**PROJECT REPORT**

***Submitted by***

**MOHAN NAGAL : RA2211003011960**

**AKSHATA GOYAL : RA2211003011951**

**KUSHAGRA AGARWAL : RA2211003011920**

***Under the Guidance of***

**Dr. V. V. RAMALINGAM**

**Associate Professor, Department of Computing Technologies**

***In partial satisfaction of the requirements for the degree of***

## BACHELOR OF TECHNOLOGY

**in**

**COMPUTER SCIENCE ENGINEERING**

## Logo, company name Description automatically generated

**SCHOOL OF COMPUTING**

# COLLEGE OF ENGINEERING AND TECHNOLOGY

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY**

**KATTANKULATHUR - 603203**

**MAY 2023**

Logo, company name

Description automatically generated

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR-603203

**BONAFIDE CERTIFICATE**

Certified that this Project Report titled **“CGPA Calculator”** is the bonafide work done by by M0HAN NAGAL – RA2211003011960, AKSHATA GOYAL – RA2211003011951, KUSHAGRA AGARWAL – RA2211003011920 who completed the project under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other work.

| **SIGNATURE**  **DR.V.V.Ramalingam**  **OODP – Course Faculty**  Associate Professor  Department of Computing Technologies  SRMIST | **SIGNATURE**  **Dr.M.Pushpalatha**  Professor & Head  Department of Computing Technologies  School of Computing  SRMIST |
| --- | --- |

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **S.No** | **CONTENTS** | **PAGE NO** |
| 1. | Problem Statement | **4** |
| 2. | Modules of Project | **7** |
| 3. | Diagrams | **11** |
|  | 1. Use case Diagram | **12** |
|  | 1. Class Diagram | **12** |
|  | 1. Behaviour Diagram | **13** |
|  | 1. State Chart Diagram | **14** |
|  | 1. Activity Diagram | **15** |
|  | 1. Package Diagram | **15** |
|  | 1. Component Diagram | **16** |
|  | 1. Deployment Diagram | **16** |
| 4. | Code/Output Screenshots | **18-27** |
| 5. | References ,Conclusion and Results | **28** |

**Chapter 1**

**INTRODUCTION**

**PREAMBLE**

As a student, keeping track of the academic performance is crucial to maintain good standing with the university. One key aspect of academic performance is Cumulative Grade Point Average (CGPA), which is a measure of the overall academic achievement. The CGPA is calculated based on the grades earned in each course and the credit hours assigned to each course.

The aim of this project is to develop a CGPA calculator using the C++ programming language. CGPA stands for Cumulative Grade Point Average and is a widely used method for calculating academic performance in universities and colleges.

In this project, we will describe the design and development process for our CGPA calculator. We will be using C++ programming language to develop the calculator, which will have a user-friendly interface and support the input of different grading systems.

The scope of our project is limited to the development of the CGPA calculator software application. We will not be addressing issues related to user adoption, marketing, or hardware compatibility.

* 1. **PROBLEM STATEMENT**

Manually calculating your CGPA can be a tedious and error-prone process, particularly when you have taken a large number of courses. Therefore, there is a need for an automated CGPA calculator program that can efficiently and accurately calculate the CGPA based on the grades obtained in each course.

Our task is to develop a CGPA calculator program using C++ that will automate the process of calculating the CGPA for a student based on the grades obtained in multiple courses. The program will prompt the user to input the number of courses and the grades obtained in each course, and then calculate the CGPA based on the grading system of the university. The program will also handle errors and exceptions gracefully.

The program is user-friendly, with clear prompts and easy-to-read output. It is also scalable and handles a large number of courses and grades. This program is useful for students who want to monitor their academic performance, professors who want to calculate the CGPA of their students, and administrators who want to track the academic progress of their students. With the development of such a CGPA calculator program, students can save time and reduce errors in the calculation process, allowing them to focus on their academic performance and progress towards graduation.

**1.2 METHODOLOGY**

The methodology for developing the CGPA calculator program using C++ involves a technical approach that includes several steps. These steps include:

1. Requirement gathering

2. Design and architecture

3. Implementation

4. Testing and validation

5. Deployment

The proposed method for the CGPA calculator program involves using C++ programming language, which is widely used and offers fast execution speeds. The program will be designed with a user-friendly interface, allowing users to easily input their grades and credit hours and obtain their CGPA. The program will also be designed to handle errors and provide appropriate feedback to the user.

The data entered by the user will be stored securely in the program's database, ensuring the privacy and confidentiality of the user's data. The program will also be designed to handle different grading systems used by universities and colleges, allowing it to be used by a wide range of students.

Overall, the proposed method for the CGPA calculator program involves an efficient technical approach that will deliver a reliable, user-friendly solution to the problem of calculating CGPA.

**1.3 SIGNIFICANCE OF THE WORK**

The CGPA calculator program developed using C++ has significant importance and application in the academic domain. Students, professors, and administrators can use this program to monitor and track academic performance, make informed decisions about course selection, and plan for graduation. By automating the CGPA calculation process, the program saves time and reduces errors, allowing students to focus on their academic performance and goals.

