

Springboot

Database

File *application.properties*

```
spring.datasource.url=jdbc:mysql://localhost:3306/
    department_employee
spring.datasource.username=root
spring.datasource.password=root
spring.jpa.hibernate.ddl-auto=update
debug=true
```

Model

Department(1)

```
@Entity
@Table(name="departments")
public class Department {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)

    Long id;
    String name;
    String faculty;
    String location;

    @OneToMany(mappedBy = "department", cascade =
        CascadeType.ALL)
    private List<Department> departments = new ArrayList<>()
        ;
    ...
}
```

Employee(M)

```
@Entity
@Table(name="employees")
public class Employee {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)

    Long id;
    String name;
    String lastname;
    Double salary;

    @ManyToOne(cascade = CascadeType.ALL)
    @JoinColumn(name="department")
    Department department;
    ...
}
```

Repository

Department(1)

```
public interface DepartmentRepository extends JpaRepository<
    Department, Long>{}
```

Employee(M)

```
public interface EmployeeRepository extends JpaRepository<
    Employee, Long>{

    default void increaseSalary(Long id, Integer percentage)
    {
        Employee e = findById(id).orElse(null);
        e.setSalary(e.getSalary() * (1+percentage/100.0));
        this.save(e);
    }
}
```

```

/* UTILITIES
List<Product> findByGiacenzaGreaterThan(Integer
    giacenza);

default void decreaseAllPrices(Integer percentage) {
    findAll().stream()
        .forEach(p -> {
            p.setPrezzo(p.getPrezzo() * (1 -
                percentage/100.0));
            this.save(p);
        });
}

default void decreaseStock(Long id) {
    Product product = this.findById(id).get();
    product.setGiacenza(product.getGiacenza() - 1);
    this.save(product);
}
*/

```

View

Welcome.html

```

<html>
  <body>
    <h2>Welcome</h2>
    <a href="/departments"><button>Departments</button><
      /a>
    <a href="/employees"><button>Employees</button></a>
  </body>
</html>

```

Department(1)

Index.html

```
<html>
  <body>
    <h2>Index</h2>
    <form th:each="d:${departments}">
      <p>Name: <span th:text="${d.name}"></span></p>
      <button formaction="/departments/show">Show</button>
      <button formaction="/departments/edit">Edit</button>
      <input type="hidden" name="id" th:value="${d.id}">
    </form>
    <br>
    <form action="/departments/create">
      <button>Create new deparment</button>
    </form>
    <br><a href="/"><button>Back at Welcome</button></a>
  </body>
</html>
```

Create.html

```
<html>
  <body>
    <h2>Create</h2>
    <form action="/departments/store" method="POST">
      Name:<input name="name" required>
      Faculty:<input name="faculty" required>
      Location:<input name="location" required>
      <button>Create</button>
    </form>

    <br><a href="/departments"><button>Back at Home</button></a>
  </body>
</html>
```

Show.html

```
<html>
  <body>
    <h2>Show</h2>

    <p>ID: <span th:text="${d.id}"></span></p>
    <p>Name: <span th:text="${d.name}"></span></p>
    <p>Faculty: <span th:text="${d.faculty}"></span></p>
    <p>Location: <span th:text="${d.location}"></span></p>

    <br>
    <form action="/departments/destroy" method="POST">
      <button>Delete</button>
      <input type="hidden" name="id" th:value="${d.id}">
    </form>

    <br><a href="/departments"><button>Back at Home</button></a>
  </body>
</html>
```

Edit.html

```
<html>
  <body>
    <h2>Edit</h2>
    <form action="/departments/update" method="POST" th:object="${d}">
      Name: <input name="name" th:field="*{name}"><br>
      Faculty: <input name="faculty" th:field="*{faculty}"><br>
      Location: <input name="location" th:field="*{location}"><br><br>
      <button>Update</button>
      <input type="hidden" name="id" th:field="*{id}">
    </form>
    <br><a href="/departments"><button>Back at Home</button></a>
  </body>
</html>
```

Employee(M)

Index.html

```
<html>
  <body>
    <h2>Index</h2>
    <form th:each="e:${employees}">
      <p>Name: <span th:text="${e.name}"></span></p>
      <button formation="/employees/show">Show</button>
      <button formation="/employees/edit">Edit</button>
      <button formation="/employees/increaseSalary">
        Increase Salary of 20%</button>
      <input type="hidden" name="id" th:value="${e.id}">
      <input type="hidden" name="percentage" th:value="20">
    </form>
    <br>
    <form action="/employees/create">
      <button>Create new employee</button>
    </form>
    <br><a href="/"><button>Back at Welcome</button></a>
  </body>
</html>
```

