DESENVOLVIMENTO DE SOFTWARE PARA A WEB 1

Prof. Delano M. Beder (UFSCar)

Atividade AA-3: Sistema para oferta de vagas de estágios/empregos

Obs 1: Essa atividade deve ser baseada na atividade AA-2. Ou seja, deve-se apenas implementar os novos requisitos (funcionalidades providas em uma REST API) aqui mencionados -- levando em consideração o que já foi desenvolvido na atividade AA-2.

O sistema deve incorporar os seguintes requisitos:

- REST API -- CRUD[1] de profissionais
  - Cria um novo profissional [**C**reate - **CRUD**]

    POST http://localhost:8080/profissionais

    Body: raw/JSON (application/json)
  - Retorna a lista de profissionais [**R**ead - **CRUD**]

    GET http://localhost:8080/profissionais
  - Retorna o profissional de id = {id} [**R**ead - **CRUD**]

    GET http://localhost:8080/profissionais/{id}
  - Atualiza o profissional de id = {id} [**U**pdate - **CRUD**]

    PUT http://localhost:8080/profissionais/{id}

    Body: raw/JSON (application/json)
  - Remove o profissional de id = {id} [**D**elete - **CRUD**]

    DELETE http://localhost:8080/profissionais/{id}
- REST API -- CRUD de empresas
  - Cria uma nova empresa [**C**reate - **CRUD**]

    POST http://localhost:8080/empresas

    Body: raw/JSON (application/json)
  - Retorna a lista de empresas [**R**ead - **CRUD**]

    GET http://localhost:8080/empresas
  - Retorna a empresa de id = {id} [**R**ead - **CRUD**]

    GET http://localhost:8080/empresas/{id}
  - Retorna a lista de todas as empresas da cidade de nome = {nome}

    GET http://localhost:8080/empresas/cidades/{nome}
  - Atualiza a empresa de id = {id} [**U**pdate - **CRUD**]

    PUT http://localhost:8080/empresas/{id}

    Body: raw/JSON (application/json)
  - Remove a empresa de id = {id} [**D**elete - **CRUD**]

    DELETE http://localhost:8080/empresas/{id}

- REST API -- Retorna a lista de vagas [**R**ead - **CRUD**]

  GET http://localhost:8080/vagas
- REST API -- Retorna a locação de id = {id} [**R**ead - **CRUD**]

  GET http://localhost:8080/vagas/{id}
- REST API -- Retorna a lista de vagas (em aberto) da empresa de id = {id} [**R**ead - **CRUD**]

  GET http://localhost:8080/vagas/empresas/{id}

Obs 2: Em todas as funcionalidades mencionadas acima, não há necessidade de autenticação (login)

Dica: Na configuração do *Spring Security* utilize algo semelhante ao apresentado no código abaixo:

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.csrf().disable().authorizeRequests()
    // Controladores REST
    .antMatchers("/profissionais", "/empresas", "/vagas").permitAll()
    .antMatchers("/profissionais/{\\d+}", "/empresas/{\\d+}").permitAll()
    .antMatchers("/vagas/{\\d+}").permitAll()
    .antMatchers("/empresas/cidades/{\\w+}").permitAll()
    .antMatchers("/vagas/empresas/{\\d+}").permitAll()
    // Demais linhas
    .anyRequest().authenticated()
    .and()
        .formLogin().loginPage("/login").permitAll()
    .and()
        .logout().logoutSuccessUrl("/").permitAll();
}
```

Arquitetura: Modelo-Visão-Controlador

Tecnologias

- Spring MVC (Controladores REST), Spring Data JPA, Spring Security & Thymeleaf (Lado Servidor)

Ambiente de Desenvolvimento

- A compilaçao e o *deployment* deve ser obrigatoriamente ser realizado via *maven*.
- Os arquivos fonte do sistema devem estar hospedados obrigatoriamente em um repositório (preferencialmente github).

---

1. CRUD: **C**reate, **R**ead, **U**pdate & **D**elete. ↩