The application area of the CGPA calculator program extends beyond academia. It can be used by employers to assess the academic performance of job applicants, by scholarship providers to evaluate eligibility criteria, and by regulatory bodies to ensure compliance with academic standards. The program's ability to handle different grading systems used by universities and colleges makes it useful for a wide range of users.

The study of developing a CGPA calculator program using C++ brings value to the broader field of computer science and software development. It demonstrates the application of programming concepts and algorithms to solve real-world problems, such as automating academic processes. The program's user-friendly interface, secure data storage, and ability to handle different grading systems are significant contributions to the development of efficient and reliable software solutions.

In summary, the main contribution of this study is the development of a CGPA calculator program using C++ that provides an efficient, secure, and user-friendly solution to the problem of calculating CGPA. The program's ability to handle a large number of courses and grades, its user-friendly interface, and secure data storage system make it a valuable tool for students, professors, and administrators in the academic domain, as well as employers, scholarship providers, and regulatory bodies outside of academia.

**Chapter 2**

**IMPLEMENTATION**

**PREAMBLE**

This chapter will cover the programming language, tools and techniques used in the development process. We will also discuss the various functions and algorithms that have been implemented to achieve the desired functionality of the application.

**2.1** **PROPOSED METHODOLOGY**

The proposed methodology involves using the C++ programming language to develop a CGPA calculator that takes in input from the user, performs the necessary calculations, and outputs the final result. The program will be designed to be user-friendly and efficient, with clear instructions and error messages for the user. The development process will involve several stages, including designing and implementing the user interface, designing and implementing the calculation algorithms, testing and debugging the code, and refining the program based on user feedback.

**2.2** **PROPOSED METHODOLOGY MODULE**

1. User Interface Module

2. Calculation Module

3. Data Storage Module

4. Reporting Module

5. Testing Module

6. Deployment and Maintenance Module

**2.2.1** **User Interface Module**

The User Interface module of the CGPA Calculator system is responsible for providing an easy-to-use and intuitive interface for users to interact with the system. This module includes various graphical elements such as windows, dialog boxes, buttons, text boxes, and menus to enable users to input their academic information and retrieve their calculated CGPA.

The User Interface module will be implemented using C++ graphical user interface (GUI) libraries such as Qt or wxWidgets, which provide a range of widgets and graphical tools for developing cross-platform applications. The user interface will be designed to be user-friendly, visually appealing, and responsive to user actions. Additionally, the user interface will be designed to provide error handling and feedback to ensure that the user can input their academic information accurately and receive their calculated CGPA in a timely manner.

**2.2.2** **Calculation Module**

The calculation module is responsible for performing the necessary calculations to determine the CGPA of the student. This module takes input from the user interface module and uses the data to calculate the CGPA based on the formula provided by the university or college. The calculation module also validates the input data to ensure that it is accurate and complete before proceeding with the calculations.

The calculation module is implemented using C++ programming language, which provides a wide range of mathematical functions and operators. It uses arrays and loops to store and process the input data, and then computes the final CGPA value based on the specified formula. The output of the calculation module is then passed on to the user interface module for display to the user.

**2.2.3** **Data Storage Module**

The data storage module is responsible for storing the student's academic data in a persistent manner, so that it can be retrieved later on. This module will create and maintain a file that will store the student's personal information and their academic performance, including their grades and courses taken. The data will be stored in a structured format, which can be easily read and manipulated by the calculation module.

This module will also provide functionalities such as adding and deleting student information, updating grades and courses, and generating reports based on the stored data. The data storage module will be implemented using file handling in C++, which will ensure the secure and efficient storage of data.

**2.2.4 Reporting Module**

The reporting module is responsible for generating reports and displaying the calculated CGPA to the user. It will take input from the calculation module and display it in a user-friendly format. This module will also allow users to generate reports in various formats such as PDF, CSV, or Excel. The reporting module will be designed to be easily customizable so that users can choose which data to include in their reports. The module will also allow users to export their reports to various cloud-based storage solutions such as Google Drive or Dropbox.

**2.2.5 Testing Module**

The testing module is responsible for testing the various modules of the CGPA calculator system to ensure that they meet the specified requirements and perform as expected. This module involves creating test cases and executing them to identify any defects or errors in the system. The testing module also includes functionality testing, performance testing, and security testing to ensure that the system is reliable and secure.

The testing module is crucial in ensuring that the CGPA calculator system is free of errors and performs efficiently. Testing helps to identify and fix issues in the system before it is deployed to end-users, which can prevent potential problems and reduce maintenance costs. By ensuring that the system functions as expected and meets the requirements, the testing module helps to ensure that users can trust and rely on the CGPA calculator system.

**2.2.6 Deployment and Maintenance Module**

The deployment and maintenance module involves the final stage of the software development process, which includes the installation of the software system on the target system and the provision of support services to ensure its proper functioning. This module also covers the management of updates and bug fixes and the provision of technical support to users.

