

Java Web – Server & Client

Desenvolvimento de Aplicações Java Web

NOME DO ESTUDANTE 1

NOME DO ESTUDANTE 2

DISCIPLINA

Tecnologias para Desenvolvimento Web

1 Contextualização

Ao longo da disciplina de Tecnologias para Desenvolvimento Web, vimos exemplos de aplicações Java Web com JSF (JavaServer Faces), interagindo com tecnologias AJAX e JSON. Logo, como base nos exemplos e tutoriais já praticados, temos condições de desenvolver alterações nesses exemplos, para consolidar nossas experiências com as tecnologias mencionadas.

Para isso, é preciso que a equipe (dupla ou trio) se organize para realizar os exercícios propostos no presente documento. Ao final dos trabalhos, a equipe deverá entregar como trabalho realizado:

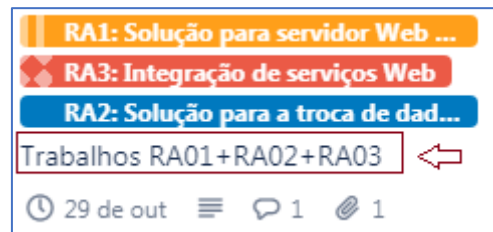
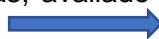
1. Os códigos-fonte dos trabalhos solicitados no GitHub, avaliado no RA5;



No Repositório do GitHub

- RA01+RA02+RA03
- Trabalho01-JSF+CRUD
- Trabalho02-JSF+AJAX
- Trabalho03-JSF+JSON

2. Criar cartão no Trello que detalhe como a equipe organizou a confecção das aplicações Java Web solicitadas, avaliado no RA4; e



RA1: Solução para servidor Web ...

RA3: Integração de serviços Web

RA2: Solução para a troca de dad...

Trabalhos RA01+RA02+RA03

29 de out 1 1

3. Três vídeos que apresentam em detalhes os exercícios desenvolvidos e funcionais, que devem ser confeccionados de acordo com o item **4 Orientações – Apresentação dos Trabalhos**, deste documento, para avaliação dos RA1+RA2+RA3.

Na sequência, detalharemos os trabalhos que deverão ser realizados pela Equipe, como deverá ser exibido em um vídeo cada um dos trabalhos executado conforme o esperado e os critérios de correção.

2 Trabalhos RA1, RA2 e RA3 – Tema 01 e Tema02

Faremos 3 trabalhos de desenvolvimento, alterando os exemplos dos tutoriais:

1. **Aplicação Java Web JSF com CRUD** – alteração no tutorial 03, no qual é solicitado o acréscimo de 2 campos na tabela **book**, da base de dados **crud**, no MySQL, e então refazer o CRUD (Create-Read-Update-Delete), de forma a contemplar os novos campos da tabela mencionada.
2. **Aplicação Java Web JSF com AJAX** – alteração no tutorial 04, no qual é solicitado o acréscimo de 2 campos no formulário de entrada, que são tratados de forma individual quando ocorre a troca de dados entre cliente e servidor.
3. **Aplicação Java Web JSF com JSON** – alteração no tutorial 05, no qual é solicitado modificar o formulário inicial para tratar dos dados usados na identificação de livro (idêntico aos campos da tabela book).

A seguir, cada novo trabalho será detalhado, indicando as alterações esperadas.

2.1 Aplicação Java Web JSF com CRUD

Alteração da aplicação JSF + CRUD pela inclusão de mais duas colunas ou campos na tabela **book**:

1. **coAuthor** (o livro tem mais um autor colaborador) e
2. **isbn** (*International Standard Book Number*, ou identificação de livros).

Passo 1) Faça essa alteração usando o **phpmyadmin** do MySQL. Inicialmente, adicione as duas colunas, conforme indicado na Figura 1.

