DESENVOLVIMENTO DE SOFTWARE PARA A WEB 1

Prof. Delano M. Beder (UFSCar)

Atividade AA-3: Sistema para criação de promoções de quarto de hotel em sites de reservas

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 sites de reserva
  - Cria um novo site de reserva [**C**reate - **CRUD**]

    POST http://localhost:8080/sites

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

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

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

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

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

    DELETE http://localhost:8080/sites/{id}
- REST API -- CRUD de hotéis
  - Cria uma novo hotel [**C**reate - **CRUD**]

    POST http://localhost:8080/hoteis

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

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

    GET http://localhost:8080/hoteis/{id}
  - Retorna a lista de todos os hotéis da cidade de nome = {nome}

    GET http://localhost:8080/hoteis/cidades/{nome}
  - Atualiza o hotel de id = {id} [**U**pdate - **CRUD**]

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

    Body: raw/JSON (application/json)
  - Remove o hotel de id = {id} [**D**elete - **CRUD**]
    DELETE http://localhost:8080/hoteis/{id}

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

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

  GET http://localhost:8080/promocoes/{id}
- REST API -- Retorna a lista de promoções do site de reserva de id = {id} [**R**ead - **CRUD**]

  GET http://localhost:8080/promocoes/sites/{id}
- REST API -- Retorna a lista de promoções do hotel de id = {id} [**R**ead - **CRUD**]

  GET http://localhost:8080/promocoes/hotel/{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("/sites", "/hoteis", "/promocoes").permitAll()
    .antMatchers("/sites/{\\d+}", "/hoteis/{\\d+}").permitAll()
    .antMatchers("/promocoes/{\\d+}").permitAll()
    .antMatchers("/hoteis/cidades/{\\w+}").permitAll()
    .antMatchers("/promocoes/sites/{\\d+}").permitAll()
    .antMatchers("/promocoes/hoteis/{\\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. ↩