

Object Oriented Analysis and Design using Java (UE18CS353)

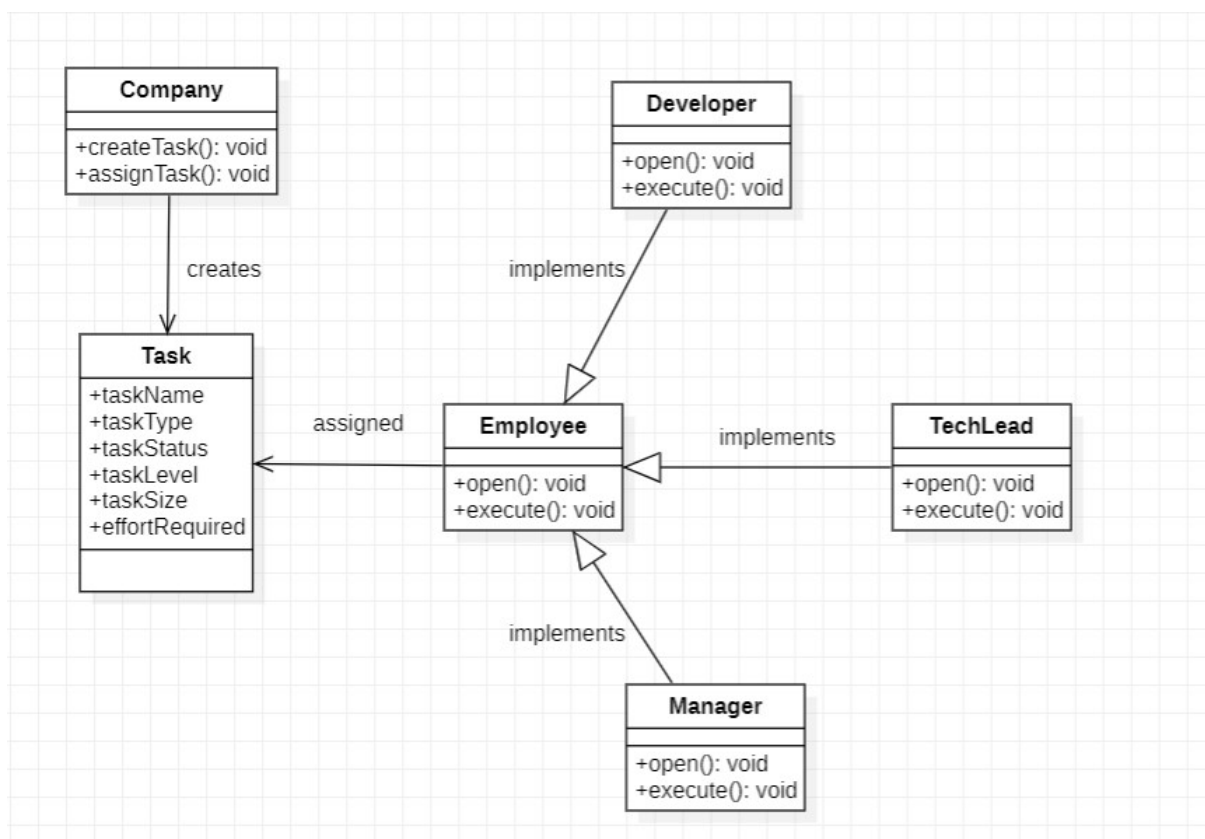
Assignment-2: Design Patterns

Name – B Pravena

Sec - B

SRN – PES2UG19CS076

Class Diagram -:



Design Patter Information -:

In object-oriented programming, a singleton class is a class that can have only one object (an instance of the class) at a time. The purpose of the singleton class is to control object creation, limiting the number of objects to only one. The singleton allows only one entry point to create the new instance of the class called the global access point.

Output -:

```
C:\Windows\System32\cmd.exe

Microsoft Windows [Version 10.0.19043.1645]
(c) Microsoft Corporation. All rights reserved.

D:\6th_sem\OOADJ_LAB\Assignment2>javac Company.java

D:\6th_sem\OOADJ_LAB\Assignment2>java Company
Company created the task
Company opened the task
Company executed the task

D:\6th_sem\OOADJ_LAB\Assignment2>
```

