UE20CS352-OOADJ Lab-9&10

NAME	SRN	CLASS & SECTION
Vijay J	PES2UG20CS815	6 - J

PROBLEM STATEMENT:

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 processes 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.

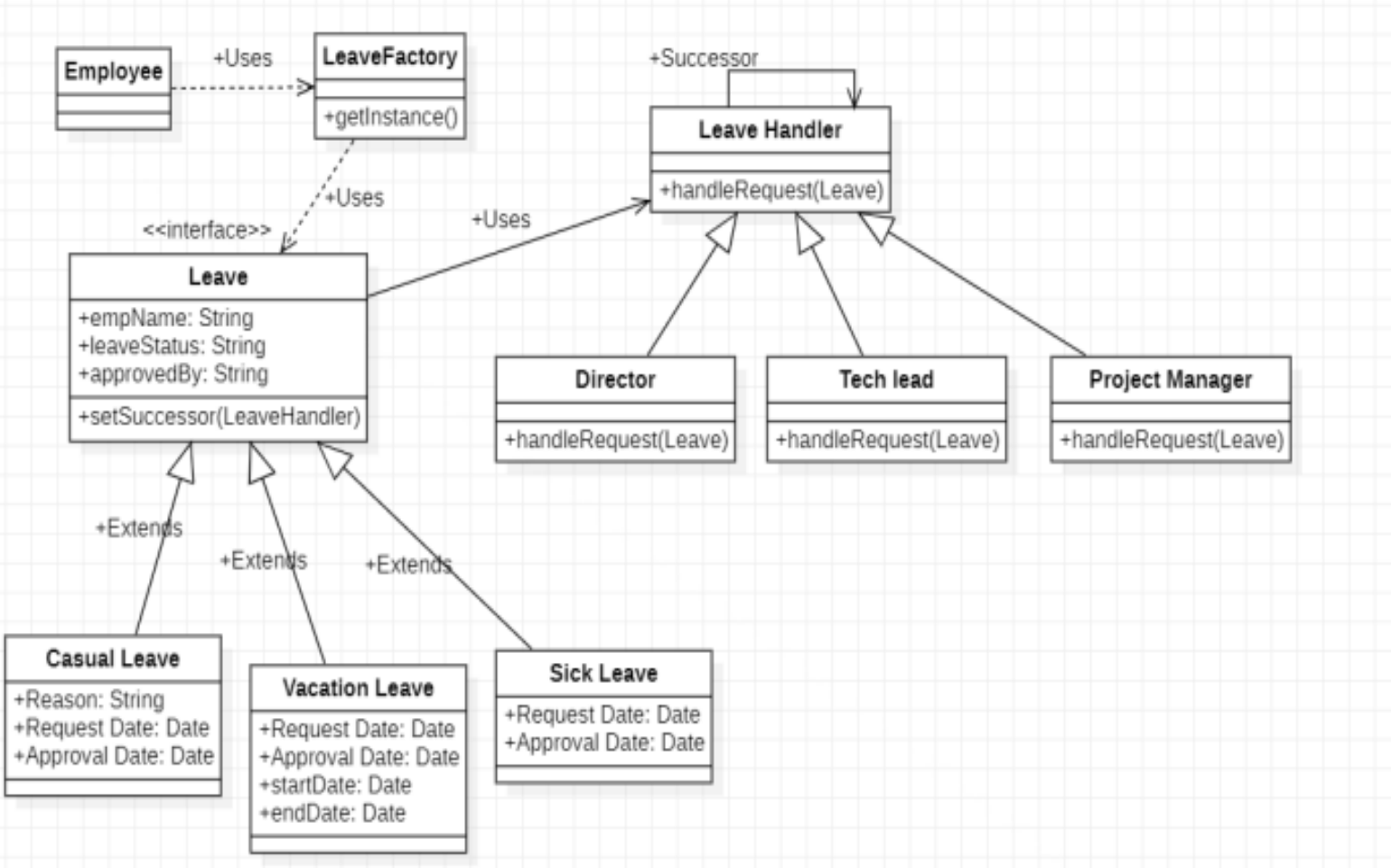
Design patterns considered:

- Chain of Responsibility Pattern.
- Facade Method
- Builder Method
- Factory Method

Design patterns used:

- Chain of Responsibility Pattern.
- Factory Method

UML Class Model:



