PROJECT REPORT

Submitted by

Tanisha Dhoot-RA2211056010009

Tahsheen Fatima-RA2211056010031

Riti Matangi-RA2211056010035

Dhivya-RA2211056010021

Under the Guidance of

Dr.K.DHANASEKARAN

Assistant Professor, Department of Data Science and business systems(DSBS)

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 "College **Area Calculation system**" is the bonafide work done by TAHSHEEN FATIMA RA2211056010031 , TANISHA DHOOT RA2211056010009 , RITI MATANGI RA2211056010035 , DHIVYA RA221106010021 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.K.DHANASEKARAN** **OODP – Course Faculty** Assistant Professor Department of Data Sciencs and Business systems (DSBS) School of Computing SRMIST	**SIGNATURE** **Dr.M.LAKSHMI** Professor & Head Department of Data Sciencs and Business systems (DSBS) School of Computing SRMIST

TABLEOFCONTENTS

Problem statement:-

- The basic role of making the online area calculation the system is to make a programmed online based system which will provide a simple and exchange approach to calculate area.
- This system can calculate area of each department and the total area.

Aim:

- In this project we are going to code a simple program to calculate college are a department wise using C++.

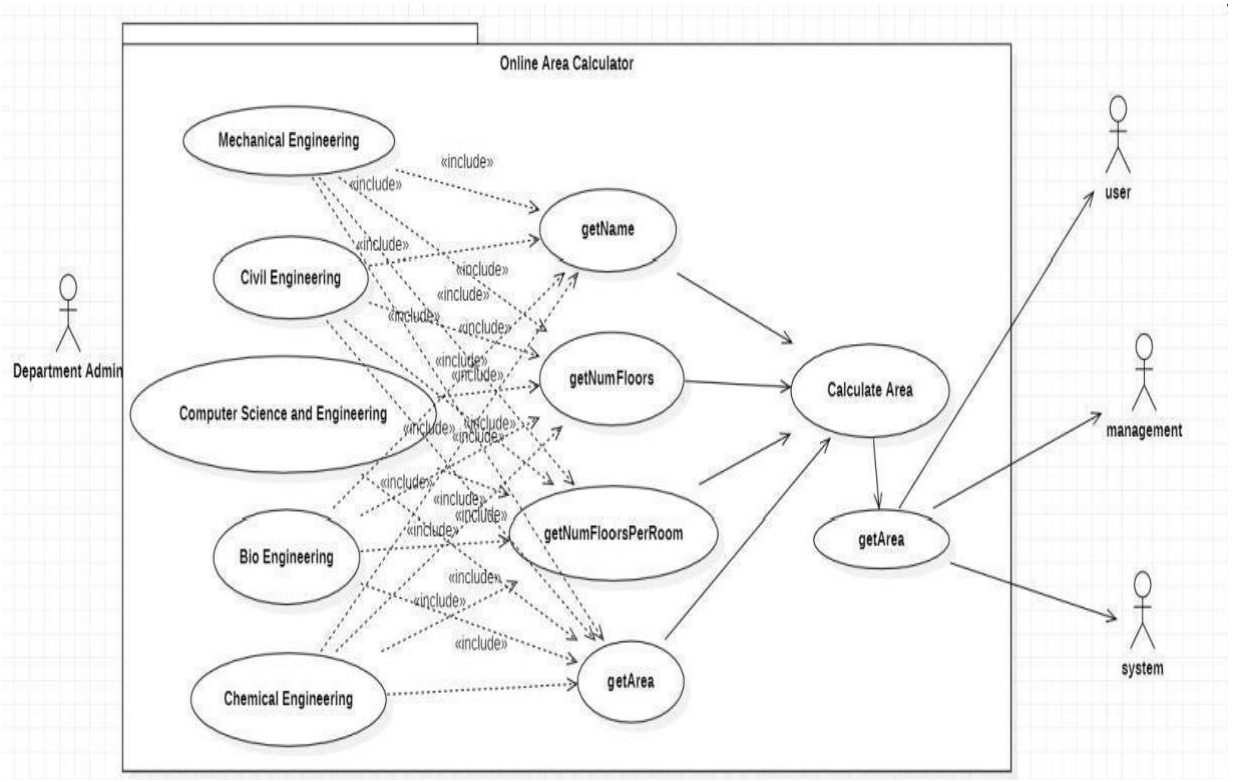
Methodology:-

The basic role of making the online area calculation the system is to make a programmed online based system which will provide a simple and exchange approach to calculate area.

This system can calculate area of each department and the total area. In this project we are going to code a simple program to calculate college are a department wise using C++.

We used the concepts of OODP that is Classes , Objects , Override , friend function ,Virtual Function , Exceptional Handling.

