Java MVC Application

Part II Custom MVC Framework - Integrating with Database

1. Modificati metoda *getOneEmployee()* din clasa *EmployeeController.java,*  astfel incat sa returneze o instanta a clasei *Employee.java*.
2. Adaugati in proiect urmatoarea dependinta:

<dependency>

<groupId>org.codehaus.jackson</groupId>

<artifactId>jackson-jaxrs</artifactId>

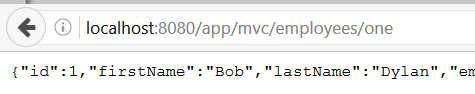
<version>${jackson-jaxrs.version}</version>

</dependency>

Unde jackson-jaxrs.version = 1.9.13

1. Modificati DispatcherServlet-ul astfel incat sa returneze catre client raspunsul in format JSON.

TEST: <http://localhost:8080/app/mvc/employees/one>



1. In continuare, ne propunerm sa modificam metoda *getAllEmployees()* din clasa *EmployeeController.java* pentru a intoarce lista angajatilor din baza de date. Pentru a respecta architectura Onion, vom introduce un layer suplimentar intre Application Controller si DAO, layer-ul de Service.
   1. In pachetul ro.teamnet.zth.appl.service creati interfata EmployeeService.java. Interfata va contine metoda findAllEmployees() si va returna o lista de angajati.
   2. In pachetul ro.teamnet.zth.appl.service creati clasa *EmployeeServiceImpl.java,* care implementeaza interfata *EmployeeService.java*. Metoda *findAllEmployees()* va returna toti utilizatorii din baza de date.
   3. Modificati metoda *getAllEmployees()* din clasa *EmployeeController.java* astfel incat sa returneze toti angajatii din baza de date, trecand prin layerul de service al aplicatiei.
2. (Pentru acasa) Modificati metoda *getAllDepartments()* din clasa *DepartmentController.java* astfel incat sa returneze lista de departamente din baza de date. Procedati ca in exercitiul anterior. In mod similar, creati si utilizati layerul de service aferent procesarilor de forma “/jobs”, “/locations”.
3. Realizati implementarile necesare pentru a cauta un angajat in baza de date. Se vor urma pasii de mai jos:
   1. In interfata *EmployeeService.java* adaugati metoda *findOneEmployee.* Aceasta va primi ca parametru id-ul unui angajat si va returna inregistrarea aferenta din baza de date. Implementati metoda.
   2. Modificati metoda *getOneEmployee()* din clasa *EmployeeController.java* astfel incat sa utilizeze layerul de service implementat la punctul a.
   3. Modificati metoda *getOneEmployee()* din clasa *EmployeeController.java* astfel incat sa primeasca un parametru care sa specifice id-ul angajatului cautat. Cum se poate transmite din browser acest id catre serverul nostru?
4. Modificati DispatcherServlet-ul astfel incat, daca primeste parametri pe request, ca in exemplul de mai jos, sa transmita acesti parametri metodei corespunzatoare din controller.

TEST: <http://localhost:8080/app/mvc/employees/one?id=1>

1. Adaugati metoda *deleteOneEmployee()* in clasa *EmployeeController.java*. Implementati metoda astfel incat sa stearga un angajat din baza de date. Id-ul angajatului este primit ca parametru. Respectati layerele aplicatiei (controller – service – dao) la implementarea functionalitatii. Ce metoda HTTP va folosi browserul pentru a trimite solicitarea de stergere a unui angajat catre serverul de aplicatie? Cum va trata DispatcherServlet acest request?
2. Ce s-ar intampla daca am incerca sa transmitem request-uri cu metode HTTP diferite catre acelasi URL? Testati acest scenariu cu metoda DELETE. Actualizati registrul de Application Controllere pentru a tine cont si de metoda HTTP din request. Utilizati aceasta informatie in mecanismul de regasire a unui Controller, pentru a delega executarea sarcinii de business catre metoda corespunzatoare.

Hint: Pentru a testa diverse tipuri de request-uri HTTP, puteti folosi plugin-ul REST Client din IDEA. Mai multe despre acest plugin puteti citi aici: <https://www.jetbrains.com/help/idea/2016.1/testing-restful-web-services.html#composeAndSubmitRequest>

1. Pentru salvarea unui nou angajat in baza de date, este necesara transmiterea informatiilor despre noul angajat din browser catre serverul de aplicatie. Ce metoda HTTP va fi folosita in acest caz? Cum putem transmite informatiile? (tip metoda HTTP si locul incapsularii datelor in pachetul de date).
2. In clasa *EmployeeController.java*, adaugati metoda *saveEmployee* care sa insereze un angajat.
3. Gasiti o modalitate prin care DispatcherServletul, in etapa de dispatch, poate transfera informatiile primite la punctul **10** catre metoda creata la punctul **11**.
4. Testati metoda de salvare a unui angajat. Cum s-ar putea actualiza informatiile despre un angajat existent?
5. (Pentru acasa) Implementati metode de cautare, stergere si salvare in cadrul controllerelor asociate entitatilor Department, Job si Location. Respectati separarea layerelor (controller – service – DAO).