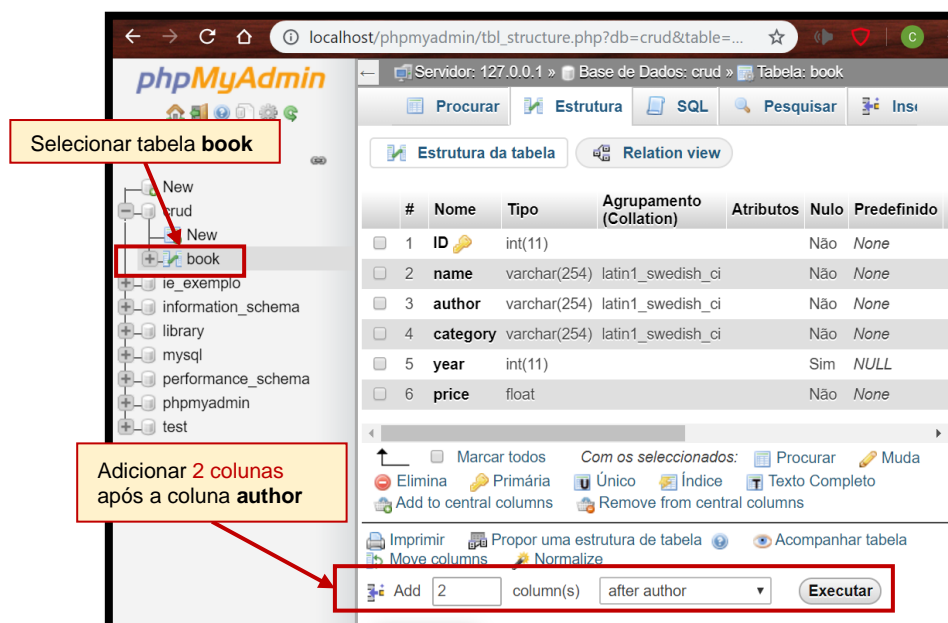


Figura 1: Adicionando 2 colunas à tabela **book**.

Passo 2) Após, configure os novos campos conforme indicado na Figura 2. Ao finalizar, verifique se a nova estrutura da tabela **book** corresponde à da Figura 3.



Figura2: Colunas adicionadas à tabela **book**.

#	Nome	Tipo	Agrupamento (Collation)	Atributos	Nulo	Predefinido	Comentários	Extra	Ações
1	ID	int(11)			Não	None		AUTO_INCREMENT	Muda Elimina Mais
2	name	varchar(254)	latin1_swedish_ci		Não	None			Muda Elimina Mais
3	author	varchar(254)	latin1_swedish_ci		Não	None			Muda Elimina Mais
4	coAuthor	varchar(100)	latin1_swedish_ci		Sim	NULL			Muda Elimina Mais
5	isbn	varchar(100)	latin1_swedish_ci		Sim	NULL			Muda Elimina Mais
6	category	varchar(254)	latin1_swedish_ci		Não	None			Muda Elimina Mais
7	year	int(11)			Sim	NULL			Muda Elimina Mais
8	price	float			Não	None			Muda Elimina Mais

Figura 3: Estrutura da tabela **book** após alteração.

Passo 3) Agora, faça a alteração nos códigos-fonte. Primeiro, **exclua** o **entity bean** e suas correspondentes **interfaces facade** do projeto (botão direito sobre os arquivos e **Delete**), conforme indicado na Figura 4.

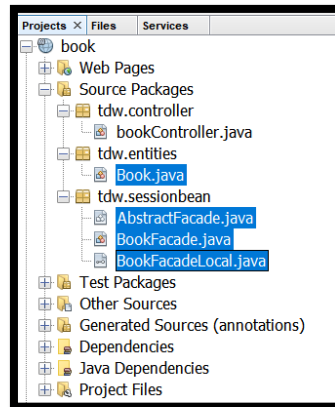


Figura 4: Exclua os arquivos indicados, que serão criados a partir na estrutura alterada da tabela **book**.

Passo 4) Após a exclusão, adicione novamente o **entity bean** pela opção **New > Entity Classes from Database**. Procure o **data source Jcrud**, do tutorial 03, e selecione a tabela **book** – mesmos passos do TDW (tutorial 03) - JSF + CRUD. Da mesma forma, adicione novamente as **facades** pela opção **New > Session Beans for Entity Classes** – mesmos passos do TDW (tutorial 03) - JSF + CRUD.

IMPORTANTE: ao final dessas alterações, a estrutura de arquivos da aplicação terá os mesmos arquivos excluídos, voltando à estruturação apresentada na Figura 4.

Passo 5) Após, altere o **managed bean controller.bookController.java**, conforme indicado na Figura 5.


```
private final Book book = new Book();
private String name;
private String author;
private String coAuthor;
private String isbn;
private Integer year;
private String category;
private float price;

public String getCoAuthor() {
    return coAuthor;
}

public void setCoAuthor(String coAuthor) {
    this.coAuthor = coAuthor;
}

public String getIsbn() {
    return isbn;
}

public void setIsbn(String isbn) {
    this.isbn = isbn;
}

public void emptyVariables() {
    this.author = "";
    this.coAuthor = "";
    this.isbn = "";
    this.category = "";
    this.name = "";
    this.price = 0;
    this.year = 0;
}

public String createBook() {
    this.book.setAuthor(this.author);
    this.book.setCoAuthor(this.coAuthor);
    this.book.setIsbn(this.isbn);
    this.book.setCategory(this.category);
    this.book.setName(this.name);
    this.book.setPrice(this.price);
    this.book.setYear(this.year);
    this.booksFacade.create(this.book);
    this.emptyVariables();
    return "index.xhtml?faces-redirect=true";
}
```

Figura 5. Alterações no **managed bean** da aplicação Java Web com JSF.

Passo 6) Após, altere o **index.xhtml**, conforme indicado na Figura 6: acrescente os campos **coAuthor** e **isbn** em todos os formulários.

Passo 7) Você pode alterar a exibição dos botões e Edição e Deleção, acrescentando um “label”, conforme indicado na Figura 7.

