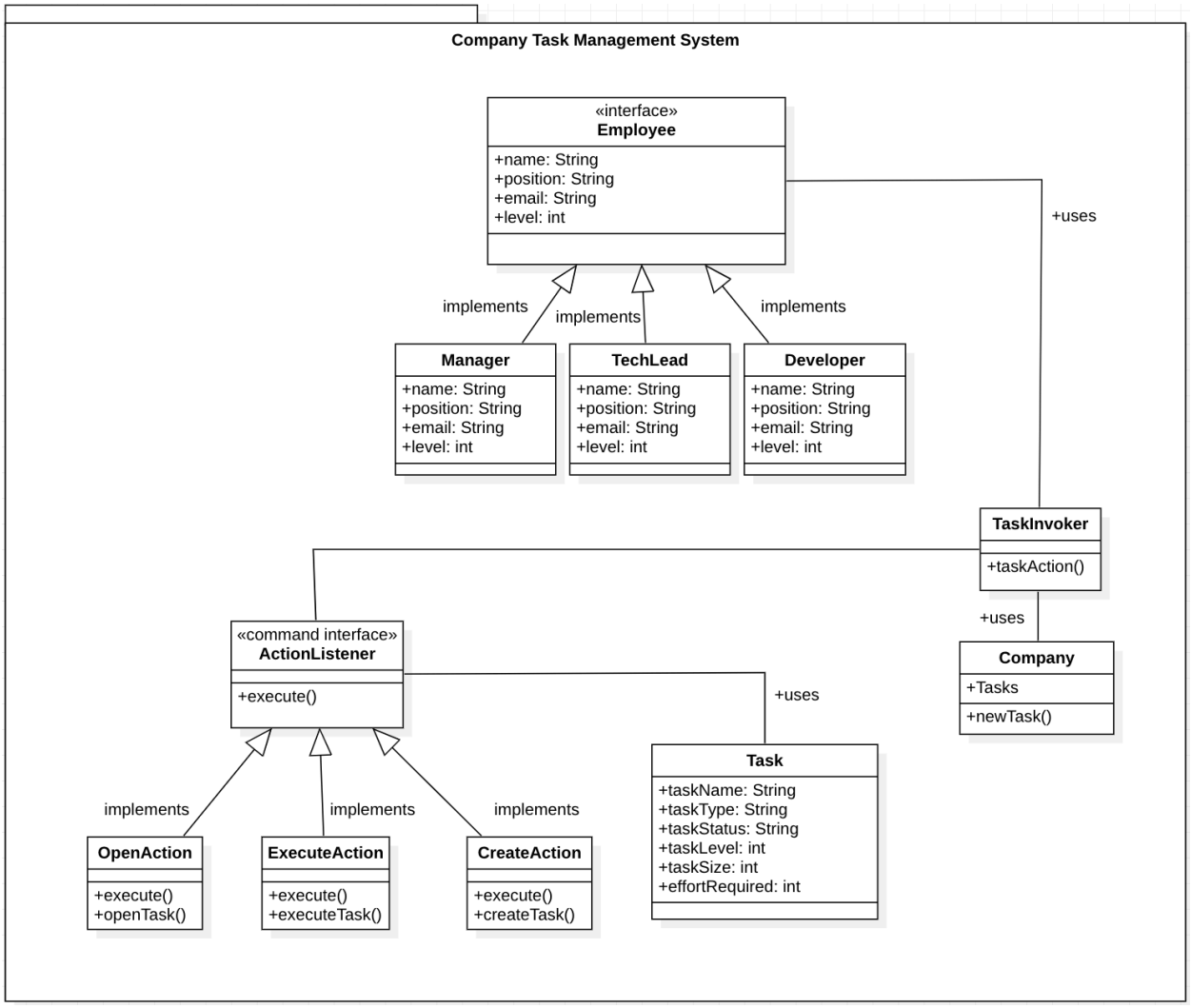
Object Oriented Analysis and Design Laboratory
Assignment - 2

Name : Abhishek Aditya BS
SRN : PES1UG19CS019
Sem & Section : VI Sem & A Section

A Company has a Manager, TechLead and Developer as its Employee. The Company creates Task for its Employee. This Task contains attributes such as taskName, taskType, taskStatus, taskLevel, taskSize, effortRequired. After the Task gets created by the Company it will be given to its Employee. Task can be Opened and Executed by the Employee. If the taskLevel is 3, then it will be Opened and Executed by Manager, if it is 2, it will be Opened by the Manager (meaning manager has a task clearance level of 3 and 2) and Executed by the TechLead, and if 1 will be Opened by the TechLead and Executed by the Developer. Design the UML and implement the same using appropriate design patterns.

