Sprint-2 Plan

Employee Detail Management

1. Establishing Project Folder Architecture
2. Server App Initialization
3. **Sprint-2 Execution** [2(a), 2(b), **2(c)**, 2(d), 2(e), 2(f)]
4. Completing Sprint-1 Objectives

2(a) Employee Module - Analysis, Design & DB-Preparation

2(b) Employee View

2(c) Employee Search

Design Search Criteria
Server App Modifications
Client App Modifications

2(d) Employee Insert
2(e) Employee Update
2(f) Employee Delete

==========

2(c) Employee Search

1. Design Search Criteria
   - **Define Search Attributes**
   - **Define Lists Required for Search Panels**
   - Define SQL Quarries Needed

2. Server-App Modification

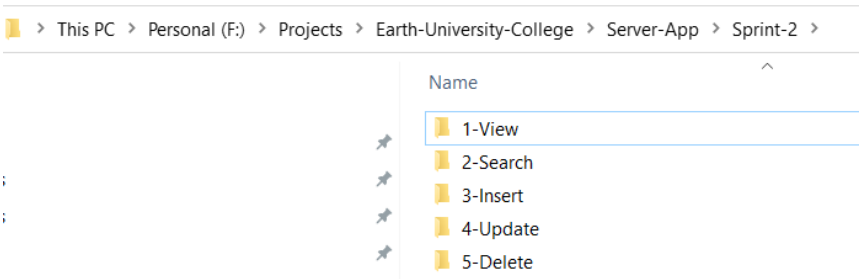
   Implement Listing Services
   Implement Searching Service
   - **(1) Using Controller Filtering**
   - (2) Search by QBE
   - (3) Search by Named Queries
   - (4) Search by Native Queries

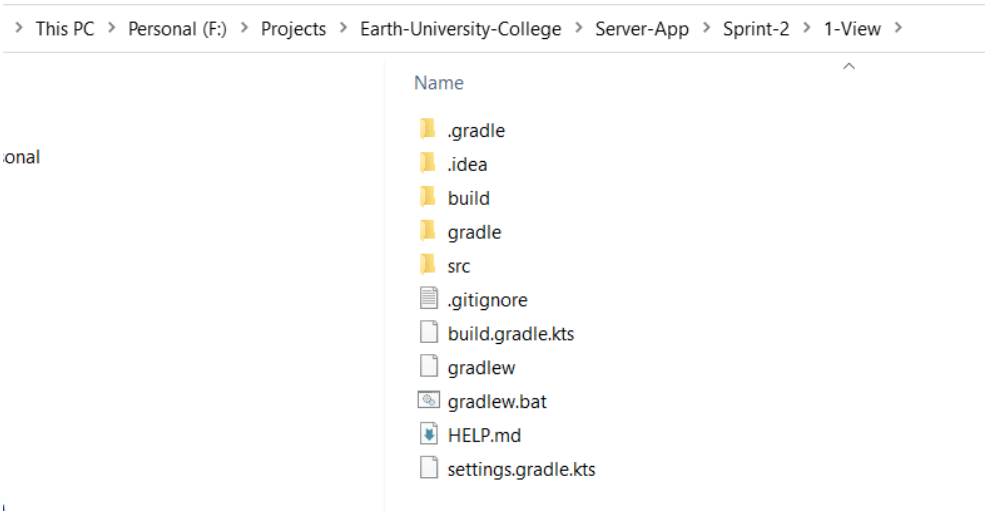
3. Client-App Modification

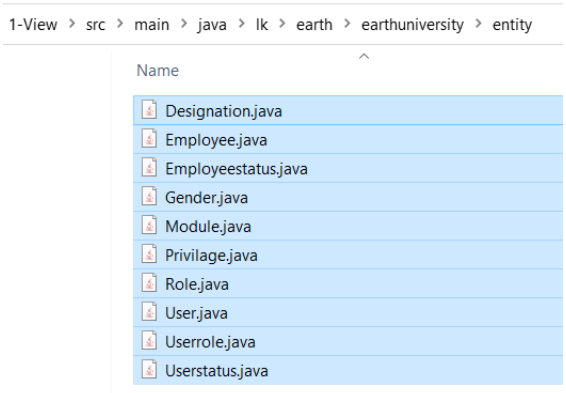
==========

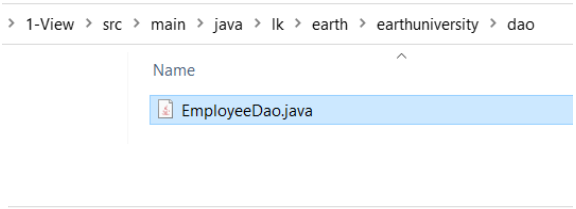
1. Content of the “Server-App” folder at the End of Sprint-2 “View” Completion

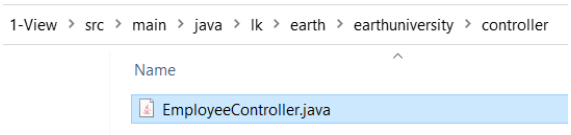
(Make sure you must start from here)