Passo 8) Revise todos os códigos-fontes e veja se todas as alterações para incluir os dois novos campos / colunas na tabela **book** foram realizadas. Após a revisão cuidadosa, você poderá realizar um **Clean and Build** no projeto e executá-lo com **Run**.

A execução do projeto correspondente à alteração solicitada neste trabalho deverá ficar semelhante ao exibido na Figura 8.

Para melhor visualização do formulário de criação de livro, indique **columns = "2"**, como na Figura 6.

```

<h:head>
<title>
</h:head>
<h:body>
<h1>CRUD</h1>
<h:form>
<h:panelGrid columns="2" cellpadding="3">
<h:outputText value="#{bookController.name}" />
<p:inputText value="#{bookController.name}" />
<h:outputText value="#{bookController.author}" />
<p:inputText value="#{bookController.author}" />
<h:outputText value="#{bookController.coAuthor}" />
<p:inputText value="#{bookController.coAuthor}" />
<h:outputText value="#{bookController.isbn}" />
<p:inputText value="#{bookController.isbn}" />
<h:outputText value="#{bookController.category}" />
<p:inputText value="#{bookController.category}" />
<h:outputText value="#{bookController.year}" />
<p:inputText value="#{bookController.year}" />
<h:outputText value="#{bookController.price}" />
<p:inputText value="#{bookController.price}" />
<p:commandButton value="Add" icon="fa fa-fw fa-plus" action=
</h:panelGrid>
</h:form>
</h:body>
</h:html>

```

```

</h:form>
<h:form id="form">
<p:dataTable value="#{bookController.getAllBooks()}" var="
<p:column headerText="Name">
<h:outputText value="#{book.name}" />
</p:column>
<p:column headerText="Year">
<h:outputText value="#{book.year}" />
</p:column>
<p:column headerText="Author">
<h:outputText value="#{book.author}" />
</p:column>
<p:column headerText="CoAuthor">
<h:outputText value="#{book.coAuthor}" />
</p:column>
<p:column headerText="ISBN">
<h:outputText value="#{book.isbn}" />
</p:column>
<p:column headerText="Category">
<h:outputText value="#{book.category}" />
</p:column>
<p:column headerText="Price">
<h:outputText value="#{book.price}" />
</p:column>

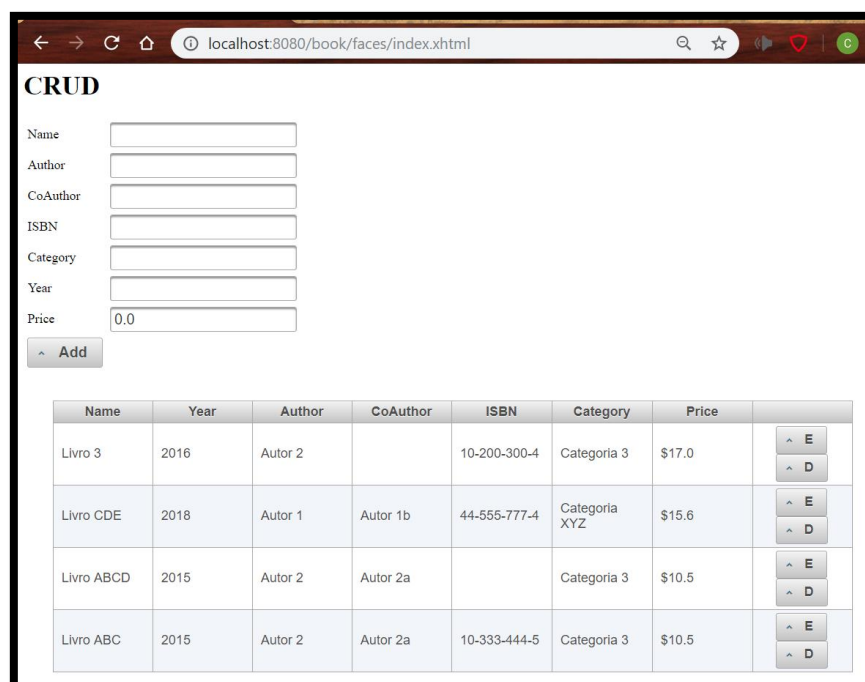
```

Figura 6. Alterações no **index.xhtml**.

```

<p:column style="width:100px;text-align: center">
<p:commandButton value="E" icon="fa-pencil" update="form:bookEdit" oncomplete="PF('editDialog').show()" />
<f:setPropertyActionListener value="#{book}" target="#{bookController.selectedBook}" />
</p:commandButton>
<p:commandButton value="D" action="#{bookController.deleteBook(book)}" icon="fa-trash" />
</p:column>

```

Figura 7. No **index.xhtml**, alterando *labels* dos botões de **Editar** e **Deletar**.


