PROJECT REPORT

Submitted by

P MANVITHA RAYAL RA2211003010001
GOPIKRISHNAN PRABHAKARAN RA2211003010043
ABDUL VAHAB K RA2211003010049
LUKE SAJEEV KOSHY RA2211003010032

Under the Guidance of

Dr.B.ARTHI

Associate Professor, Department of Computing Technologies

In partial satisfaction of the requirements for the degree of

BACHELOR OF TECHNOLOGY
in
COMPUTER SCIENCE ENGINEERING

SCHOOL OF COMPUTING

COLLEGE OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

KATTANKULATHUR - 603203

MAY 2023

SRM INSTITUTION OF SCIENCE AND TECHNOLOGY
KATTANKULATHUR-603203

BONAFIDE CERTIFICATE

Certified that this Project Report titled **"STUDENT DATABASE MANAGEMENT"** is the bonafide work done by P Manvitha Rayal (RA2211003010001), Luke Sajeev Koshy (RA2211003010032), Gopikrishnan Prabhakaran (RA2211003010043), Abdul Vahab (RA2211003010049) 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.B.Arthi** **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

1. PROBLEM STATEMENT

The student must register by entering the name and password to login the form. The admin selects the particular student to view the details about that student and maintains the student details. This process of the student information system is described sequentially through following steps. The student registers the system. The admin login to the student information system. He/she searches for the list of students. Then select the particular student. Then view the details of that student. After displaying the student details then logout.

- **Effective for Administration Purpose**
- **Cheap**
- **Better Service**

MODULES OF THE PROJECT

LOGIN : It's a multiple login interface where you can login as ADMIN or as a CLERK.

Admin: The admin will have the privileges to access all the functions in the applications.
Clerk: The Clerk is restricted to some privileges of the application.

MANAGE STUDENT : The Manage Student module will contain two Sub Modules, that is Manage Existing Student and Add new Student. In Manage Existing Student Module we can manipulate the Existing Student Information and Update. In Add new Student Module we can add the new Student to the Database.

MANAGE SUBJECT : In Manage Subject Module we have two Sub Modules. In Manage Existing Subject Module we can manipulate and update the Existing Subjects. In Add new subject Module we can add the New Subjects which contains the Subject name, Subject Code and the Department in which the subject is related to.

MANAGE FEES : This module contains the Fees amount of each Department and this Module is Available only to the Admin. The Admin can Change the Fees Amount for Every Department

REPORT : This Module Contains the All Reports related to the Students such as Student Details, Marks of Student, Fees Information etc.. The Admin or the Clerk can print the Reports.

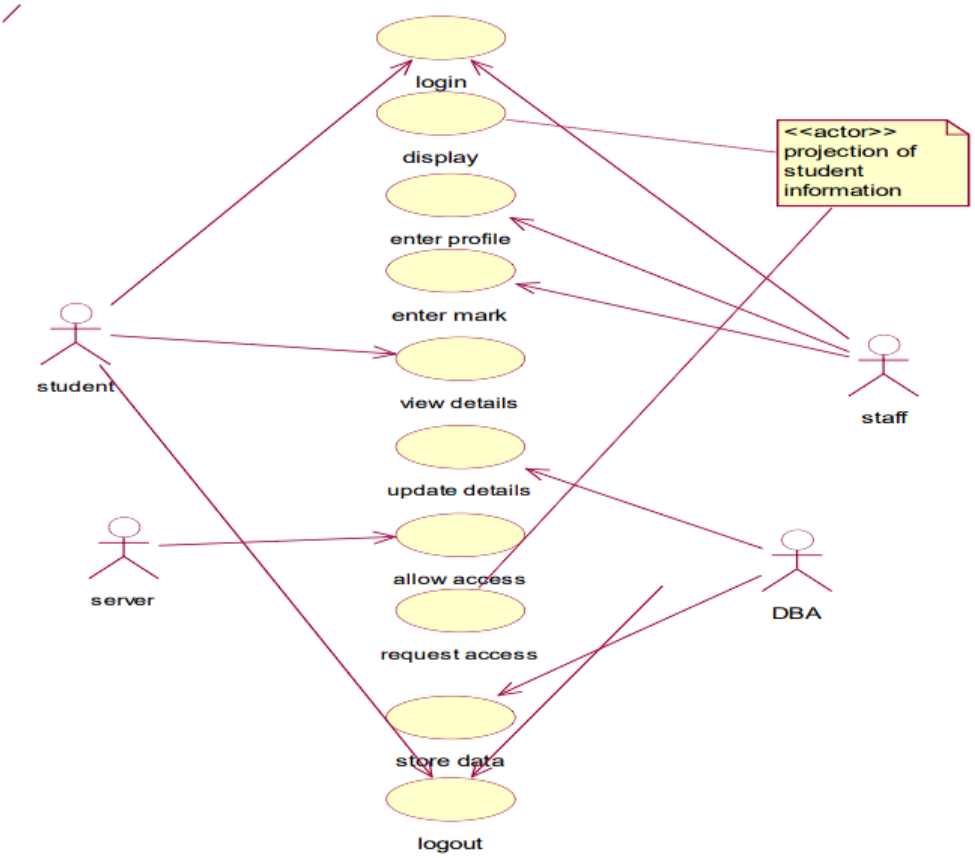
SETTING : In this Module every user can Change their password and the User Name.

BACK UP : This module deals with backing up data for future references or to recover the database tables.

