**STUDENT MANAGEMENT SYSTEM**

* **Introduction**

The student class marks management system deals with the complete academic details of the student. It comprises of student name, rollno, marks, total and average. It can be accessed by the faculty and student. Faculty can change or update the marks of the student and student can only view results the faculty members try to maintain the records and administrator generates the report cards. The faculty should consider the mid marks, semester marks, assignment marks and lab marks for calculating the total marks based on the performance of the student.

* **DOMAIN ANALYSIS**

 Domain analysis is a process by which we learn about the domain to understand the problem better.

* **Hardware requirements :**

(Intel Pentium 4 processor 256 MB RAM 80 GB hardware)

* **Programming language:**

**C** language

* **PROBLEM STATEMENT :**

The management of marks is always done by the faculty person, that person should be careful, not to be error prone as the system maintains information of the student. The regular updating of marks of students after each examination for the ultimate

result provides a problem if any mistake occurs.

* **Requirement Analysis**

Requirements give the short and concise piece of information about the system and helps to solve the customer’s problem.

* **Background Information**

Students marks are being stored and analyzed for result manually by every time maintaining the records but today communication have the way of database technology for maintaining the different models for the relations and finding the respective entries of users in the database. So simply the method is to maintain or create a database, enter records, insert values, update every time and obtain the results.

* **Functional Requirement**

Functional requirements describe what the system designed should work. It also provides information for inputs, outputs, data stored, computations and timing. They define the fundamental sections that system must perform. The fundamental requirements for the student marks management consists of students, class administrator, faculty, results and further details referring usecases like: admin login,

login id,

login pwd, etc.

* **Non Functional Requirement**

Non-functional requirements are included during the development of the system according to necessity like categories of response time, availability, reliability and throughput. The constraints must be of three types: constraining reflecting, technology of system and project plan and development.

For every new user login there would be email verification designed for the security reasons so such extra considering constraints for development are non-functionals.**Non-Functional Requirements are**“ any Requirement that specifies **how** the system performs a certain function.”

* Architecture Using UML

**1. Use case Diagram**

A cornerstone part of the system is the functional requirements that the system fulfills. Use Case diagrams are used to analyze the system’s high-level requirements. These requirements are expressed through different use cases. We notice three main components of this UML diagram:

**Functional requirements** – represented as use cases; a verb describing an action

**Actors** – they interact with the system; an actor can be a human being, an organization or an internal or external application

**Relationships** between actors and use cases – represented using straight arr

**2. Class Diagram**

Class UML diagram is the most common diagram type for software documentation. Since most software being created nowadays is still based on the Object-Oriented Programming paradigm, using class diagrams to document the software turns out to be a common-sense solution. This happens because OOP is based on classes and the relations between them

**3. Sequence Diagram & Collaboration Diagram**

Sequence diagrams are probably the most important UML diagrams among not only the computer science community but also as design-level models for business application development. Lately, they have become popular in depicting business processes, because of their visually self-explanatory nature.

As the name suggests, sequence diagrams describe the sequence of messages and interactions that happen between actors and objects. Actors or objects can be active only when needed or when another object wants to communicate with them. All communication is represented in a chronological manner

**4. State Diagram**

State machine UML diagrams, also referred to as Statechart diagrams, are used to describe the different states of a component within a system. It takes the name state machine because the diagram is essentially a machine that describes the several states of an object and how it changes based on internal and external events.

**5. Activity Diagram**

Activity diagrams are probably the most important UML diagrams for doing business process modeling. In software development,

it is generally used to describe the flow of different activities and actions. These can be both sequential and in parallel. They describe the objects used, consumed or produced by an activity and the relationship between the different activities. All the above are essential in business process modeling.

**6. Component Diagram**

A component represents a modular part of a system that encapsulates its contents and whose manifestation is replaceable within its environment. In UML 2, a component is drawn as a rectangle with optional compartments stacked vertically.

Student marks analysis system is useful for regularly updating the students marks easily by the lecturers and these analysing systems can be extended to include many more functions for further improvement.

* **IMPLEMENTATION :**

**Prerequisites:**

  operations:

1. Store the First name of the student.
2. Store the Last name of the student.
3. Store the unique Roll number for every student.
4. Store the CGPA of every student.
5. Store the courses registered by the student.

* **CODE:**

void add\_student()

{

printf("Add the Students Details\n");

printf("-------------------------\n");

printf("Enter the first name of student\n");

// First name of the student

st[i].fname = "Rahul";

printf("Enter the last name of student\n");

// Last name of the student

st[i].lname = "Kumar";

printf("Enter the Roll Number\n");

// Roll Number of the student

st[i].roll = 1;

printf("Enter the CGPA you obtained\n");

// CGPA of the student

st[i].cgpa = 8;

printf("Enter the course ID"

" of each course\n");

// Storing the courses every student

// is enrolled in

for (int j = 0; j < 5; j++) {

st[i].cid[j] = j;

}

i = i + 1;