CRUD

Name:

Author:

CoAuthor:

ISBN:

Category:

Year:

Price:

Name	Year	Author	CoAuthor	ISBN	Category	Price	
Livro 3	2016	Autor 2		10-200-300-4	Categoria 3	\$17.0	^ E ^ D
Livro CDE	2018	Autor 1	Autor 1b	44-555-777-4	Categoria XYZ	\$15.6	^ E ^ D
Livro ABCD	2015	Autor 2	Autor 2a		Categoria 3	\$10.5	^ E ^ D
Livro ABC	2015	Autor 2	Autor 2a	10-333-444-5	Categoria 3	\$10.5	^ E ^ D

Figura 8. Aplicação JSF + CRUD alterada, com mais 2 campos na tabela **book**.

2.2 Aplicação Java Web JSF com AJAX

Alteração da aplicação JSF + AJAX com a inclusão de mais **dois** campos no formulário de entrada (input), que deverão ser tratados na **página JSF** (**helloAjax.xhtml**) e no **managed bean** (**helloBean.java**):

1. `"#{helloBean.program}"` (campo no JSF e no managed bean) e
2. `"#{helloBean.someDate}"` (campo no JSF e no managed bean).

Passo 1) Faça as alterações no **helloAjax.xhtml**, conforme indicado na Figura 9. Observe que, para exibir o formulário com maior organização, utilizamos a Tag JSF - **h:panelGrid**, que renderiza em uma tabela HTML. Observe, ainda na Figura 9, como essa tag deve ser posicionada na página JSF, em relação a tag JSF **h:form**.

Continuando na Figura 9, utilizamos a tag JSF **h:selectOneMenu**, que renderiza como um menu de opções **select** do HTML. Da mesma forma, a tag JSF **f:convertDateTime** é usada para converter uma string em uma data, no formato requerido. Ela também atua como um validador, com um formato de data obrigatório – verificar o atributo **pattern**.

Por fim, a Figura 9 ainda mostra como executar o AJAX para múltiplos campos, em um único botão de ação. Detalhe que cada campo renderizará apenas se houver dados recebidos do servidor, não atualizando toda a página.

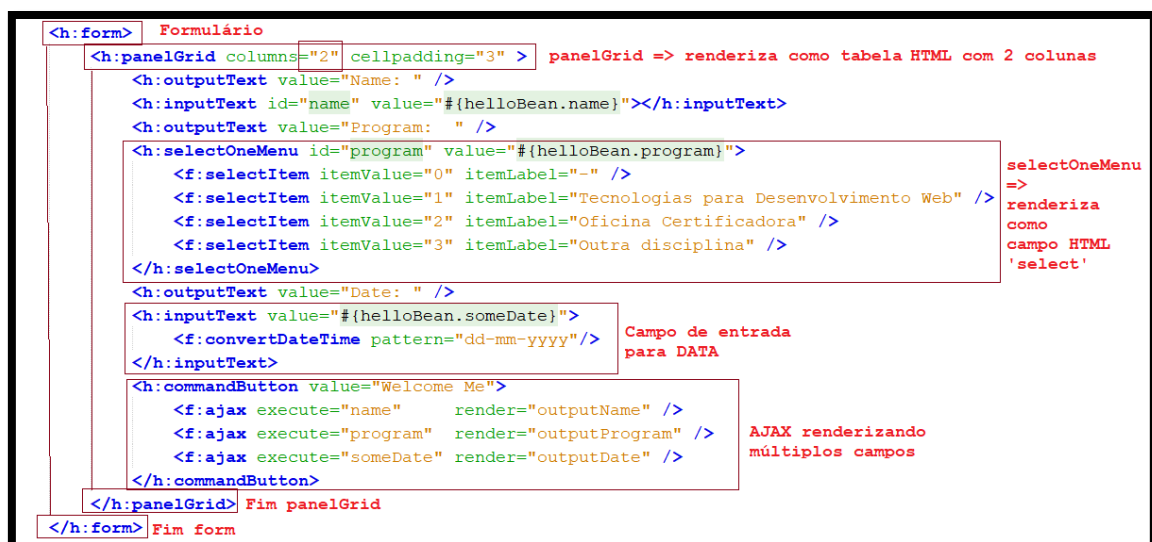


Figura 9. **HelloAjax.xhtml**: form com campo menu de opções e campo data.

Passo 2) Continue realizando as alterações no **helloAjax.xhtml**, conforme indicado na Figura 10. Observe é apresentado o final do arquivo, com código forma do formulário (fechado em **</h:form>**). No código, cada campo de output tem um identificador (**id**) referenciado na chamada AJAX de execução da Figura 9. No código do managed bean **helloBean** será preciso acrescentar os tratamentos indicados para os valores (**value =**) da Figura 10.

```
</h:form>
<h2><h:outputText id="outputName" value="#{helloBean.sayWelcome}" /></h2>
<h2><h:outputText id="outputProgram" value="#{helloBean.sayProgram}" /></h2>
<h2><h:outputText id="outputDate" value="#{helloBean.someDate}" >
    <f:convertDateTime pattern = "dd-mm-yyyy" />
</h:outputText>
</h2>
</h:body>
</html>
```

Figura 10. **HelloAjax.xhtml**: campos de saída com respectivos valores.