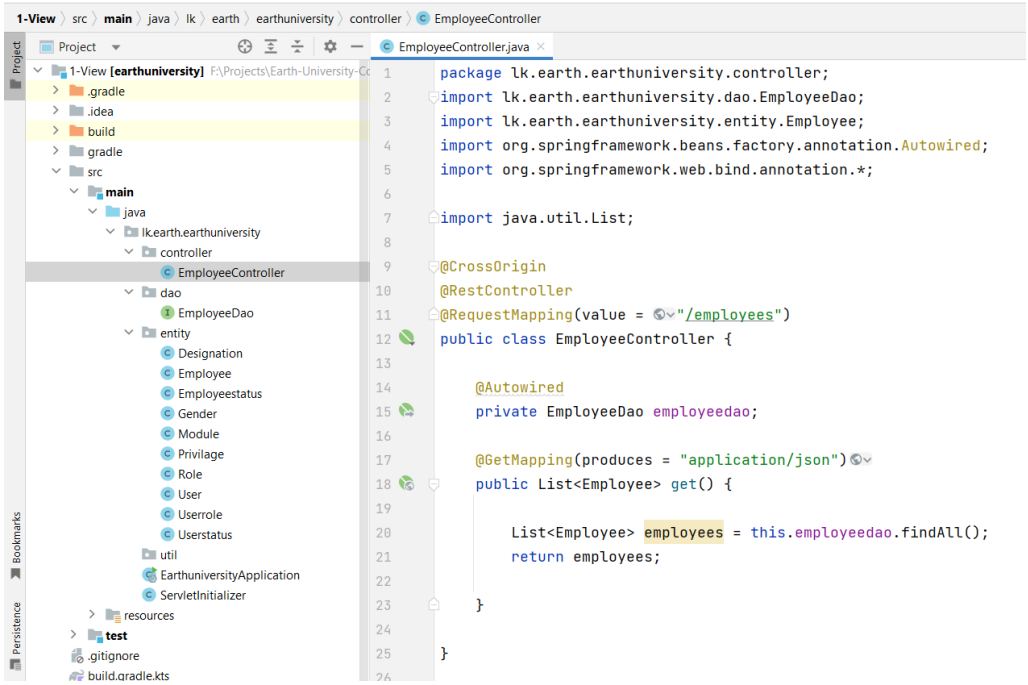






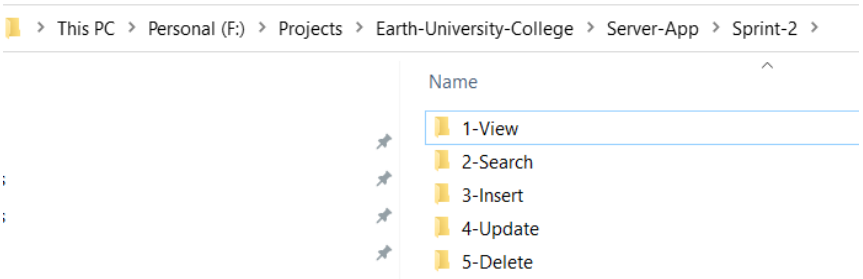
EmployeeDao.java

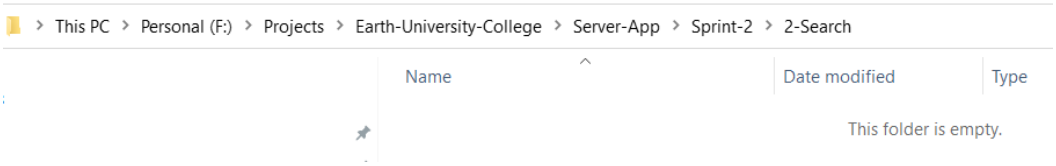
```java
package lk.earth.earthuniversity.dao;

import lk.earth.earthuniversity.entity.Employee;
import org.springframework.data.jpa.repository.JpaRepository;

public interface EmployeeDao extends JpaRepository<Employee,Integer> {

}
```

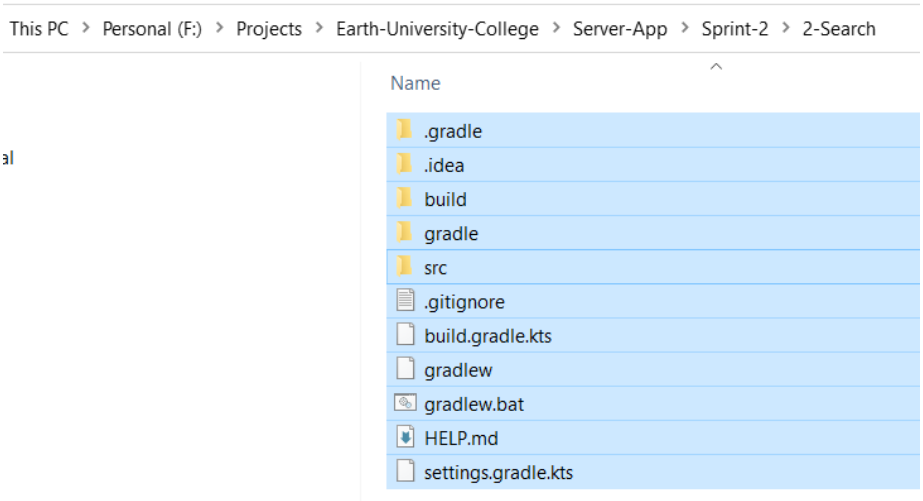


Entities ware auto generated and have not done any modification

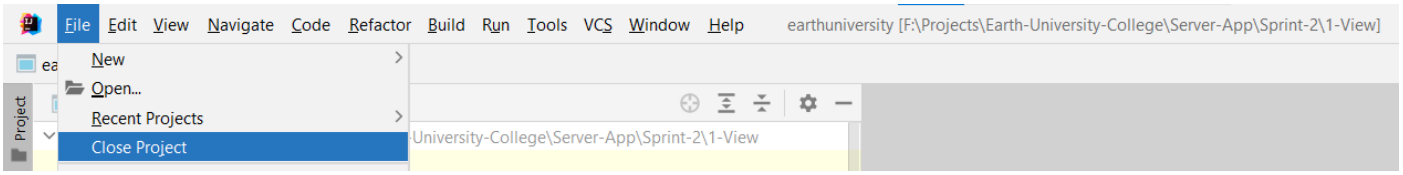
2. Copy the Content of “1-View” into “2-Search”

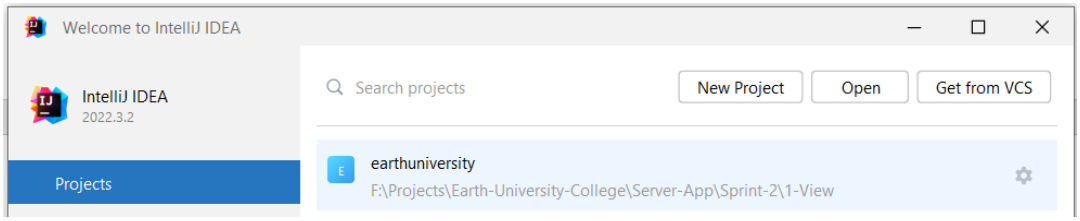






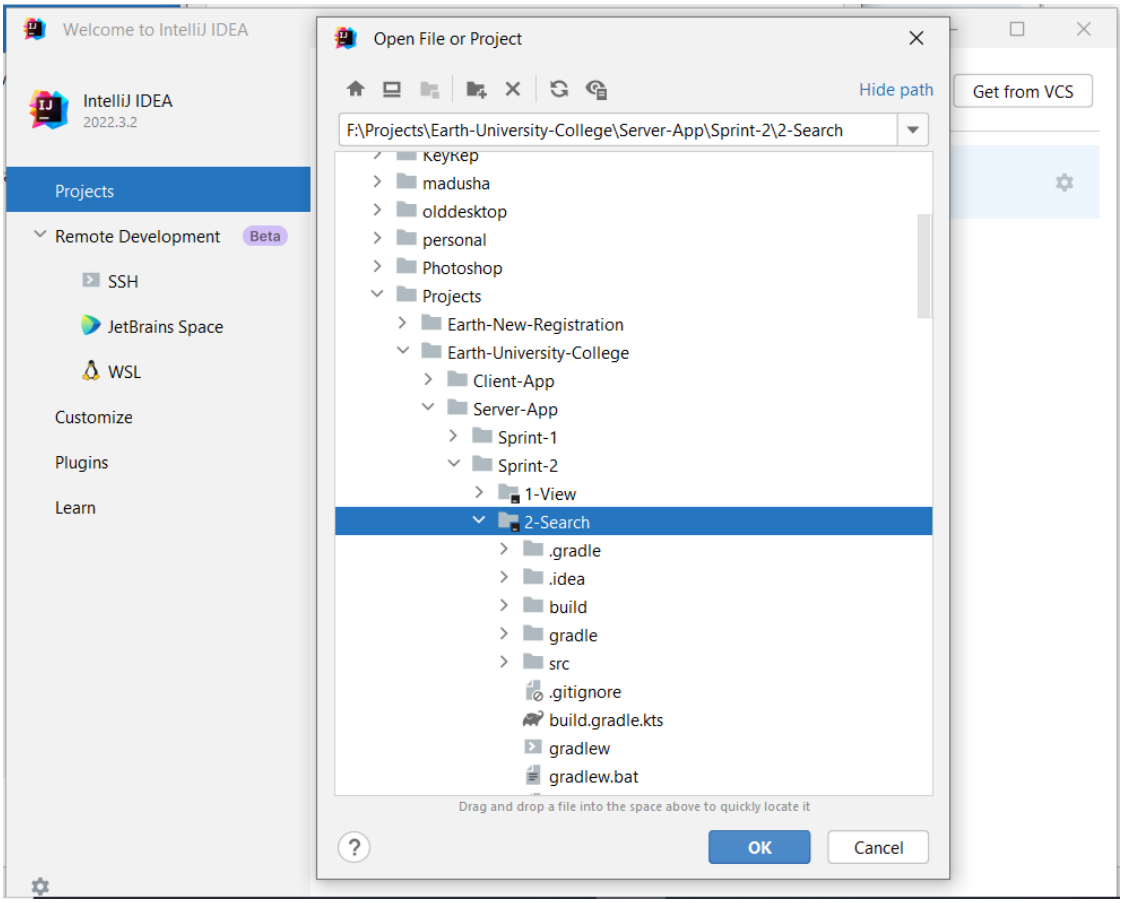
3. Open the IntelliJ-IDEA Project from the content in the “2-Search” folder