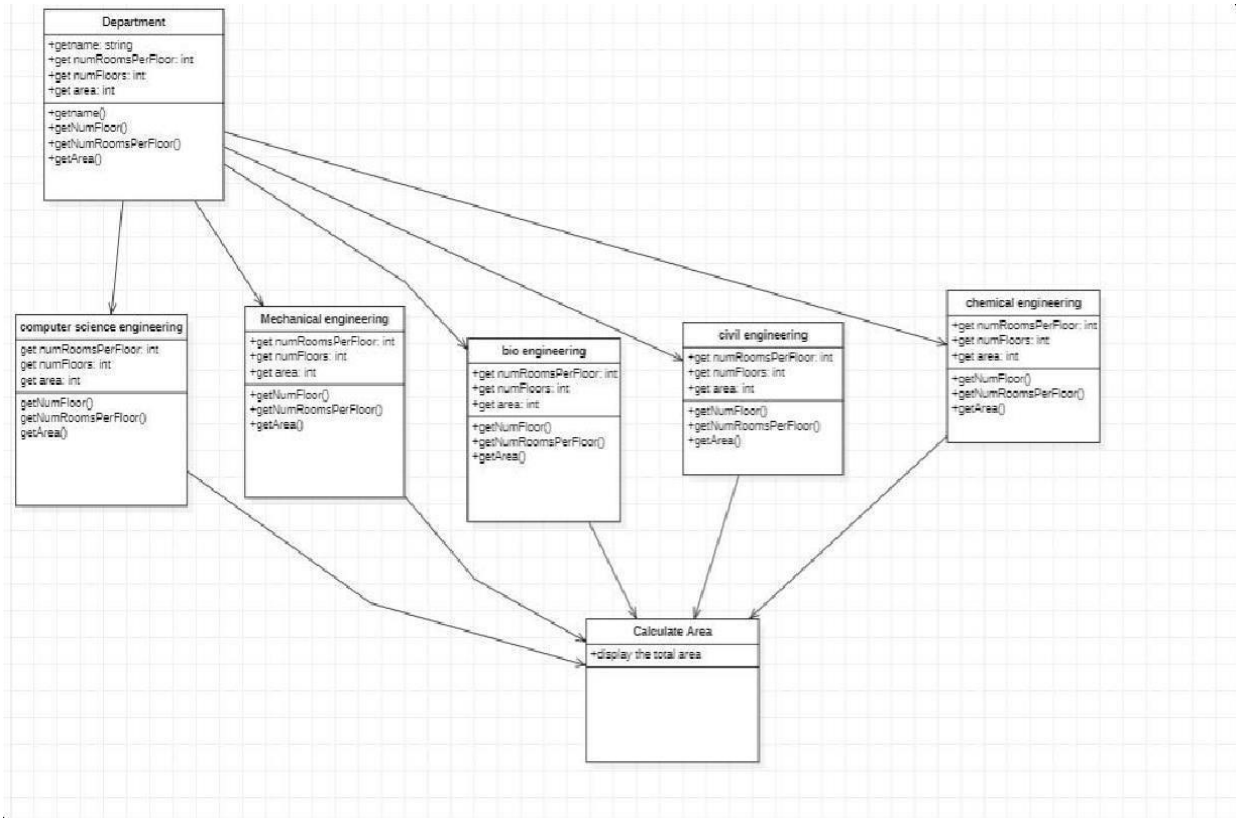
USE CASE DIAGRAM



In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. An effective use case diagram can help your team discuss and represent:

- Scenarios in which your system or application interacts with people, organizations, or external systems
- Goals that your system or application helps those entities (known as actors) achieve
- The scope of your system

ClassDiagram:

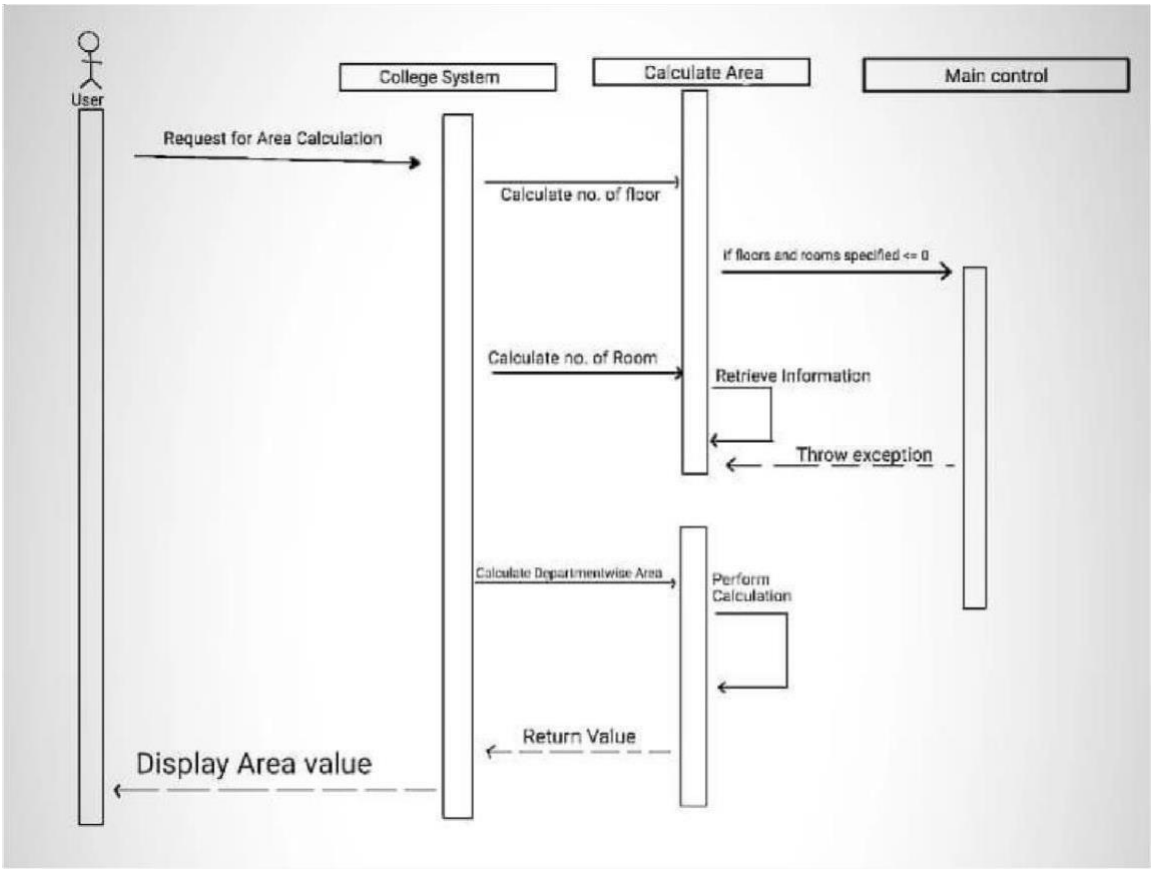


Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of as system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

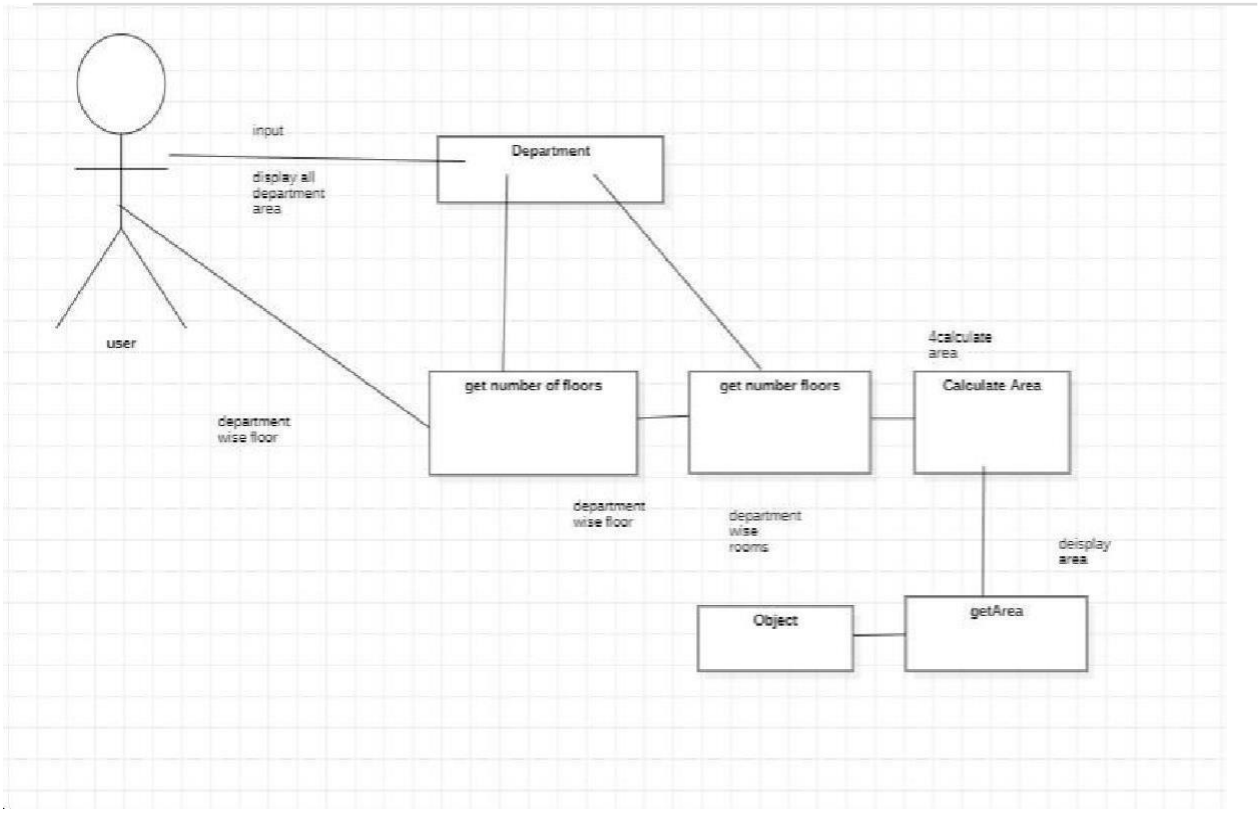
Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.

Sequence Diagram



A sequence diagram is a type of interaction diagram because it describes how—and in what order—a group of objects works together. These diagrams are used by software developers and business professionals to understand requirements for a new system or to document an existing process. Sequence diagrams are sometimes known as event diagrams or event scenarios.