Passo 3) No código do **helloBean.java**, devemos acrescentar os novos atributos da classe, com respectivos métodos *getter* e *setter*, conforme indicado na Figura 11.

```
private String program;
private Date someDate;

public helloBean() {
}

public String getProgram() {
    return program;
}

public void setProgram(String program) {
    this.program = program;
}

public Date getSomeDate() {
    return someDate;
}

public void setSomeDate(Date someDate) {
    this.someDate = someDate;
}
```

Figura 11. **helloBean.java**: novos atributos, com respectivos *getter* e *setter*.

Passo 4) Ainda no código do **helloBean.java**, precisamos tratar o retorno do menu de opções recebido do campo de entrada **h:selectOneMenu**. Isso é feito conforme indicado na Figura 12, com um **switch** do Java.

Passo 5) Revise todos os códigos-fontes e veja se todas as alterações para incluir os dois novos campos de entrada no formulário do XHTML foram realizadas. Após a revisão cuidadosa, você poderá realizar um **Clean and Build** no projeto, selecionar o arquivo **helloAjax.xhtml** e executá-lo com **Run File**.

A execução do projeto correspondente à alteração solicitada neste trabalho deverá ficar semelhante ao exibido na Figura 13.

```
public String getSayProgram() {  
    String outStr = "";  
    //check if null?  
    if ("".equals(program) || program == null) {  
        return "";  
    } else {  
        int num = Integer.parseInt(program);  
        switch (num) {  
            case 1:  
                outStr = "Programa: Tecnologias para Desenvolvimento Web";  
                break;  
            case 2:  
                outStr = "Programa: Oficina Certificadora";  
                break;  
            case 3:  
                outStr = "Programa: Outra disciplina";  
                break;  
        }  
        return outStr;  
    }  
}
```

Figura 12. **helloBean.java**: tratamento do input **h:selectOneMenu**.

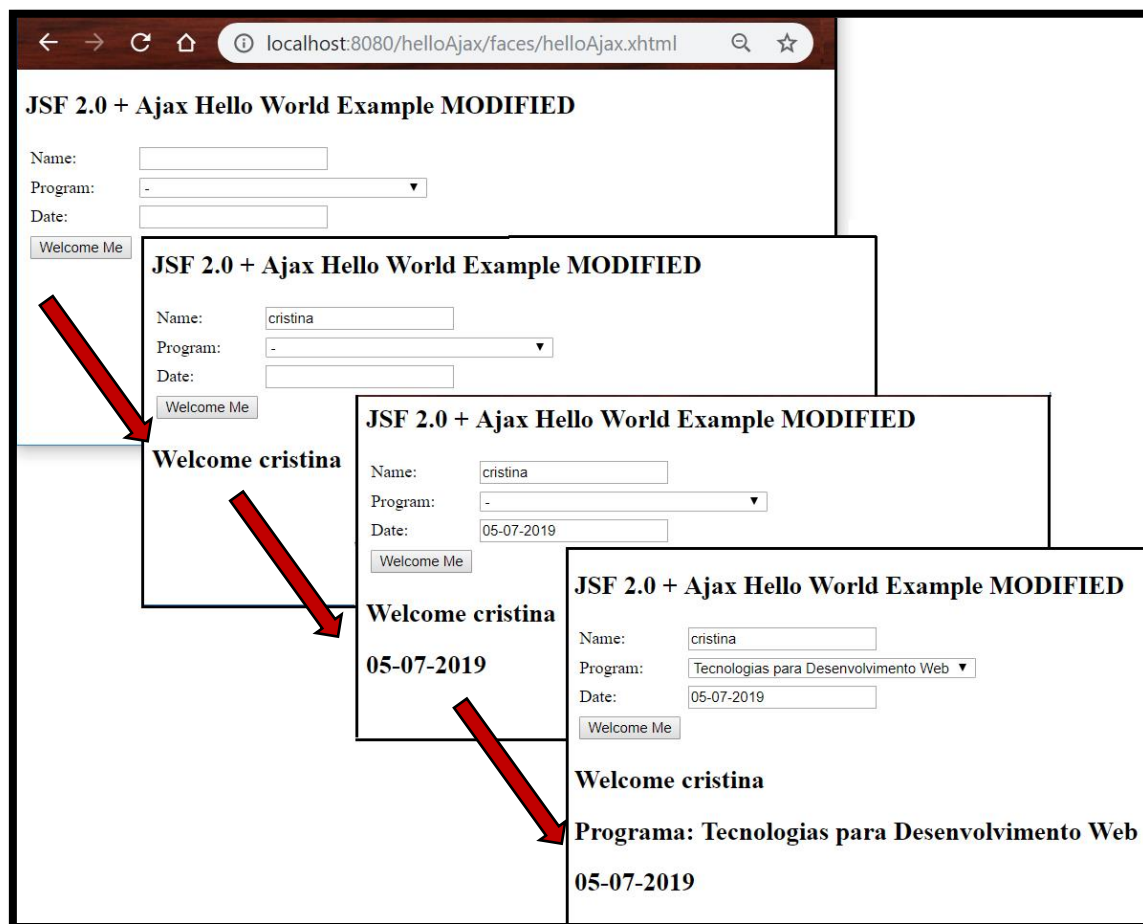


Figura 13. **helloAjax**: transferência progressiva de dados entre cliente-servidor.