The deployment and maintenance module is crucial for the success of the software system as it ensures the system's reliability, scalability, and maintainability over its entire lifespan. Effective deployment and maintenance reduce the likelihood of system downtime and enable the system to adapt to changing user requirements and new technological advancements.

**2.3** **APPLICATION**

The CGPA calculator project can be applied in various educational institutions, such as schools, colleges, and universities, to calculate the CGPA and average CGPA of students. Here are some potential applications of the project:

1. Student evaluation

2. Progress reports

3. Curriculum planning

4. Recruitment

5. Research

**Chapter 3**

**SYSTEM REQUIREMENTS AND SYSTEM DESIGN**

**PREAMBLE**

This part focuses on the technical aspects of the program. It will discuss the system requirements and system design of the CGPA calculator program, including the hardware and software requirements, program architecture, and user interface design. This chapter aims to provide readers with a detailed understanding of the program's technical aspects and serve as a guide for future development and improvements.

**3.1 SOFTWARE REQUIREMENTS/ HARDWARE REQUIREMENTS**

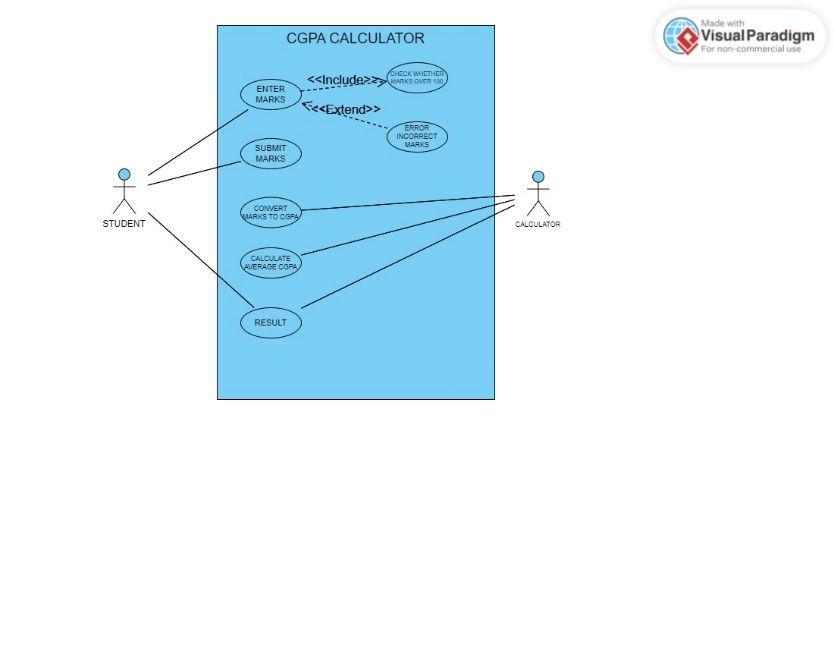
**3.1.1** **Software Requirements**

The software requirements for the CGPA calculator program include an operating system that supports C++ programming language, a development environment, and a C++ compiler. The program may also require the use of third-party libraries or utilities.

**3.1.2** **Hardware Requirements**

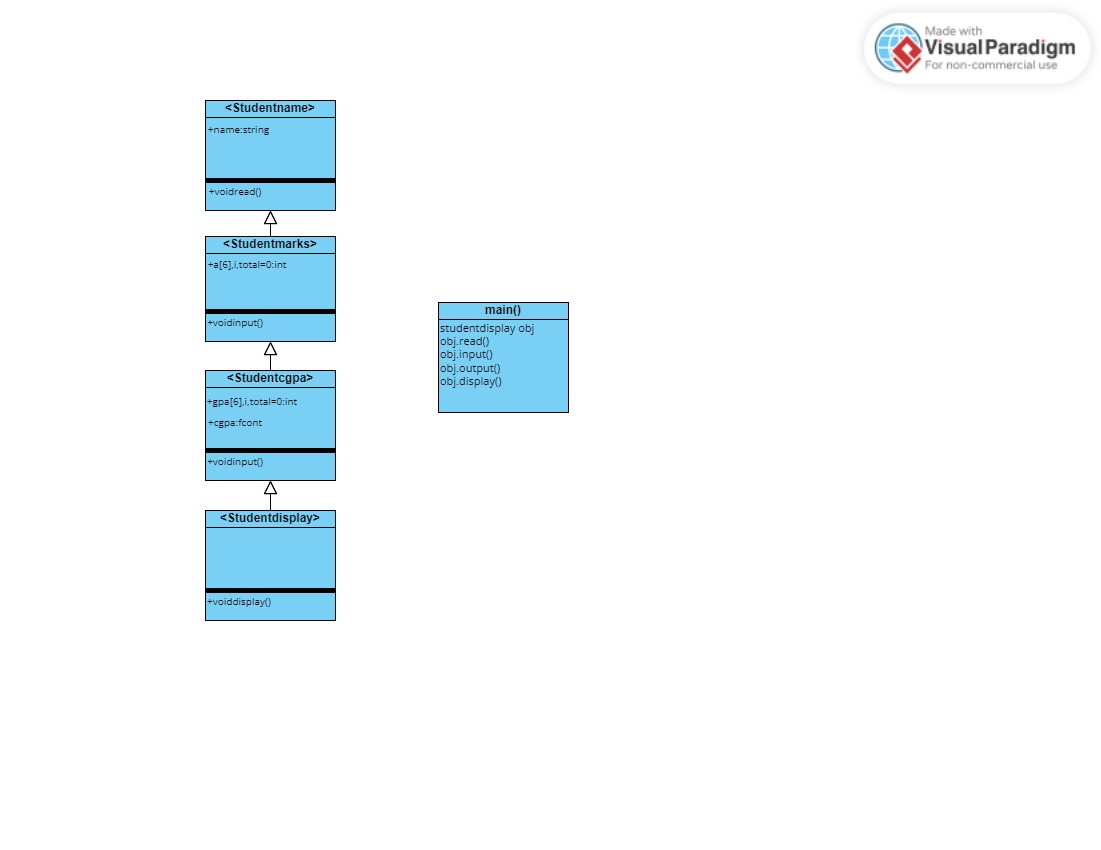
The hardware requirements for the CGPA calculator program are minimal and include a 1GHz processor, 2GB of RAM, and sufficient storage space. A standard monitor, keyboard, and pointing device are also necessary.