Note: Design the application in such a way that extensibility is easy. It should be easy to add new types of Employee, new actions that can be performed on the task (Opened, Executed, etc.).

UML Diagram



Design Patterns Identified

- **Factory Pattern** - This pattern is used for employee creation. All employees created will have the same properties but each employee will perform different actions.
- **Command Pattern** - This pattern is used for invoking the open and execute actions to different employees. Employees acts as receivers of action.

Implementation

Employee.java

```
public abstract class Employee {
    String Name;
    String Post;
    String Email;

    abstract void openOrExecute(boolean oe, String tn);
}
```

Manager.java

```
public class Manager extends Employee{
    Manager(String n, String e){
        this.Name = n;
        this.Post = "MANAGER";
        this.Email = e;
    }

    public void openOrExecute(boolean oe, String tn){
        TaskInvoker.taskAction("MANAGER", tn, oe);

    }
}
```

Developer.java

```
public class Developer extends Employee{
    Developer(String n, String e){
        this.Name = n;
        this.Post = "DEVELOPER";
        this.Email = e;
    }

    public void openOrExecute(boolean oe, String tn){
        TaskInvoker.taskAction("DEVELOPER", tn, oe);
    }
}
```

TechLead.java

```
public class TechLead extends Employee{
    TechLead(String n, String e){
        this.Name = n;
        this.Post = "TECHLEAD";
```

```
        this.Email = e;
    }

    public void openOrExecute(boolean oe, String tn){
        TaskInvoker.taskAction("TECHLEAD", tn, oe);
    }
}
```

Task.java

```
public class Task{
    protected String taskName;
    protected String taskType;
    protected String taskStatus;
    protected int taskLevel;
    protected int taskSize;
    protected int effortRequired;

    Task(String tn, String tt, String ts, int tl, int tS, int er){
        this.taskName = tn;
        this.taskType = tt;
        this.taskStatus = ts;
        this.taskLevel = tl;
        this.taskSize = tS;
        this.effortRequired = er;
    }

    Task(){}
}
```

TaskInvoker.java

```
import java.util.*;

public class TaskInvoker {
    static ArrayList<Task> Tasks = new ArrayList<Task>();
    public static void taskAction(String s, String tn, boolean oe){
        if(s.equals("COMPANY")){
            CreateAction c = new CreateAction();
            c.execute(5,"");
            return;
        }
        Task T = getTask(tn);
        if(T != null){
            if(oe){
                OpenAction o = new OpenAction();
                if(s.equals("MANAGER") && T.taskLevel == 3 || T.taskLevel == 2){
                    o.execute(3, T.taskName);
```

```
            }
            else if(s.equals("TECHLEAD") && T.taskLevel == 1){
                o.execute(1, T.taskName);
            }
            else{
                System.out.println("Access Denied for Employee.");
            }

        }
        else{
            ExecuteAction e = new ExecuteAction();
            if(s.equals("MANAGER") && T.taskLevel == 3){
                e.execute(3, T.taskName);
            }
            else if(s.equals("TECHLEAD") && T.taskLevel == 2){
                e.execute(2, T.taskName);
            }
            else if(s.equals("DEVELOPER") && T.taskLevel == 1){
                e.execute(1, T.taskName);
            }
            else{
                System.out.println("Access Denied for Employee.");
            }
        }
    }
}

public static void getAllTasks(boolean p){
    CreateAction c = new CreateAction();
    Tasks = c.getTasks();
    if(p){
        for(Task t : Tasks){
            System.out.println("Task :");
            System.out.println("Task Name: "+ t.taskName);
            System.out.println("Task Type: "+ t.taskType);
            System.out.println("Task Status: "+ t.taskStatus);
            System.out.println("Task Level: "+ t.taskLevel);
            System.out.println("Task Size: "+ t.taskSize);
            System.out.println("Effort Required: "+ t.effortRequired);
            System.out.println("\n");
        }
    }

}

private static Task getTask(String TN){
    getAllTasks(false);
    ListIterator<Task> lt = Tasks.listIterator();
    Task Tk;
```

```
        while(It.hasNext()){
            Tk = It.next();
            if(Tk.taskName.equals(TN)){
                return Tk;
            }
        }
        System.out.println("Task Not Found");
        return new Task();
    }

}
```

