Nome: _____ Data: ____/____/____

Projeto – Utilizando Hibernate

1. Criar uma pasta chamada LojaVirtual.
2. Criar uma pasta chamada BD para salvar o script do Banco de Dados.
3. Abrir o PGAdmin III, criar um novo banco de dados com o nome: BDLojaVirtual
4. Abrir o editor SQL (botão SQL na barra de tarefas do PGAdmin) e criar o seguinte script:

```
CREATE TABLE produto
(
  pro_id          serial          NOT NULL        PRIMARY KEY,
  pro_nome        varchar(60),
  pro_preco       float
);
```

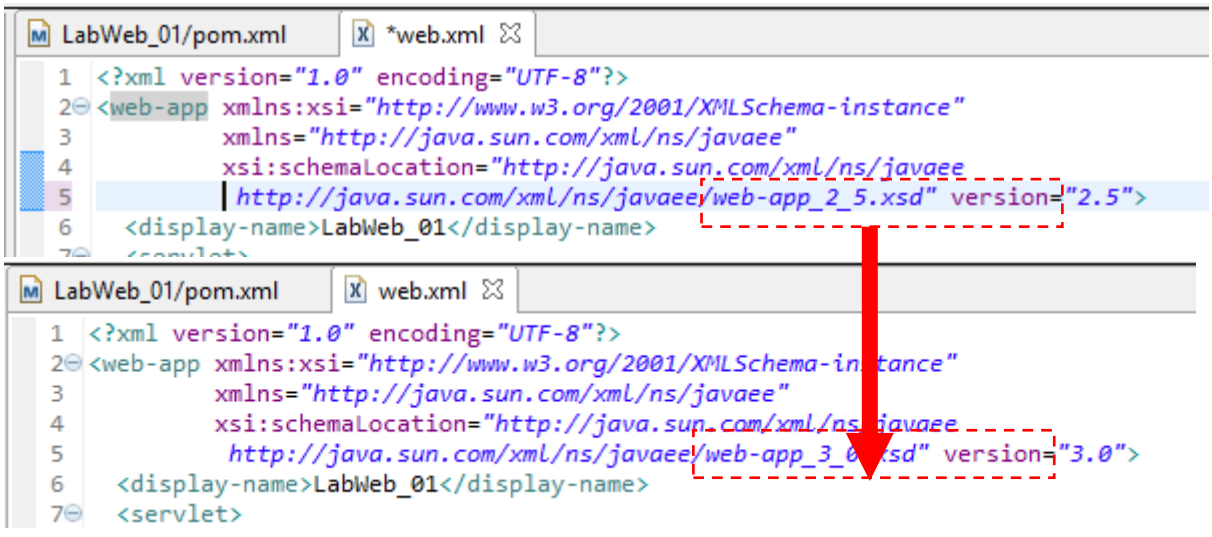
5. Criar um novo projeto Maven no Eclipse com o nome LojaVirtual.
6. Configurar o arquivo pom.xml conforme a imagem a seguir:

Nome: _____ Data: ____/____/____

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>br.com.exemplo</groupId>
  <artifactId>ExemploProdutoHibernate</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>war</packaging>
  <properties>
        <project.build.sourceEncoding>
            UTF-8
        </project.build.sourceEncoding>
  </properties>

  <build>
        <plugins>
            <plugin>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.0</version>
                <configuration>
                    <source>1.8</source>
                    <target>1.8</target>
                </configuration>
            </plugin>
        </plugins>
  </build>

  <dependencies>
        <dependency>
            <groupId>com.sun.faces</groupId>
            <artifactId>jsf-api</artifactId>
            <version>2.2.10</version>
        </dependency>
        <dependency>
            <groupId>com.sun.faces</groupId>
            <artifactId>jsf-impl</artifactId>
            <version>2.2.10</version>
        </dependency>
        <dependency>
            <groupId>org.postgresql</groupId>
            <artifactId>postgresql</artifactId>
            <version>9.4-1201-jdbc41</version>
        </dependency>
        <dependency>
            <groupId>org.hibernate</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>4.3.4.Final</version>
        </dependency>

  </dependencies>

</project>
```

7. Acesse as propriedades do Projeto e marque a opção JavaServer Faces (versão 2.2).

8. Altere o arquivo web.xml conforme a figura abaixo.



Nome: _____ Data: ____/____/____

9. Crie um novo arquivo com o nome: hibernate.cfg.xml no pacote default (src/main/java) com o seguinte conteúdo:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-configuration PUBLIC
        "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
        "http://www.hibernate.org/dtd/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <!-- Configuração da conexão com o banco PostgreSQL e dialeto -->
        <property name="hibernate.dialect">org.hibernate.dialect.PostgreSQLDialect</property>
        <property name="hibernate.connection.driver_class">org.postgresql.Driver</property>
        <property name="hibernate.connection.username">postgres</property>
        <property name="hibernate.connection.password">123</property>
        <property name="hibernate.connection.url">jdbc:postgresql://localhost:5432/hibernatedb</property>

        <!-- Usando as configurações do C3PO para pool de conexões -->
        <property name="c3po.min_size">5</property>
        <property name="c3po.max_size">20</property>
        <property name="c3po.timeout">300</property>
        <property name="c3po.max_statements">50</property>
        <property name="c3po.idle_test_period">3000</property>
        <!-- Configurações de debug -->
        <property name="show_sql">true</property>
        <property name="format_sql">true</property>
        <property name="generate_statistics">true</property>
        <property name="use_sql_comments">true</property>

        <!-- Mapeamento das Classes -->
        <mapping class="beans.Produto"/>
    </session-factory>

</hibernate-configuration>
```

Ao digitar faça as devidas adequações nas propriedades: password e url, digitando a senha conforme instalação e fazendo a referência correta do seu banco de dados.