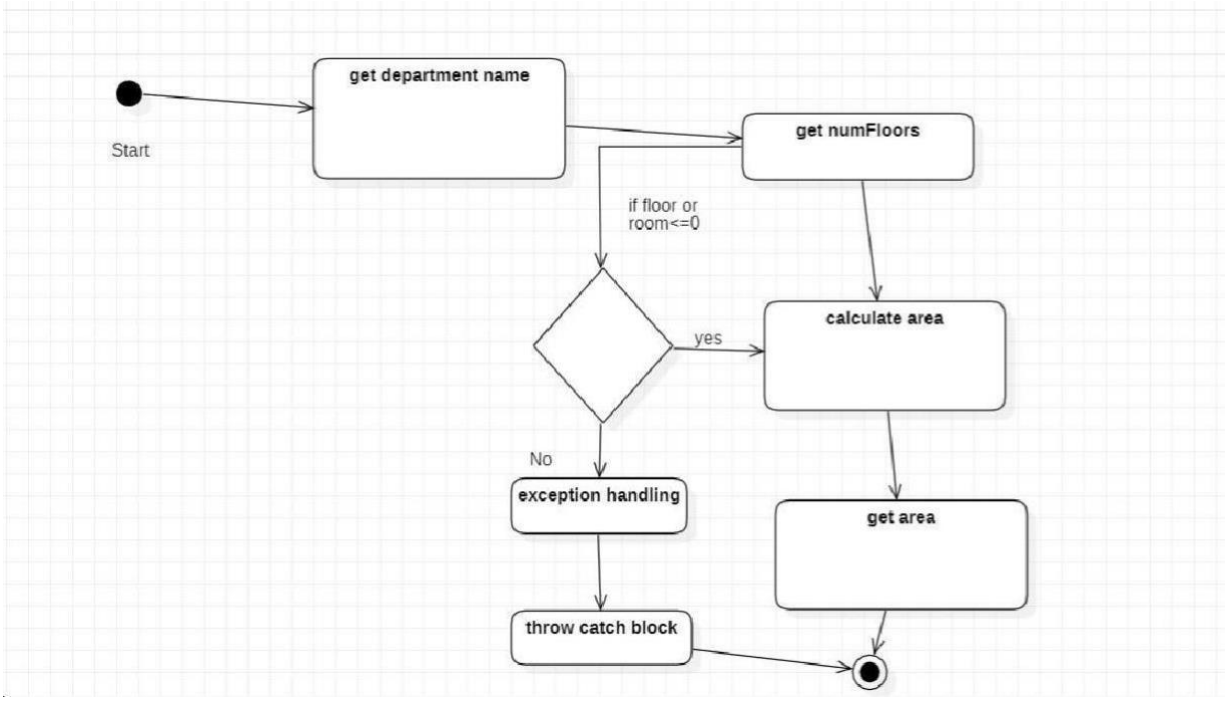
COLLABORATION DIAGRAM:



A collaboration diagram, also known as a communication diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML).Developers can use the diagrams to portray the dynamic behavior of a particular use case and define the role of each object.

To create a collaboration diagram, first identify the structural elements required to carry out the functionality of an interaction. Then build a model using the relationships between those elements. Several vendors offer software for creating and editing collaboration diagrams.