2.3 Aplicação Java Web JSF com JSON

Alteração da aplicação JSF + JSON com a alteração na aplicação do tutorial 05, no qual é preciso modificar o formulário inicial para tratar dos dados usados na identificação de livro (idêntico aos campos da tabela book).

Passo 1) Faça as alterações no **index.xhtml**, conforme indicado na Figura 14, considerando que os dados deste formulário devem ser idênticos aos utilizados para manter a tabela **book**, da aplicação apresentada no tutorial 03:

```
private String name;
private String author;
private String coAuthor;
private String isbn;
private Integer year;
private String category;
private float price;
```

```
<h:form>
  <table>
    <tr>
      <td align="right">Name:</td>
      <td><h:inputText value="#{objectModelBean.name}"/></td>
    </tr><tr>
      <td align="right">Author:</td>
      <td><h:inputText value="#{objectModelBean.author}"/></td>
    </tr><tr>
      <td align="right">CoAuthor:</td>
      <td><h:inputText value="#{objectModelBean.coAuthor}"/></td>
    </tr><tr>
      <td align="right">ISBN:</td>
      <td><h:inputText value="#{objectModelBean.isbn}"/></td>
    </tr><tr>
      <td align="right">Category:</td>
      <td><h:inputText value="#{objectModelBean.category}"/></td>
    </tr><tr>
      <td align="right">Price:</td>
      <td><h:inputText value="#{objectModelBean.price}"/></td>
    </tr>
  </table>
  <p><h:commandButton value="Create a JSON Object" action="#{objectModelBean.buildJson()}" /></p>
</h:form>
```

Figura 14. **Index.xhtml**: novo formulário para aplicação JSON.

Passo 2) Faça as alterações no managed bean **ObjectModelBean.java**, conforme indicado na Figura 15, para tratar dos novos dados do formulário alterado em **index.xhtml**.

```
/* Form properties */
private String name = "Duke Book";
private String author = "Duke Sr.";
private String coAuthor = "Duke Jr.";
private String isbn = "10-202-303-5";
private int year = 2014;
private String category = "Duku Duke";
private float price = (float) 20.5;
protected String jsonTextArea = "";
```

Figura 15. **ObjectModelBean.java**: novas propriedades para manter o novo formulário de **index.xhtml**.

Passo 3) Ainda no managed bean **ObjectModelBean.java**, acrescentar os respectivos *getter* e *setter*, conforme indicado na Figura 16, para tratar das novas propriedades da classe.

```
/* Getters and setters */  
public String getName() { return name; }  
  
public void setName(String name) { this.name = name; }  
public String getAuthor() { return author; }  
public void setAuthor(String author) { this.author = author; }  
  
public String getCoAuthor() { return coAuthor; }  
  
public void setCoAuthor(String coAuthor) { this.coAuthor = coAuthor; }  
  
public String getIsbn() { return isbn; }  
  
public void setIsbn(String isbn) { this.isbn = isbn; }  
  
public int getYear() { return year; }  
  
public void setYear(int year) { this.year = year; }  
  
public String getCategory() { return category; }  
  
public void setCategory(String category) { this.category = category; }  
  
public float getPrice() { return price; }  
  
public void setPrice(float price) { this.price = price; }
```

Figura 16. **ObjectModelBean.java**: novos *getter* e *setter* para as novas propriedades.

Passo 4) Revise todos os códigos-fontes e veja se todas as alterações para alterar o formulário de entrada, que trata agora os dados de **book**, foram realizadas. Após a revisão cuidadosa, você poderá realizar um **Clean and Build** no projeto e executá-lo com **Run File**.

A execução do projeto correspondente à alteração solicitada neste trabalho deverá ficar semelhante ao exibido na Figura 17.

