Student management system

A python console system

[School]

[Course title]

# TABLE OF CONTENTS

Contents

[Introduction and assumptions. 3](#_Toc134603880)

[Design of the program – using pseudocode or flowcharts – which adheres to the requirements provided above 4](#_Toc134603881)

[Programming Concepts with source code for explanation 5](#_Toc134603882)

[Screenshots of sample input/output and explanation 6](#_Toc134603883)

[Conclusion 11](#_Toc134603884)

[References 11](#_Toc134603885)

# TABLE OF FIGURES

[Figure 1: Student management system flowchart 4](#_Toc134603994)

[Figure 2: SMS main menu 6](#_Toc134603995)

[Figure 3: SMS selecting option 1 7](#_Toc134603996)

[Figure 4: SMS adding student with wrong details 7](#_Toc134603997)

[Figure 5: SMS updating student record 8](#_Toc134603998)

[Figure 6: SMS updating records with invalid data 8](#_Toc134603999)

[Figure 7: SMS listing all student records 9](#_Toc134604000)

[Figure 8: SMS searching for a student record 9](#_Toc134604001)

[Figure 9: SMS deleting a student record 10](#_Toc134604002)

[Figure 10: SMS exiting the system 10](#_Toc134604003)

# Introduction and assumptions.

**Introduction**

The Python-based Student Management System is designed to manage student records. Users can add, view, search, amend, and delete student data. The program is designed for educational institutions, such as schools, colleges, and universities, that require an effective data management system for student information.

**Assumptions**

The following Assumptions were made during program development:

1. The program presumes that the user has a fundamental understanding of the Python programming language.
2. The program assumes the user has a thorough comprehension of student management systems and their requirements.
3. Python is presumptively installed on the user's computer or device.
4. The program presume that the user is familiar with the command line interface and is able to navigate the system using it.
5. The program assumes the user has the necessary permissions and privileges to add, view, amend, search, and delete student information.
6. While utilizing the program, the program presumes that the user will provide valid input. Invalid input may result in errors or behavior that is unanticipated.
7. The program presume that the user will utilize the menus and not directly input commands.

## Design of the program – using pseudocode or flowcharts – which adheres to the requirements provided above

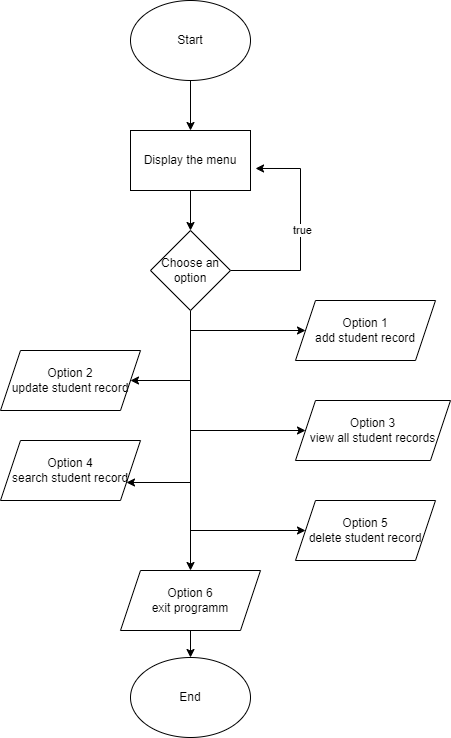


Figure : Student management system flowchart

## Programming Concepts with source code for explanation

**Overview:**

This application is a student management system that enables users to add, view, amend, search for, and delete student records. The program is written in Python and is composed of multiple functions that interact with a list of dictionaries representing student records. The program operates in a loop until the user chooses to terminate.

**Source Code:**

The source code is divided into several parts. The first part is the displayMenu() function that prints the program menu. The second part consists of functions that add, view, and validate student records. The third part is responsible for listing and searching records. The final part includes functions that delete and update records. The main program loop handles user input and calls the appropriate function based on the user's choice.

