**Object Oriented Analysis and Design using Java (UE20CS352)**

**Lab Assignment - 9 & 10**

**Name : Tushar J Section : H**

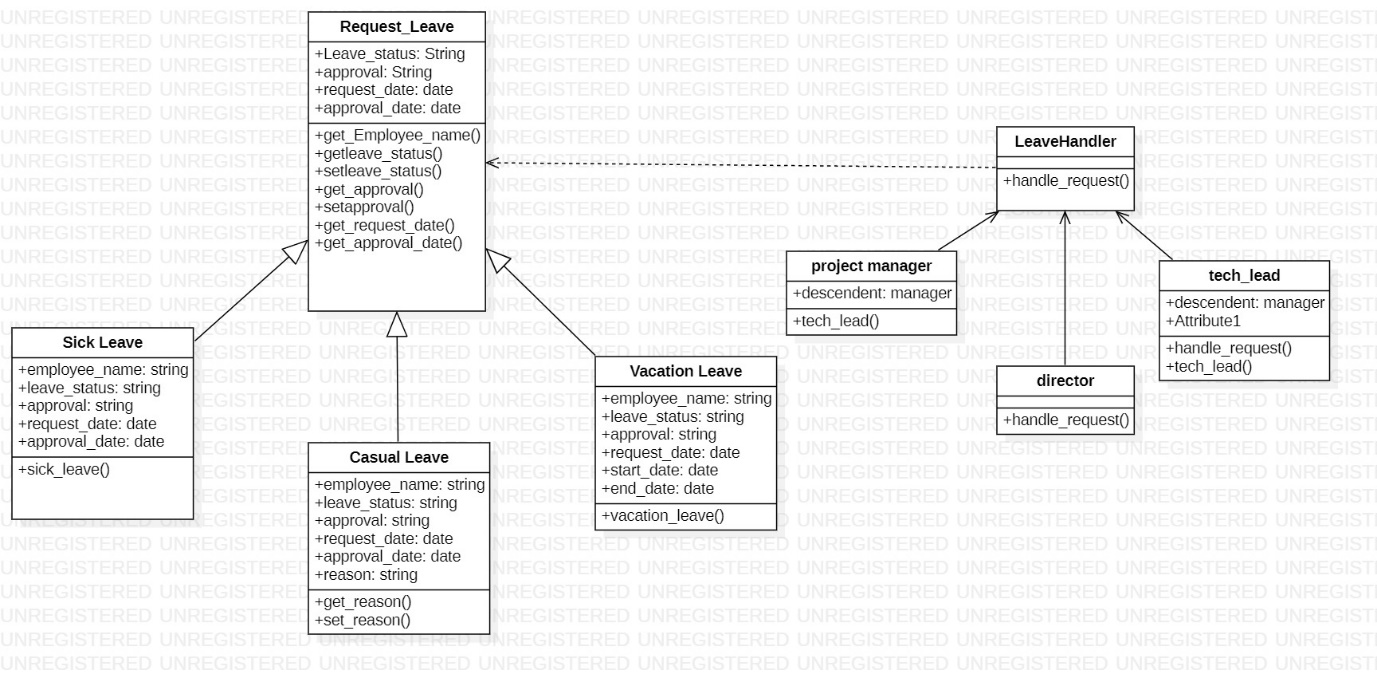
**Roll No : 10 SRN : PES1UG20CS472**

**Date : 21-04-2023**

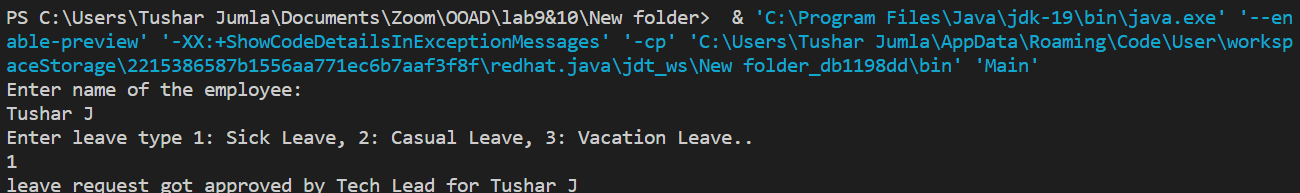
**Problem:-**

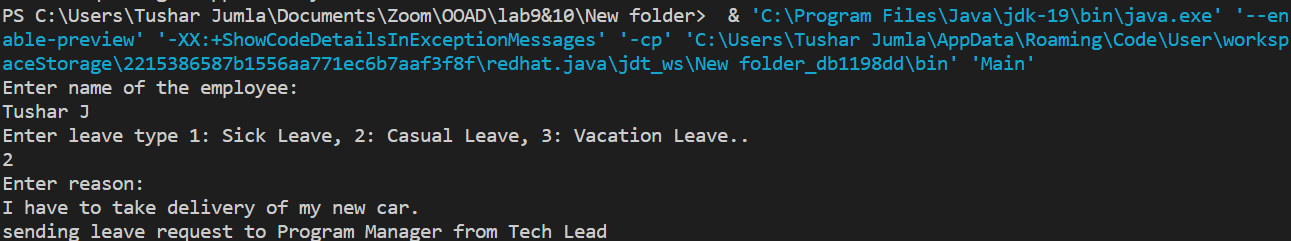
A Company’s Leave Management System has the following features. An Employee (client) can apply for Casual Leave (CL), Sick Leave (SL) and Vacation Leave (VL). The roles in the hierarchy who are responsible for approving or rejecting the leave using the process specified are Director, Project Manager and Tech Lead. The Leave request contains the following details: empName, leaveStatus, approvedBy, requestDate and approvalDate. A CL and SL are for only one day. A VL will have a startDate and endDate. A CL will also need a reason to be specified. The Leave created by the client is assigned a “New” status. If the leave is SL, then it will be processed by Tech Lead, if it is CL, it will be processed by the Project Manager, and if it is VL, will be processed by the Director. The Leave when created is sent to Tech Lead for processing, if it is not SL, the Tech Lead will just pass the request to the next higher level. Similarly, Project Manager will process a CL request or forward the VL request to the next higher level. Once the request is processed, a message should be displayed on the console showing request details and approval details. Note: Design the application in such a way that extensibility is easy. It should be easy to add new types of Employee and new types of Leave.