DIAGRAMS :

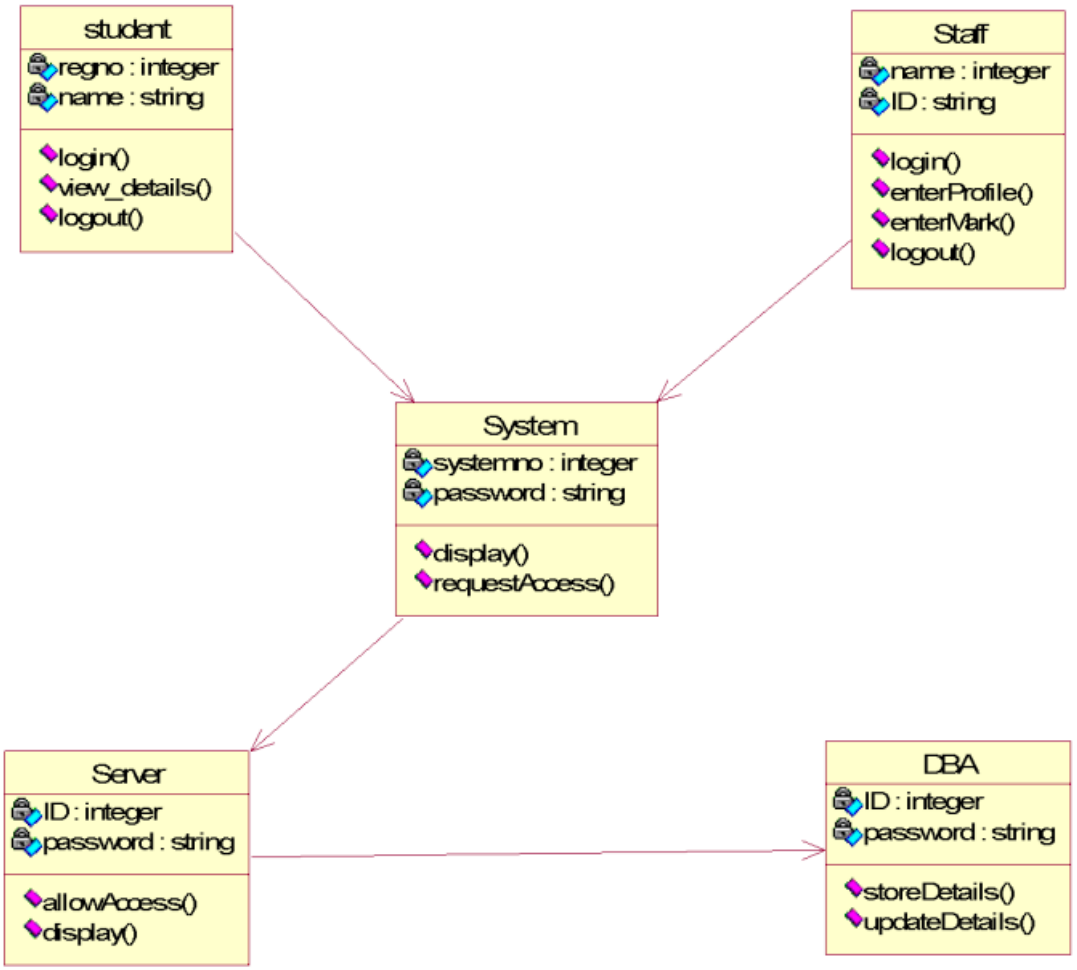
USE CASE DIAGRAM

Use case diagram is a graph of actors, a set of use cases, association between the actors and the use cases and generalization among the cases. Use case diagram is a list of actions or events. Use case diagram was drawn to represent the static design view of the system. Steps typically define the interactions between a role and a system to achieve a goal. The use case diagram consists of various functionality performed by the actors like student, staff, system, DBA and server. The use case diagram consists of various functionality like login, display, enter profile, enter mark, view details, update details, allow access, request access, store details, logout.