**3.2**  **USECASE DIAGARM**



A use case diagram for a CGPA calculator program would likely have two main actors: the user and the system. The user would interact with the system to input their course information, such as course names, credit hours, and grades. The system would then calculate the CGPA based on the input and display it to the user. Overall, the use case diagram would provide a high-level overview of the interactions between the user and the system and the various functions and features of the CGPA calculator program.

**3.3** **CLASS DIAGRAM**

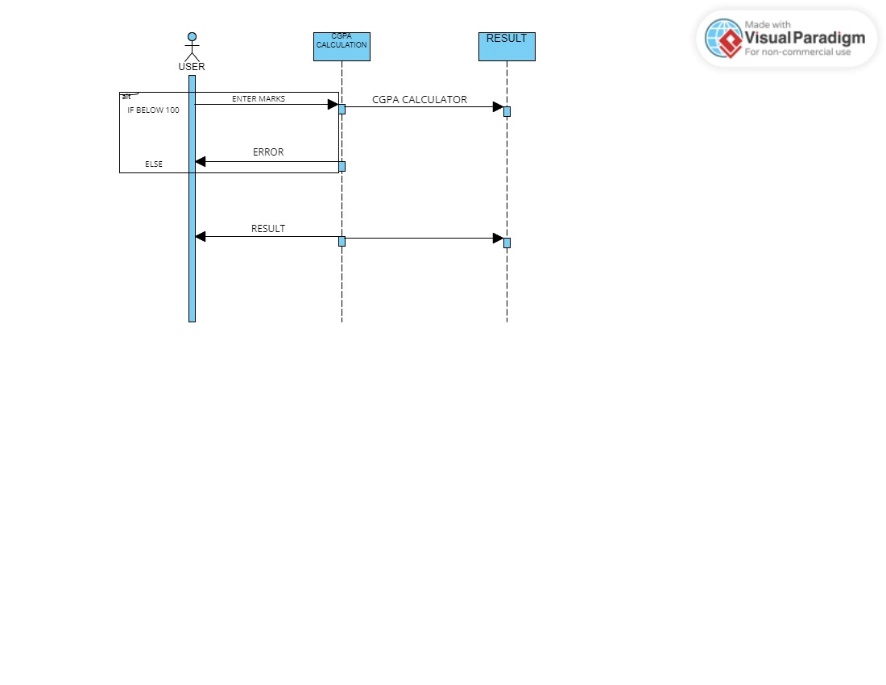


A class diagram for a CGPA calculator program would have three classes: Course, Student, and Calculator. The Student class would have a list of Course objects, and the Calculator class would calculate the CGPA based on the courses and grades associated with a given student. The Course class would not have any relationships with other classes.

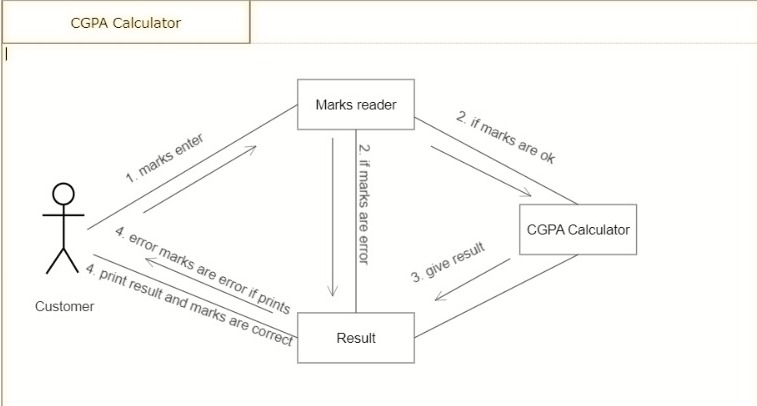
Overall, the class diagram would provide a visual representation of the relationships between the different classes in the CGPA calculator program and their attributes and methods.