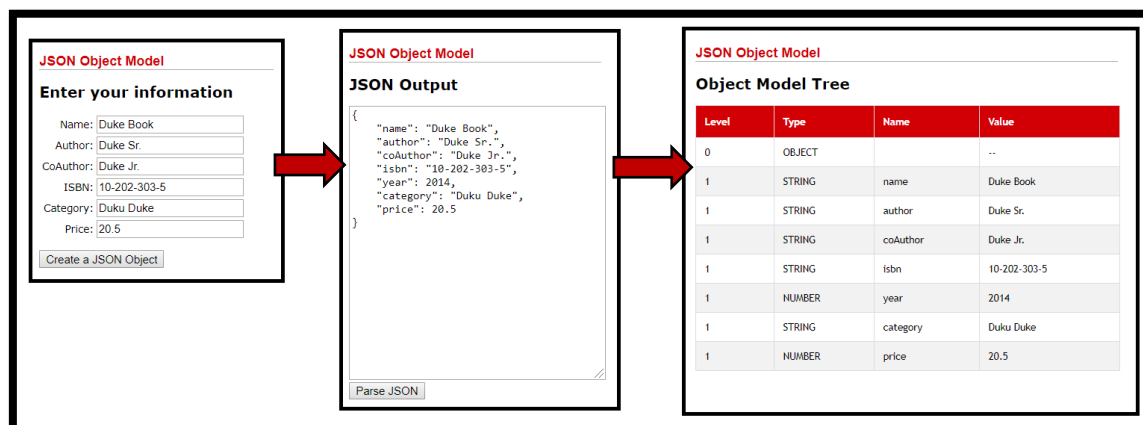


Figura 17. Nova execução do projeto JSF + JSON.

3 Orientações Gerais

O professor tutor irá avaliar este trabalho de acordo com o solicitado neste documento. Assim, combine com seus colegas de equipe para revisar os itens solicitados, verificando se todos foram atendidos.

Sugestão: crie mais uma lista ou mais cartões no Trello, nos quais cada membro da equipe verifica se as tarefas foram criadas para todos, se receberam anexos, se foram realizadas com seus respectivos registros gravados.

4 Orientações - Apresentação dos Trabalhos em Vídeo

4.1 Vídeo para demonstrar o Java Web JSF com CRUD

Os seguintes itens deverão estar contemplados no vídeo do **Trabalho 1 JSF + CRUD**, que deve ter **duração entre 3min e 7 min.**

- Identificação de cada um dos componentes da equipe: nome apresentado na tela e narrado pelo próprio estudante – não é necessário aparecer nos vídeos.
- Breve apresentação da tabela alterada na máquina local e no **phpmyadmin**, indicando os campos incluídos
- Breve apresentação das alterações realizadas nos códigos-fonte, indicando como tratou no código os campos incluídos

- d) A aplicação JSF aparece com formatação de tabela, com todos os elementos solicitados (botões, campos, dados)
- e) Exibição do ciclo CRUD completo, apresentando todos os campos originais da tabela, mais os dois novos campos:
 - I. Criação de novo livro;
 - II. Exibição completa da lista de livros gravados em tabela;
 - III. Edição de um livro e correspondente exibição dos seus valores editados / alterados;
 - IV. Exclusão de livro e correspondente atualização de exibição da lista de livros.

4.2 Vídeo para demonstrar o Java Web JSF com AJAX

Os seguintes itens deverão estar contemplados no vídeo do **Trabalho 2 JSF + AJAX**, que deve ter **duração entre 3min e 5 min**.

- a) Identificação de cada um dos componentes da equipe: nome apresentado na tela e narrado pelo próprio estudante – não é necessário aparecer nos vídeos.
- b) Breve apresentação do formulário de entrada alterado, no IDE NEtBeans, indicando os campos incluídos e seu respectivo tratamento no managed bean.
- c) A aplicação JSF aparece com formatação de tabela, com todos os elementos solicitados (botões, campos, dados)
- d) Exibição da execução completa, apresentando todos os campos do formulário sendo enviados e tratados no navegador, conforme Figura 13.
 - I. Envio e tratamento do campo **name**;
 - II. Envio e tratamento do campo **program**;
 - III. Envio e tratamento do campo **someDate**;

4.3 Vídeo para demonstrar o Java Web JSF com JSON

Os seguintes itens deverão estar contemplados no vídeo do **Trabalho 2 JSF + JSON**, que deve ter **duração entre 3min e 5 min**.

- a) Identificação de cada um dos componentes da equipe: nome apresentado na tela e narrado pelo próprio estudante – não é necessário aparecer nos vídeos.

- b) Breve apresentação do formulário de entrada alterado, no IDE NEtBeans, indicando os campos incluídos e seu respectivo tratamento no managed bean.
- c) A aplicação JSF aparece com formatação de tabela, com todos os elementos solicitados (botões, campos, dados)
- d) Exibição da execução completa, apresentando todos os campos do novo formulário sendo enviados e tratados no navegador, conforme Figura 17 (indicar, no vídeo, a interação entre as páginas XHTML e managed bean, conforme material de apoio do tutorial 05):
 - a. 1º Cria um modelo a partir de um formulário;
 - b. 2º Retorna String com estrutura JSON;
 - c. 3º Faz a análise gramatical (**parse**) da String;
 - d. 4º Retorna a String decomposta em um Object Model

5 Critérios de Correção – Aplicações Java Web

O presente trabalho é avaliativo e será corrigido conforme detalhado a seguir.