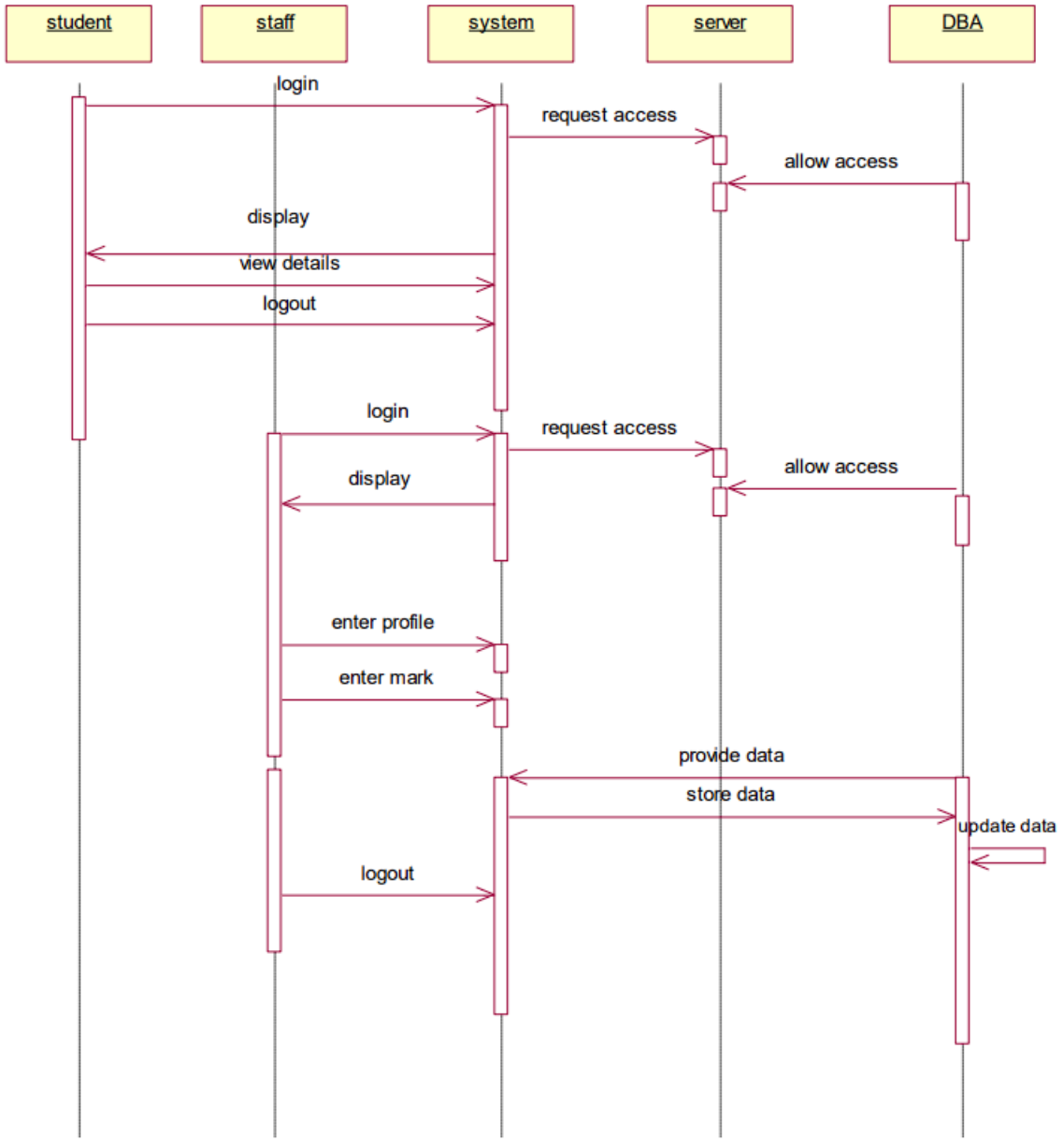
CLASS DIAGRAM

The class diagram is the graphical representation of all classes used in the system. The class diagram is drawn as a rectangular box with three components or compartments like class name, attributes and operations. The student information system makes use of the following classes like student, staff, system, DBA and server.



