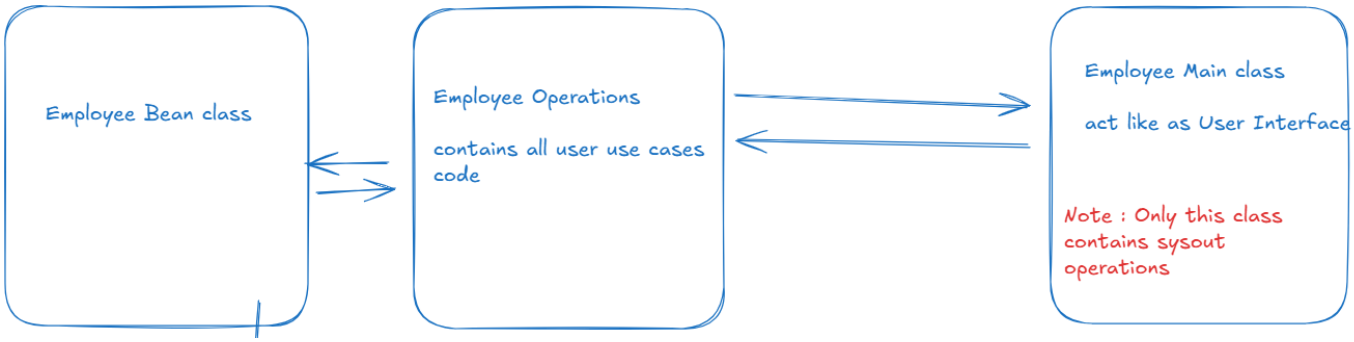
#Problem 1 : Implement the below problem statement through Java List Collection

```
package p4;

import java.util.List;

public class Demo {

    List<Integer> list;

    public static void main(String[] args) {
        // write code to insert the element into the list
        // print :: x element available in the list  if item x is present in the list
        // else print x Element is not available
        // x is the value of element user is looking for
    }
    public void insert(int x)
    {

    }

    public boolean findElement(int searchElement)
    {
        // implement code to search the element and return boolean
        return false;
    }
}
```

#Problem 2 : Implement the following Use cases

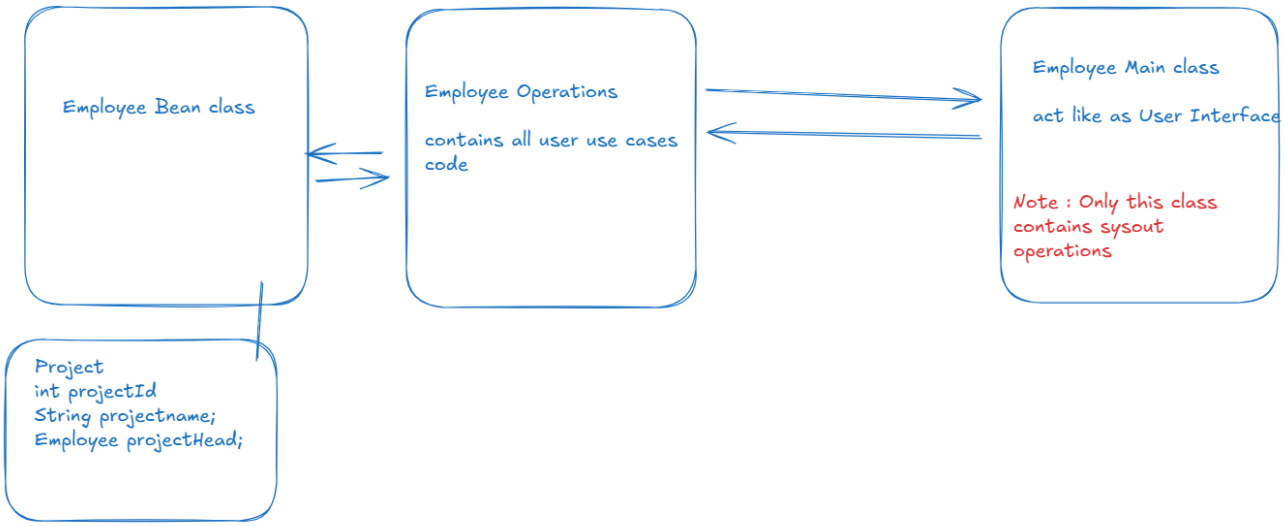



```java
public class Employee {

    private int empId;
    private String name;
    private int salary;
    private String baseLocation;
    private String currentLocation;
    private LocalDate joiningDate;


```

```java
public class EmployeeRunner {

    public static void main(String[] args) {

        System.out.println(" ---- MENU ----");
        System.out.println("1. Insert Employee");
        System.out.println("2. Get All Employees");
        System.out.println("3. Search Employee By id");
        System.out.println("4. Search Employee by Department");
        System.out.println("5. Search Employee by Baselocation & CurrentLocation are Same");
        System.out.println("6. Search Employee by Date of Joinning more than 1 year");
        System.out.println("0. EXIT");
        // add more search and filter operations over employee


    }

}
```

```java
package p1;

import java.util.ArrayList;
import java.util.List;

public class EmployeeOperation {

    List<Employee> allEmployees;

    public void insertEmployee(Employee e)
    {

    }

    public List<Employee> getEmployeeByBaseLocation(String searchLocation)
    {
        List<Employee> resultList = new ArrayList<>();

        // code

        return resultList;
    }
    // many more...
}
```

#Problem 3 : Add Project Class in the Above code



Implement below Use cases.

```
System.out.println("7. Allocate Project to an Employee");
System.out.println("8. Deallocate Project ");
System.out.println("9. Get list of Employees Under Project Head");
System.out.println("10. Get All Employees based on Project Name");
```

#Problem 4 :

Implement two following Custom Exceptions

```
2
3  public class EmployeeAlreadyOccupiedException {
4
5      //An employee cannot be allocated to a new project
6      //if they are already assigned to an existing one
7  }
8
```

```
3  public class InvalidLocationException {
4
5      // Only Valid locations are
6      // Delhi-NCR , Banglore , Chennai , Pune
7
8  }
```