**3.4** **BEHAVIOUR DIAGRAM**

**3.4.1** **Interaction Diagram (Sequence Diagram and Collaboration Diagram)**

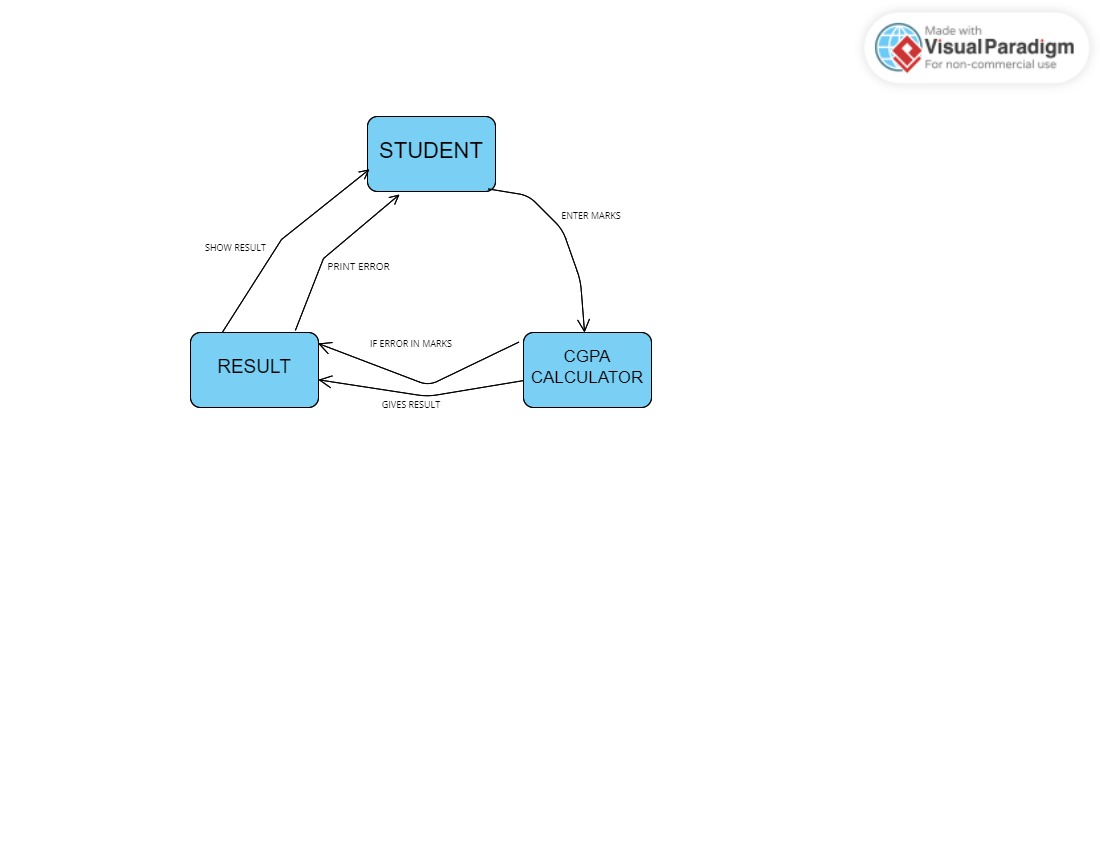


A sequence diagram for a CGPA calculator would show the flow of events when a user enters course information and calculates their CGPA. It would include the user, the CGPA calculator system, and the relevant classes involved in storing course information and calculating the CGPA. The diagram would demonstrate how the user's inputs trigger the system's calculations to produce the final result.



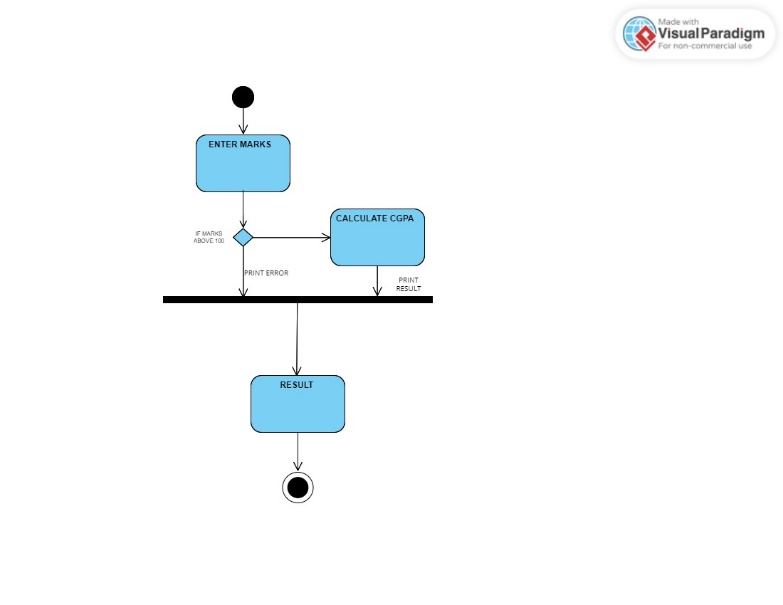
The collaboration diagram is used to show the relationship between the objects in a system. Both the sequence and the collaboration diagrams represent the same information but differently. Instead of showing the flow of messages, it depicts the architecture of the object residing in the system as it is based on object-oriented programming.