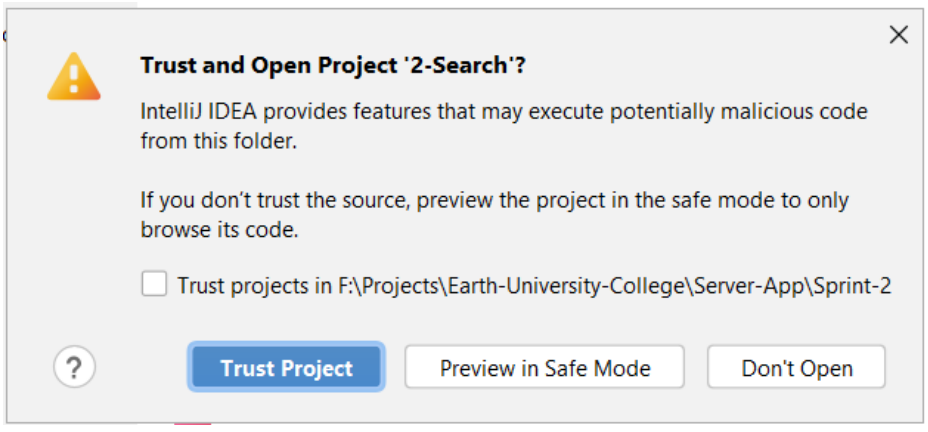




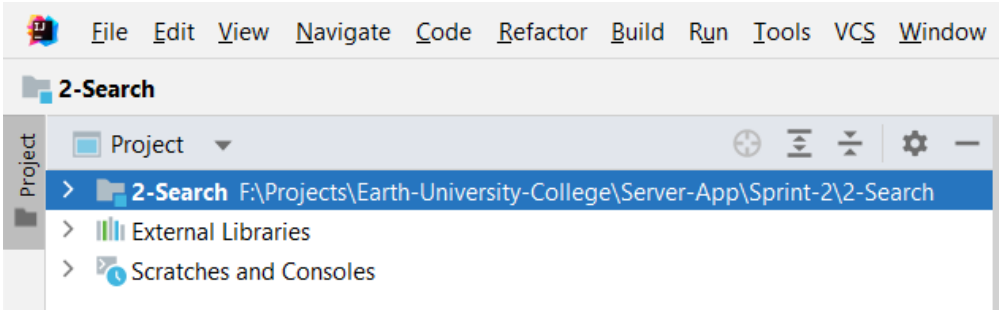
Click on Open

Select the Project and Click OK

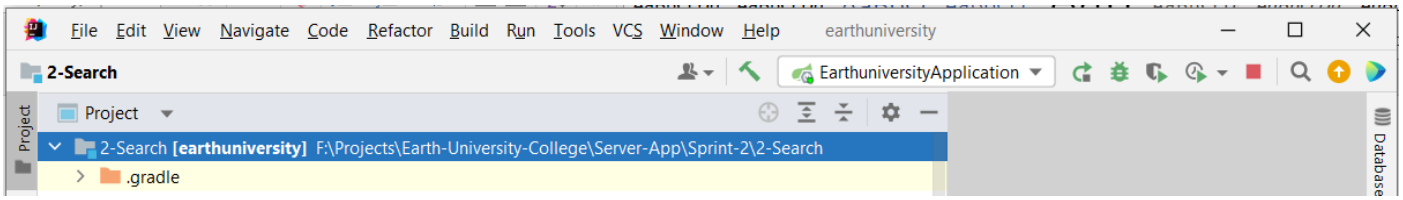


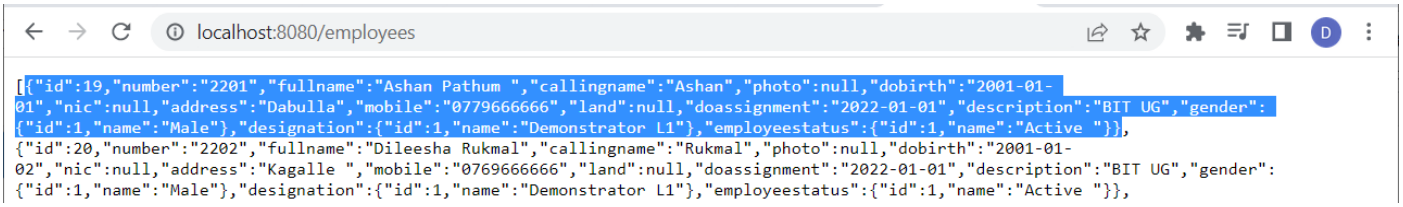


Click on “Trust Project”



Run the Project and Test the Output as follows





==========

2(c) Employee Search

4. Design Search Criteria
   - **Define Search Attributes**
   - **Define Lists Required for Search Panels**
   - Define SQL Quarries Needed

5. Server-App Modification

   Implement Listing Services
   Implement Searching Service
   - **(5) Using Controller Filtering**
   - (6) Search by QBE
   - (7) Search by Named Queries
   - (8) Search by Native Queries

6. Client-App Modification

==========

Define Search Attributes required Object Lists for Search Combos

1	number	Exact Search	
2	fullname	Like Search	
3	nic	Like Search	
4	designation	designation.id	Designation List
5	gender	gender.id	Gender List

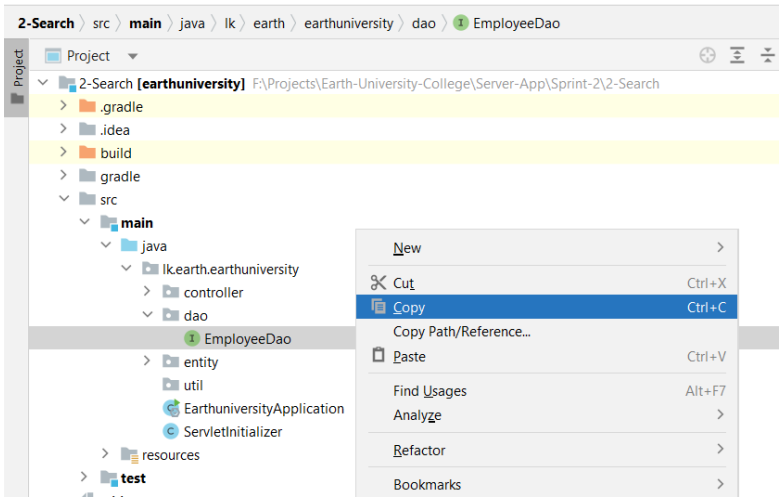
Object Lists – Vertically projected attribute list mostly with the “id” and “name” attributes.
These lists are important for Combo Boxes in both “Server-Search” & “Add-New” Forms.

