all the methods for adding, editing, deleting, and displaying student records. This helped me keep everything organized and easier to work with.

I decided to use a List<Student> collection instead of arrays, stacks, or queues because using a list made it easier to add, remove, and update student records. It was more flexible and simple to manage compared to other data structures.

I also created an InputHelper class to handle input validation and error handling. I used try-catch blocks so the program wouldn’t crash when the user entered something wrong, like a letter instead of a number. I added validations to make sure the user doesn’t leave fields empty, that the age is between 18 to 40, and that the student ID is exactly five digits. I also made sure it checks for duplicate student IDs.

I used if and while statements to control the flow of the program and make sure only valid inputs go through. I also added user-friendly messages so the user knows if something went wrong or if the operation was successful. Additionally, I used switch case for calling the methods based on the operation they want to perform. Honestly, this project was pretty time-consuming and kind of overwhelming at times, so I had to take breaks just to reset and think clearly. But in the end, I learned a lot about organizing my code, using object-oriented concepts, and handling input properly in C#.