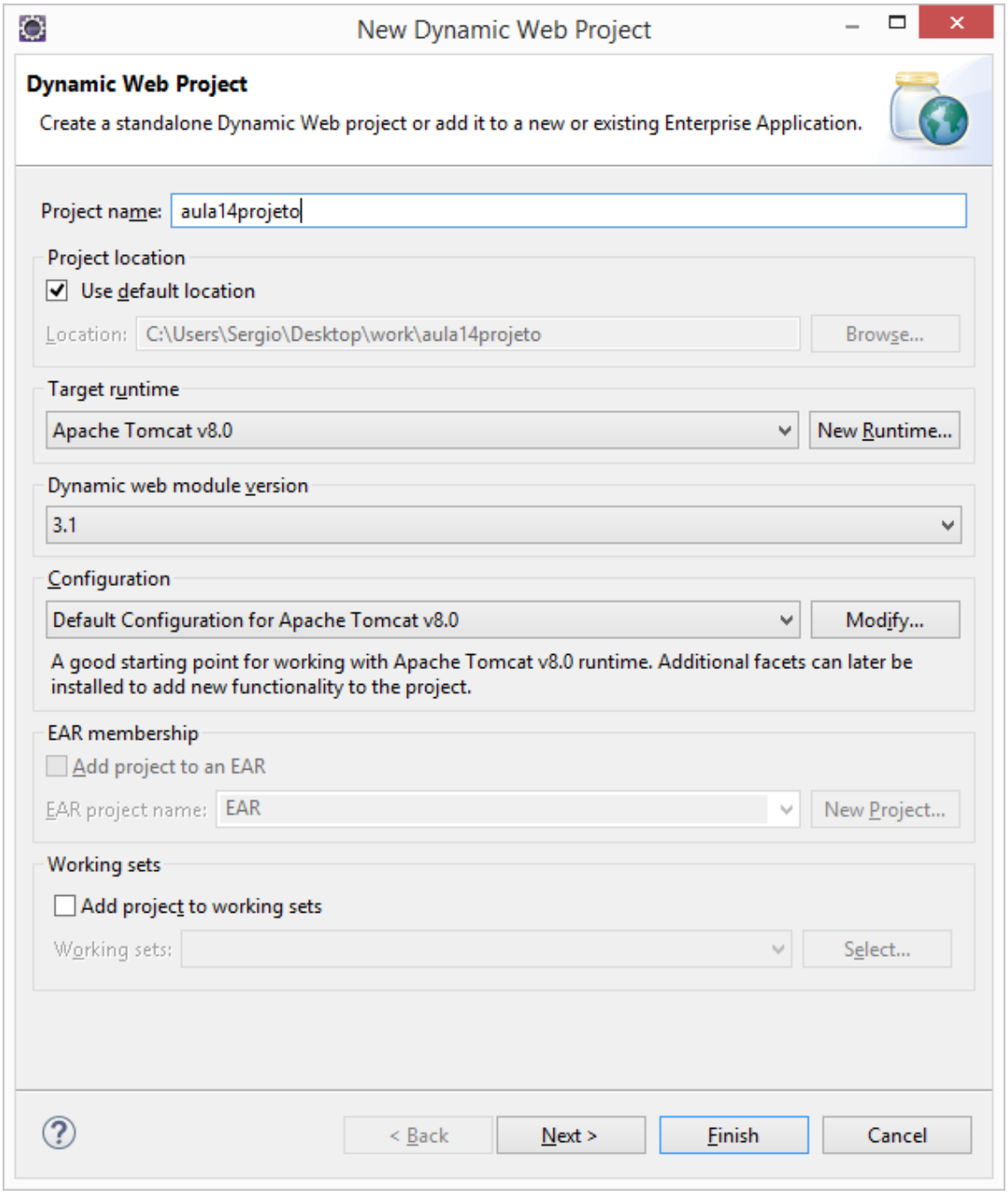
Criando o projeto:

File > New > Dynamic Web Project



Criando a entidade Pessoa com JPA

Java Persistence API

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

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.Table;

@Entity
@Table(name = "pessoa")
@NamedQueries(
		{
			@NamedQuery(name  = Pessoa.FINDBY_NOME,
			query = "select p from Pessoa as p
				where p.nome like :p1 order by p.nome"),

			@NamedQuery(name  = Pessoa.HAS_EMAIL,
			query = "select count(p) from Pessoa as p where p.email = :p1")
		}
)
public class Pessoa {

	//constantes..
	public static final String FINDBY_NOME = "pessoa.findbynome";
	public static final String HAS_EMAIL = "pessoa.hasemail";