CODE:
PES2UG20CS815.java

Open | PES2UG20CS815.java | ~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10 | Save

```java

// This class represents the main class that creates and tests the chain of responsibility
import java.util.Scanner; // for user input

public class PES2UG20CS815 {

    public static void main(String[] args) {

        // Create a chain of handlers: Tech Lead → Project Manager → Director
        Handler techLead = new TechLead(new ProjectManager(new Director()));

        // Create a scanner object for user input
        Scanner scanner = new Scanner(System.in);

        // Ask the user for the number of leave requests to test
        System.out.println("How many leave requests do you want to test?");
        int n = scanner.nextInt(); // read an integer from the user

        // Loop n times to create and test n leave requests
        for (int i = 0; i < n; i++) {

            // Ask the user for the details of the leave request
            System.out.println("Enter the employee name:");
            String empName = scanner.next(); // read a string from the user
            System.out.println("Enter the leave type (SL, CL, or VL):");
            String leaveType = scanner.next(); // read a string from the user
            System.out.println("Enter the request date (dd/mm/yyyy):");
            String requestDate = scanner.next(); // read a string from the user

            // Create a leave request object based on the leave type
            LeaveRequest request;
            if (leaveType.equals("SL") || leaveType.equals("CL")) {
                // Ask the user for the reason for SL or CL
                System.out.println("Enter the reason for " + leaveType + ":");
                String reason = scanner.next(); // read a string from the user
                // Create a SL or CL request with the reason
                request = new LeaveRequest(empName, leaveType, requestDate, reason);
            } else if (leaveType.equals("VL")) {
                // Ask the user for the start date and end date for VL
                System.out.println("Enter the start date (dd/mm/yyyy) for VL:");
                String startDate = scanner.next(); // read a string from the user
                System.out.println("Enter the end date (dd/mm/yyyy) for VL:");
                String endDate = scanner.next(); // read a string from the user
                // Create a VL request with the start date and end date
                request = new LeaveRequest(empName, leaveType, requestDate, startDate, endDate);
            } else {
                // Invalid leave type entered by the user
                System.out.println("Error: Invalid leave type entered");
                continue; // skip this iteration and go to the next one
            }

            // Test the chain of responsibility with the leave request
            techLead.handle(request);
        }

        // Close the scanner object
        scanner.close();
    }
}
```

Handler.java

Handler.java
~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10

```java
// This interface represents a handler in the chain of responsibility
public interface Handler {

    // This method handles the leave request or passes it to the next handler
    public void handle(LeaveRequest request);
}
```

Director.java

Director.java
~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10

```java
// This class represents a Director handler in the chain of responsibility
public class Director implements Handler {

    // The constructor that does not take any parameter
    public Director() {
        // No need to set the next handler as this is the last handler in the chain
    }

    @Override
    public void handle(LeaveRequest request) {

        System.out.println("Director received the leave request from " +
request.getEmpName());

        // If the leave type is VL, process it and approve or reject it based on the
        // duration
        if (request.getLeaveType().equals("VL")) {
            System.out.println("Director processed the VL request");
            // Assuming some logic to calculate the duration and decide whether to
approve
            // or reject
            int duration = calculateDuration(request.getStartDate(),
request.getEndDate());
            if (duration <= 10) { // assuming 10 days is the maximum allowed VL
duration
                System.out.println("Director approved the VL request");
                request.setLeaveStatus("Approved");
                request.setApprovedBy("Director");
                request.setApprovalDate("Today"); // assuming today's date
            } else {
                System.out.println("Director rejected the VL request");
                request.setLeaveStatus("Rejected");
```

```
                request.setApprovedBy("Director");
                request.setApprovalDate("Today"); // assuming today's date
            }
        }
        // Otherwise, print an error message as there is no more handler in the chain
        else {
            System.out.println("Error: Invalid leave type for Director handler");
        }
    }

    // A helper method to calculate the duration of VL in days
    private int calculateDuration(String startDate, String endDate) {
        // Assuming some logic to parse and compare the dates and return the
difference
        // in days
        return 5; // for simplicity, return 5 for now
    }
}
```

TechLead.java

Open | TechLead.java ~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10 | Save

```
// This class represents a Tech Lead handler in the chain of responsibility
public class TechLead implements Handler {

    // The next handler in the chain
    private Handler next;

    // The constructor that takes the next handler as a parameter
    public TechLead(Handler next) {
        this.next = next;
    }

    @Override
    public void handle(LeaveRequest request) {

        System.out.println("Tech Lead received the leave request from " +
request.getEmpName());

        // If the leave type is SL, process it and approve it
        if (request.getLeaveType().equals("SL")) {
            System.out.println("Tech Lead approved the SL request");
            request.setLeaveStatus("Approved");
            request.setApprovedBy("Tech Lead");
            request.setApprovalDate("Today"); // assuming today's date
        }
        // Otherwise, forward it to the next handler
        else {
            System.out.println("Tech Lead forwarded the request to the next handler");
            next.handle(request);
        }
    }
}
```