10. Crie os pacotes: beans, negocio, persistência e teste.
11. Crie uma nova classe no pacote beans com o nome Produto.java e digite o seguinte código:

Nome: _____ Data: ____/____/____

```java
package beans;

import javax.persistence.*;

@Entity
@Table(name = "produto")
public class Produto {

    @Id
    @GeneratedValue
    @Column(name = "pro_id")
    private int id;
    @Column(name = "pro_nome", length = 60, nullable = true)
    private String nome;
    @Column(name = "pro_preco", nullable = true)
    private float preco;

    public int getId() {
        return id;
    }
    public void setId(int id) {
        this.id = id;
    }
    public String getNome() {
        return nome;
    }
    public void setNome(String nome) {
        this.nome = nome;
    }
    public float getPreco() {
        return preco;
    }
    public void setPreco(float preco) {
        this.preco = preco;
    }
}
```

12. No pacote persistência, crie o HibernateUtil.java e digite o código a seguir:

Nome: _____ Data: ____/____/____

```java
package persistencia;

import org.hibernate.boot.registry.StandardServiceRegistry;

public class HibernateUtil {
    private static final SessionFactory sessionFactory  = buildSessionFactory();

    private static SessionFactory buildSessionFactory() {
        try {
            Configuration cfg = new Configuration();
            cfg.configure("hibernate.cfg.xml");

            StandardServiceRegistryBuilder registradorServico = new StandardServiceRegistryBuilder();
            registradorServico.applySettings(cfg.getProperties());
            StandardServiceRegistry servico = registradorServico.build();

            return cfg.buildSessionFactory(servico);
    } catch (Throwable e) {
            System.out.println("Criação inicial do objeto SessionFactory falhou. Erro: " + e);
            throw new ExceptionInInitializerError(e);
        }
    }
    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
}
```

13. Ainda no pacote persistência crie o arquivo ProdutoDAO.java com o seguinte código:

```java
package persistencia;

import java.io.Serializable;
import java.util.List;
import org.hibernate.*;
import beans.Produto;

public class ProdutoDAO implements Serializable {

    private static final long serialVersionUID = 1L;

    public static void inserir(Produto produto) {
        Session sessao = HibernateUtil.getSessionFactory().openSession();
        Transaction t = sessao.beginTransaction();
        sessao.save(produto);
        t.commit();
        sessao.close();
    }

    public static void alterar(Produto produto) {
        Session sessao = HibernateUtil.getSessionFactory().openSession();
        Transaction t = sessao.beginTransaction();
        sessao.update(produto);
        t.commit();
        sessao.close();
    }
```

```
28      public static void excluir(Produto produto) {
29          Session sessao = HibernateUtil.getSessionFactory().openSession();
30          Transaction t = sessao.beginTransaction();
31          sessao.delete(produto);
32          t.commit();
33          sessao.close();
34      }
35
36      public static List<Produto> listagem(String filtro) {
37          Session sessao = HibernateUtil.getSessionFactory().openSession();
38          Query consulta;
39          if (filtro.trim().length() == 0) {
40              consulta = sessao.createQuery("from Produto order by pro_nome");
41          }
42          else {
43              consulta = sessao.createQuery("from Produto "
44                      + "where pro_nome like :parametro order by pro_nome");
45              consulta.setString("parametro", "%" + filtro + "%");
46          }
47          List lista = consulta.list();
48          sessao.close();
49          return lista;
50      }
```

14. Criar uma nova classe com o nome: ProdutoCtrl.java no pacote negocio e digitar o código a seguir:

```
1  package negocio;
2
3  import java.io.Serializable;
4  import java.util.List;
5  import javax.faces.bean.*;
6  import persistencia.ProdutoDAO;
7  import beans.Produto;
8
9  @ManagedBean
10 @SessionScoped
11 public class ProdutoCtrl implements Serializable{
12
13         private static final long serialVersionUID = 1L;
14         private Produto produto;
15
16         public Produto getProduto() {
17             return produto;
18         }
19         public void setProduto(Produto produto) {
20             this.produto = produto;
21         }
22
23         public List<Produto> getListagem() {
24             return ProdutoDAO.listagem();
25         }
26
```

Nome: _____ Data: ____/____/____

```
27          public String actionGravar() {
28              if (produto.getId() == 0) {
29                  ProdutoDAO.inserir(produto);
30                  return actionInserir();
31              }
32              else {
33                  ProdutoDAO.alterar(produto);
34                  return "lista_produto";
35              }
36          }
37
38          public String actionInserir() {
39              produto = new Produto();
40              return "form_produto";
41          }
42
43          public String actionExcluir(Produto p) {
44              ProdutoDAO.excluir(p);
45              return "lista_produto";
46          }
47
48          public String actionAlterar(Produto p) {
49              produto = p;
50              return "form_produto";
51          }
52  }
```

15. Criar as páginas web: index.xhtml, lista_produto.xhtml e form_produto.xhtml.