**Functionality:**

1. Display Menu - The displayMenu() function prints a menu with options for the user to choose from.
2. Add, View and Validate Student’s Record The addStudent() function prompts the user to enter a new student's information and validates the input. The function then creates a dictionary with the student's information and adds it to the studentList.
3. The viewAllStudents() function lists all the students in the studentList. If the studentList is empty, the function prints a message indicating that no records were found.
4. List and Search Record The searchStudent() function prompts the user to enter a student ID and searches for a student with that ID in the studentList. If a student is found, the function prints the student's information. If no student is found, the function prints a message indicating that no records were found.
5. Delete Record - The deleteStudent() function prompts the user to enter a student ID and deletes the student's record if found in the studentList. If no student is found, the function prints a message indicating that no records were found.
6. The updateStudent() function prompts the user to enter a student ID and updates the student's information if found in the studentList. The function validates the input and updates the corresponding values in the dictionary.

**User Interaction:**

The program starts by displaying a menu. The user can choose an option by entering the corresponding number. If the user enters an invalid choice, the program prompts the user to enter a valid choice. The program continues to run until the user chooses to exit.

## Screenshots of sample input/output and explanation

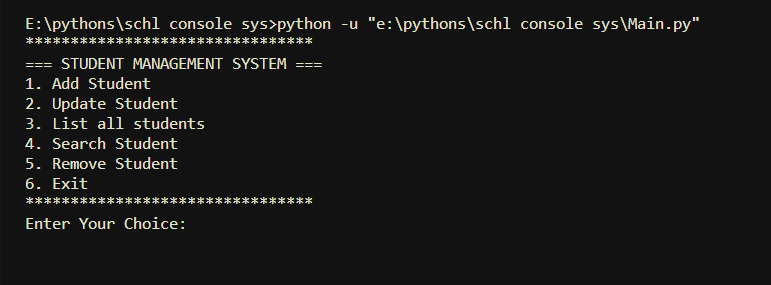


Figure : SMS main menu

The system on first running it, the main menu from which one can interact with the system.

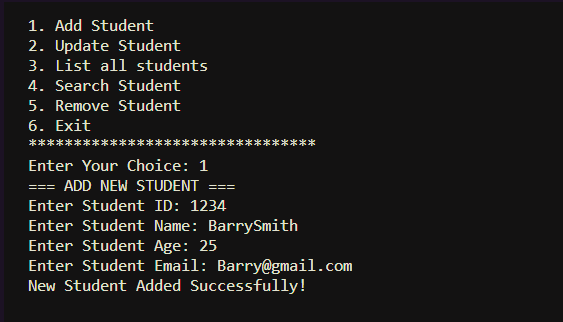


Figure : SMS selecting option 1

Adding a new student, while filling in the correct data, the student gets added successfully.

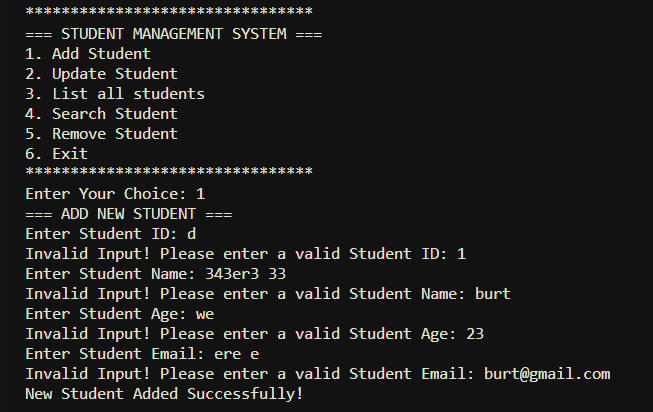


Figure : SMS adding student with wrong details

Adding a new student while inputting the wrong data, the system will ask you to input the correct data, will only accept correct data.

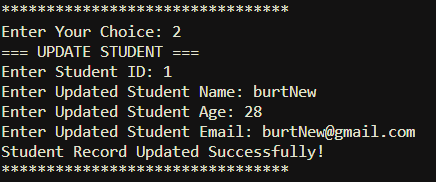


Figure : SMS updating student record

Selecting the option 2 to update the student record, and entering the correct data, this successfully updated the record.

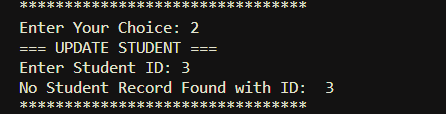


Figure : SMS updating records with invalid data

Selecting option 2 and inserting invalid student id displays an error.