**3.4.2** **State Diagram**



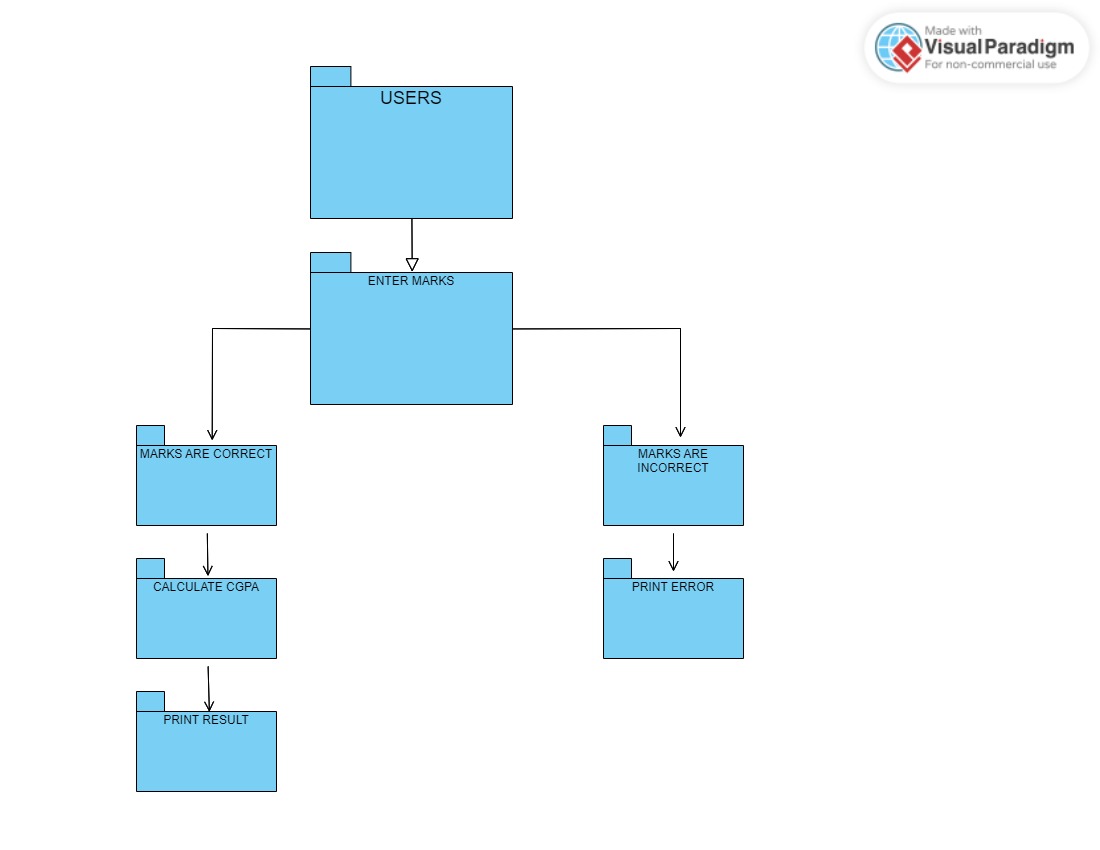
A state diagram for a CGPA calculator would show the different states a user can be in while using the program. It would include states such as “Student”, “CGPA Calculator”, “Result” and demonstrates how the user can move between these states based on their inputs and choices.

**3.4.3** **Activity Diagram**



An activity diagram for a CGPA calculator would show the different activities and decision points involved in using the program, such as entering marks, calculating the CGPA, and viewing the result. It provides a visual representation of how these activities flow together to produce the desired result of calculating and viewing the user's CGPA.

