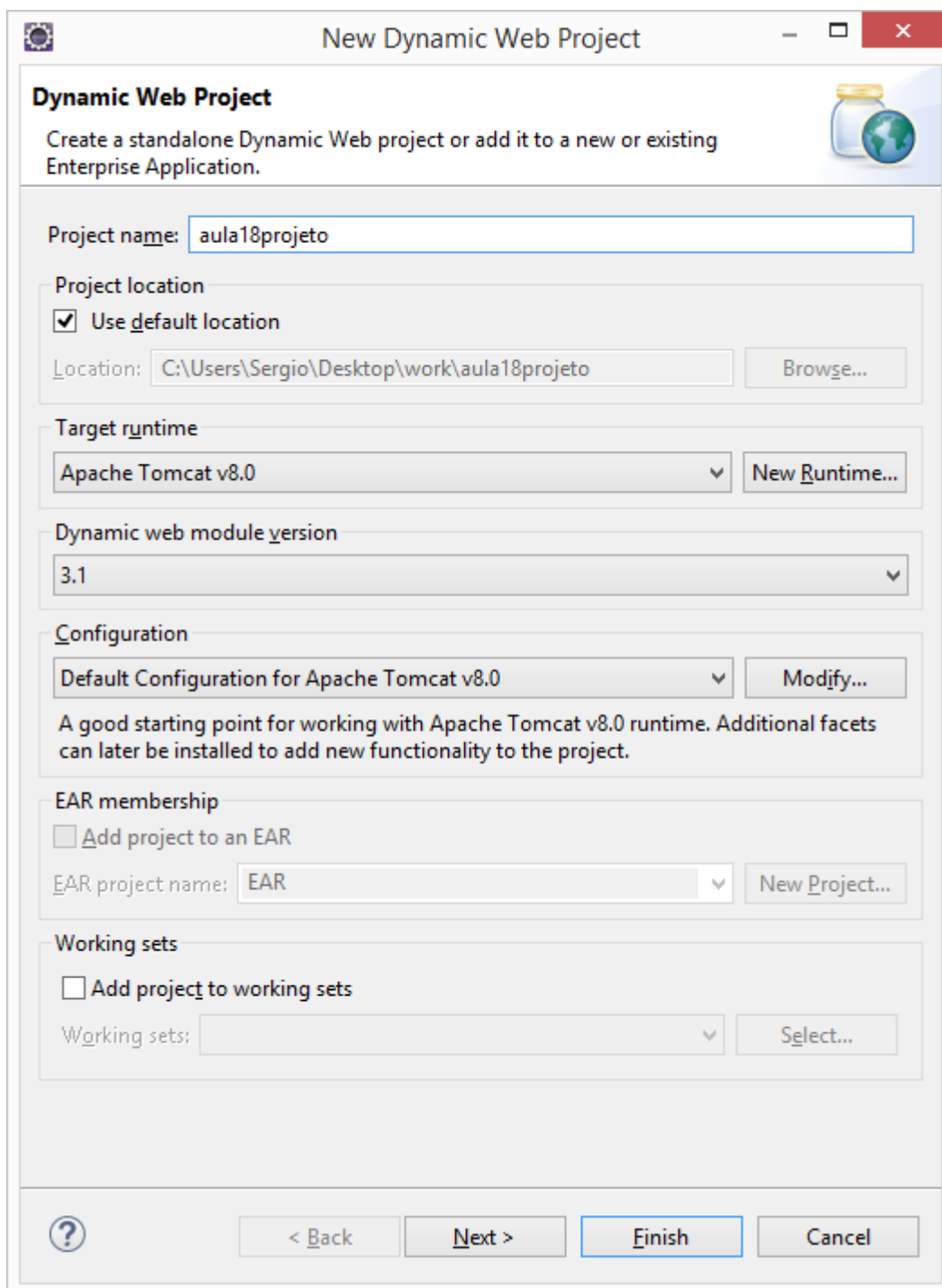


Criando o projeto:

File > New > Dynamic Web Project



The screenshot shows the 'New Dynamic Web Project' dialog box in the Eclipse IDE. The dialog is titled 'New Dynamic Web Project' and has a close button (X) in the top right corner. It contains several sections for configuring the project:

- Dynamic Web Project**: A section with a description: 'Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.' and a small icon of a globe and a jar.
- Project name:** A text field containing 'aula18projeto'.
- Project location**: A section with a checkbox 'Use default location' which is checked. Below it is a text field for 'Location' containing 'C:\Users\Sergio\Desktop\work\aula18projeto' and a 'Browse...' button.
- Target runtime**: A dropdown menu showing 'Apache Tomcat v8.0' and a 'New Runtime...' button.
- Dynamic web module version**: A dropdown menu showing '3.1'.
- Configuration**: A dropdown menu showing 'Default Configuration for Apache Tomcat v8.0' and a 'Modify...' button. Below this is a descriptive text: 'A good starting point for working with Apache Tomcat v8.0 runtime. Additional facets can later be installed to add new functionality to the project.'
- EAR membership**: A section with a checkbox 'Add project to an EAR' which is unchecked. Below it is a text field for 'EAR project name' containing 'EAR' and a 'New Project...' button.
- Working sets**: A section with a checkbox 'Add project to working sets' which is unchecked. Below it is a text field for 'Working sets' and a 'Select...' button.

At the bottom of the dialog, there is a row of buttons: a help button (question mark), '< Back', 'Next >', 'Finish' (highlighted with a blue border), and 'Cancel'.

Criando as entidades e mapeando com JPA

Java Persistence API

```
package br.com.brq.entities;

import java.util.List;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.OneToOne;
import javax.persistence.Table;

@Entity
@Table(name = "usuario")
@NamedQueries(
    {
        @NamedQuery(name = Usuario.HAS_LOGIN,
            query = "select count(u) from Usuario as u where u.login = :p1"),

        @NamedQuery(name = Usuario.AUTHENTICATE,
            query = "select u from Usuario as u where u.login = :p1 and u.senha = :p2")
    }
)
public class Usuario {

    public static final String HAS_LOGIN = "usuario.haslogin";
    public static final String AUTHENTICATE = "usuario.authenticate";

    @Id
    @GeneratedValue(strategy=GenerationType.AUTO)
    @Column(name = "idusuario")
    private Integer idUsuario;

    @Column(name = "nome", length = 50, nullable = false)
    private String nome;

    @Column(name = "login", length = 25, nullable = false, unique = true)
    private String login;

    @Column(name = "senha", length = 50, nullable = false)
    private String senha;
```

```
@OneToMany(mappedBy = "usuario") //1 usuario tem muitas tarefas..  
private List<Tarefa> tarefas;
```

```
public Usuario() {  
    // TODO Auto-generated constructor stub  
}
```

```
public Usuario(Integer idUsuario, String nome, String login, String senha) {  
    super();  
    this.idUsuario = idUsuario;  
    this.nome = nome;  
    this.login = login;  
    this.senha = senha;  
}
```

```
public Usuario(Integer idUsuario, String nome, String login, String senha,  
                List<Tarefa> tarefas) {  
    this(idUsuario, nome, login, senha);  
    this.tarefas = tarefas;  
}
```

```
public Integer getIdUsuario() {  
    return idUsuario;  
}
```

```
public void setIdUsuario(Integer idUsuario) {  
    this.idUsuario = idUsuario;  
}
```

```
public String getNome() {  
    return nome;  
}
```

```
public void setNome(String nome) {  
    this.nome = nome;  
}
```

```
public String getLogin() {  
    return login;  
}
```

```
public void setLogin(String login) {  
    this.login = login;  
}
```

```
public String getSenha() {  
    return senha;  
}
```

```
        public void setSenha(String senha) {
            this.senha = senha;
        }

        public List<Tarefa> getTarefas() {
            return tarefas;
        }

        public void setTarefas(List<Tarefa> tarefas) {
            this.tarefas = tarefas;
        }

        @Override
        public String toString() {
            return "Usuario [idUserario=" + idUsuario + ", nome=" + nome
                + ", login=" + login + ", senha=" + senha + "]\n";
        }
    }

package br.com.brq.entities;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import br.com.brq.entities.types.Status;

@Entity
@Table(name = "tarefa")
@NamedQueries({
    {
        @NamedQuery(name = Tarefa.FINDBY_DATA,
            query = "select t from Tarefa as t where t.dataEntrega between :p1 and :p2")
    }
})
```

```
    }  
)  
public class Tarefa {  
  
    public static final String FINDBY_DATA = "tarefa.findbydata";  
  
    @Id  
    @GeneratedValue(strategy = GenerationType.AUTO)  
    @Column(name = "idtarefa")  
    private Integer idTarefa;  
  
    @Column(name = "nome", length = 50, nullable = false)  
    private String nome;  
  
    @Column(name = "descricao", length = 255, nullable = false)  
    private String descricao;  
  
    @Temporal(TemporalType.DATE)  
    @Column(name = "dataentrega", nullable = false)  
    private Date dataEntrega;  
  
    @Enumerated(EnumType.STRING)  
    @Column(name = "status", length = 25, nullable = false)  
    private Status status;  
  
    @ManyToOne //muitas tarefas para 1 usuario..  
    @JoinColumn(name = "idusuario", nullable = false) //chave estrangeira  
    private Usuario usuario; // TER-1  
  
    public Tarefa() {  
        // TODO Auto-generated constructor stub  
    }  
  
    public Tarefa(Integer idTarefa, String nome, String descricao,  
        Date dataEntrega, Status status) {  
        super();  
        this.idTarefa = idTarefa;  
        this.nome = nome;  
        this.descricao = descricao;  
        this.dataEntrega = dataEntrega;  
        this.status = status;  
    }  
  
    public Tarefa(Integer idTarefa, String nome, String descricao,  
        Date dataEntrega, Status status, Usuario usuario) {  
        this(idTarefa, nome, descricao, dataEntrega, status);  
        this.usuario = usuario;  
    }  
}
```

```
public Integer getIdTarefa() {  
    return idTarefa;  
}  
  
public void setIdTarefa(Integer idTarefa) {  
    this.idTarefa = idTarefa;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public String getDescricao() {  
    return descricao;  
}  
  
public void setDescricao(String descricao) {  
    this.descricao = descricao;  
}  
  
public Date getDataEntrega() {  
    return dataEntrega;  
}  
  
public void setDataEntrega(Date dataEntrega) {  
    this.dataEntrega = dataEntrega;  
}  
  
public Status getStatus() {  
    return status;  
}  
  
public void setStatus(Status status) {  
    this.status = status;  
}  
  
public Usuario getUsuario() {  
    return usuario;  
}  
  
public void setUsuario(Usuario usuario) {  
    this.usuario = usuario;  
}
```

```
@Override
public String toString() {
    return "Tarefa [idTarefa=" + idTarefa + ", nome=" + nome
        + ", descricao=" + descricao + ", dataEntrega="
        + dataEntrega + ", status=" + status + "]\n";
}
}
```