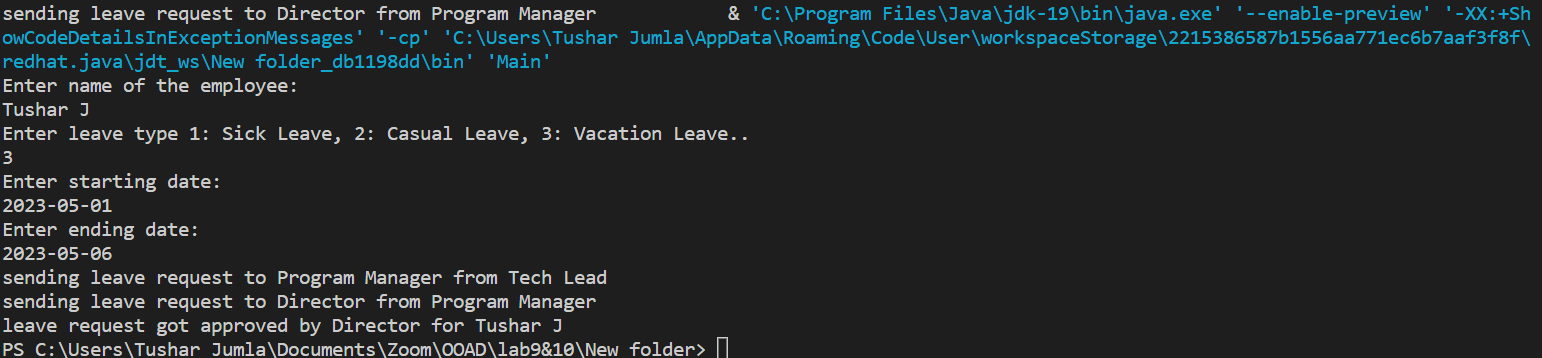
**UML Class Model :-**



**Output:-**







Request\_Leave.java :-

import java.time.LocalDate;

public abstract class Request\_Leave

{

    private String name\_of\_employee;

    private String leave\_status;

    private String approval;

    private LocalDate date\_of\_request;

    private LocalDate date\_of\_approval;

    public Request\_Leave(String name\_of\_employee, LocalDate date\_of\_request)

    {

        this.name\_of\_employee = name\_of\_employee;

        this.leave\_status = "New";

        this.approval = null;

        this.date\_of\_request = date\_of\_request;

    }

    public String getname\_of\_employee()

    {

        return name\_of\_employee;

    }

    public String getleave\_status()

    {

        return leave\_status;

    }

    public void setleave\_status(String leave\_status)

    {

        this.leave\_status = leave\_status;

    }

    public String getapproval()

    {

        return approval;

    }

    public void setapproval(String approval)

    {

        this.approval = approval;

    }

    public LocalDate getdate\_of\_request()

    {

        return date\_of\_request;

    }

    public LocalDate getdate\_of\_approval()

    {

        return date\_of\_approval;

    }

    public void setdate\_of\_approval(LocalDate date\_of\_approval)

    {

        this.date\_of\_approval = date\_of\_approval;