State Chart Diagram

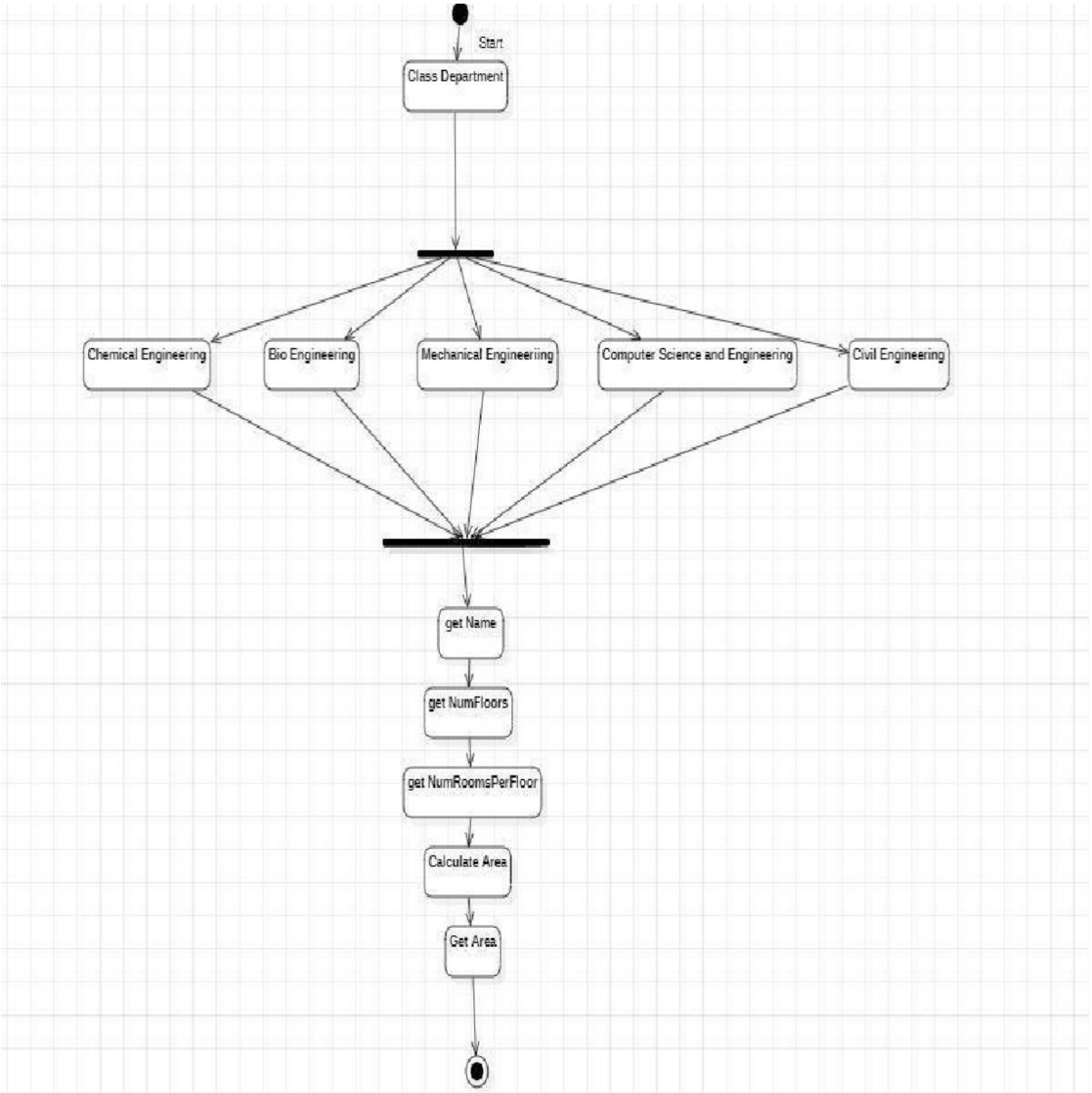


The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

A State chart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object and these states are controlled by external or internal events.

Activity diagram explained in the next chapter, is a special kind of a State chart diagram. As State chart diagram defines the states, it is used to model the lifetime of an object.

ACTIVITY DIAGRAM

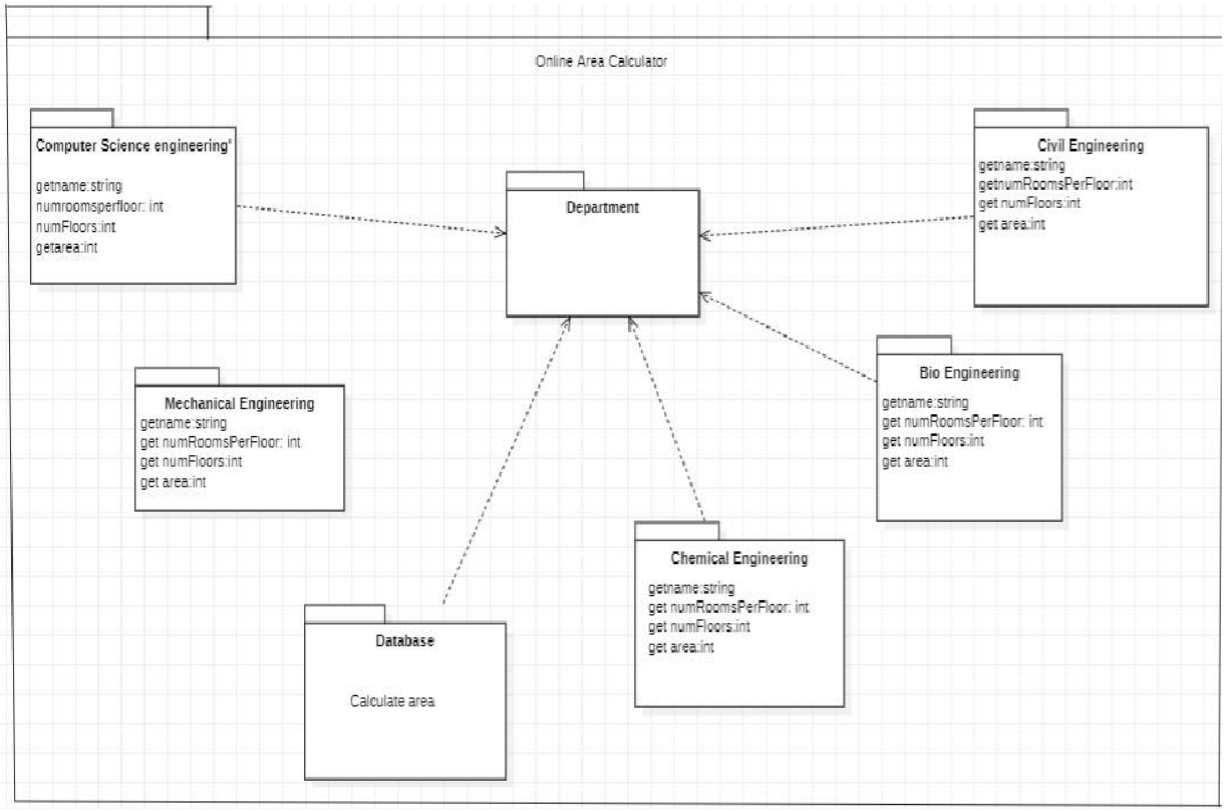


Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.

Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

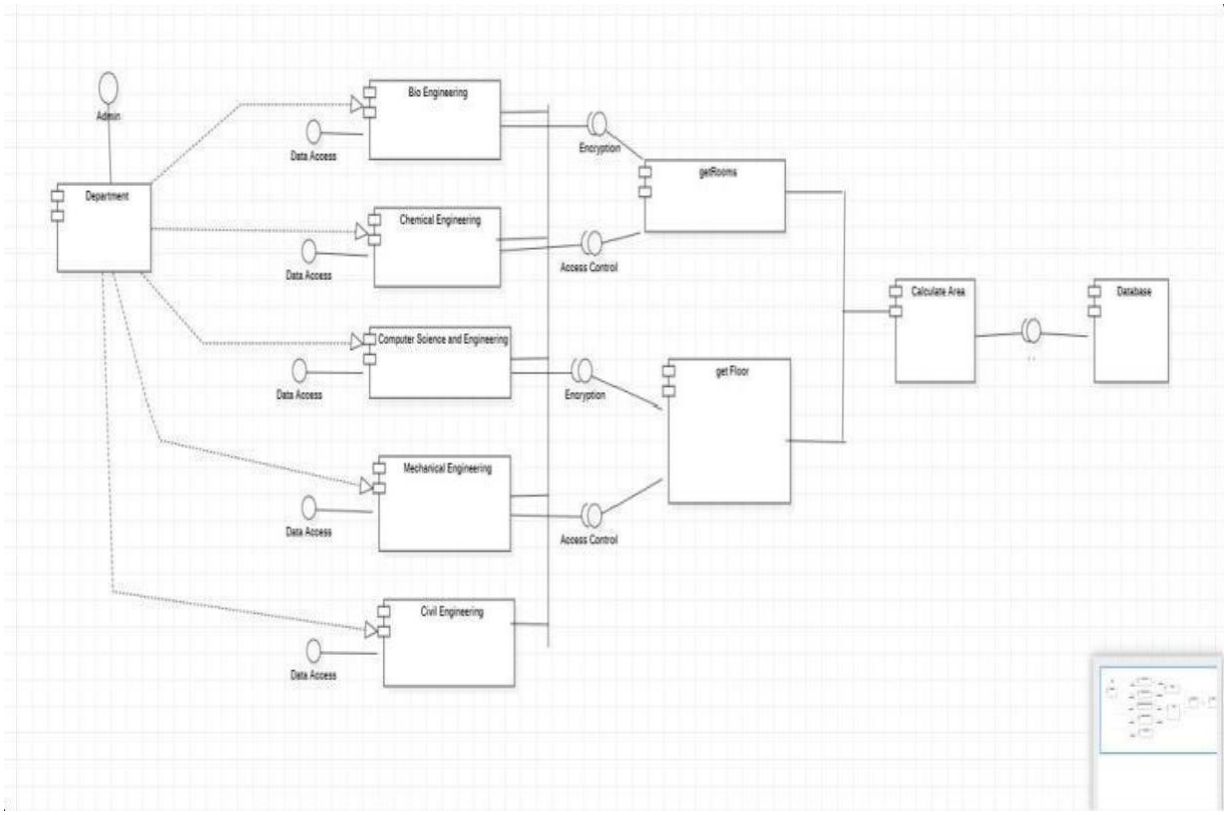
Package



Package diagrams are structural diagrams used to show the organization and arrangement of various model elements in the form of packages.

A package is a grouping of related UML elements, such as diagrams, documents, classes, or even other packages. Each element is nested within the package, which is depicted as a file folder within the diagram, then arranged hierarchically within the diagram. Package diagrams are most commonly used to provide a visual organization of the layered architecture within any UML classifier, such as a software system.