Configurando a conexão com o banco de dados:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.dialect">
            org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.connection.driver_class">
            com.mysql.jdbc.Driver</property>
        <property name="hibernate.connection.url">
            jdbc:mysql://localhost:3306/aula17</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">brqbrq</property>

        <property name="hibernate.format_sql">true</property>
        <property name="hibernate.show_sql">true</property>

        <mapping class="br.com.brq.entities.Usuario"/>
        <mapping class="br.com.brq.entities.Tarefa"/>
    </session-factory>
</hibernate-configuration>
```

Criando um DAO Generico para a camada de persistência de dados:

```
package br.com.brq.generics;

import java.io.Serializable;
import java.util.List;

public interface IDAOGeneric<T, K extends Serializable> {

    void saveOrUpdate(T obj) throws Exception;
    void delete(T obj) throws Exception;
    List<T> findAll() throws Exception;
    T findById(K id) throws Exception;
}
```

```
package br.com.brq.generics;

import java.io.Serializable;
import java.util.List;

import org.hibernate.Query;
import org.hibernate.Session;
import org.hibernate.Transaction;

import br.com.brq.persistence.HibernateUtil;

public abstract class DAOGeneric<T, K extends Serializable> implements IDAOGeneric<T, K> {

    protected Session session;
    protected Transaction transaction;
    protected Query query;

    private Class<T> tipo;

    //construtor..
    public DAOGeneric(Class<T> tipo) {
        this.tipo = tipo;
    }