ProjectManager.java

Open | ProjectManager.java | ~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10 | Save

```java
// This class represents a Project Manager handler in the chain of responsibility
public class ProjectManager implements Handler {

    // The next handler in the chain
    private Handler next;

    // The constructor that takes the next handler as a parameter
    public ProjectManager(Handler next) {
        this.next = next;
    }

    @Override
    public void handle(LeaveRequest request) {

        System.out.println("Project Manager received the leave request from " + request.getEmpName());

        // If the leave type is CL, process it and approve or reject it based on the
        // reason
        if (request.getLeaveType().equals("CL")) {
            System.out.println("Project Manager processed the CL request");
            // Assuming some logic to check the reason and decide whether to approve or
            // reject
            boolean validReason = checkReason(request.getReason());
            if (validReason) {
                System.out.println("Project Manager approved the CL request");
                request.setLeaveStatus("Approved");
                request.setApprovedBy("Project Manager");
                request.setApprovalDate("Today"); // assuming today's date
            } else {
                System.out.println("Project Manager rejected the CL request");
                request.setLeaveStatus("Rejected");
                request.setApprovedBy("Project Manager");
                request.setApprovalDate("Today"); // assuming today's date
            }
        }
        // Otherwise, forward it to the next handler
        else {
            System.out.println("Project Manager forwarded the request to the next handler");
            next.handle(request);
        }
    }

    // A helper method to check the reason for CL and return true or false
    private boolean checkReason(String reason) {
        // Assuming some logic to validate the reason
        return true; // for simplicity, return true for now
    }
}
```

LeaveRequest.java

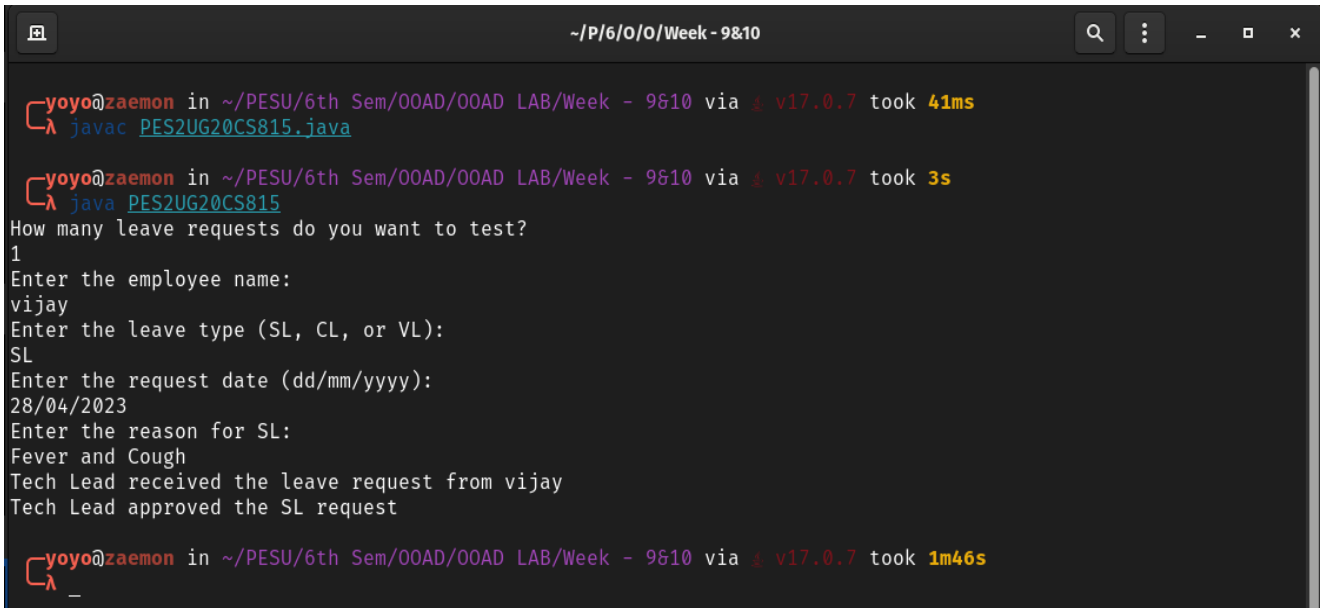
Open | LeaveRequest.java | ~/PESU/6th Sem/OOAD/OOAD LAB/Week - 9&10 | Save

```java
// This class represents a leave request with the details
public class LeaveRequest {
    // The employee name
    private String empName;
    // The leave type: SL, CL, or VL
    private String leaveType;
    // The leave status: New, Approved, or Rejected
    private String leaveStatus;
    // The name of the handler who approved or rejected the request
    private String approvedBy;
    // The date of the request
    private String requestDate;
    // The date of the approval or rejection
    private String approvalDate;
    // The start date of the VL
    private String startDate;
    // The end date of the VL
    private String endDate;
    // The reason for the CL
    private String reason;

    // Constructor for SL and CL requests
    public LeaveRequest(String empName, String leaveType, String requestDate, String reason) {
        this.empName = empName;
        this.leaveType = leaveType;
        this.requestDate = requestDate;
        this.reason = reason;
        this.leaveStatus = "New"; // default status
        this.approvedBy = null; // default value
        this.approvalDate = null; // default value
        this.startDate = null; // not applicable
        this.endDate = null; // not applicable
    }

    // Constructor for VL requests
    public LeaveRequest(String empName, String leaveType, String requestDate, String startDate, String endDate) {
        this.empName = empName;
        this.leaveType = leaveType;
        this.requestDate = requestDate;
        this.startDate = startDate;
        this.endDate = endDate;
        this.leaveStatus = "New"; // default status
        this.approvedBy = null; // default value
        this.approvalDate = null; // default value
        this.reason = null; // not applicable
    }

    // Getters and setters for the fields

    public String getEmpName() {
        return empName;
    }

    public void setEmpName(String empName) {
        this.empName = empName;
    }

    public String getLeaveType() {
        return leaveType;
    }

    public void setLeaveType(String leaveType) {
```

