1. Design the Employee Application with Model class Employee with following fields.

private Long id;

private Long organizationId;

private Long departmentId;

private String name;

private int age;

private String position;

private float salary;

private Date joiningDate;

private Date endDate; // end of thecontract.

Organization :

Id ,

Name,

Location,

Noofbranches

1. Create the public default and parameterized constructor.
2. Write an EmployeeService class to perform the operations.
3. Public String addEmployee(Employee employee)
4. Public String updateEmployee(long id)
5. Public String deleteEmployee(long id)
6. Public Optional<Employee> findById(long id)
7. Public Optional<List<Employee>> getEmployees()
8. Public Optional<List<Employee>> findByOrganizationId(long id)
9. Public Optional<List<EmplOrgMapper>> findByEmpDetailsBasedOnOrg();
10. Public Optional<List<Employee>> contractEndson(Date date);
11. Public Boolean isContractEnd(long empId);
12. Write a EmployeeRepository class to perform the DB operations on Employee.

Public String addEmployee(Employee employee)

Public String updateEmployee(long id)

Public String deleteEmployee(long id)

Public Optional<Employee> findById(long id)

Public Optional<List<Employee>> getEmployees()

Public Optional<List<Employee>> findByOrganizationId(long id)

1. Write a menu driven program for this.
2. Design the Department class with following fields.

private Long id;

private Long organizationId;

private String name;

private List<Employee> employees;

1. Repeat steps 3 to 5 for Department.
2. Design the Organization class with following field.

|  |
| --- |
|  |
| private Long id; |
|  | private String name; |
|  | private String address; |
|  | private List<Department> departments = new ArrayList<>(); |
|  | private List<Employee> employees = new ArrayList<>(); |
|  |  |

1. Repeat the steps from 3 to 5 for Organization.
2. Implement Singleton practice for all service and DAOs.