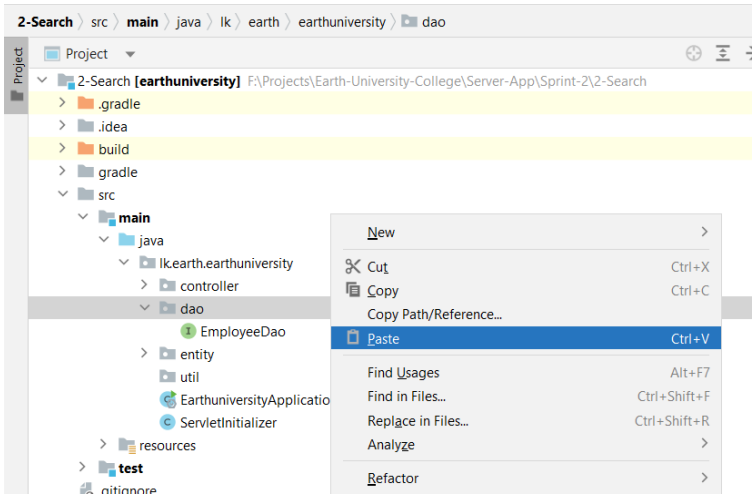
(1) – Implementing Listing Service
(2) – Implementing Searching Service

(1) Implement “listing” services

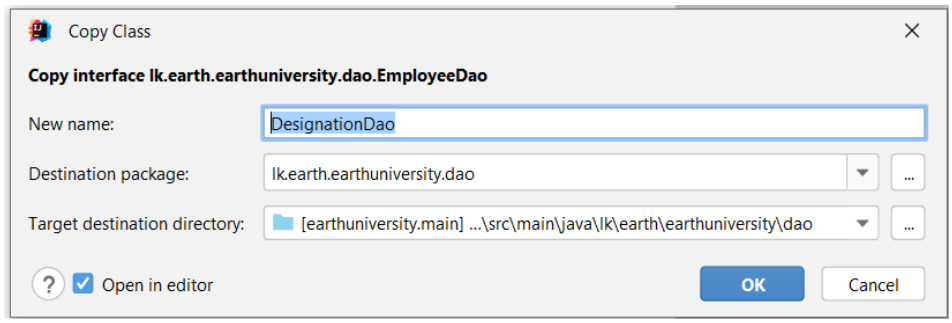
(a) DAO – DesignationDao, GenderDao
(b) Controller – DesignationController, GenderController
(c) Testing “genders/list”, “designation/list”

(a) DAO – DesignationDao, GenderDao

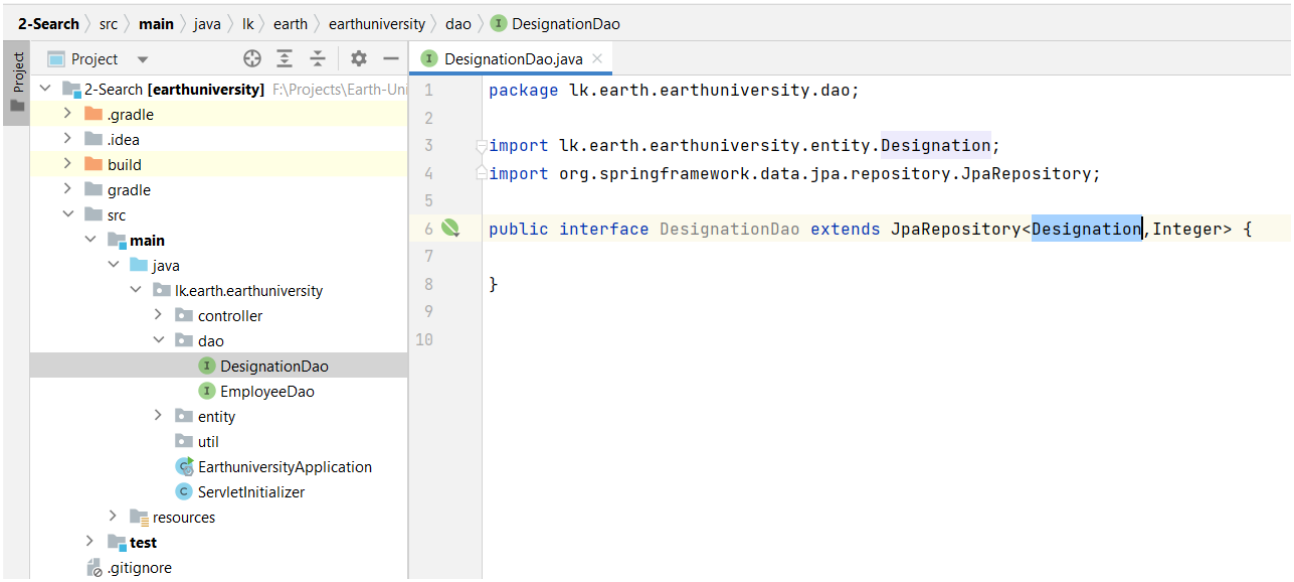




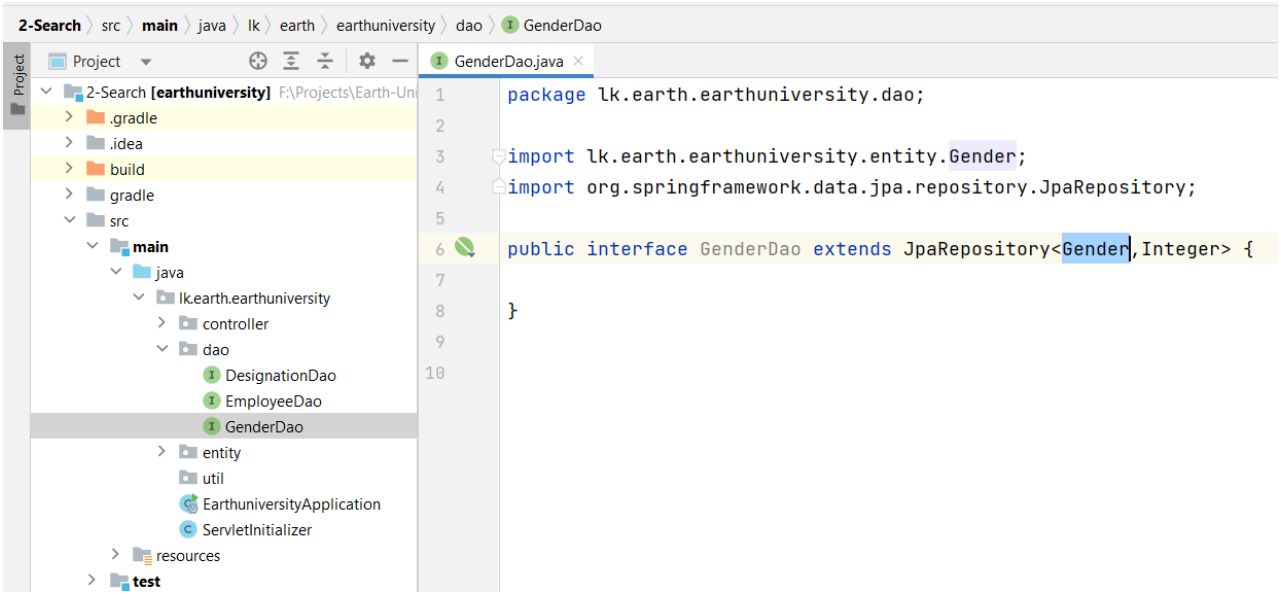
Change “EmployeeDao” to “DesignationDao”