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

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

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

```
@Column(name = "email", length = 100, nullable = false, unique = true)
private String email;

public Pessoa() {
}

public Pessoa(Integer idPessoa, String nome, String sobrenome, String email) {
	this.idPessoa = idPessoa;
	this.nome = nome;
	this.sobrenome = sobrenome;
	this.email = email;
}

public Integer getIdPessoa() {
	return idPessoa;
}

public void setIdPessoa(Integer idPessoa) {
	this.idPessoa = idPessoa;
}

public String getNome() {
	return nome;
}

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

public String getSobrenome() {
	return sobrenome;
}

public void setSobrenome(String sobrenome) {
	this.sobrenome = sobrenome;
}

public String getEmail() {
	return email;
}

public void setEmail(String email) {
	this.email = email;
}
```

```
        @Override
        public String toString() {
                return "Pessoa [idPessoa=" + idPessoa + ", nome=" + nome + ",
                sobrenome=" + sobrenome + ", email=" + email
                                + "]";
        }
}
```

Configuração para acesso ao banco de dados:
mysql_hibernate.cfg.xml

```
<?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/aula14</property>
    <property name="hibernate.connection.username">root</property>
    <property name="hibernate.connection.password">brqbrq</property>

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

    <mapping class="br.com.brq.entities.Pessoa"/>

 </session-factory>
</hibernate-configuration>
```

Gerando as tabelas no banco de dados:

```
package br.com.brq.util;

import org.hibernate.cfg.AnnotationConfiguration;
import org.hibernate.cfg.Configuration;
import org.hibernate.tool.hbm2ddl.SchemaExport;

public class GenerateTables {

        public static void main(String[] args) {
```

```
		try{

			Configuration cfg = new AnnotationConfiguration();
			cfg.configure("br/com/brq/config/mysql_hibernate.cfg.xml");

			SchemaExport s = new SchemaExport(cfg);
			s.create(true, true);
		}
		catch(Exception e){
			System.out.println(e.getMessage());
		}
	}
}
```

Criando um DAO Generico com Hibernate:

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

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

//T -> generico para representar a entidade
//K -> generico para representar a chave primaria
public interface IDAOGeneric<T, K extends Serializable> {

	void insert(T obj) throws Exception;

	void update(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.persistence.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> {

	//componentes do hibernate..
	protected Session session;
	protected Transaction transaction;
	protected Query query;

	//atributo utilizado para as consultas genericas..
	private Class<T> tipo; //tipo da entidade utilizada na consulta..

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

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

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

		session.close();
	}

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

		transaction = session.beginTransaction();
		session.update(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;
	}
}
```

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

import java.util.List;

import br.com.brq.entities.Pessoa;
import br.com.brq.persistence.generics.DAOGeneric;


public class DAOPessoa extends DAOGeneric<Pessoa, Integer>{
```

```
	public DAOPessoa() {
		super(Pessoa.class); //construtor da classe abstrata..
	}

	public List<Pessoa> findByNome(String nome) throws Exception{
		session = HibernateUtil.getSessionFactory().openSession();

		query = session.getNamedQuery(Pessoa.FINDBY_NOME);
		query.setString("p1", "%" + nome + "%");
		List<Pessoa> lista = query.list();

		session.close();
		return lista;
	}

	public boolean hasEmail(String email) throws Exception{
		session = HibernateUtil.getSessionFactory().openSession();

		query = session.getNamedQuery(Pessoa.HAS_EMAIL);
		query.setString("p1", email);
		Long qtd = (Long) query.uniqueResult();

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

Ajax

Assyncronous JavaScript and XML

Ajax (acrônimo em língua inglesa de Asynchronous Javascript and XML, em português "Javascript e XML Assíncrono") é o uso metodológico de tecnologias como Javascript e XML, providas por navegadores, para tornar páginas Web mais interativas com o usuário, utilizando-se de solicitações assíncronas de informações. Apesar do nome, a utilização de XML não é obrigatória (JSON é frequentemente utilizado) e as solicitações também não necessitam ser assíncronas.

Criando a página .jsp realizando requisições assíncronas ao Servlet:

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

	<script type="text/javascript" src="js/jquery-1.12.4.min.js"></script>

	<script type="text/javascript">

	$(document).ready( //quando a página for carregada..
	function(){ //faça..

	//criar um evento quando o botão 'btncadastro' for clicado..
	$("#btncadastro").click( //quando o botão for clicado..
		function(){ //faça..

		//JSON (JavaScript Object Notation)
		var dados = {
			"nome"		: $("#nome").val(),
			"sobrenome"	: $("#sobrenome").val(),
			"email"		: $("#email").val()
		};

		//alert(JSON.stringify(dados));
		$("#mensagem").html("Enviando requisição...");

		//ajax..
		$.ajax(
		{
			type : "POST",
			url  : "ControlePessoa?action=cadastrar",
			data : dados,
			success : function(msg){
				$("#mensagem").html(msg);

				//limpar os campos..
				$("#nome").val("");
				$("#sobrenome").val("");
				$("#email").val("");

		},
		error : function(e){
				$("#mensagem").html("Erro: " + e.status);
		}
		}
```

```
		);

		}
		);

		//criar um evento quando digitarmos no campo de pesquisa
		$("#nomepesquisa").keyup( //quando digitarmos no campo..
		function(){ //faça..

		//JSON (Javascript object notation)
		var dados = {
			"nome" : $("#nomepesquisa").val()
		};

		//função ajax..
		$.ajax(
		{
			type : "POST",
			url  : "ControlePessoa?action=pesquisar",
			data : dados,
			success : function(resultado){
				$("#tabela tbody").html(resultado);
			},
			error : function(e){
				$("#mensagem").html("Erro: " + e.status);
			}
		}
		);

	}
);

}
);

</script>

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

	<div class="well well-sm">
		<h2>Projeto Controle de Pessoas</h2>
		JSP, Servçets, Ajax (JQuery) e Hibernate
	</div>

	<div class="col-md-12">

		<div class="col-md-4">

			<h3>Cadastro de Pessoas</h3>
			<hr/>

			<label>Nome da Pessoa:</label>
			<input type="text" id="nome" class="form-control"
				placeholder="Digite aqui"/>
```

```
                <br/>

                <label>Sobrenome:</label>
                <input type="text" id="sobrenome" class="form-control"
                        placeholder="Digite aqui"/>
                <br/>

                <label>Email:</label>
                <input type="text" id="email" class="form-control"
                        placeholder="Digite aqui"/>
                <br/>

                <button id="btncadastro" class="btn btn-success">
                      Cadastrar Pessoa
                </button>
                <br/><br/>
                <span id="mensagem"></span>
          </div>

          <div class="col-md-8">

                <h3>Pesquisa de Pessoas</h3>
                <hr/>

                <label>Informe o nome desejado:</label>
                <input type="text" id="nomepesquisa" class="form-control"
                        placeholder="Digite aqui"/>
                <br/><br/>

                <table id="tabela" class="table table-hover">
                      <thead>
                            <tr>
                                  <th>Código</th>
                                  <th>Nome da Pessoa</th>
                                  <th>Sobrenome</th>
                                  <th>Email</th>
                            </tr>
                      </thead>
                      <tbody>

                      </tbody>
                      <tfoot>
                            <tr>
                                  <td colspan="4">
                                        Registros encontrado(s):
                                        <span id="quantidade">0</span>
                                  </td>
                            </tr>
                      </tfoot>
                </table>
          </div>
    </div>
</body>
</html>
```