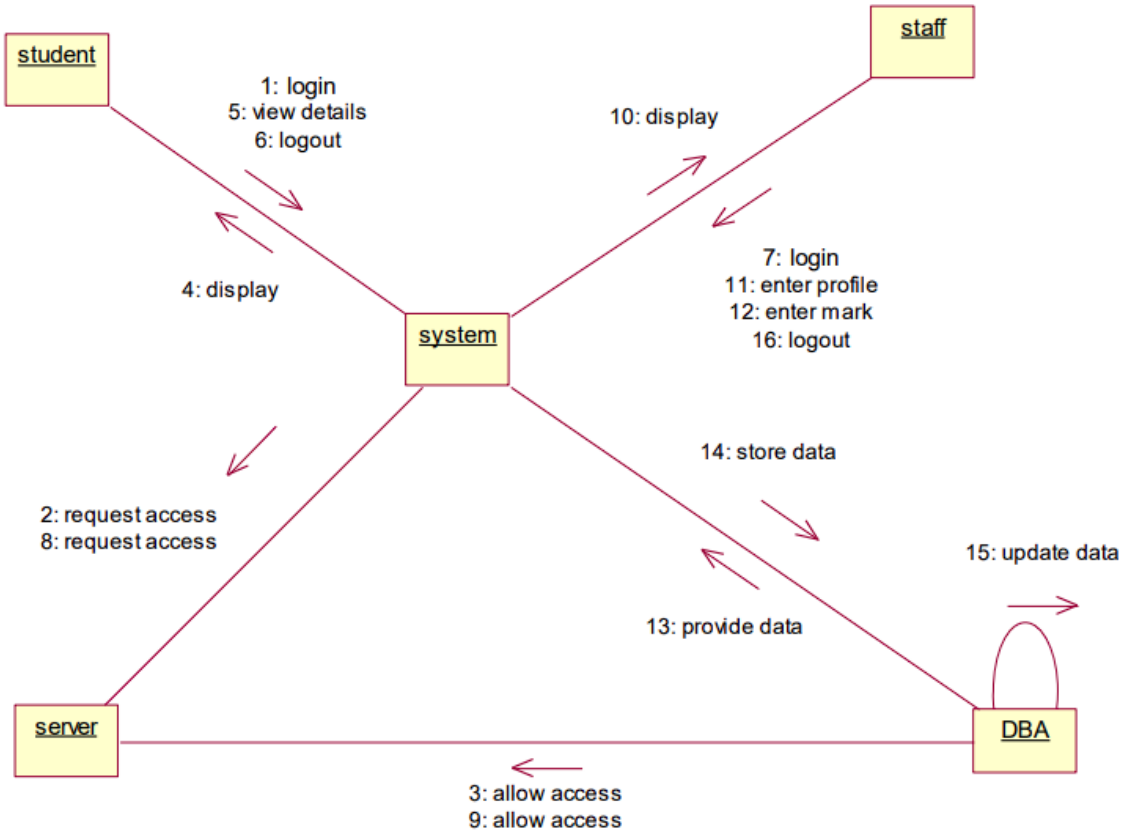
SEQUENCE DIAGRAM

A Sequence diagram represents the sequence and interaction of a given use case or scenario. Sequence diagrams capture most of the information about the system. Here the sequence starts between the student and the system. The second half of interaction takes place between staff and system then by police and followed by database. The student first login to the system and then view the details. Staff login to the system, enter a mark and enter the details of the student. DBA store and update the details of the student.



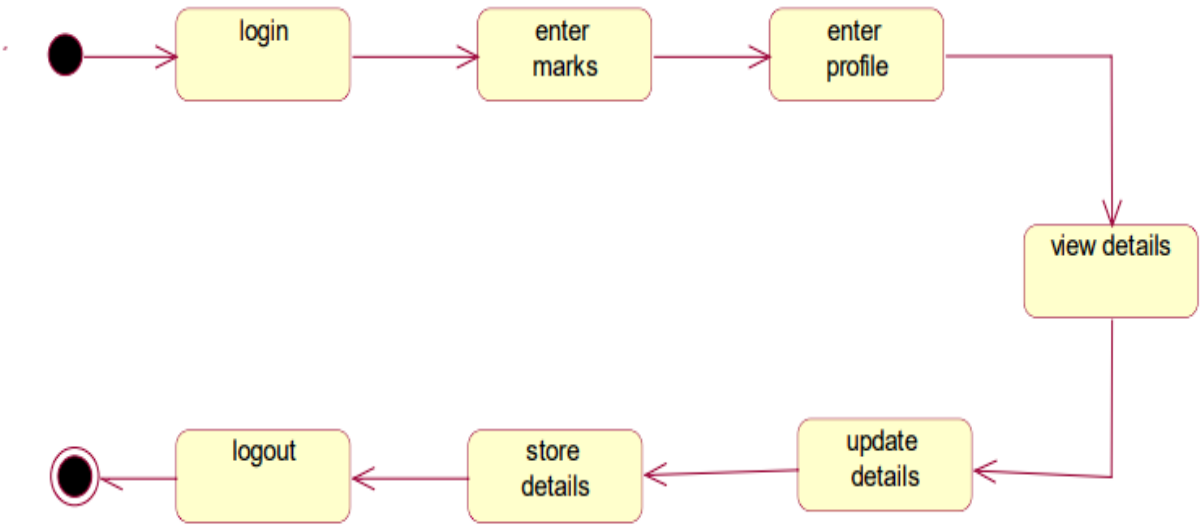
COLLABORATION DIAGRAM

A Collaboration diagram represents the collaboration in which is a set of objects related to achieve a desired outcome. In collaboration, the sequence is indicated by numbering the message several numbering schemes are available. Login, request access, allow access, display, view details, logout, login, request access, allow access, display, enter profile, enter mark, provide data, logout, store data, update data.