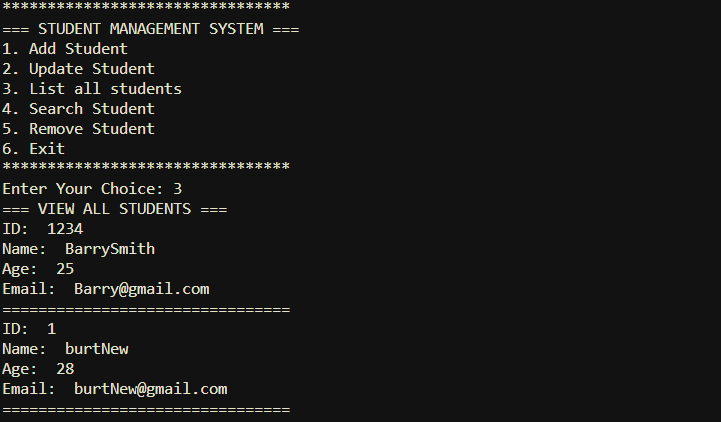


Figure : SMS listing all student records

Option 3 lists all students entered into the system with all their details listed.

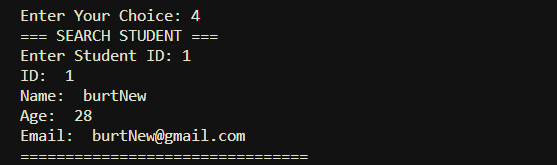


Figure : SMS searching for a student record

Option 4 searches for a student based on the id passed, then returns the student details matching the passed id and if the id is invalid it displays an error.

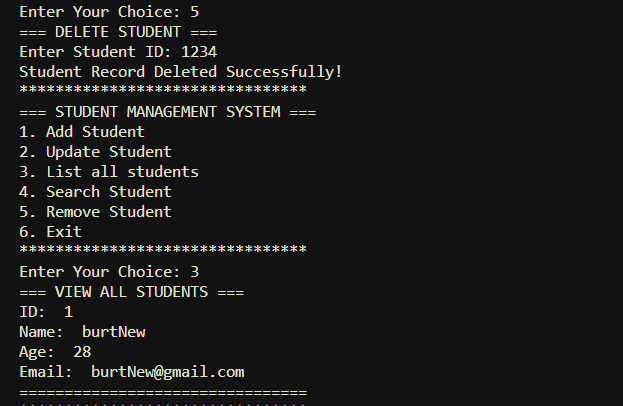


Figure : SMS deleting a student record

Option 5 deletes a student based on the passed id. If the id is invalid it displays an error indicating invalid id.

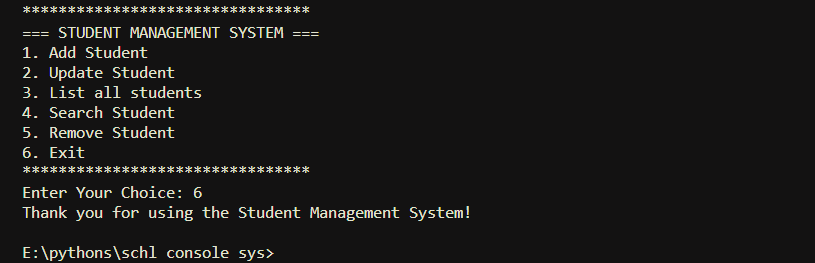


Figure : SMS exiting the system

The last option, option 6 exits the system gracefully.

## Conclusion

The system has features for adding, updating, searching, listing, and removing student records. The program's overall objective is to manage student records. The code is structured and well-organized, with each function carrying out a particular task. By employing input validation, it is made sure that user input is correctly structured and suitable for the intended application.

## References

1. Python, R. (n.d.). *Functional Programming in Python: When and How to Use It – Real Python*. Functional Programming in Python: When and How to Use It – Real Python. <https://realpython.com/python-functional-programming/>
2. *Understanding Code Reuse and Modularity in Python 3 - GeeksforGeeks*. (2017, April 19). GeeksforGeeks. https://www.geeksforgeeks.org/understanding-code-reuse-modularity-python-3/