CreateAction.java

```
import java.util.*;

public class CreateAction implements ActionListener{
    static ArrayList<Task> Tasks = new ArrayList<Task>();
    public void execute(int cl, String n){
        System.out.println("-----------------------------------------------------");
        System.out.println("Company Call to Create New Task");
        createTask();
    }

    public void createTask(){

            Scanner sc = new Scanner(System.in);
            System.out.println("Enter the Task Name: ");
            String tn = sc.nextLine();
            System.out.println("Enter the Task Type: ");
            String tt = sc.nextLine();
            System.out.println("Enter the Task Status: ");
            String ts = sc.nextLine();
            System.out.println("Enter the Task Level: ");
            int tl = Integer.parseInt(sc.nextLine());
            System.out.println("Enter the Task Size: ");
            int tS = Integer.parseInt(sc.nextLine());
            System.out.println("Enter the Effort Required: ");
            int er = Integer.parseInt(sc.nextLine());

            Task Tk = new Task(tn, tt, ts, tl, tS, er);
            Tasks.add(Tk);
            System.out.println(String.format("Task %s Created",tn));
            sc.close();
    }

    public ArrayList<Task> getTasks(){
        return Tasks;
    }
}
```

OpenAction.java

```
public class OpenAction implements ActionListener{

    public void execute(int cl, String n){
        System.out.println("-----------------------------------------------------");
        System.out.println("Function to execute the Open Task");
        openTask(cl, n);
    }

    public void openTask(int cl, String n){
        if(cl == 3 || cl == 2)
            System.out.println(String.format("Manager can OPEN the Task
%s\n\n----------------------",n));
        else if(cl == 1)
            System.out.println(String.format("TechLead can OPEN the Task %s\n\n----------------------",
n));
    }
}
```

ExecuteAction.java

```
public class ExecuteAction implements ActionListener{

    public void execute(int cl, String n){
        System.out.println("-----------------------------------------------------");
        System.out.println("Function to execute the execute Task");
        executeTask(cl, n);
    }

    public void executeTask(int cl, String n){
        if(cl == 3)
            System.out.println(String.format("Manager can EXECUTE the Task
%s\n\n--------------------", n));
        else if(cl == 2)
            System.out.println(String.format("TechLead can EXECUTE the Task
%s\n\n--------------------",n));
        else
            System.out.println(String.format("Developer can EXECUTE the Task
%s\n\n--------------------",n));

    }
}
```

ActionListener.java

```
public interface ActionListener {
```

```
    public void execute(int cl, String n);
}
```

Company.java

```
import java.util.*;

public class Company {
    static ArrayList<Task> Tasks = new ArrayList<Task>();

    public void newTask(){
        TaskInvoker.taskAction("COMPANY","", false);
    }

    public void getTasks(){
        TaskInvoker.getAllTasks(true);
    }
}
```

Main.java

```
public class Main {
    public static void main(String[] args) {
        Company company = new Company();
        Manager manager = new Manager("Bailey", "bailey@yahoo.in");
        TechLead techLead = new TechLead("John", "john@gmail.com");
        Developer developer = new Developer("Summer", "summer@rediff.com");

        company.newTask();

        manager.openOrExecute(true, "Login");
        manager.openOrExecute(false, "Admin");

        company.newTask();

        techLead.openOrExecute(true, "Login");
        techLead.openOrExecute(false, "Admin");

        developer.openOrExecute(true, "Login");
        developer.openOrExecute(false, "Admin");

        company.getTasks();

    }
}
```

Output

```
Company Call to Create New Task
Enter the Task Name:
Server Management
Enter the Task Type:
Beginner
Enter the Task Status:
Pending
Enter the Task Level:
3
Enter the Task Size:
2
Enter the Effort Required:
2
Task Server Management Created
Task Not Found
Access Denied for Employee.
Task Not Found
Access Denied for Employee.
```

```
Company Call to Create New Task
Enter the Task Name:
College Management
Enter the Task Type:
Senior
Enter the Task Status:
Open
Enter the Task Level:
1
Enter the Task Size:
1
Enter the Effort Required:
1
Task College Management Created
Task Not Found
Access Denied for Employee.
Task Not Found
Access Denied for Employee.
Task Not Found
Access Denied for Employee.
Task Not Found
Access Denied for Employee.
Task :
Task Name: Server Management
Task Type: Beginner
Task Status: Pending
Task Level: 3
Task Size: 2
Effort Required: 2
```

```
Task :
Task Name: College Management
Task Type: Senior
Task Status: Open
Task Level: 1
Task Size: 1
Effort Required: 1
```