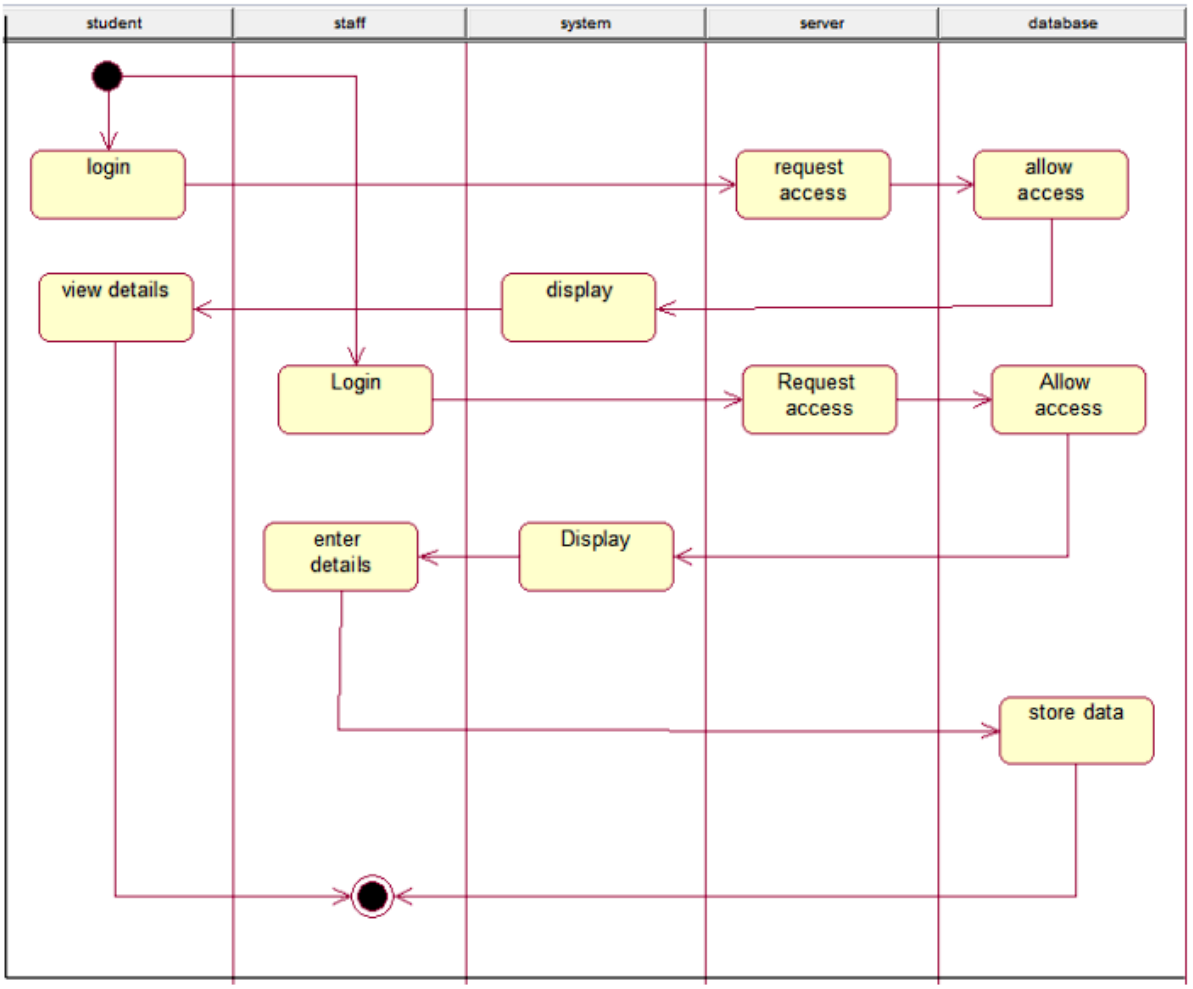
STATE CHART DIAGRAM

A State chart diagram is also called a state machine diagram. The state chart contains the states in the rectangular boxes and the states are indicated by the dot enclosed. The state chart diagram describes the behavior of the system. The state chart involves six stages such as login, enter mark, enter profile, view details, provide details, update details, store details and logout.