    @Override
    public void saveOrUpdate(T obj) throws Exception {
        session = HibernateUtil.getSessionFactory().openSession();

        transaction = session.beginTransaction();
        session.saveOrUpdate(obj);
        transaction.commit();

        session.close();
    }

    @Override
    public void delete(T obj) throws Exception {
        session = HibernateUtil.getSessionFactory().openSession();

        transaction = session.beginTransaction();
        session.delete(obj);
        transaction.commit();

        session.close();
    }
}
```



```
@Override
public List<T> findAll() throws Exception {
    session = HibernateUtil.getSessionFactory().openSession();

    @SuppressWarnings("unchecked")
    List<T> lista = session.createCriteria(tipo).list();

    session.close();
    return lista;
}

@Override
public T findById(K id) throws Exception {
    session = HibernateUtil.getSessionFactory().openSession();

    @SuppressWarnings("unchecked")
    T obj = (T) session.get(tipo, id);

    session.close();
    return obj;
}
}
```

Classe para criptografia da senha em MD5

Utilitarios

```
package br.com.brq.security;

import java.math.BigInteger;
import java.security.MessageDigest;

public class Criptografia {

    // método para encriptar senha em MD5..
    public static String encriptarSenha(String senha) throws Exception {

        // criando um algoritmo de criptografia MD5/SHA1
        MessageDigest m = MessageDigest.getInstance("MD5");

        // preparando a senha para ser criptografada
        m.update(senha.getBytes(), 0, senha.length());

        // fazendo a criptografia e retornando o resultado..
        return new BigInteger(1, m.digest()).toString(16);
    }
}
```

Criando as classes de persistência para Tarefa e Usuario

```
package br.com.brq.persistence;

import java.util.Date;
import java.util.List;

import br.com.brq.entities.Tarefa;
import br.com.brq.generics.DAOGeneric;

public class DAOTarefa extends DAOGeneric<Tarefa, Integer>{

    public DAOTarefa() {
        super(Tarefa.class);
    }

    public List<Tarefa> findByData(Date dataIni, Date dataFim) throws Exception{
        session = HibernateUtil.getSessionFactory().openSession();

        query = session.getNamedQuery(Tarefa.FINDBY_DATA);
        query.setDate("p1", dataIni);
        query.setDate("p2", dataFim);
        List<Tarefa> lista = query.list();

        session.close();
        return lista;
    }
}
```

```
package br.com.brq.persistence;

import br.com.brq.entities.Usuario;
import br.com.brq.generics.DAOGeneric;

public class DAOUsuario extends DAOGeneric<Usuario, Integer>{

    public DAOUsuario() {
        super(Usuario.class);
    }

    public boolean hasLogin(String login) throws Exception{
        session = HibernateUtil.getSessionFactory().openSession();

        query = session.getNamedQuery(Usuario.HAS_LOGIN);
        query.setString("p1", login);
    }
}
```

```
Long qtd = (Long) query.uniqueResult();

session.close();
return qtd > 0; //true / false..
}

public Usuario authenticate(String login, String senha) throws Exception{
    session = HibernateUtil.getSessionFactory().openSession();

    query = session.getNamedQuery(Usuario.AUTHENTICATE);
    query.setString("p1", login);
    query.setString("p2", senha);
    Usuario u = (Usuario) query.uniqueResult();

    session.close();
    return u; //retornando o usuario..
}
}
```

Criando as páginas de cadastro e autenticação de Usuario:

cadastro.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!-- taglib do struts -->
<%@ taglib uri="/struts-tags" prefix="s" %>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css"
        href="/aula17_projeto/css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css"
        href="/aula17_projeto/css/bootstrap-theme.min.css"/>

    <style>
        .label { color: #000; }
    </style>

</head>
<body class="container">

    <h2>Cadastro de Usuarios</h2>
    <a href="/aula17_projeto/index.jsp">Voltar</a> para a página inicial.
</body>
```

```
<!-- Formulário para cadastro do cliente.. -->
<s:form id="formulario" method="post" action="cadastrarusuario">

    <s:textfield label="Nome do Usuario:" name="usuario.nome"/>
    <s:textfield label="Login de Acesso:" name="usuario.login"/>
    <s:password label="Informe sua Senha:" name="usuario.senha"/>

    <s:submit value="Cadastrar Usuario"
              cssClass="btn btn-success btn-sm"/>

</s:form>

<p>
    ${mensagem}
</p>

</body>
</html>
```

login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!-- taglib do struts -->
<%@ taglib uri="/struts-tags" prefix="s" %>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css"
          href="/aula17_projeto/css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css"
          href="/aula17_projeto/css/bootstrap-theme.min.css"/>