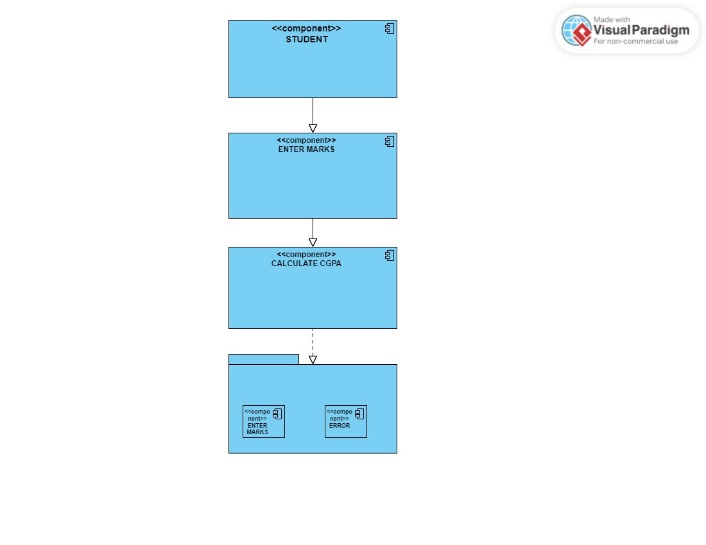
**3.5**  **PACKAGE DIAGRAM**



A package diagram for a CGPA calculator shows the different modules and components of the system, organized into packages. Each package contains relevant classes and components. The diagram provides a high-level overview of the program's architecture and organization, making it easier to maintain and modify the system in the future.

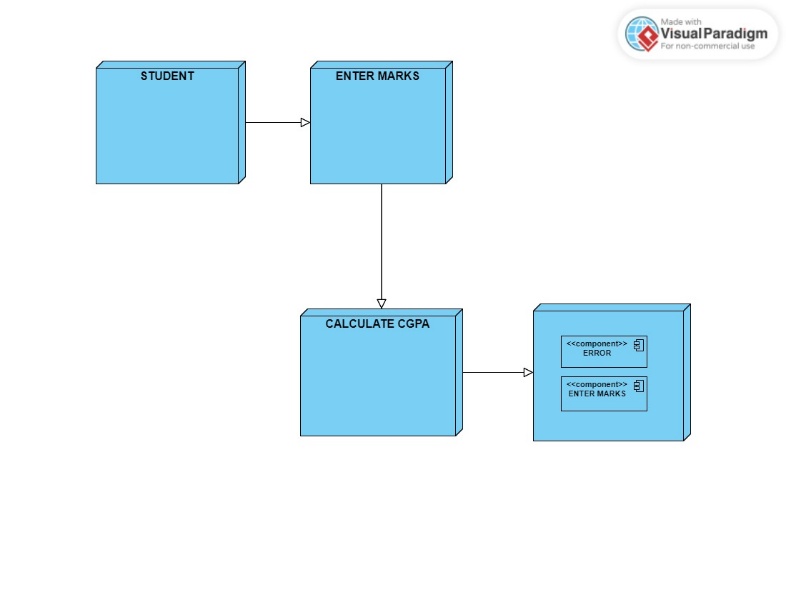
**3.6** **IMPLEMENTATION DIAGRAM**

**3.6.1** **Component Diagram**



A component diagram for a CGPA calculator program shows the physical and logical components of the system and their interconnections. It provides a detailed view of the program's architecture, helping to identify areas for improvement and ensure that the system meets its requirements.

**3.6.2** **Deployment Diagram**



A deployment diagram shows how software components are deployed on hardware components in a system. It helps ensure that the software is correctly installed and configured on the appropriate hardware and is an important tool for system administrators and developers.

**Appendix A**

**SOURCE CODE**

#include<iostream>

#include<bits/stdc++.h>

#include<iomanip>

#include<cstdio>

#include<stdlib.h>

#include<conio.h>

using namespace std;

class studentname

{

public:

string name;

void read()

{

cout<<"Enter name=";

cin>>name;

}

void login()

{

char ui[]="kushagra";

char psw[]="123";

char gui[10], gpsw[10];

int i;

cout<<"Enter Teacher's ID :";

cin>>gui;

cout<<"Enter Teacher's Password :";

i = 0;

do

{

gpsw[i] = getch(); // read password character without echoing

if(gpsw[i] != '\r') // ignore enter key

cout << ""; // print '' instead of the password character

i++;

}

while(gpsw[i-1] != '\r' && i < 10); // read until enter key or max password length

gpsw[i-1] = '\0'; // add null terminator at the end of password

if(strcmp(ui,gui)==0 && strcmp(psw,gpsw)==0)

{

cout<<"\n\n\t\t\t\t\t\t\tLOGGED IN SUCCESSFULLY"<<endl<<endl;

}

else

{

cout<<"\n\n\t\t\t\t\t\t\tINVALID ID or PASSWORD";

getch();

exit(0);

}

}

};

class studentmarks:public studentname

{

public:

int a[6],total=0,i,sum=0;

void input()

{

for(i=0;i<6;i++)

{

if(i==0)

{

cout<<"Enter Marks of Telugu=";

}

else if(i==1)

{

cout<<"Enter Marks of Hindi=";

}

else if(i==2)

{

cout<<"Enter Marks of English=";

}

else if(i==3)

{

cout<<"Enter Marks of Maths=";

}

else if(i==4)

{

cout<<"Enter Marks of Sceince=";

}

else

{

cout<<"Enter Marks of Social=";

