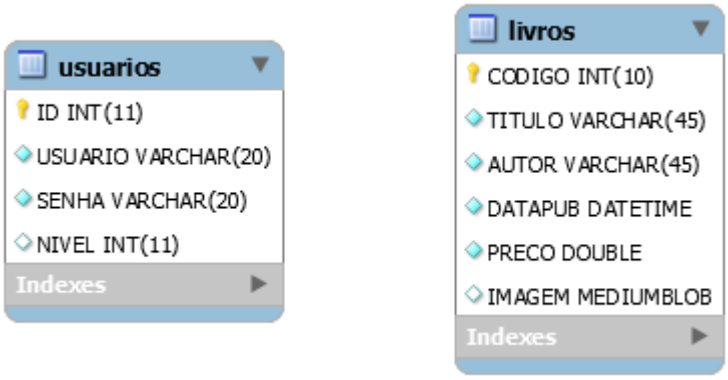
Roteiro para desenvolvimento da aula sobre Servlets, JSP e JSTL

Este documento tem como objetivo apresentar um roteiro para o desenvolvimento das aulas referentes a disciplina Java na WEB.

Camada de Acesso a Dados

Para esta aula usaremos o seguinte banco de dados (script disponível no portal da Fiap):



Obs.: Tamanhos dos campos blob:

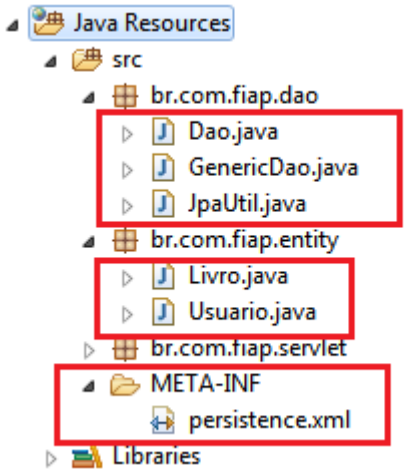
TINYBLOB: 255 (2^8 - 1) bytes.

BLOB: 65535 (2^16 - 1) bytes.

MEDIUMBLOB: 16777215 (2^24 - 1) bytes.

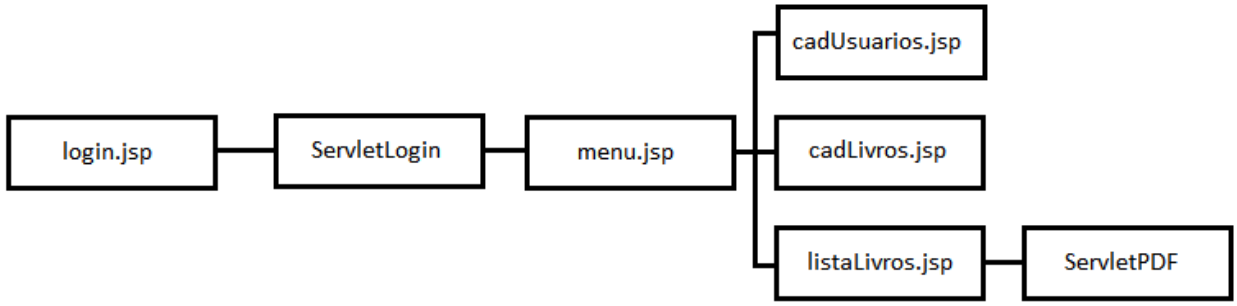
LONGBLOB: 4294967295 (2^32 - 1) bytes.

O script deste banco de dados está disponível no portal da Fiap, juntamente com as classes e o arquivo de configurações do JPA representados abaixo:



Aplicação – Cadastro e Consulta de Livros usando páginas jsp

O diagrama de navegação ilustrativo é apresentado a seguir:



Os itens são descritos a seguir:

login.jsp: conteúdo HTML para entrada do usuário e senha tradicionais.

ServletLogin: recebe os dados do formulário em **login.jsp**, autentica o usuário (valida e armazena em sessão) e direciona para **menu.jsp**. Nota: armazenar na sessão o objeto **Usuario**, pois seus dados serão recuperados nas páginas seguintes.

O método **doGet()** deste formulário deve transferir a requisição para **login.jsp** no sentido de incluir seu conteúdo.

Se houver erro na validação, transferir a requisição para uma página chamada **erro.jsp** (não indicada no esquema). Esta página deverá apresentar a mensagem de erro e ter um link para **login**.

menu.jsp: Nesta página deverá ter apenas três links para as páginas seguintes, conforme especificado no modelo. NA verdade, cada link deverá redirecionar o usuário para um servlet, cujo método doGet() incluirá o conteúdo do JSP. O método doPost() receberá os dados do formulário quando este for submetido.

cadUsuarios.jsp: Uma página que permite realizar o cadastro de novos usuários. Esta página deve ser elaborada com conteúdo HTML.

cadLivros.jsp: Formulário com os dados dos livros, com elementos HTML, JavaBeans e JSTL.

listaLivros.jsp: Apresenta os dados dos livros em forma tabular. O código deve estar envolvido por um link que, quando selecionado pelo usuário, apresenta os detalhes do livro incluindo sua imagem. Deverá haver também um link para um documento PDF contendo um resumo do livro.

As páginas menu.jsp, cadUsuarios.jsp, cadLivros.jsp e listaLivros.jsp deverão ser criadas na pasta /admin, abaixo de WebContent. Porém é uma boa idéia coloca-las em uma pasta abaixo de /WEB-INF. Saberia dizer porque?

A seguir apresentaremos os esquemas que deverão ser seguidos para o desenvolvimento dos servlets.