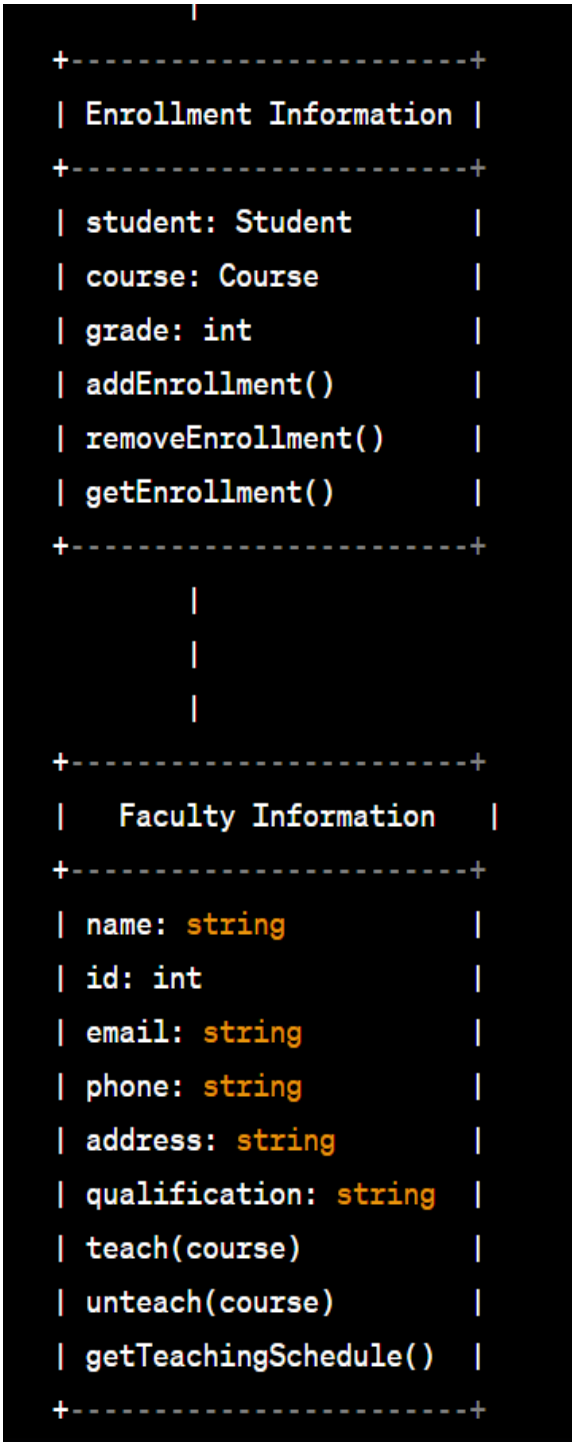
ACTIVITY DIAGRAM

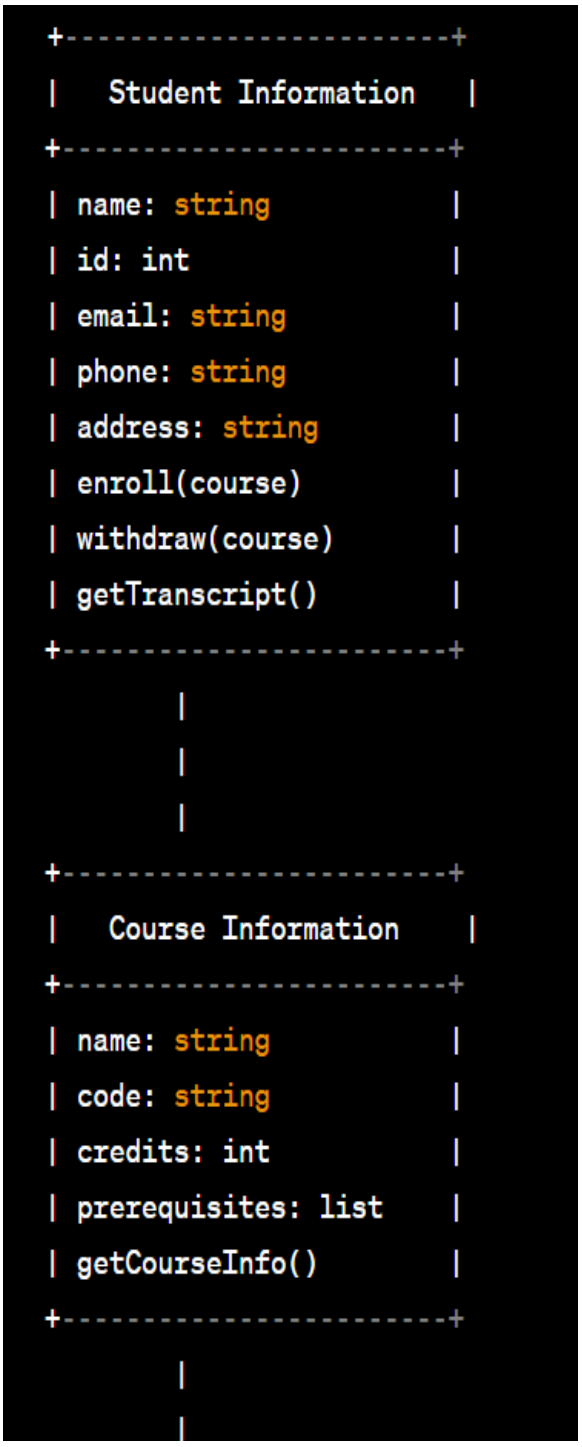
Activity diagrams are graphical representations of stepwise activities and actions with support for choice, interaction and concurrency. Here in the activity diagram the student login to the system and view the details of the student. The staff login to the system for entering the student details and update the details in the database. The final interaction is the DBA stores the details of the student.



PACKAGE DIAGRAM

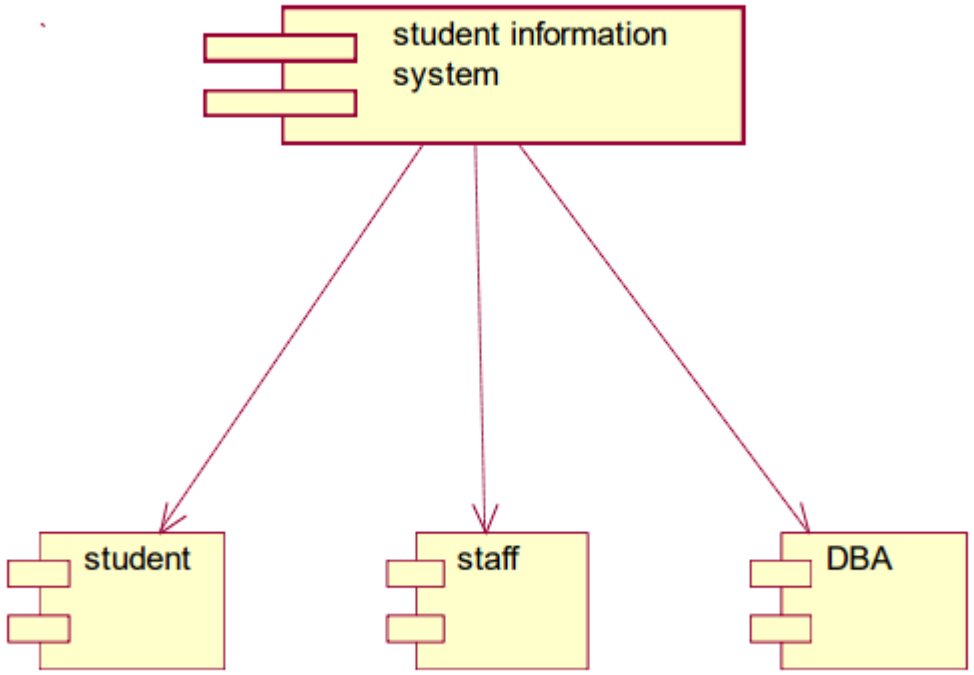
Package diagram, a kind of structural diagram, shows the arrangement and organization of model elements in a middle to large scale project. Package diagram can show both structure and dependencies between subsystems or modules, showing different views of a system, for example, as multi-layered (aka multi-tiered) application - multi-layered application model.





