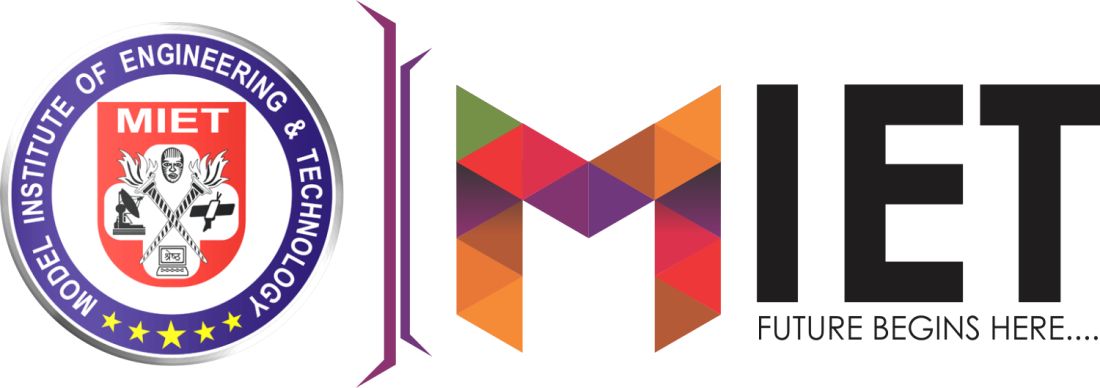
JAVA ASSIGNMENT 2



SUBMITTED TO ::Mrs Sheerin Zandoo

SUBMITTED BY:: Bhawna Sharma

CSE(A1)

1606/17

4 Semester

CONDITIONS FOR THE MINI PROJECT

Micro Project: Student Recommender System

Problem Statement: A system is needed which recommends the

Placement Officer the students who have 0 backlog count and have overall

percentage more than 60%. This system will help placement officer to

further groom the recommended students and align them for the various

companies’ placement requirements which comes to Campus. Following

artifacts should be prepared in Google Document submission:

1. Requirements Specification: A list of requirements specifications

which student will frame based on what they perceive from the high

level problem statement. List should not contain more than 5 Use

Cases 1 Marks

2. Design Specification: Class diagram &amp; Sequence Diagram for the

defined problem statement should be prepared in the Draw.io tool

and prepared images should be added in the main Google

Document. Diagrams should be easy to understand with proper notes

in it. Not more than 2 pages for this section. 2

Marks

3. Program Specification: This section should contain only the

important algorithms / functions logic. Not more than 3 pages for this

section. 2 Marks

4. Implementation: Create one GitHub Repository and commit your

complete code implementation in the repository. Copy paste the link

of the GitHub Code in the document 2 Marks

5. Testing: Unit Testing Specifications and execution logs should be

prepared. Execution logs must contain the problems students faced

during compilation and execution and how they resolved it. Not more

than 3 pages for this section. 2 Marks

6. Demo: Students should be able to create the video of the working

demo and should upload it in YouTube. Once uploaded the link of the

video should be added in the document. Just copy paste the link in

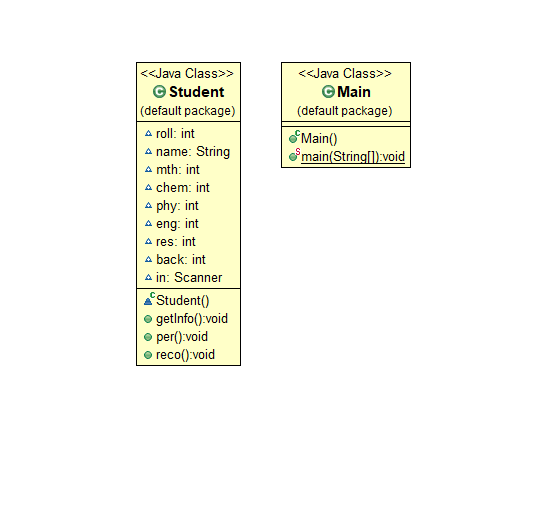
this section.

* CLASS DIAGRAM

In [software engineering](https://en.wikipedia.org/wiki/Software_engineering" \o "Software engineering), a **class diagram** in the [Unified Modeling Language](https://en.wikipedia.org/wiki/Unified_Modeling_Language" \o "Unified Modeling Language) (UML) is a type of static structure diagram that describes the structure of a system by showing the system's [classes](https://en.wikipedia.org/wiki/Class_(computer_science)" \o "Class (computer science)), their attributes, operations (or methods), and the relationships among objects.