}

cin>>a[i];

total=total+a[i];

if(a[i]>100){

sum=sum+1;}

}

}

};

class studentcgpa:public studentmarks

{

public:

int gpa[6],total1=0;

float cgpa;

void output()

{

if(sum==0){

for(i=0;i<6;i++){

if(a[i]>90)

{

gpa[i]=10;

}

else if(a[i]>80&&a[i]<91)

{

gpa[i]=9;

}

else if(a[i]>70&&a[i]<81)

{

gpa[i]=8;

}

else if(a[i]>60&&a[i]<71)

{

gpa[i]=7;

}

else if(a[i]>50&&a[i]<61)

{

gpa[i]=6;

}

else if(a[i]>40&&a[i]<51)

{

gpa[i]=5;

}

else if(a[i]>34&&a[i]<41)

{

gpa[i]=4;

}

else

{

gpa[i]=0;

}

total1=total1+gpa[i];

}}

cgpa=float(total1)/6;

}

};

class studentdisplay:public studentcgpa

{

public:

void display(){

if(sum==0){

cout<<"Student Name:"<<name<<endl;

cout<<"Total Marks:"<<total<<endl;

cout<<"CGPA in Telugu:"<<gpa[0]<<endl;

cout<<"CGPA in Hindi:"<<gpa[1]<<endl;

cout<<"CGPA in English:"<<gpa[2]<<endl;

cout<<"CGPA in Maths:"<<gpa[3]<<endl;

cout<<"CGPA in Science:"<<gpa[4]<<endl;

cout<<"CGPA in Social:"<<gpa[5]<<endl;

cout<<"Total CGPA:"<<setprecision(3)<<cgpa;

}

else{cout<<"ERROR";}}

};

int main()

{

studentdisplay obj;

obj.login();

obj.read();

obj.input();

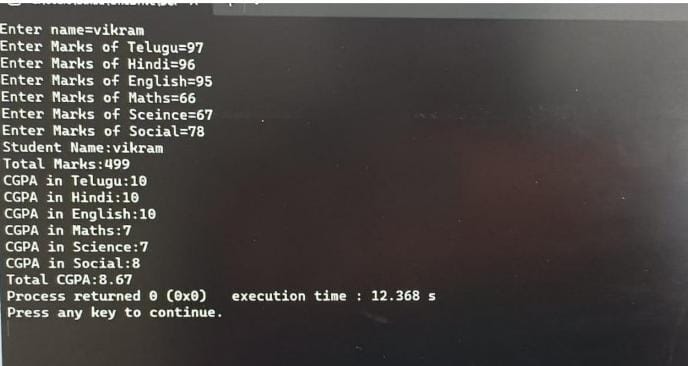
obj.output();

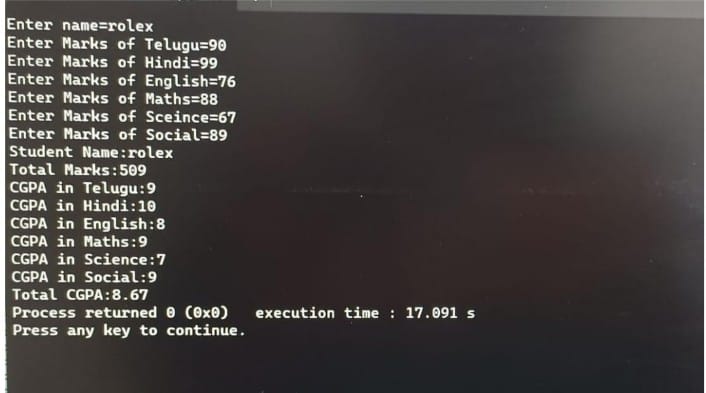
obj.display();

}

**Appendix B**

**SCREENSHOTS**





**5.1 REFERENCES**

YouTube

W3schools

GeeksforGeeks

**5.2** **CONCLUSION**

The CGPA calculator project is a software tool designed to calculate the CGPA and average CGPA of students in educational institutions. The project can help to automate the process of CGPA calculation and generate progress reports more efficiently, thereby improving productivity and efficiency in schools.

**5.3**  **RESULTS**

As the project involved creating a CGPA calculator using C++, the main outcome is a fully functional software application that can calculate the CGPA of a student based on their grades. The application has a user-friendly interface that allows students to input their grades easily, and the calculation module can perform the necessary calculations accurately.

The data storage module stores the input data securely, and the reporting module can generate a report of the calculated CGPA. The testing module ensures that the application performs as expected and that there are no errors or bugs.

The comparison of the result of the CGPA calculator project can be made with other existing CGPA calculator software tools or manual methods used for calculating CGPA. Some of the potential result comparisons are Accuracy, Speed, User-friendliness, Functionality, Maintenance and Support.

Overall, the project outcome is a successful CGPA calculator that can help students calculate their CGPA easily and accurately. The project has achieved its objective of providing a useful tool for students, and it can be used in various academic institutions to simplify the calculation of CGPA.