Component Diagram



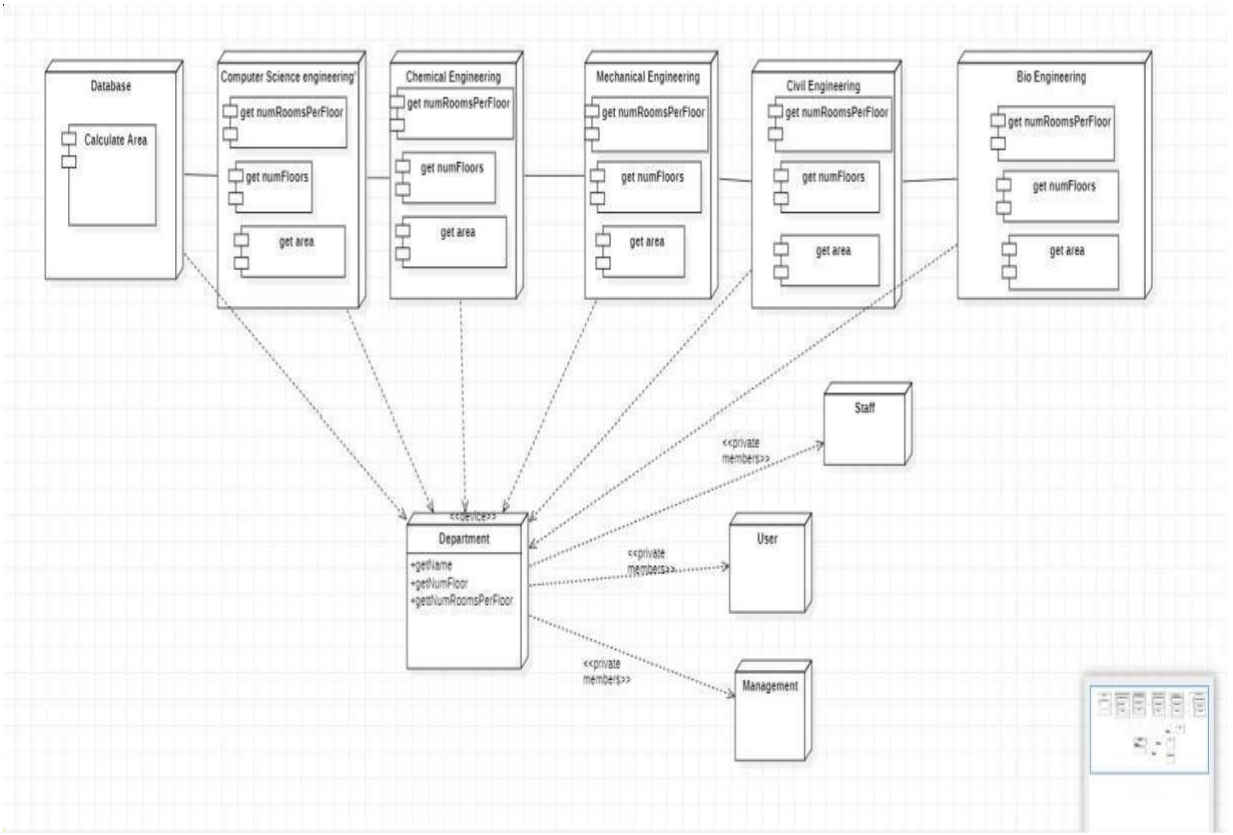
The purpose of a component diagram is to show the relationship between different components in a system. For the purpose of UML2.0, the term "component" refers to a module of classes that represent independent systems or subsystems with the ability to interface with the rest of the system.

There exists a whole development approach that revolves around components: component-based development (CBD).In this approach, component diagrams allow the planner to identify the different components so the whole system does what it's supposed to do.

More commonly, in a programming approach, the component diagram allows a senior developer to group classes together based on common purpose so that the developer and others can look at a software development project at a high level.

\

DEPLOYMENTDIAGRAM



A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.

Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.