The class diagram is the main building block of [object-oriented](https://en.wikipedia.org/wiki/Object-oriented_programming" \o "Object-oriented programming) modeling. It is used for general [conceptual modeling](https://en.wikipedia.org/wiki/Conceptual_model" \o "Conceptual model) of the structure of the application, and for detailed modeling translating the models into [programming code](https://en.wikipedia.org/wiki/Programming_code" \o "Programming code). Class diagrams can also be used for [data modeling](https://en.wikipedia.org/wiki/Data_modeling" \o "Data modeling).[[1]](https://en.wikipedia.org/wiki/Class_diagram" \l "cite_note-1) The classes in a class diagram represent both the main elements, interactions in the application, and the classes to be programmed.

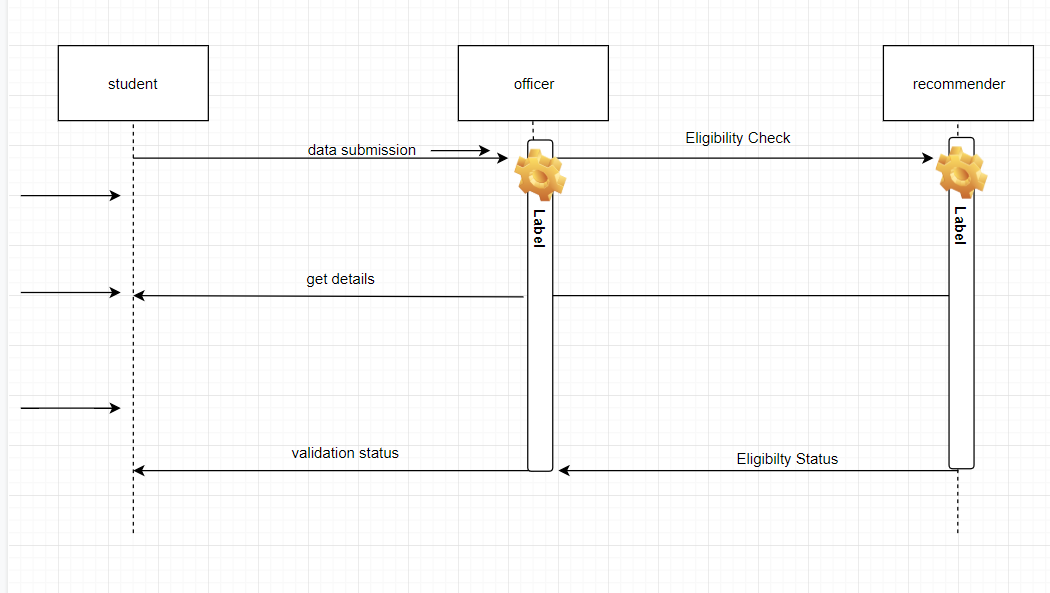
FOR THIS PROGRAM THE CLASS DIAGRAM IS :-



* SEQUENCE DIAGRAM

A **sequence diagram** simply depicts interaction between objects in a **sequential** order i.e. the order in which these interactions take place. We can also use the terms event **diagrams** or event scenarios to refer to a **sequence diagram**. **Sequence diagrams** describe how and in what order the objects in a system function.

FOR THIS PROGRAM THE SEQUENCE DIAGRAM IS:-



SOURCE CODE

import java.util.\*;

class Student

{

int roll;

String name;

int mth;

int chem;

int phy;

int eng;

int res;

int back;

Scanner in = new Scanner(System.in);

public void getInfo()

{

System.out.println("enter the name and roll no.:");

name = in.next();

roll = in.nextInt();

System.out.println("enter the marks of all years (out of 100)");

mth = in.nextInt();

chem = in.nextInt();

phy = in.nextInt();

eng = in.nextInt();

System.out.println("Enter the no. of the backlogs left {sachi sachi batana koi fark nhi padta apun bhi engineer hai}");

back = in.nextInt();

}

public void per()

{

int add= mth+chem+phy+eng;

res=add/4;

System.out.println("the result of the student is ="+res+"%");

}

public void reco()

{

if (mth>=40 && chem>=40 && phy>=40 && eng>=40 && back==0 && res>=60)

{

System.out.println (name+" is recommended to the company");

}

else

{

System.out.println (name+" is recommended not to the company");

}

}

}

public class Main

{

public static void main (String[]args)

{

Scanner in = new Scanner(System.in);

System.out.println("enter the number of Student in database");

int n = in.nextInt();

ArrayList<Student> al = new ArrayList<Student>();

for(int i=0; i<=n; i++)

{

Student s = new Student();

s.getInfo();

s.per();

s.reco();

al.add(s);

}

}

}

https://github.com/bhawnasharma112k/StudentRecommander