ServletPDF: recebe um parâmetro representando o código do livro, e busca por um arquivo PDF com o mesmo nome do código. Se não for encontrado, apresentar um documento padrão.

Código para ler o PDF:

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
	ServletOutputStream stream = null;
	BufferedInputStream buffer = null;

	String arquivo = request.getParameter("cod");

	try {
		stream = response.getOutputStream();
		File pdf = new File("D:/arquivos/" + arquivo + ".pdf");
		if(!pdf.exists()){
			pdf = new File("D:/arquivos/geral.pdf");
		}

		response.setContentType("application/pdf");
		FileInputStream input = new FileInputStream(pdf);
		buffer = new BufferedInputStream(input);
		int bytes = 0;

		while((bytes = buffer.read()) != -1){
			stream.write(bytes);
		}
	} catch (Exception e) {
		throw new ServletException(e.getMessage());
	} finally {
		if(stream != null){ stream.close(); }
		if(buffer != null){ buffer.close(); }
	}

}
```

ServletCadastroLivros, ServletCadastroUsuarios: Deverão ter, respectivamente, os valores do mapeamento configurados como **/admin/cadLivros** e **/admin/cadUsuarios**. O método doGet() de cada servlet incluirá os respectivos arquivos JSP e o método doPost() realizará a tarefa de persistência, transportando a requisição com uma mensagem para cada arquivo JSP, de acordo com o objetivo.

O cadastro de livros deverá contemplar a inclusão da imagem. Este servlet poderá ter o código sugerido a seguir:

```
@WebServlet("/admin/cadLivros")
@MultipartConfig
public class ServletCadastroLivros extends HttpServlet {
	private static final long serialVersionUID = 1L;
```

```
    public ServletUpload() {
        super();
        // TODO Auto-generated constructor stub
    }

      protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
            // TODO Auto-generated method stub
      }

      protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
            PrintWriter out = response.getWriter();
            InputStream inputStream = null;

            try {

                  int codigo = Integer.parseInt(request.getParameter("codigo"));

                  String titulo = request.getParameter("titulo");
                  String autor = request.getParameter("autor");
                  String data = request.getParameter("datapub");
                  Date datapub = new SimpleDateFormat("dd/MM/yyyy").parse(data);

                  double preco = Double.parseDouble(request.getParameter("preco"));


                  Part filePart = request.getPart("foto");
                  byte[] imagem = new byte[(int)filePart.getSize()];

                  if(filePart != null){
                        inputStream = filePart.getInputStream();
                        inputStream.read(imagem, 0, (int)filePart.getSize());
                  }

                  Livro livro = new Livro();
                  livro.setCodigo(codigo);
                  livro.setTitulo(titulo);
                  livro.setAutor(autor);
                  livro.setDataPublicacao(datapub);
                  livro.setPreco(preco);
                  livro.setImagem(imagem);

                  GenericDao<Livro> dao = new GenericDao<Livro>(Livro.class);

                  dao.adicionar(livro);
                  //restante do código

            } catch (Exception e) {
                  //tratamento do erro
            }
      }
}
```

ServletListaLivros: Criará a lista de livros obtida do banco de dados, a incluirá na requisição e a transportará para **listaLivros.jsp**, que se encarregará de exibi-la.

ServetConsultarLivros: acessado pelo link correspondente em listaLivros.jsp, este servlet busca por um livro a partir do seu código. Como modelo, sugerimos o código a seguir:

```
@WebServlet("/admin/consultarLivro")
public class ServletConsultarCliente extends HttpServlet {
	private static final long serialVersionUID = 1L;

    public ServletConsultarCliente() {
        super();
        // TODO Auto-generated constructor stub
    }

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		GenericDao<Livro> dao = new GenericDao<>(Livro.class);

		int id = Integer.parseInt(request.getParameter("codigo"));
		Livro livro = dao.buscar(id);
		request.setAttribute("livro", livro);
		request.getRequestDispatcher("mostrarLivro.jsp").forward(request,
response);

	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
	}

}
```

E o arquivo **mostrarLivro.jsp**, por sua vez, possuirá o código sugerido abaixo:

```
<body>
	<h1>Detalhes do livro</h1>
	<ul>
		<li>Codigo: ${livro.codigo}</li>
		<li>Título: ${livro.titulo}</li>
		<li>Autor: ${livro.autor}</li>

	</ul>
	<img src="admin/imagem?id=${livro.codigo}"/>
</body>
```

Observe que a exibição da imagem é processada através da chamada a outro servlet, que através do código do livro, obtém a imagem no banco de dados. O código para este servlet está representado a seguir:

```
@WebServlet("/admin/imagem")
public class ServletImagem extends HttpServlet {
	private static final long serialVersionUID = 1L;

    public ServletImagem() {
        super();
        // TODO Auto-generated constructor stub
    }

	protected void doGet(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
		int id = Integer.parseInt(request.getParameter("id"));
		GenericDao<Livro> dao = new GenericDao<>(Livro.class);
		byte[] imagem = dao.buscar(id).getImagem();
		response.setContentType("image/jpeg");
		ServletOutputStream os = response.getOutputStream();
		os.write(imagem);
		os.close();
	}

	protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
		// TODO Auto-generated method stub
	}
}
```

Na sequencia, criar um filtro para bloquear o acesso a usuários não autenticados (neste caso, não armazenados em sessão). Este filtro deverá operar sobre todos os recursos mapeados por /admin/*.

Incluir também *listeners* capazes de monitorar as operações do usuário na sua aplicação, como criação e inclusão de usuários na sessão, início da aplicação, inclusão de atributos na requisição, e outros que julgar importantes. Como sugestão, criar um arquivo de log para estas tarefas

Bom trabalho a todos!

Prof. Emilio