    <style>
        .Label { color: #000; }
    </style>

</head>
<body class="container">

    <h2>Autenticação de Usuarios</h2>
    <a href="/aula17_projeto/index.jsp">Voltar</a> para a página inicial.
    <hr/>

    <!-- Formulário para cadastro do cliente.. -->
    <s:form id="formulario" method="post"
          action="autenticarusuario">

        <s:textfield label="Login de Acesso:" name="usuario.login"/>
        <s:password label="Senha de Acesso:" name="usuario.senha"/>
```

```
<s:submit value="Acessar Sistema"
          cssClass="btn btn-primary btn-sm"/>

</s:form>

<p>
    ${mensagem}
</p>

</body>
</html>
```

Criando o controle de Usuario utilizando Annotations

**** O nome da classe deverá terminar com **Action** e a classe deverá estar em um pacote denominado **action****

```
package br.com.brq.action;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.apache.struts2.ServletActionContext;
import org.apache.struts2.convention.annotation.Action;
import org.apache.struts2.convention.annotation.Result;

import br.com.brq.entities.Usuario;
import br.com.brq.persistence.DAOUsuario;
import br.com.brq.security.Criptografia;

//Classe de controle para Usuario (MVC)
public class UsuarioAction{

    // atributo para armazenar os dados dos campo
    // das páginas de cadastro e login
    private Usuario usuario; //resgatar os dados dos formularios

    // construtor...
    public UsuarioAction() {
        usuario = new Usuario(); // espaço de memória..
    }

    //Método para executar o cadastro do usuario..
    @Action(
        value = "cadastrarusuario", //nome da ação (chamado no formulário..)
        results = { //redirecionamentos..
            @Result(name = "sucesso", location = "/cadastro.jsp")
        }
    )
}
```

```
public String cadastrar(){

    HttpServletRequest request = ServletActionContext.getRequest();

    try{
        DAOUsuario d = new DAOUsuario(); //classe de persistencia..
        if( ! d.hasLogin(usuario.getLogin())) //se o login ja foi cadastrado..
        {
            //criptografar a senha do usuario..
            usuario.setSenha(Criptografia.encryptarSenha(usuario.getSenha()));

            d.saveOrUpdate(usuario); //gravando..

            request.setAttribute("mensagem", usuario.getNome()
                + ", cadastrado com sucesso.");
            usuario = new Usuario(); //limoando espao de memoria..
        }
        else
        {
            throw new Exception("Este login ja encontra-se em uso.
                Tente outro.");
        }
    }
    catch(Exception e){
        request.setAttribute("mensagem", "Erro: " + e.getMessage());
    }

    return "sucesso";
}

@Action(
    value = "/autenticarusuario",
    results = {
        @Result( name = "sucesso", location = "/admin/index.jsp"),
        @Result( name = "acessonegado", location = "/login.jsp"),
    }
)
public String autenticar(){

    HttpServletRequest request = ServletActionContext.getRequest();

    try{

        DAOUsuario d = new DAOUsuario(); //pesistencia..
        //buscar 1 usuario pelo login e senha..

        Usuario u = d.authenticate(usuario.getLogin(),
            Criptografia.encryptarSenha(usuario.getSenha()));

        if(u != null){ //se usuario foi encontrado..
            //armazenar os dados do usuario em sessão..
            HttpSession session = request.getSession();
            session.setAttribute("usuariologado", u);
        }
    }
}
```

```
        return "sucesso";
    }
    else{
        throw new Exception("Acesso Negado. Tente novamente.");
    }
}
catch(Exception e){
    request.setAttribute("mensagem", e.getMessage());
}
return "acessonegado";
}