Change the “Employee” into “Designation”



Repeat the Process to create “GenderDao” as same as the above



(b) Controller – DesignationController, GenderController

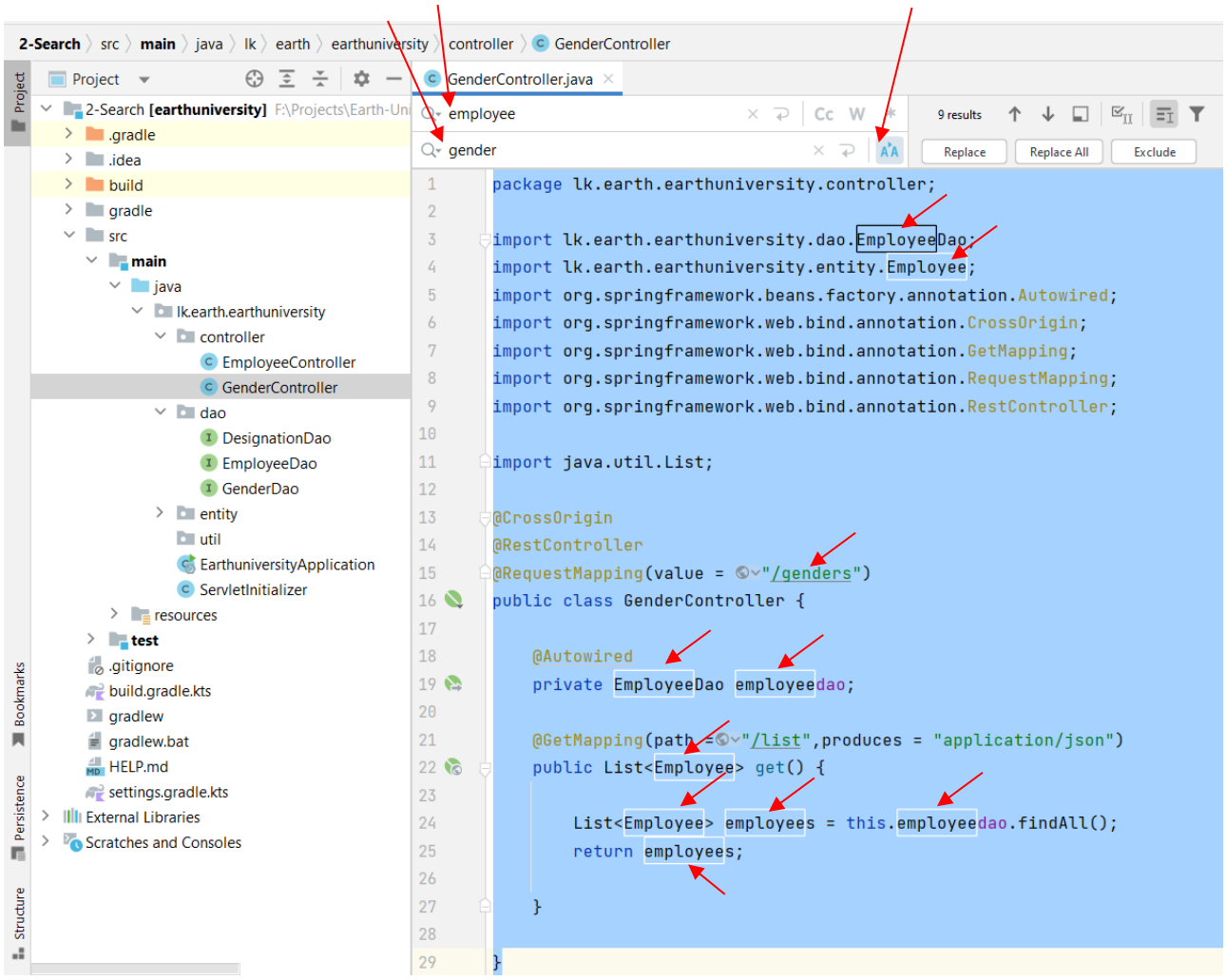
Copy the “EmployeeController” and Paste as “GenderController”
Change the codes in the “GenderController”.

Select the Codes
Click “Ctrl” + “R” → Find & Replace

Coding use shortest time period of the Project when you a member of a Software Firm with High CMM Level.

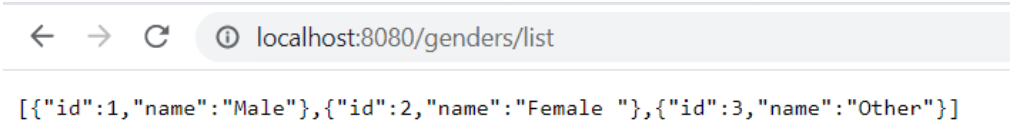
Use automotive tools and techniques as far as possible to reduce coding time.

Here, Expected time to code a Module must get less than 6Hours.

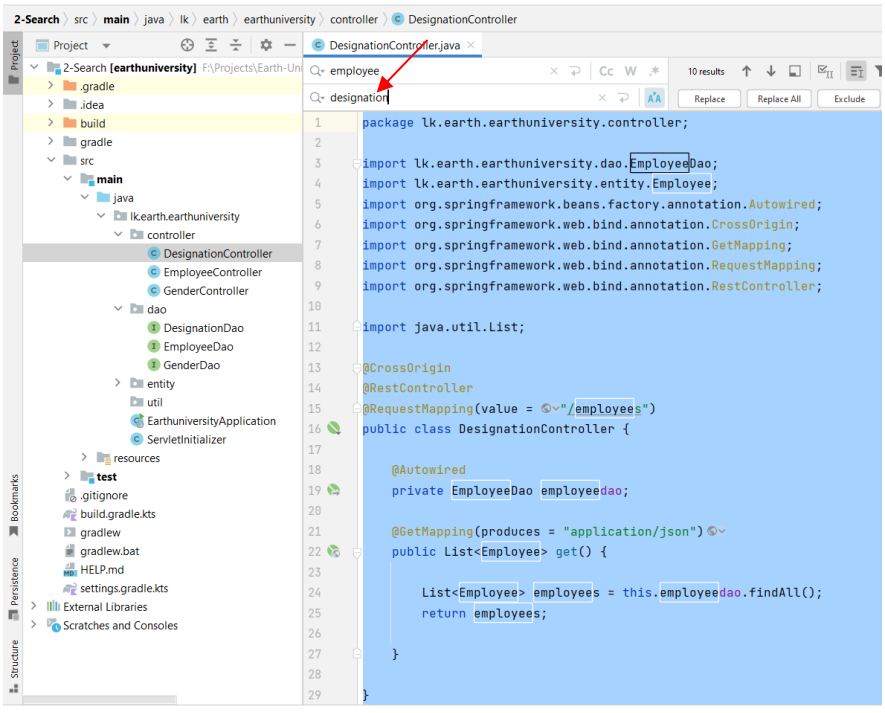


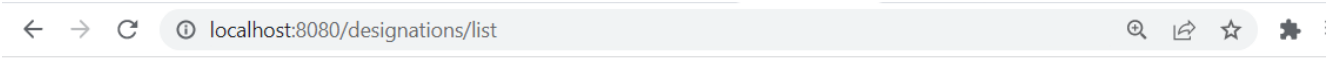
Replace All. Add → path="/list" into the @GetMapping annotation