    }

}

class sick\_leave extends Request\_Leave

{

    public sick\_leave(String name\_of\_employee, LocalDate date\_of\_request)

    {

        super(name\_of\_employee, date\_of\_request);

    }

}

class casual\_leave extends Request\_Leave

{

    private String reason\_description;

    public casual\_leave(String name\_of\_employee, LocalDate date\_of\_request, String reason\_description)

    {

        super(name\_of\_employee, date\_of\_request);

        this.reason\_description = reason\_description;

    }

    public String getreason\_description()

    {

        return reason\_description;

    }

    public void setreason\_description(String reason\_description)

    {

        this.reason\_description = reason\_description;

    }

}

class vacation\_leave extends Request\_Leave

{

    private LocalDate starting\_date;

    private LocalDate ending\_date;

    public vacation\_leave(String name\_of\_employee, LocalDate date\_of\_request, LocalDate starting\_date, LocalDate ending\_date)

    {

        super(name\_of\_employee, date\_of\_request);

        this.starting\_date = starting\_date;

        this.ending\_date = ending\_date;

    }

    public LocalDate getstarting\_date()

    {

        return starting\_date;

    }

    public LocalDate getending\_date()

    {

        return ending\_date;

    }

}

interface leave\_manager

{

    void handleRequest(Request\_Leave leave);

}

class tech\_lead implements leave\_manager

{

    private leave\_manager descendant;

    public tech\_lead(leave\_manager leavemanager)

    {

        this.descendant = leavemanager;

    }

    @Override

    public void handleRequest(Request\_Leave leave)

    {

        if(leave instanceof sick\_leave && leave.getleave\_status().equals("New"))

        {

            leave.setleave\_status("approved");

            leave.setapproval("Tech Lead");

            leave.setdate\_of\_approval(LocalDate.now());

            System.out.println("leave request got approved by Tech Lead for " + leave.getname\_of\_employee());

        }

        else

        {

            System.out.println("sending leave request to Program Manager from Tech Lead");

            this.descendant.handleRequest(leave);

        }

    }

}

class project\_manager implements leave\_manager

{

    private leave\_manager descendant;

    public project\_manager(leave\_manager leavemanager)

    {

        this.descendant = leavemanager;

    }

    @Override

    public void handleRequest(Request\_Leave leave)

    {

        if(leave instanceof casual\_leave && leave.getleave\_status().equals("New"))

        {

            leave.setleave\_status("approved");

            leave.setapproval("Project Manager");

            leave.setdate\_of\_approval(LocalDate.now());

            System.out.println("leave request got approved by Project Manager for " + leave.getname\_of\_employee());

        }

        else

        {

            System.out.println("sending leave request to Director from Program Manager");

            this.descendant.handleRequest(leave);

        }

    }

}

class director implements leave\_manager

{

    public void handleRequest(Request\_Leave leave)

    {

        if(leave instanceof vacation\_leave  && !leave.getleave\_status().equals("approved"))

        {

            leave.setleave\_status("approved");

            leave.setapproval("Director");

            leave.setdate\_of\_approval(LocalDate.now());

            System.out.println("leave request got approved by Director for "+leave.getname\_of\_employee());

        }

    }

}

Main.java :-

import java.time.LocalDate;

import java.util.Scanner;

public class Main

{

    public static void main(String[] args)

    {

        Scanner in = new Scanner(System.in);

        System.out.println("Enter name of the employee:");

        String name = in.nextLine();

        System.out.println("Enter leave type 1: Sick Leave, 2: Casual Leave, 3: Vacation Leave.. ");

        int leave\_type = in.nextInt();

        LocalDate start\_date = null;

        LocalDate end\_date = null;

        String reason = null;

        switch(leave\_type)

        {

            case 1: break;

            case 2: in.nextLine();

                    System.out.println("Enter reason:");

                    reason = in.nextLine();

                    break;

            case 3: System.out.println("Enter starting date:");

                    start\_date = LocalDate.parse(in.next());

                    System.out.println("Enter ending date:");

                    end\_date = LocalDate.parse(in.next());

                    break;

            default: System.out.println("Invalid leave type entered");

                     return;

        }

        Request\_Leave leave;

        switch(leave\_type)

        {

            case 1: leave = new sick\_leave(name, LocalDate.now());

                    break;

            case 2: leave = new casual\_leave(name, LocalDate.now(), reason);

                    break;

            case 3: leave = new vacation\_leave(name, LocalDate.now(), start\_date, end\_date);

                    break;

            default: return;

        }

        director director = new director();

        project\_manager projectmanager = new project\_manager(director);

        tech\_lead techlead = new tech\_lead(projectmanager);

        techlead.handleRequest(leave);

        in.close();

    }

}