@Action(
    value = "/logout",
    results = {
        @Result(name = "sucesso", location = "/login.jsp")
    }
)
public String logout(){
    //destruir os valores contidos na sessão..
    HttpServletRequest request = ServletActionContext.getRequest();
    HttpSession session = request.getSession();

    session.removeAttribute("usuariologado"); //excluindo..
    session.invalidate(); //limpar toda a sessão..

    return "sucesso";
}

public Usuario getUsuario() {
    return usuario;
}

public void setUsuario(Usuario usuario) {
    this.usuario = usuario;
}
}
```

Página inicial da área restrita do projeto:

/admin/index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>

<link rel="stylesheet" type="text/css"
      href="/aula17_projeto/css/bootstrap.min.css"/>
<link rel="stylesheet" type="text/css"
      href="/aula17_projeto/css/bootstrap-theme.css"/>

</head>
<body class="container">

<h2>Bem vindo ao Sistema</h2>
Área Restrita da aplicação.
<hr/>

Usuario Autenticado: <label> ${usuariologado.nome} </label> <br/>
Login de Acesso: <label> ${usuariologado.login} </label> <br/>
<br/>

<a href="/logout.action" class="btn btn-danger btn-sm">
  Sair do Sistema
</a>

<hr/>

<ul>
<li> <a href="/aula17_projeto/admin/cadastro.jsp">
  Cadastrar Tarefas</a> </li>
<li> <a href="/aula17_projeto/admin/consulta.jsp">
  Consultar Tarefas</a> </li>
</ul>

</body>
</html>
```

Modificando a consulta de tarefas para filtrar pelo Usuario:

```
package br.com.brq.entities;

import java.util.Date;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.EnumType;
import javax.persistence.Enumerated;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
```



```
import javax.persistence.Id;
import javax.persistence.JoinColumn;
import javax.persistence.ManyToOne;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
import javax.persistence.Table;
import javax.persistence.Temporal;
import javax.persistence.TemporalType;

import br.com.brq.entities.types.Status;

@Entity
@Table(name = "tarefa")
@NamedQueries(
    {
        @NamedQuery(name = Tarefa.FINDBY_DATA,
            query = "select t from Tarefa as t inner join t.usuario as u " +
                "where t.dataEntrega between :p1 and :p2 " +
                "and u.idUsuario = :p3")
    }
)
public class Tarefa {

    public static final String FINDBY_DATA = "tarefa.findbydata";

    @Id
    @GeneratedValue(strategy = GenerationType.AUTO)
    @Column(name = "idtarefa")
    private Integer idTarefa;

    @Column(name = "nome", length = 50, nullable = false)
    private String nome;

    @Column(name = "descricao", length = 255, nullable = false)
    private String descricao;

    @Temporal(TemporalType.DATE)
    @Column(name = "dataentrega", nullable = false)
    private Date dataEntrega;

    @Enumerated(EnumType.STRING)
    @Column(name = "status", length = 25, nullable = false)
    private Status status;

    @ManyToOne //muitas tarefas para 1 usuario..
    @JoinColumn(name = "idusuario", nullable = false) //chave estrangeira
    private Usuario usuario; // TER-1
```

```
public Tarefa() {  
    // TODO Auto-generated constructor stub  
}  
  
public Tarefa(Integer idTarefa, String nome, String descricao,  
    Date dataEntrega, Status status) {  
    super();  
    this.idTarefa = idTarefa;  
    this.nome = nome;  
    this.descricao = descricao;  
    this.dataEntrega = dataEntrega;  
    this.status = status;  
}  
  
public Tarefa(Integer idTarefa, String nome, String descricao,  
    Date dataEntrega, Status status, Usuario usuario) {  
    this(idTarefa, nome, descricao, dataEntrega, status);  
    this.usuario = usuario;  
}  
  
public Integer getIdTarefa() {  
    return idTarefa;  
}  
  
public void setIdTarefa(Integer idTarefa) {  
    this.idTarefa = idTarefa;  
}  
  
public String getNome() {  
    return nome;  
}  
  
public void setNome(String nome) {  
    this.nome = nome;  
}  
  
public String getDescricao() {  
    return descricao;  
}  
  
public void setDescricao(String descricao) {  
    this.descricao = descricao;  
}  
  
public Date getDataEntrega() {  
    return dataEntrega;  
}
```

```
public void setDataEntrega(Date dataEntrega) {
    this.dataEntrega = dataEntrega;
}

public Status getStatus() {
    return status;
}

public void setStatus(Status status) {
    this.status = status;
}

public Usuario getUsuario() {
    return usuario;
}

public void setUsuario(Usuario usuario) {
    this.usuario = usuario;
}

@Override
public String toString() {
    return "Tarefa [idTarefa=" + idTarefa + ", nome=" + nome
        + ", descricao=" + descricao + ", dataEntrega="
        + dataEntrega + ", status=" + status + "];"
}
}
```