Run the Project and Test as follows



Repeat the Process with DesignationController and test as the following

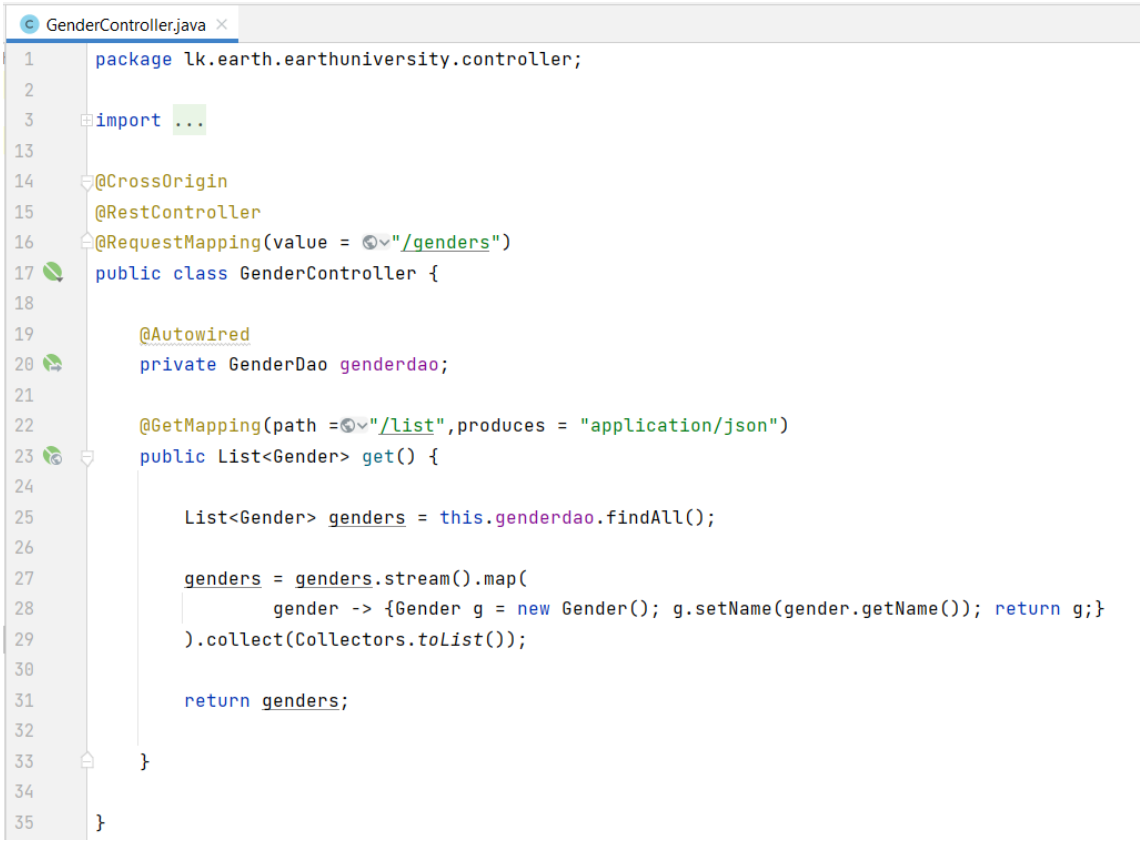




[{"id":1,"name":"Demonstrator L1"},{"id":2,"name":"Assistent G1"},{"id":3,"name":"Assistent G2"},
{"id":4,"name":"Demonstrtaor L2"},{"id":5,"name":"Demonstrator L3"}]

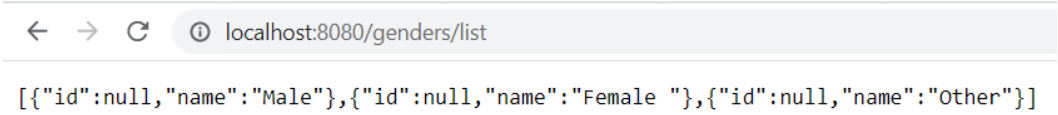
Change the controller to project only the attribute “name” as follows.

```java
@RestController
@RequestMapping(value = "/genders")
public class GenderController {

    @Autowired
    private GenderDao genderdao;

    @GetMapping(path ="/list",produces = "application/json")
    public List<Gender> get() {

        List<Gender> genders = this.genderdao.findAll();
        return genders;

    }

}
```



```java
package lk.earth.earthuniversity.controller;

import ...

@CrossOrigin
@RestController
@RequestMapping(value = "/genders")
public class GenderController {

    @Autowired
    private GenderDao genderdao;

    @GetMapping(path ="/list",produces = "application/json")
    public List<Gender> get() {

        List<Gender> genders = this.genderdao.findAll();

        genders = genders.stream().map(
                gender -> {Gender g = new Gender(); g.setName(gender.getName()); return g;}
        ).collect(Collectors.toList());

        return genders;

    }

}
```

Observe the outcome

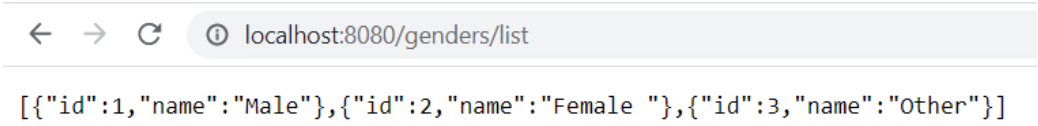


[{"id":null,"name":"Male"},{"id":null,"name":"Female "},{"id":null,"name":"Other"}]

Change the code to get Both ID and Name.

```
genders = genders.stream().map(
        gender -> { Gender g = new Gender();
                    g.setId(gender.getId());
                    g.setName(gender.getName());
                    return g; }
).collect(Collectors.toList());
```

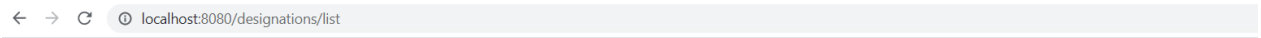
Re run the Project and observe the outcome.




```
[{"id":1,"name":"Male"},{"id":2,"name":"Female "},{"id":3,"name":"Other"}]
```


Do the Same Changers in Designation Controller and Test the outcome

```
@GetMapping(path ="/list", produces = "application/json")
public List<Designation> get() {

    List<Designation> designations = this.designationdao.findAll();

    designations = designations.stream().map(
            designation -> { Designation d = new Designation();
                d.setId(designation.getId());
                d.setName(designation.getName());
                return d; }
    ).collect(Collectors.toList());

    return designations;

}
```




```
[{"id":1,"name":"Demonstrator L1"},{"id":2,"name":"Assistent G1"},{"id":3,"name":"Assistent G2"},
{"id":4,"name":"Demonstrtaor L2"},{"id":5,"name":"Demonstrator L3"}]
```


(2) Implementing Searching Service

Using Query Parameters (**employees?designation=1&&number=2201**) ← Method used by our project

Using Path Parameters (**employees/designation/1/number/2201**)