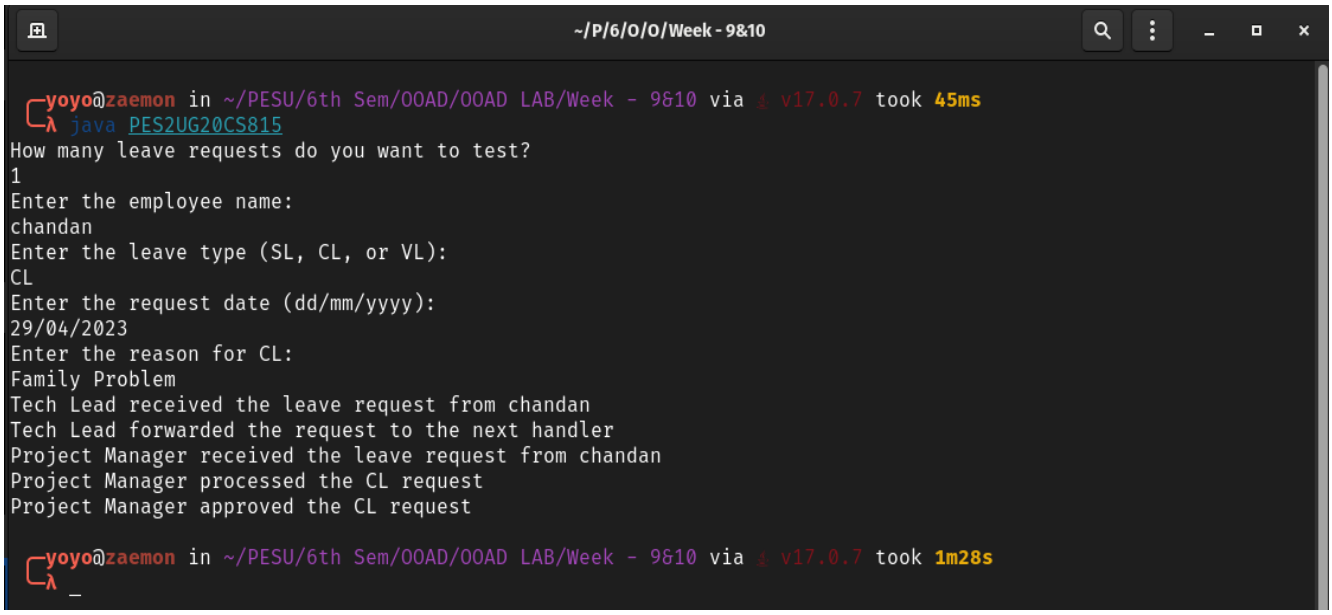
```
        this.leaveType = leaveType;
    }

    public String getLeaveStatus() {
        return leaveStatus;
    }

    public void setLeaveStatus(String leaveStatus) {
        this.leaveStatus = leaveStatus;
    }

    public String getApprovedBy() {
        return approvedBy;
    }

    public void setApprovedBy(String approvedBy) {
        this.approvedBy = approvedBy;
    }

    public String getRequestDate() {
        return requestDate;
    }

    public void setRequestDate(String requestDate) {
        this.requestDate = requestDate;
    }

    public String getApprovalDate() {
        return approvalDate;
    }

    public void setApprovalDate(String approvalDate) {
        this.approvalDate = approvalDate;
    }

    public String getStartDate() {
        return startDate;
    }

    public void setStartDate(String startDate) {
        this.startDate = startDate;
    }

    public String getEndDate() {
        return endDate;
    }

    public void setEndDate(String endDate) {
        this.endDate = endDate;
    }

    public String getReason() {
        return reason;
    }

    public void setReason(String reason) {
        this.reason = reason;
    }
}
```

OUTPUT:

Sick Leave (SL):



Casual Leave (CL):



Vacation Leave (VL):