Equipe	Estudante 1:		Estudante 2:	
Critério	No Vídeo de Defesa do Trabalho		Pontuação	
			Atende	Não Atende
Trab. 1) Vídeo 1: Aplicação Java Web JSF com CRUD	A equipe identifica, no vídeo , cada um dos seus componentes (nome aparece escrito e narrado pelo próprio estudante)		0,2	0,0
	A equipe faz breve apresentação da tabela alterada no phpmyadmin , indicando os novos campos incluídos .		0,2	0,0
	A equipe faz breve apresentação das alterações realizadas nos códigos-fonte, indicando como tratou os campos incluídos		0,5	0,0
	A aplicação JSF é apresentada no navegador organizada como uma tabela, com todos os elementos solicitados (uma linha de tabela por registro de banco de dados, botões, campos, dados) – conforme Figura 8 deste documento		0,5	0,0
	A equipe exibe ciclo CRUD completo, com os novos campos incluídos na tabela book :	I. Cria livro novo e o exibe na lista completa de livros (formato de tabela), a partir da consulta aos livros gravados em tabela;	1,0	0,0
		II. Edita dados de um livro e o exibe na lista completa de livros (formato de tabela), a partir da consulta aos livros gravados em tabela;	1,0	0,0
		III. Exclui um livro e exibe a atualização da lista completa de livros (formato de tabela), a partir da consulta aos livros gravados em tabela;	1,0	0,0
	O Vídeo tem duração entre 3 min e 7 min .		0,2	0,0
Trab. 2) Vídeo 2: Aplicação Java Web JSF com AJAX	A equipe identifica, no vídeo , cada um dos seus componentes (nome aparece escrito e narrado pelo próprio estudante)		0,2	0,0
	A equipe faz breve apresentação do formulário de entrada alterado, no IDE NEtBeans, indicando os campos incluídos e seu respectivo tratamento no managed bean .		0,2	0,0
	A equipe faz breve apresentação das alterações realizadas nos códigos-fonte, indicando como tratou os campos incluídos		0,5	0,0
	A aplicação JSF aparece com formatação de tabela, com todos os elementos solicitados (botões, campos, dados) – conforme Figura 13 deste documento.		0,5	0,0
	A equipe exibe a execução completa , apresentando todos os campos do formulário sendo enviados e tratados no navegador, conforme Figura 13 :	I. Envio e tratamento do campo name ;	0,5	0,0
		II. Envio e tratamento do campo program	0,5	0,0
		III. Envio e tratamento do campo someDate ;	0,5	0,0
	O Vídeo tem duração entre 3 min e 5 min .		0,2	0,0

Trab. 3) Vídeo 3: Aplicação Java Web JSF com JSON	A equipe identifica, no vídeo , cada um dos seus componentes (nome aparece escrito e narrado pelo próprio estudante)		0,2	0,0
	A equipe faz breve apresentação do formulário de entrada alterado, no IDE NetBeans, indicando os campos incluídos e seu respectivo tratamento no managed bean.		0,2	0,0
	A equipe faz breve apresentação das alterações realizadas nos códigos-fonte, indicando como tratou os campos incluídos		0,5	0,0
	A aplicação JSF aparece com formatação de tabela, com todos os elementos solicitados (botões, campos, dados) – conforme Figura 17 deste documento.		0,2	0,0
	A equipe faz a exibição da execução completa, apresentando todos os campos do novo formulário sendo enviados e tratados no navegador, conforme Figura 17 (indicar, no vídeo, a interação entre as páginas XHTML e managed bean, conforme material de apoio do tutorial 05):	I. Cria um modelo JSON a partir de um formulário e o apresenta como String em nova página no navegador;	0,5	0,0
		II. Faz a análise gramatical (parse) da String, a transforma em objeto JSON e a apresenta decomposta como tabela em nova página no navegador;	0,5	0,0
	O Vídeo tem duração entre 3 min e 5 min.		0,2	0,0
Pontuação dos 3 Trabalhos (máxima)			10,0	

6 REFERÊNCIAS

JSF 2.2 View Declaration Language: Facelets Variant – Tag Libraries, em

<https://docs.oracle.com/javaee/7/javaxserver-faces-2-2/vldocs-facelets/>

JSF – **h:panelGrid**, em https://www.tutorialspoint.com/jsf/jsf_panelgrid_tag.htm

JSF - **h:selectOneMenu**, em

https://www.tutorialspoint.com/jsf/jsf_selectonemenu_tag.htm

JSF - **f:convertDateTime**, em

https://www.tutorialspoint.com/jsf/jsf_convertdatetime_tag.htm

JSF AJAX Render Multiple Field, em:

<http://javahonk.com/jsf-ajax-render-multiple-field/>

The jsonmodel Example Application, em:

<https://javaee.github.io/tutorialjsonp006.html#to-run-the-jsonmodel-example-application-using-netbeans-ide>

JSON Tutorial, em: <https://www.tutorialspoint.com/json/index.htm>