Criando o método na classe de persistência:

```
package br.com.brq.persistence;

import java.util.Date;
import java.util.List;

import br.com.brq.entities.Tarefa;
import br.com.brq.generics.DAOGeneric;

public class DAOTarefa extends DAOGeneric<Tarefa, Integer>{

    public DAOTarefa() {
        super(Tarefa.class);
    }

    public List<Tarefa> findByData(Date dataIni, Date dataFim, Integer idUsuario)
    throws Exception{
        session = HibernateUtil.getSessionFactory().openSession();
    }
}
```

```
        query = session.getNamedQuery(Tarefa.FINDBY_DATA);
        query.setDate("p1", dataIni);
        query.setDate("p2", dataFim);
        query.setInteger("p3", idUsuario);
        List<Tarefa> lista = query.list();

        session.close();
        return lista;
    }
}
```

Criando a página de cadastro de Tarefas:

/admin/cadastro.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>

<!-- Taglib do Struts -->
<%@ taglib uri="/struts-tags" prefix="s" %>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css"
        href="/aula17_projeto/css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css"
        href="/aula17_projeto/css/bootstrap-theme.css"/>

    <style type="text/css">
        .label { color: #000; }
    </style>

</head>
<body class="container">

    <h2>Cadastro de Tarefas</h2>
    <a href="/aula17_projeto/admin/index.jsp">Voltar</a>
        para a página inicial
    <br/><br/>

    Usuario Autenticado: <label> ${usuariologado.nome} </label> <br/>
    Login de Acesso: <label> ${usuariologado.login} </label>
    <hr/>

    <div>
        Para cadastrar uma nova tarefa, informe os dados abaixo:
    </div>

    <s:form name="formulario" method="post" action="cadastrartarefa">
```

```
<s:textfield label="Nome da Tarefa"
             name="tarefa.nome"/>
<s:textarea label="Descrição da Tarefa" name="tarefa.descricao"
            cols="30" rows="4"/>
<s:textfield label="Data de Entrega"
             name="tarefa.dataEntrega" />

<s:submit value="Cadastrar Tarefa"
          cssClass="btn btn-success btn-sm"/>

</s:form>

<p>
    ${mensagem}
</p>

</body>
</html>
```

Página de Consulta de Tarefas:

/admin/consulta.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
      pageEncoding="ISO-8859-1"%>

<!-- TagLib do struts -->
<%@ taglib uri="/struts-tags" prefix="s" %>

<!-- TagLib do JSTL -->
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fnc" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>

<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Projeto</title>

    <link rel="stylesheet" type="text/css"
          href="/aula17_projeto/css/bootstrap.min.css"/>
    <link rel="stylesheet" type="text/css"
          href="/aula17_projeto/css/bootstrap-theme.css"/>

    <style type="text/css">
        .label { color: #000; }
    </style>

</head>
<body class="container">

    <h2>Consulta de Tarefas</h2>
    <a href="/aula17_projeto/admin/index.jsp">Voltar</a>
    para a página inicial
    <br/><br/>
```

```
Usuario Autenticado: <label> ${usuariologado.nome} </label> <br/>
Login de Acesso: <label> ${usuariologado.login} </label>
<hr/>

<div>
    Para pesquisar tarefas, informe o periodo de datas:
</div>

<s:form name="formulario" method="post" action="consultartarefas">

    <s:textfield label="Data de" name="dataInicio"/>
    <s:textfield label="Até" name="dataFim"/>

    <s:submit value="Pesquisar Tarefas"
        cssClass="btn btn-primary btn-sm"/>

</s:form>

<p>
    ${mensagem}
</p>

<c:if test="${listagemTarefas != null
    && fnc:length(listagemTarefas) > 0}">

    <table class="table">
        <thead>
            <tr>
                <th>Código</th>
                <th>Nome da Tarefa</th>
                <th>Data de Entrega</th>
                <th>Descrição</th>
                <th>Status</th>
                <th>Usuario</th>
            </tr>
        </thead>
        <tbody>

            <s:iterator value="ListagemTarefas">
                <tr>
                    <td> <s:property value="idTarefa"/> </td>
                    <td> <s:property value="nome"/> </td>
                    <td> <s:property value="dataEntrega"/> </td>
                    <td> <s:property value="descricao"/> </td>
                    <td> <s:property value="status"/> </td>
                    <td> <s:property value="usuario.nome"/> </td>
                </tr>
            </s:iterator>

        </tbody>
    </table>

</c:if>
</body>
</html>
```