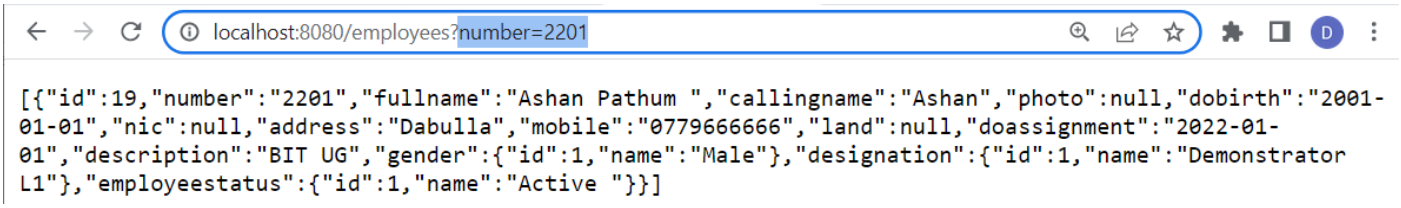
(a) Convert the Get method of the Controller as follows to get Search by Number Facility,

```
@GetMapping(produces = "application/json")
public List<Employee> get() {

    List<Employee> employees = this.employeedao.findAll();
    return employees;

}
```

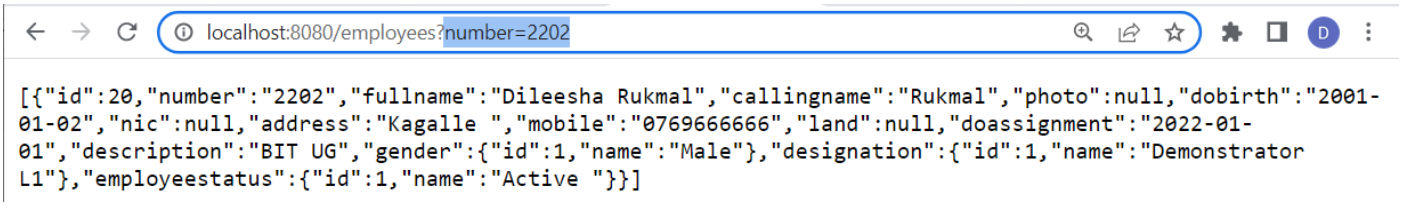
→

```java
@GetMapping(produces = "application/json")
public List<Employee> get(@RequestParam HashMap<String, String> params) {

    String number = params.get("number");

    List<Employee> employees = this.employeedao.findAll();

    if(params.isEmpty())  return employees;

    employees = employees.stream().filter(
            employee -> {
                if(number!=null) return employee.getNumber().equals(number);
                return false;

            }).collect(Collectors.toList());

    return employees;

}
```

Re Run the Project and Observe the Outcome

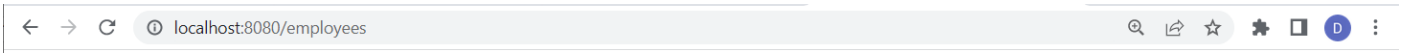



```
[{"id":19,"number":"2201","fullname":"Ashan Pathum ","callingname":"Ashan","photo":null,"dobirth":"2001-
01-01","nic":null,"address":"Dabulla","mobile":"0779666666","land":null,"doassignment":"2022-01-
01","description":"BIT UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator
L1"},"employeestatus":{"id":1,"name":"Active "}}]
```


```
[{"id":20,"number":"2202","fullname":"Dileesha Rukmal","callingname":"Rukmal","photo":null,"dobirth":"2001-
01-02","nic":null,"address":"Kagalle ","mobile":"0769666666","land":null,"doassignment":"2022-01-
01","description":"BIT UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator
L1"},"employeestatus":{"id":1,"name":"Active "}}]
```

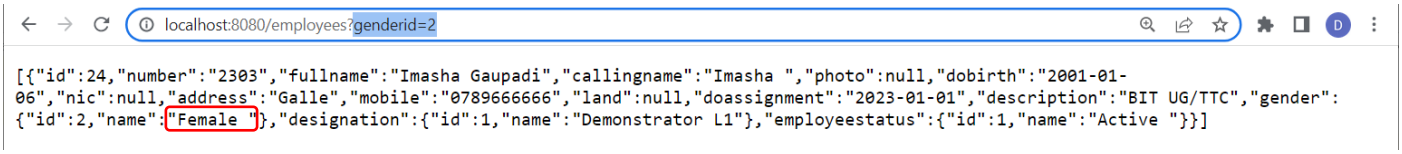

Regression Testing




```
[{"id":19,"number":"2201","fullname":"Ashan Pathum ","callingname":"Ashan","photo":null,"dobirth":"2001-01-
01","nic":null,"address":"Dabulla","mobile":"0779666666","land":null,"doassignment":"2022-01-01","description":"BIT
UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":20,"number":"2202","fullname":"Dileesha Rukmal","callingname":"Rukmal","photo":null,"dobirth":"2001-01-
02","nic":null,"address":"Kagalle ","mobile":"0769666666","land":null,"doassignment":"2022-01-01","description":"BIT
UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":21,"number":"2203","fullname":"Thahir Imran","callingname":"Imran ","photo":null,"dobirth":"2001-01-
03","nic":null,"address":"Waththala","mobile":"0719666666","land":null,"doassignment":"2022-11-01","description":"BIT
UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":22,"number":"2301","fullname":"Lakshan Ruwinda","callingname":"Lakshan ","photo":null,"dobirth":"2001-01-
04","nic":null,"address":"Kirindiwala","mobile":"0729666666","land":null,"doassignment":"2023-01-01","description":"BIT
UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":23,"number":"2302","fullname":"Arshad Ahamad","callingname":"Arshad","photo":null,"dobirth":"2001-01-
05","nic":null,"address":"Galle ","mobile":"0759666666","land":null,"doassignment":"2023-01-01","description":"BIT/BSC
UG","gender":{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":24,"number":"2303","fullname":"Imasha Gaupadi","callingname":"Imasha ","photo":null,"dobirth":"2001-01-
06","nic":null,"address":"Galle","mobile":"0789666666","land":null,"doassignment":"2023-01-01","description":"BIT
UG/TTC","gender":{"id":2,"name":"Female "},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":
{"id":1,"name":"Active "}}]
```

(3) Convert the above Controller to Filter employees both based on “gender” and “number”