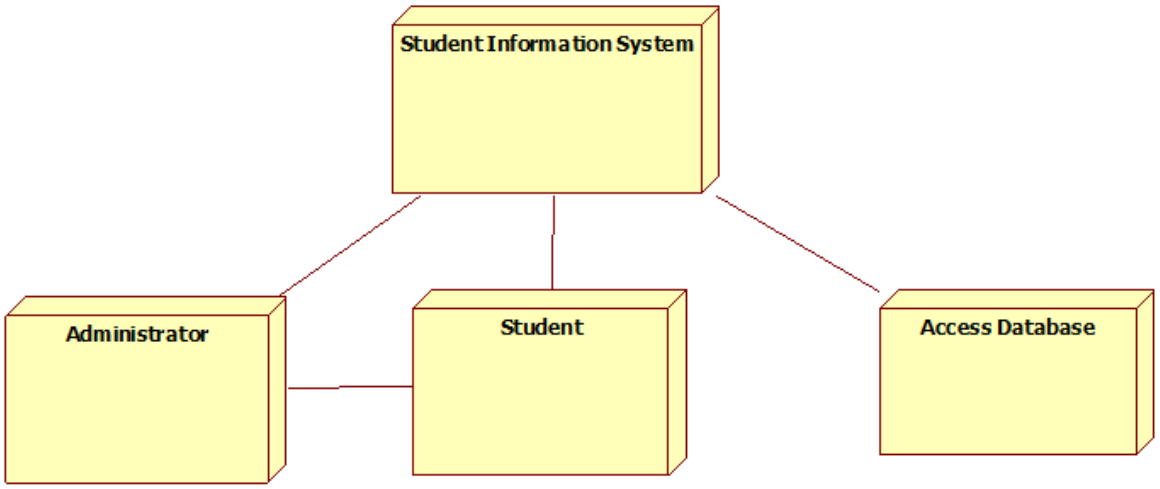
COMPONENT DIAGRAM

Component diagram carries the major living actors of the system. The component diagram's main purpose is to show the structural relationship between components of the system.The main component of the system is the student information system and the other components of the system are student, staff and DBA.



DEPLOYMENT DIAGRAM

Deployment diagram shows the configuration of runtime processing elements and the software components processes and objects that live in them. Component diagrams are used in conjunction with deployment diagrams to show how physical modules code are distributed on various hardware platforms. The processor node in the system is student information system and the execution environment nodes or device nodes are student, staff and DBA.



CODE

```
#include<iostream>
#include<string>
#include<conio.h>
#include<stdlib.h>
using namespace std;

int main();
void show_data(int searchkey);     //function used to show data to the end-user.
void get_data(int i);          //function used to obtain data from the end-user.
void search_student(int searchkey);
void add_student();      //This function is used to add a record of new students.
void edit_student(int idnumber);   //function is used to edit existing records.
void fullscreen();
int ts;

struct student    //Structure student is made to store student attributes.
{
  int rollno;
  string name;
  string fname;
  string cell;
  string dob;
  string address;
};

student rec[50];  //This is a basic array of defined structure to store data.
int main()
{
  system("CLS");
  system("color B1");
  int choice;   //int variable used to determine which operation the user want to do.
  int idnumber;   //int variable used to record ID number with user want to edit.
  int searchkey;  //int variable to store student roll_no by which user can search.
```

```
 cout<<"Enter The Total Number of Student(s)- Max 50: ";
 cin>>ts;

while(ts--)
{
 cout<<"\n\t\tWhat do you want to do?"<<endl;
 cout<<"\t\t----------------------"<<endl;
 cout<<"\t\t1-Add student"<<endl;
 cout<<"\t\t2-Edit student"<<endl;
 cout<<"\t\t3-Search student"<<endl;
 cout<<"\t\t4-Quit Program"<<endl;
 cout<<"\t\t----------------------"<<endl;
 cout<<"Enter your choice: ";

 cin>>choice;
 switch(choice)
{
  case 1:       //If user presses 1 then the student adding module would be displayed.
   add_student();
   break;
  case 2:       //If there are no records in the array then it will ask the user to input
records first.
   if(rec[0].rollno==0)
  {
    cout<<"Please Add students first."<<endl;
    system("pause");
   main();
  }
   else       //If records are present in an array then it will show a table.
  {
    cout<<endl;
    cout<<"--------------------------------------------------------------------------------"<<endl;
    cout<<"-------------------------Student record Table--------------------------------"<<endl;
    cout<<"--------------------------------------------------------------------------------"<<endl;
    cout<<"ID   "<<"Roll   "<<"Name      "<<"Father\tCell no.      "<<"DOB
"<<"Address\n\n";
```

```
 cout<<"-------------------------------------------------------------------------------"<<endl;

 for(int i=0;i<=ts;i++)
{
  show_data(i);     //function is called with index value to show data.
 }

 cout<<"-------------------------------------------------------------------------------"<<endl;
 cout<<"Which ID number your want to edit: ";

 cin>>idnumber;           //Asking the user at which ID he wants to make a change.

if(idnumber>ts || idnumber<0)  //Validating the ID number.
{
     cout<<"\nInvalid ID Number."<<endl;
  }
  else
{
     edit_student(idnumber);    //Passing ID number to Edit Function.
  }
}
break;

case 3:
 if(rec[0].rollno==0)     //If no record exists then ask the user to enter records first.
{
 cout<<"Please Add students first."<<endl;
 system("pause");
 main();         //Return to main so the user can input a new record.
}
else
{
 cout<<"Enter roll_no of student you want to search: ";
 cin>>searchkey;     //roll_no as the search key can be entered by the user.
 search_student(searchkey);}
 break;
case 4:
```

```
    return 0;        //Terminating the records.
    break;
   default:         //Default value for invalid Input.
    cout<<"Invalid number."<<endl;
    system("pause");
  main();
  }
 }
  return 0;
}


void get_data(int i)        //Function for receiving data from users and populating the
variables with values.
{
  cout<<"Enter student roll number in  format(1XXX): ";
  cin>>rec[i].rollno;
  cout<<"Enter student name: ";
  cin>>rec[i].name;
  cout<<"Enter student's Father name: ";
  cin>>rec[i].fname;
  cout<<"Enter student's cell phone number: ";
  cin>>rec[i].cell;
  cout<<"Enter student's Date of Birth(dd/mm/yyyy): ";
  cin>>rec[i].dob;
  cout<<"Enter student's Address: ";
  cin>>rec[i].address;
}

void show_data(int searchkey)      //Function for showing data on the screen.
{
  int i=searchkey;
  cout<<i<<"    ";
  cout<<rec[i].rollno<<"   ";
  cout<<rec[i].name<<"     ";
  cout<<rec[i].fname<<"\t";
  cout<<rec[i].cell<<"   ";
```

```
 cout<<rec[i].dob<<"   ";
 cout<<rec[i].address<<"\n\n";
}

void search_student(int searchkey)
{
 for(int i=0;i<=ts;i++)       //Loop through complete array.
 {
  if(rec[i].rollno==searchkey)    //If roll number matches to search term.
 {
   cout<<"ID   "<<"Roll   "<<"Name      "<<"Father\tCell no.      "<<"DOB
"<<"Address\n\n";
   show_data(i);       //A function is used inside another function.
  }
 }
}

void add_student()         //This function is used to add a record of new students.
{
 for(int i=0;i<=ts;i++)
 {
  get_data(i);          //Loop was processed 5 times to get input for 5 students.
 }
 system("CLS");
 cout<<endl;
 cout<<"-------------------------------------------------------------------------------"<<endl;
 cout<<"-------------------------Student record Table-------------------------------"<<endl;
 cout<<"-------------------------------------------------------------------------------"<<endl;
 cout<<"ID   "<<"Roll   "<<"Name      "<<"Father\tCell no.      "<<"DOB
"<<"Address\n\n";
 cout<<"-------------------------------------------------------------------------------"<<endl;

 for(int i=0;i<=ts;i++)
 {
  show_data(i);        //Loop was processed for 5 times to show obtained records.
 }
 cout<<"-------------------------------------------------------------------------------"<<endl;
```

```
  cout<<"--------------------------------FINISH-------------------------------------"<<endl;
  cout<<"---------------------------------------------------------------------------"<<endl;
  system("pause");

  main();          //Return to main function to again show the menu.
}

void edit_student(int idnumber)     //function is used to edit existing records.
{
  for(int i=0;i<=ts;i++)       //Loop is started.
 {
   if(idnumber==i)       //Through loop every value is compared with search term.
  {
    cout<<"\nExisted information about this record.\n\n";
    cout<<"---------------------------------------------------------------------------"<<endl;
    cout<<"ID   "<<"Roll   "<<"Name      "<<"Father\tCell no.      "<<"DOB
"<<"Address\n\n";
    cout<<"---------------------------------------------------------------------------"<<endl;
    show_data(i);       //Load information for existing records.
    cout<<"\n\nEnter new data for above shown record.\n\n";
    get_data(i);         //Inputting data for that specific record.
    cout<<"\n\nRecord updated successfully."<<endl;
    system("pause");
    main();           //Return to main function.
   }
  }
}
```