Classe Action para gerenciamento de tarefas:

```
package br.com.brq.action;

import java.util.Date;
import java.util.List;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;

import org.apache.struts2.ServletActionContext;
import org.apache.struts2.convention.annotation.Action;
import org.apache.struts2.convention.annotation.Result;

import br.com.brq.entities.Tarefa;
import br.com.brq.entities.Usuario;
import br.com.brq.entities.types.Status;
import br.com.brq.persistence.DAOTarefa;

public class TarefaAction {

    private Tarefa tarefa; // atributo..
    private List<Tarefa> listagemTarefas; //exibir a consulta..
    private Date dataInicio; //filtro..
    private Date dataFim; //filtro..

    // construtor..
    public TarefaAction() {
        tarefa = new Tarefa(); // instanciando..
    }

    @Action(
        value = "/cadastrartarefa",
        results = {
            @Result(name = "sucesso", location = "/admin/cadastro.jsp")
        }
    )
    public String cadastrar(){

        HttpServletRequest request = ServletActionContext.getRequest();
        HttpSession session = request.getSession();

        try{

            tarefa.setStatus(Status.EM_ABERTO); //definindo o status..

            //resgatar o usuario da sessão...
            Usuario u = (Usuario) session.getAttribute("usuariologado");
            tarefa.setUsuario(u); //relacionando o usuario com a tarefa..

            DAOTarefa d = new DAOTarefa(); //persistencia..
            d.saveOrUpdate(tarefa); //gravando..
```

```
request.setAttribute("mensagem", "Tarefa " + tarefa.getNome()
                                + ", cadastrado com sucesso.");

    tarefa = new Tarefa(); //novo espaço de memoria..
}
catch(Exception e){
    request.setAttribute("mensagem", e.getMessage());
}

return "sucesso";
}

@Action(
    value = "/consultartarefas",
    results = {
        @Result(name = "sucesso", location = "/admin/consulta.jsp")
    }
)
public String consultar(){

    HttpServletRequest request = ServletActionContext.getRequest();
    HttpSession session = request.getSession();

    try{
        //resgatar o usuario da sessão..
        Usuario u = (Usuario) session.getAttribute("usuariologado");

        //executar a busca..
        DAOTarefa d = new DAOTarefa();
        listagemTarefas = d.findByData(dataInicio, dataFim, u.getIdUsuario());

        request.setAttribute("mensagem", "Tarefas encontradas: "
                                + listagemTarefas.size());
    }
    catch(Exception e){
        request.setAttribute("mensagem", e.getMessage());
    }

    return "sucesso";
}

public Tarefa getTarefa() {
    return tarefa;
}

public void setTarefa(Tarefa tarefa) {
    this.tarefa = tarefa;
}
}
```



```
public List<Tarefa> getListagemTarefas() {  
    return listagemTarefas;  
}  
  
public void setListagemTarefas(List<Tarefa> listagemTarefas) {  
    this.listagemTarefas = listagemTarefas;  
}  
  
public Date getDataInicio() {  
    return dataInicio;  
}  
  
public void setDataInicio(Date dataInicio) {  
    this.dataInicio = dataInicio;  
}  
  
public Date getDataFim() {  
    return dataFim;  
}  
  
public void setDataFim(Date dataFim) {  
    this.dataFim = dataFim;  
}  
}
```