}

|  |
| --- |
| // C program for the implementation of student management system  #include <math.h>  #include <stdio.h>  #include <stdlib.h>  #include <string.h>  int i = 0;  struct sinfo {      char fname[50];      char lname[50];      int roll;      float cgpa;      int cid[10];  } st[55];   // Function to add the student  void add\_student()  {  printf("Add the Students Details\n");      printf("-------------------------\n");      printf("Enter the first name of student\n");      scanf("%s", st[i].fname);      printf("Enter the last name of student\n");      scanf("%s", st[i].lname);      printf("Enter the Roll Number\n");      scanf("%d", &st[i].roll);      printf("Enter the CGPA you obtained\n");      scanf("%f", &st[i].cgpa);      printf("Enter the course ID of each course\n");      for (int j = 0; j < 5; j++) {         scanf("%d", &st[i].cid[j]);      }      i = i + 1;  }    // Function to find the student by the roll number  void find\_rl()  {  int x;      printf("Enter the Roll Number of the student\n");      scanf("%d", &x);      for (int j = 1; j <= i; j++) {          if (x == st[i].roll) {              printf( "The Students Details are\n");              printf( "The First name is %s\n", st[i].fname);              printf("The Last name is %s\n", st[i].lname);              printf("The CGPA is %f\n”,st[i].cgpa);              printf("Enter the course ID of each course\n");          }          for (int j = 0; j < 5; j++) {              printf("The course ID are %d\n",st[i].cid[j]);          }          break;      }  }    // Function to find the student  // by the first name  void find\_fn()  {      char a[50];      printf("Enter the First Name"             " of the student\n");      scanf("%s", a);      int c = 0;  for (int j = 1; j <= i; j++) {          if (!strcmp(st[j].fname, a))  { printf( "The Students Details are\n");              Printf( "The First name is %s\n",st[i].fname);              Printf( "The Last name is %s\n".st[i].lname);              printf( "The Roll Number is %d\n ",st[i].roll);              printf("The CGPA is %f\n",st[i].cgpa);              printf( "Enter the course ID of each course\n");  for (int j = 0; j < 5; j++) {                  printf("The course ID are %d\n",st[i].cid[j]);              }c = 1;          }else             printf("The First Name not Found\n");      }  }    // Function to find  // the students enrolled  // in a particular course  void find\_c()  {      int id;      printf("Enter the course ID \n");      scanf("%d", &id);      int c = 0;        for (int j = 1; j <= i; j++) {          for (int d = 0; d < 5; d++) {              if (id == st[j].cid[d]) {                    printf( "The Students Details are\n");                  printf("The First name is %s\n”,st[i].fname);                  printf( "The Last name is %s\n",st[i].lname);                  printf( "The Roll Number is %d\n ",st[i].roll);                  printf( "The CGPA is %f\n",st[i].cgpa);  c = 1;    break;              }              else                  printf( "The First Name not Found\n");          }      }  }    // Function to print the number of students  void tot\_s()  {      printf("The total number of Student is %d\n", i);      printf("\n you can have max of 50 students\n");      printf("you can have %d more students\n",50 - i);  }    // Function to delete a student by the roll number  void del\_s()  {      int a;      printf("Enter the Roll Number which you want "to delete\n");      scanf("%d", &a);      for (int j = 1; j <= i; j++) {          if (a == st[j].roll) {              for (int k = j; k < 49; k++)                  st[k] = st[k + 1];              i--;          }      }      printf("The Roll Number is removed Successfully\n");  }    // Function to update a students data  void up\_s()  {        printf("Enter the roll number to update the entry : ");      long int x;      scanf("%ld", &x);      for (int j = 0; j < i; j++) {          if (st[j].roll == x) {              printf("1. first name\n"                     "2. last name\n"                     "3. roll no.\n"                     "4. CGPA\n"                     "5. courses\n");              int z;              scanf("%d", &z);              switch (z) {              case 1:                  printf("Enter the new first name : \n");                  scanf("%s", st[j].fname);                  break;              case 2:                  printf("Enter the new last name : \n");                  scanf("%s", st[j].lname);                  break;              case 3:                  printf("Enter the new roll number : \n");                  scanf("%d", &st[j].roll);                  break;              case 4:                  printf("Enter the new CGPA : \n");                  scanf("%f", &st[j].cgpa);                  break;              case 5:                  printf("Enter the new courses \n");                  scanf(                      "%d%d%d%d%d", &st[j].cid[0],                      &st[j].cid[1], &st[j].cid[2],                      &st[j].cid[3], &st[j].cid[4]);                  break;              }              printf("UPDATED SUCCESSFULLY.\n");          }      }  }    // Driver code  void main()    {      int choice, count;      while (i = 1) {          printf("The Task that you want to perform\n");          printf("1. Add the Student Details\n");          printf("2. Find the Student Details by Roll Number\n");          printf("3. Find the Student Details by First Name\n");          printf("4. Find the Student Details by Course Id\n");          printf("5. Find the Total number of Students\n");          printf("6. Delete the Students Details by Roll Number\n");          printf("7. Update the Students Details by Roll Number\n");          printf("8. To Exit\n");          printf("Enter your choice to find the task\n");          scanf("%d", &choice);          switch (choice) {          case 1:              add\_student();              break;          case 2:              find\_rl();              break;          case 3:              find\_fn();              break;          case 4:              find\_c();              break;          case 5:              tot\_s();              break;          case 6:              del\_s();              break;          case 7:              up\_s();              break;          case 8:              exit(0);              break;          } }} |

**Output:** The Task that you want to perform

1. Add the Student Details

2. Find the Student Details by Roll Number

3. Find the Student Details by First Name

4. Find the Student Details by Course Id

5. Find the Total number of Students

6. Delete the Students Details by Roll Number

7. Update the Students Details by Roll Number

8. TO Exit

Enter your choice to find the task 1

Add the Students Details ————————-

Enter the first name of student pravallika

Enter the last name of student kuppi

Enter the Roll Number 64

Enter the CGPA you obtained 8

Enter the course ID of each course 1 2 3 4 5

The Task that you want to perform

1. Add the Student Details

2. Find the Student Details by Roll Number

3. Find the Student Details by First Name

4. Find the Student Details by Course Id

5. Find the Total number of Students

6. Delete the Students Details by Roll Number

7. Update the Students Details by Roll Number

8. TO Exit

* **Conclusion:**

Student marks analysis system is useful for regularly updating the students marks easily by the lecturers and these analysing systems can be extended to include many more functions for further improvement.