Create.html

```
<html>
  <body>
    <h2>Create</h2>
    <form action="/employees/store" method="POST">
      Name:<input name="name" required>
      Lastname:<input name="lastname" required>
      Salary:<input name="salary" required>
      Department:
      <select name="department" required>
        <div th:each="d:${departments}">
          <option th:value="${d.id}"><span th:text="${d.name}"></span></option>
        </div>
      </select>
      <button>Create</button>
    </form>

    <br><a href="/employees"><button>Back at Home</button></a>
  </body>
</html>
```

Show.html

```
<html>
  <body>
    <h2>Show</h2>

    <p>ID: <span th:text="${e.id}"></span></p>
    <p>Name: <span th:text="${e.name}"></span></p>
    <p>Salary: <span th:text="${e.salary}"></span></p>
    <p>Department: <span th:text="${e.department.name}">
      </span></p>

    <br>
    <form action="/employees/destroy" method="POST">
      <button>Delete</button>
      <input type="hidden" name="id" th:value="${e.id}"
        ">
    </form>

    <br><a href="/employees"><button>Back at Home</
      button></a>
  </body>
</html>
```

Edit.html

```
<html>
  <body>
    <h2>Edit</h2>
    <form action="/employees/update" method="POST" th:
      object="${e}">
      Name: <input name="name" th:field="*{name}"><br>
      Lastname: <input name="lastname" th:field="*{
        lastname}"><br>
      Salary: <input name="salary" th:field="*{salary}"
        "><br>
      Department:
      <select name="department">
        <div th:each="d:${departments}">
          <option th:value="${d.id}"><span th:text="${
            d.name}"></span></option>
        </div>
      </select><br><br>
      <button>Update</button>
      <input type="hidden" name="id" th:field="*{id}">
    </form>
    <br><a href="/employees"><button>Back at Home</
      button></a>
  </body>
</html>
```

Controller

Department(1)

```
@Controller
public class DepartmentController {
    private final DepartmentRepository repo;

    public DepartmentController(DepartmentRepository repo){
        this.repo = repo;
    }

    @GetMapping("/")
    String welcome(){
        return "/welcome";
    }

    @GetMapping("/departments")
    String index(Model model){
        model.addAttribute("departments", repo.findAll());
        return "/departments/index";
    }

    @GetMapping("/departments/create")
    String create(){
        return "/departments/create";
    }

    @PostMapping("/departments/store")
    String store(Department department){
        repo.save(department);
        return "redirect:/departments";
    }

    @GetMapping("/departments/show")
    String show(Model model, Long id){
        model.addAttribute("d", repo.findById(id).orElse(
            null));
        return "/departments/show";
    }

    @GetMapping("/departments/edit")
    String edit(Model model, Long id){
        model.addAttribute("d", repo.findById(id).orElse(
            null));
        return "/departments/edit";
    }
}
```



```

@PostMapping("/departments/update")
String update(Department department){
    repo.save(department);
    return "redirect:/departments";
}

@PostMapping("/departments/destroy")
String destroy(Department department){
    repo.deleteById(department.getId());
    return "redirect:/departments";
}
}

```

Employee(M)

```

@Controller
public class EmployeeController {

    private final EmployeeRepository repo;
    private final DepartmentRepository departmentRepo;

    public EmployeeController(EmployeeRepository repo,
        DepartmentRepository departmentRepo) {
        this.repo = repo;
        this.departmentRepo = departmentRepo;
    }

    @GetMapping("/employees")
    String index(Model model){
        model.addAttribute("employees", repo.findAll());
        return "employees/index";
    }

    @GetMapping("/employees/create")
    String create(Model model){
        model.addAttribute("departments", departmentRepo.findAll());
        return "employees/create";
    }

    @PostMapping("/employees/store")
    String store(Employee employee){
        repo.save(employee);
        return "redirect:/employees";
    }
}

```

```

@GetMapping("/employees/show")
String show(Model model, Long id){
    model.addAttribute("e", repo.findById(id).orElse(
        null));
    return "/employees/show";
}

@GetMapping("/employees/edit")
String edit(Model model, Long id){
    model.addAttribute("departments", departmentRepo.
        findAll());
    model.addAttribute("e", repo.findById(id).orElse(
        null));
    return "/employees/edit";
}

@PostMapping("/employees/update")
String update(Employee employee){
    repo.save(employee);
    return "redirect:/employees";
}

@PostMapping("/employees/destroy")
String destroy(Employee employee){
    repo.deleteById(employee.getId());
    return "redirect:/employees";
}

@GetMapping("/employees/increaseSalary")
String increaseSalary(Integer percentage, Model model,
    Long id){
    repo.increaseSalary(id, percentage);
    return show(model, id);
}
}

```