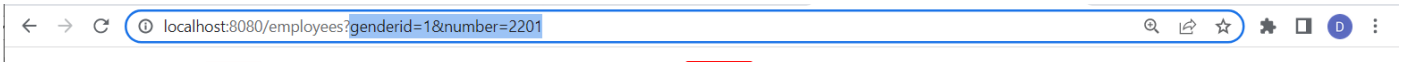
```
@GetMapping(produces = "application/json")
public List<Employee> get(@RequestParam HashMap<String, String> params) {

    String number = params.get("number");
    String genderid= params.get("genderid");

    List<Employee> employees = this.employeedao.findAll();

    if(params.isEmpty())  return employees;

    employees = employees.stream().filter(
            employee -> {
                if(number!=null) return employee.getNumber().equals(number);
                if(genderid!=null) return employee.getGender().getId()==Integer.parseInt(genderid);
                return false;

            }).collect(Collectors.toList());

    return employees;

}
```




```
[{"id":24,"number":"2303","fullname":"Imasha Gaupadi","callingname":"Imasha ","photo":null,"dobirth":"2001-01-
06","nic":null,"address":"Galle","mobile":"0789666666","land":null,"doassignment":"2023-01-01","description":"BIT UG/TTC","gender":
{"id":2,"name":"Female "},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}}]
```



```
[{"id":19,"number":"2201","fullname":"Ashan Pathum ","callingname":"Ashan","photo":null,"dobirth":"2001-01-
01","nic":null,"address":"Dabulla","mobile":"0779666666","land":null,"doassignment":"2022-01-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":20,"number":"2202","fullname":"Dileesha Rukmal","callingname":"Rukmal","photo":null,"dobirth":"2001-01-
02","nic":null,"address":"Kagalle ","mobile":"0769666666","land":null,"doassignment":"2022-01-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":21,"number":"2203","fullname":"Thahir Imran","callingname":"Imran ","photo":null,"dobirth":"2001-01-
03","nic":null,"address":"Waththala","mobile":"0719666666","land":null,"doassignment":"2022-11-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":22,"number":"2301","fullname":"Lakshan Ruwinda","callingname":"Lakshan ","photo":null,"dobirth":"2001-01-
04","nic":null,"address":"Kirindiwala","mobile":"0729666666","land":null,"doassignment":"2023-01-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}},
{"id":23,"number":"2302","fullname":"Arshad Ahamad","callingname":"Arshad","photo":null,"dobirth":"2001-01-
05","nic":null,"address":"Galle ","mobile":"0759666666","land":null,"doassignment":"2023-01-01","description":"BIT/BSC UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}}]
```


```
[{"id":19,"number":"2201","fullname":"Ashan Pathum ","callingname":"Ashan","photo":null,"dobirth":"2001-01-
01","nic":null,"address":"Dabulla","mobile":"0779666666","land":null,"doassignment":"2022-01-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}}]
```



```
[{"id":20,"number":"2202","fullname":"Dileesha Rukmal","callingname":"Rukmal","photo":null,"dobirth":"2001-01-
02","nic":null,"address":"Kagalle ","mobile":"0769666666","land":null,"doassignment":"2022-01-01","description":"BIT UG","gender":
{"id":1,"name":"Male"},"designation":{"id":1,"name":"Demonstrator L1"},"employeestatus":{"id":1,"name":"Active "}}]
```

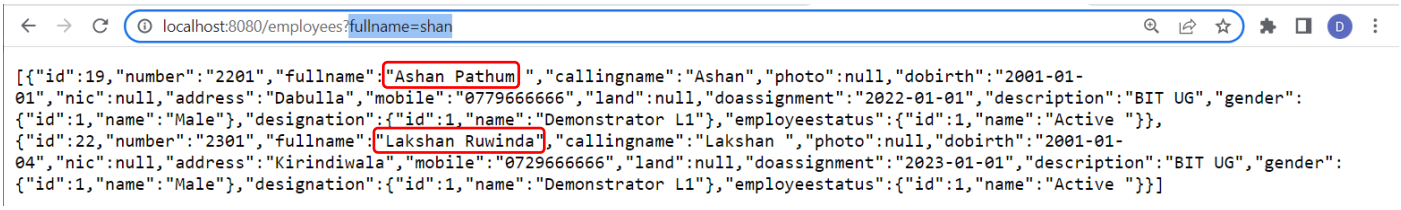

Regression Testing : Test all possible combinations including all employees

- All,
- By Number,
- By Gender ID,
- By Number & Gender ID

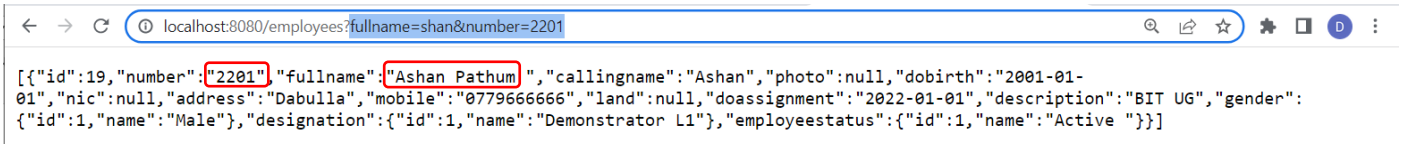
(4) Convert the above Controller to Filter employees both based on “gender”, “number” & “fullname”

```
@GetMapping(produces = "application/json")
public List<Employee> get(@RequestParam HashMap<String, String> params) {

    String number = params.get("number");
    String genderid= params.get("genderid");
    String fullname= params.get("fullname");

    List<Employee> employees = this.employeedao.findAll();

    if(params.isEmpty())  return employees;

    employees = employees.stream().filter(
            employee -> {
                if(number!=null) return employee.getNumber().equals(number);
                if(genderid!=null) return employee.getGender().getId()==Integer.parseInt(genderid);
                if(fullname!=null) return employee.getFullname().contains(fullname);
                return false;
            }).collect(Collectors.toList());

    return employees;

}
```

By Fullname



By Fullname & Number



Regression Testing – Run all the previous testing relevant to the change

- All,
- By Number,
- By Gender ID,
- By Fullname
- By Number & Gender ID
- By Number & Fullname
- By Gender ID & Fullname
- By Gender ID & Fullname & Number

(5) Repeat the Process with Designation ID

Use Copy and Paste the most relevant code segment strategy to enhance coding speed

```
String number = params.get("number");
String genderid= params.get("genderid");
String fullname= params.get("fullname");
String designationid= params.get("designationid");
```

```
employee -> {
    if(number!=null) return employee.getNumber().equals(number);
    if(genderid!=null) return employee.getGender().getId()==Integer.parseInt(genderid);
    if(fullname!=null) return employee.getFullname().contains(fullname);
    if(designationid!=null) return employee.getDesignation().getId()==Integer.parseInt(designationid);
    return false;
}).collect(Collectors.toList());
```

(6) Repeat the Process with NIC

(Before testing with NIC, insert values into NIC column in the Database if do not have)

```
String number = params.get("number");
String genderid= params.get("genderid");
String fullname= params.get("fullname");
String designationid= params.get("designationid");
String nic= params.get("nic");
```

```
employee -> {
    if(number!=null) return employee.getNumber().equals(number);
    if(genderid!=null) return employee.getGender().getId()==Integer.parseInt(genderid);
    if(fullname!=null) return employee.getFullname().contains(fullname);
    if(designationid!=null) return employee.getDesignation().getId()==Integer.parseInt(designationid);
    if(nic!=null) return employee.getFullname().contains(nic);
    return false;
}).collect(Collectors.toList());
```

The above algorithm suffers from Bugs as the predicate will be applied separately and if one predicate get true, the entire search will get true. This can be solved using 2 techniques

(1) Using Boolean controlled such as && operators

(2) Using Pipe & Filter Design Pattern (Enhanced Layard) on a continues filtered stream ← Best Solution

```
@GetMapping(produces = "application/json")
public List<Employee> get(@RequestParam HashMap<String, String> params) {

    String number = params.get("number");
    String genderid= params.get("genderid");
    String fullname= params.get("fullname");
    String designationid= params.get("designationid");
    String nic= params.get("nic");

    List<Employee> employees = this.employeedao.findAll();

    if(params.isEmpty())  return employees;

    Stream<Employee> estream = employees.stream();

    if(number!=null) estream = estream.filter(e -> e.getNumber().equals(number));
    if(genderid!=null) estream = estream.filter(e -> e.getGender().getId()==Integer.parseInt(genderid));
    if(fullname!=null) estream = estream.filter(e -> e.getFullname().contains(fullname));
    if(designationid!=null) estream = estream.filter(e -> e.getDesignation().getId()==Integer.parseInt(designationid));
    if(nic!=null) estream = estream.filter(e -> e.getNic().contains(nic));

    return estream.collect(Collectors.toList());

}
```