**Objectives**

* Demonstrate integration of RESTful Web Service of type GET and test the service-using postman.
* REST Web Service architecture with Controller, Service and Dao, service methods.Page Break

**Hands-on 1**

**Problem Statement –**

**Display Employee List and Edit Employee Details using RESTful Web Service**    
   
This service should be developed to get the data from RESTful Web Service developed in Spring. The following are the high-level activities that needs to be done to accomplish this:  

* Create static employee list data using spring xml configuration
* Create a REST Service that reads data from xml configuration and returns it

***NOTE*** *: There is no specific activity as part of this hands on, refer the next hands-on’s that covers above three*

*activities in detail.*

**Create static employee list data using spring xml configuration** 

Follow steps below to accomplish this activity:  

* Incorporate the following in employee.xml:
* Create one or two more departments
* Create four more instances of Employee.  (use employee sample data from angular)
* Reuse existing skills instead of creating new ones
* Include all four employee instances in an ArrayList.
* In EmployeeDao, incorporate the following:
* Create static variable with name EMPLOYEE\_LIST of type ArrayList<Employee>
* Include constructor that reads employee list from xml config and set the EMPLOYEE\_LIST
* Create method getAllEmployees() that returns the EMPLOYEE\_LIST

**Create REST service to gets all employees**    
  
Follow steps below to accomplish this activity:

* In EmployeeService, incorporate the following:
* Change the annotation for this class from @Component to @Service
* Create method getAllEmployees() that invokes employeeDao.getAllEmployees() and return the employee list
* Define @Transactional annotation for this method.
* In EmployeeController, incorporate the following:
* Include a new get method with name getAllEmployees() that returns the employee list
* Mark this method as GetMapping annotation with the URL as '/employees'
* Within this method invoke employeeService.getAllEmployees() and return the same.

​​

* Test ​the service using postman.

**Hands-on 2**

**Create REST service for department**

Create a new service to get all the departments.

Follow steps below to achieve this:

* Create a new REST Service, define below list of classes and respective methods:
* DepartmentController
* getAllDepartments() with URL "/departments", this method will return array of departments
* DepartmentService
* getAllDepartments()
* DepartmentDao
* getAllDepartments() - Create a static variable DEPARTMENT\_LIST, this should be populated from spring xml configuration
* Test ​the service using postman.
* Also verify if department REST service is called by looking into the logs.