Code -:

```
Company.java 3 X
D: > 6th_sem > OOADJ_LAB > Assignment2 > Company.java > Company > main(String[])
1  interface Employee
2  {
3      public void showEmployeeDetails();
4  }
5
6  class Task
7  {
8      private String taskName;
9      private String taskType;
10     private String taskStatus;
11     private int taskLevel;
12     private int taskSize;
13     private int effortRequired;
14
15     public Task(String taskName, String taskType, String taskStatus, int taskLevel, int taskSize, int effortRequired)
16     {
17         this.taskName = taskName;
18         this.taskType = taskType;
19         this.taskStatus = taskStatus;
20         this.taskLevel = taskLevel;
21         this.taskSize = taskSize;
22         this.effortRequired = effortRequired;
23     }
24
25     public String getTaskName()
26     {
27         return taskName;
28     }
29
30     public void setTaskName(String taskName)
31     {
32         this.taskName = taskName;
33     }
```

```

35     public String getTaskType()
36     {
37         return taskType;
38     }
39
40     public void setTaskType(String taskType)
41     {
42         this.taskType = taskType;
43     }
44
45     public String getTaskStatus()
46     {
47         return taskStatus;
48     }
49
50     public void setTaskStatus(String taskStatus)
51     {
52         this.taskStatus = taskStatus;
53     }
54
55     public int getTaskLevel()
56     {
57         return taskLevel;
58     }
59
60     public void setTaskLevel(int taskLevel)
61     {
62         this.taskLevel = taskLevel;
63     }
64
65     public int getTaskSize()
66     {
67         return taskSize;
68     }

```

```

70     public void setTaskSize(int taskSize)
71     {
72         this.taskSize = taskSize;
73     }
74
75     public int getEffortRequired()
76     {
77         return effortRequired;
78     }
79
80     public void setEffortRequired(int effortRequired)
81     {
82         this.effortRequired = effortRequired;
83     }
84
85     public void showTaskDetails()
86     {
87         System.out.println("Task Name: " + taskName);
88         System.out.println("Task Type: " + taskType);
89         System.out.println("Task Status: " + taskStatus);
90         System.out.println("Task Level: " + taskLevel);
91         System.out.println("Task Size: " + taskSize);
92         System.out.println("Effort Required: " + effortRequired);
93     }
94 }

```

```

97  class Manager implements Employee
98  {
99      private String name;
100     private long empId;
101     private String position;
102
103     public Manager(long empId, String name, String position)
104     {
105         this.empId = empId;
106         this.name = name;
107         this.position = position;
108     }
109
110     @Override
111     public void showEmployeeDetails()
112     {
113         System.out.println(empId + " " + name + " " + position);
114     }
115
116     public void createTask(Task task)
117     {
118         System.out.println("Manager created the task");
119     }
120
121     public void openTask(Task task)
122     {
123         System.out.println("Manager opened the task");
124     }
125
126     public void executeTask(Task task)
127     {
128         System.out.println("Manager executed the task");
129     }
130 }

```

```

131
132  class Developer implements Employee
133  {
134      private String name;
135      private long empId;
136      private String position;
137
138      public Developer(long empId, String name, String position)
139      {
140          this.empId = empId;
141          this.name = name;
142          this.position = position;
143      }
144
145      @Override
146      public void showEmployeeDetails()
147      {
148          System.out.println(empId + " " + name + " " + position);
149      }
150
151      public void createTask(Task task)
152      {
153          System.out.println("Developer created the task");
154      }
155
156      public void openTask(Task task)
157      {
158          System.out.println("Developer opened the task");
159      }
160
161      public void executeTask(Task task)
162      {
163          System.out.println("Developer executed the task");
164      }
165  }

```

```

168 class TechLead implements Employee
169 {
170     private String name;
171     private long empId;
172     private String position;
173
174     public TechLead(long empId, String name, String position)
175     {
176         this.empId = empId;
177         this.name = name; this.position = position;
178     }
179
180     @Override
181     public void showEmployeeDetails()
182     {
183         System.out.println(empId + " " + name + " " + position);
184     }
185
186     public void createTask(Task task)
187     {
188         System.out.println("TechLead created the task");
189     }
190
191     public void openTask(Task task)
192     {
193         System.out.println("TechLead opened the task");
194     }
195
196     public void executeTask(Task task)
197     {
198         System.out.println("TechLead executed the task");
199     }
200 }

```

```

202 class CompanyDirectory
203 {
204     private Manager manager;
205     private TechLead techLead;
206     private Developer developer;
207
208     public CompanyDirectory(Manager manager, TechLead techLead, Developer developer)
209     {
210         this.manager = manager;
211         this.techLead = techLead;
212         this.developer = developer;
213     }
214
215     public void createTask(Task task)
216     {
217         System.out.println("Company created the task");
218     }
219
220     public void openTask(Task task)
221     {
222         System.out.println("Company opened the task");
223     }
224
225     public void executeTask(Task task)
226     {
227         System.out.println("Company executed the task");
228     }
229 }

```

```

231 class Company
232 {
233     public static void main(String[] args)
234     {
235         Manager manager = new Manager(12345, "John", "Manager");
236         TechLead techLead = new TechLead(12346, "Peter", "TechLead");
237         Developer developer = new Developer(12347, "Paul", "Developer");
238         CompanyDirectory companyDirectory = new CompanyDirectory(manager, techLead, developer);
239         Task task = new Task("Task1", "TaskType1", "TaskStatus1", 1, 1, 1);
240         companyDirectory.createTask(task);
241         companyDirectory.openTask(task);
242         companyDirectory.executeTask(task);
243     }
244 }
245

```