OUTPUT

```
                What do you want to do?
                ----------------------
                1-Add student
                2-Edit student
                3-Search student
                4-Quit Program
                ----------------------
Enter your choice: 1
Input student roll number in  format(1XXX): 1001
Input student name: Kamal
input student's Father name: Wasib
Input student's cell phone number: 03078568605
Input student's Date of Birth(dd/mm/yyyy): 04/09/1996
Input student's Address: Noonawali
```

```
-------------------------------------------------------------------------------------
-------------------------Student record Table----------------------------------------
-------------------------------------------------------------------------------------
ID   Roll   Name      Father    Cell no.      DOB          Address
-------------------------------------------------------------------------------------
0    1001   Kamal     Wasib     03078568605   04/09/1996   Noonawali
1    1002   Ashar     Naseer    03025632147   02/02/1997   Lalamusa
2    1003   Usama     Bashir    03029685741   02/05/1994   Jalalpur
3    1004   Ammar     Ahmed     03059685321   02/04/1995   Lalamusa
4    1005   Adnan     Ahmed     03015632147   01/03/1995   Gujrat
-------------------------------------------------------------------------------------
-----------------------------------FINISH--------------------------------------------
-------------------------------------------------------------------------------------
```

```
Which ID number your want to edit: 1

Existed information about this record.

-------------------------------------------------------------------------------------
ID   Roll   Name      Father    Cell no.      DOB          Address
-------------------------------------------------------------------------------------
1    1002   Ashar     Naseer    03025632147   02/02/1997   Lalamusa


Enter new data for above shown record.

Input student roll number in  format(1XXX): 1002
Input student name: Nasir
input student's Father name: Ahmed
Input student's cell phone number: 03025696854
Input student's Date of Birth(dd/mm/yyyy): 05/05/1998
Input student's Address:
```

CONCLUSION

Thus the various UML diagrams for the student information system were drawn and code was generated successfully.

REFERENCES

1. https://www.programmingwithbasics.com
2. https://www.geeksforgeeks.org
3. https://hackr.io