INPUT

```
#include<iostream>
#include<string> #include<vector>
usingnamespacestd;


//defineDepartmentclass
template<typenameT>
classDepartment{ private:
    stringname; intnumRoomsPerFloor;
    intnumFloors;
    Tarea;
  public:
    Department(stringdeptName,intfloors,introoms){
//ParamaterizedConstructor if(floors<=0||rooms<=0){
throwinvalid_argument("Numberoffloorsandroomsper
floormustbepositive.");
      } name=deptName;
      numFloors=floors;
      numRsPerFloor=rooms;
      area=calculateArea()
      ;
    }
    virtualstringgetName(){
      returnname;
    }
    inlineintgetNumFloors(){
      returnnumFloors;
    }
    intgetNumRoomsPerFloor(){
    returnnumRoomsPerFloor;
    }
    TgetArea(){
      returnarea;
    }
     //overloadthe"+"operatortocalculatethetotalareaoftwo
departments friendDepartmentoperator+(constDepartment&dept1,const
Department&dept2){
    Departmentresult(dept1.name+"&"+dept2.name,dept1.numFloors
+dept2.numFloors,dept1.numRoomsPerFloor);
```

```
        result.area=dept1.area+dept2.area; returnresult;
      }


  private:
    TcalculateArea(){


      TfloorArea=numRoomsPerFloor*50;//
      roomis50sq.m.TtotalArea=floorArea*numFloors;
      returntotalArea;
    }
};



//Inheritance class ComputerScience : public
Department<double> {public:
    ComputerScience(int floors, int rooms) :
Department("ComputerScience",floors,rooms){}
     stringgetName()override{re
      turn"CSDepartment";
    }
};



class Mechanical : public Department<int>
  {public:
    Mechanical(int floors, int rooms) :
Department("MechanicalEngineering",floors,rooms){}
};
 class Chemical : public Department<double>
{public:
    Chemical(int floors, int rooms) :
Department("ChemicalEngineering",floors,rooms){}
};
class Civil : public Department<double>
{public:
    Civil(intfloors,introoms):Department("CivilEngineering",floors,rooms
){}
};
classBio:publicDepartment<double>{
  public:
    Bio(intfloors,introoms):Department("BioEngineering",floors, rooms){}
};
```

```
intmain(){
 try{
  cout<<"************OnlineAreaCalculator******************\n"<<endl;

  //createdepartmentsandcalculatetheirareasComputerSciencecsDep
  t(7,20);
  MechanicalmDept(3,15);
  ChemicalchemDept(5,18);C
  ivilcDept(9,15);
  BioBDept(6,14);

  //printtheinformationaboutthedepartments
  cout<<"DepartmentName:"<<csDept.getName()<<endl;
  cout<<"NumberofFloors:"<<csDept.getNumFloors()<<endl;
  cout<<"NumberofRoomsperFloor:"<<csDept.getNumRoomsPerFloor()
<<endl; cout<<"Each room is of 50 sq. m. So the
  Area will
be:"<<csDept.getNumFloors()<<"X50X"<<csDept.getNumRoomsPerFloor()<<endl;
  cout<<"Area:"<<csDept.getArea()<<"sq.m."<<endl;

  cout<<"                                                    \n";


  cout<<"DepartmentName:"<<mDept.getName()<<endl;
  cout<<"NumberofFloors:"<<mDept.getNumFloors()<<endl; cout
  <<"Number of Rooms per Floor: "<<mDept.getNumRoomsPerFloor()
<<endl; cout<<"Each room is of 50 sq. m. So the Area
    will
be:"<<mDept.getNumFloors()<<"X50X"<<mDept.getNumRoomsPerFloor()<<endl;
  cout<<"Area:"<<mDept.getArea()<<" sq.m."<<endl;

   cout<<"                                                    \n";


  cout<<"DepartmentName:"<<chemDept.getName()<<endl;
  cout<<"NumberofFloors:"<<chemDept.getNumFloors()<<endl;
  cout<<"NumberofRoomsperFloor:"<<chemDept.getNumRoomsPerFloor()
<<endl;
    cout<<"Eachroomisof50sq.m.SotheAreawillbe:"<<chemDept.getNumFloors()<<"X50
    X "<<chemDept.getNumRoomsPerFloor()<<endl;
    cout<<"Area:"<<chemDept.getArea()<<"sq.m."<<endl; cout<<"
    ----------------------------------------------------------------------------------------------
                                                               \n";

  cout<<"DepartmentName:"<<cDept.getName()<<endl;
  cout<<"NumberofFloors:"<<cDept.getNumFloors() <<endl;
```

```
  cout <<"Number of Rooms per Floor: "<<cDept.getNumRoomsPerFloor()
<<endl; cout<<"Each room is of 50 sq. m. So the Area
    will
be:"<<cDept.getNumFloors()<<"X50X"<<cDept.getNumRoomsPerFloor()<<endl;
  cout<<"Area:"<<cDept.getArea()<<"sq.m."<<endl;
```

```
   cout<<"----------------------------------------------------------------------------------------------------\n";

cout<<"DepartmentName:"<<BDept.getName()<<endl;
  cout<<"NumberofFloors:"<<BDept.getNumFloors()<<endl; cout
  <<"Number of Rooms per Floor: "<<BDept.getNumRoomsPerFloor()
<<endl; cout<<"Each room is of 50 sq. m. So the Area
    will
be:"<<BDept.getNumFloors()<<"X50X"<<BDept.getNumRoomsPerFloor()<<endl;
  cout<<"Area:"<<BDept.getArea()<<"sq.m."<<endl;

   cout<<"                                                                                \n";




cout<<"                                                                                \n";
  Department<double>dept1("ComputerScience",7,20);
  Department<double>dept2("ChemicalEngineering",5,18);
  Department<double>dept3("CivilEngineering",9,15);
  Department<double>dept4("MechanicalEngineering",3,15);
  Department<double>dept5("Bio Engineering", 6,
  14);Department<double>dept6=dept1+dept2+dept3+dept4+dept5;
cout<<"TotalAreaofthedepartment:"<<dept6.getArea()<<"sq.m."<<endl;
 }
 catch(invalid_argument&e){ cerr <<"Invalid
    argument: "<<e.what() << endl;return1;
 }
 return0;
 }
```

Output

```
**********Online Area Calculator**********

Department Name: Computer Science
Number of Floors: 7
Number of Rooms per Floor: 20
Each room is of 50 sq. m. So the Area will be: 7 X 50 X 20
Area: 7000 sq. m.
------------------------------------------------------------
Department Name: Mechanical Engineering
Number of Floors: 3
Number of Rooms per Floor: 15
Each room is of 50 sq. m. So the Area will be: 3 X 50 X 15
Area: 2250 sq. m.
------------------------------------------------------------
Department Name: Chemical Engineering
Number of Floors: 5
Number of Rooms per Floor: 18
Each room is of 50 sq. m. So the Area will be: 5 X 50 X 18
Area: 4500 sq. m.
------------------------------------------------------------
Department Name: Civil Engineering
Number of Floors: 9
Number of Rooms per Floor: 15
Each room is of 50 sq. m. So the Area will be: 9 X 50 X 15
Area: 6750 sq. m.
------------------------------------------------------------
Department Name: Bio Engineering
Number of Floors: 6
Number of Rooms per Floor: 14
Each room is of 50 sq. m. So the Area will be: 6 X 50 X 14
Area: 4200 sq. m.
------------------------------------------------------------
------------------------------------------------------------
Total Area of the department: 24700 sq. m.
```

```
In Exception Case:-
If we give negative value of floors then:--

    // create departments and calculate their areas
    ComputerScience csDept(-7, 20);
    Mechanical mDept(3, 15);
    Chemical chemDept(5, 18);
    Civil cDept(9,15);
    Bio BDept(6,14);
```

```
**********Online Area Calculator**********

Invalid argument: Number of floors and rooms per floor must be positive.
```

Conclusion and Results

Aim is successfully proved..