**A**

**PROJECT REPORT**

**ON**

**“Student Attendance management stystem”**

SUBMITTED BY:

**Mr. Tejas Narayan Satpute (2124UCEM1022)** SUBJECT:

# Programming in Problem

# Solving using c++

Under the guidance of

**Miss. Ishwari Tirse**



**Department of Computer Science and Engineering**

**Sanjivani Rural Education Society’s**

# SANJIVANI UNIVERSITY

**KOPARGAON – 423603, DIST : AHMEDNAGAR**

**2024-2025**

# 

# INDEX

|  |  |  |
| --- | --- | --- |
| **SR.**  **NO** | **CONTENT** | **PAGE NO.** |
| **1.** | **INTRODUCTION** | **3** |
| **2.** | **CODE** | **4** |
| **3.** | **OUTPUT** | **9** |
| **4.** | **CONCLUSION** | **11** |

**INTRODUCTION**

A Student Attendance Management System is a software application designed to streamline the process of tracking student attendance. It automates the recording, management, and reporting of student attendance data, providing a user-friendly interface for both teachers and administrators. The system enables easy monitoring of student attendance, including features such as attendance marking, attendance reports, and generating statistics for better analysis. It can also help in minimizing manual errors and reducing paperwork, making the management of large student data more efficient.

C++ is a powerful choice for developing a Student Attendance Management System due to its high performance, flexibility, and control over system resources. As a widely used object-oriented programming language, C++ provides the necessary tools to create efficient, modular, and maintainable software.

There are several advantages of a C++-based Student Attendance Management System, including high performance, flexibility, strong memory management, and a rich standard library that supports various data structures and algorithms.

|  |  |
| --- | --- |
| **CODE**    #include <iostream>  #include <fstream>  #include <iomanip>  #include <string>  using namespace std;  class Student {  private:  string rollNo;  string name;  int attendanceCount;  public:  Student() : attendanceCount(0) {}  void getStudentDetails() {  cout << "Enter Roll Number: ";  cin >> rollNo;  cout << "Enter Name: ";  cin.ignore();  getline(cin, name);  attendanceCount = 0;  }  void displayStudentDetails() const {  cout << left << setw(15) << rollNo << setw(25) << name << setw(10) << attendanceCount << endl;  }  void markAttendance() {  attendanceCount++;  }  string getRollNo() const {  return rollNo;  }  };  void addStudent() {  ofstream outFile("students.dat", ios::binary | ios::app);  Student student;  student.getStudentDetails();  outFile.write(reinterpret\_cast<char\*>(&student), sizeof(student));  outFile.close();  cout << "Student added successfully!\n";  }  void displayAllStudents() {  ifstream inFile("students.dat", ios::binary);  Student student;  cout << left << setw(15) << "Roll No" << setw(25) << "Name" << setw(10) << "Attendance" << endl;  cout << "----------------------------------------------------------\n";  while (inFile.read(reinterpret\_cast<char\*>(&student), sizeof(student))) {  student.displayStudentDetails();  }  inFile.close();  }  void markAttendanceForStudent() {  string rollNo;  cout << "Enter Roll Number to mark attendance: ";  cin >> rollNo;  fstream file("students.dat", ios::binary | ios::in | ios::out);  Student student;  bool found = false;  while (!file.eof() && !found) {  streampos pos = file.tellg(); // Get current position  file.read(reinterpret\_cast<char\*>(&student), sizeof(student));  if (student.getRollNo() == rollNo) {  student.markAttendance();  file.seekp(pos); // Go back to the position to update the record  file.write(reinterpret\_cast<char\*>(&student), sizeof(student));  cout << "Attendance marked for " << rollNo << "!\n";  found = true;  }  }  if (!found) {  cout << "Roll Number not found!\n";  }  file.close();  }  // Function to display the menu  void menu() {  int choice;  do {  cout << "\n==== Student Attendance Management System ====\n";  cout << "1. Add Student\n";  cout << "2. Display All Students\n";  cout << "3. Mark Attendance\n";  cout << "4. Exit\n";  cout << "Enter your choice: ";  cin >> choice;  switch (choice) {  case 1:  addStudent();  break;  case 2:  displayAllStudents();  break;  case 3:  markAttendanceForStudent();  break;  case 4:  cout << "Exiting program...\n";  break;  default:  cout << "Invalid choice. Try again.\n";  }  } while (choice != 4);  }  int main() {  menu();  return 0;  output  === Student Attendance Management System ====  1. Add Student  2. Display All Students  3. Mark Attendance  4. Exit  Enter your choice: 1  Enter Roll Number: 22  Enter Name: tejas  Student added successfully!  **CONCLUSION**  In conclusion, a C++ scientific calculator is a useful tool for solving math problems.  It can handle basic operations like addition and subtraction.  It also performs advanced functions like square roots, powers, and trigonometry.  The calculator uses the <cmath> library for these operations.  Users can easily input numbers and get results quickly.  It's simple to use and efficient for solving complex problems.  This program helps understand how coding can make calculations easier.  Overall, it shows how C++ can be used to create practical tools. | 4 |
|  |  |