**Project Documentation**

**Project Title: Student Management System (CLI-Based in C++)**

**1. Introduction**

The Student Management System (SMS) is a simple command-line program that helps manage student records. The system allows users to **add, display, search, update, and delete student records**.  
It is designed using **basic Object-Oriented Programming (OOP) concepts** such as **classes and objects**, without advanced features like vectors. Instead, arrays are used to store records.

**2. Objectives**

* To provide a simple student record-keeping system.
* To practice and implement OOP concepts in C++.
* To build a beginner-friendly project.

**3. Features**

* **Add Student**: Add a new student record.
* **Display Students**: Show all stored student records.
* **Search Student**: Find a student by roll number.
* **Update Student**: Modify details of an existing student.
* **Delete Student**: Remove a student record.

**4. Tools & Technologies**

* Programming Language: **C++**
* Development Environment: Any C++ compiler (e.g., Dev-C++, Code::Blocks, Visual Studio Code)
* Data Storage: **Array** (fixed size for simplicity)

**5. System Design**

* **Class Used**: Student
  + Attributes: rollNo, name, age, course
  + Functions: setData(), displayData()
* **Main Program**: Handles menu-driven interaction with the user.
* **Array**: Stores multiple Student objects.

**6. Flow of Program**

1. User runs the program.
2. A **menu** is displayed with various options (Add, Display, Search, Update, Delete, and Exit).
3. User selects an option → respective function is executed.
4. Loop continues until the user exit the program.

**7. Code Explanation (Brief)**

* **Class Student**: Represents student records.
* **Global Array of Students**: Used to store multiple student records.
* **Switch Menu in main ()**: Provides options to the user.
* **Functions**: Handle operations in the system like adding, updating, searching, and deleting.

**8. Limitations**

* Fixed number of students (array size).
* Data is not stored permanently (clears on program exit).
* CLI-based only, no graphical interface.

**9. Future Enhancements**

* Replace arrays with dynamic structures (e.g., vectors).
* Add file handling for permanent storage.
* Create a GUI-based version.

**10. Conclusion**

This project demonstrates how basic OOP concepts in C++ can be applied to build a functional and friendly Student Management System.