Criando o Servlet para responder as requisições da página:

```
package br.com.brq.control;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.List;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import br.com.brq.entities.Pessoa;
import br.com.brq.persistence.DAOPessoa;

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

	public ControlePessoa() {
		super();
	}

	protected void execute(HttpServletRequest request, HttpServletResponse response)
			throws ServletException, IOException {

		String action = request.getParameter("action");
		PrintWriter out = response.getWriter();
```

```
if("cadastrar".equalsIgnoreCase(action)){
	try{

		Pessoa p = new Pessoa(); //entidade..
		p.setNome(request.getParameter("nome"));
		p.setSobrenome(request.getParameter("sobrenome"));
		p.setEmail(request.getParameter("email"));

		DAOPessoa d = new DAOPessoa(); //persistencia..
		if( ! d.hasEmail(p.getEmail())){ //se email nao existe..

			d.insert(p); //gravando..

			out.println("Pessoa " + p.getNome() + ", cadastrado
				com sucesso.");
		}
		else{
			throw new Exception("Este email ja encontra-se
				cadastrado, tente outro.");
		}
	}
	catch(Exception e){
		out.print(e.getMessage()); //mensagem de erro..
	}
}
else if("pesquisar".equalsIgnoreCase(action)){

	try{
		String nome = request.getParameter("nome");

		DAOPessoa d = new DAOPessoa(); //persistencia..
		List<Pessoa> lista = d.findByNome(nome);
		//executando a busca..

		if(lista.size() > 0){ //se alguma pessoa foi encontrada..

			for(Pessoa p : lista){
			//varrendo cada pessoa contida na lista..
			out.println("<tr>");
			out.println("<td>" + p.getIdPessoa() + "</td>");
			out.println("<td>" + p.getNome() + "</td>");
			out.println("<td>" + p.getSobrenome() + "</td>");
			out.println("<td>" + p.getEmail() + "</td>");
			out.println("</tr>");
			}
		}
		else{
```

```
                                        out.println("Nenhum registro foi encontrado.");
                                }
                        }
                        catch(Exception e){
                                out.println(e.getMessage());
                        }
                }

        }

        protected void doGet(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                execute(request, response);
        }

        protected void doPost(HttpServletRequest request, HttpServletResponse response)
                        throws ServletException, IOException {
                execute(request, response);
        }
}
```
