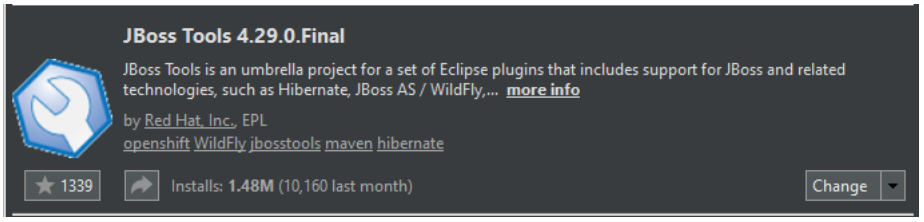
TP8 : JEE

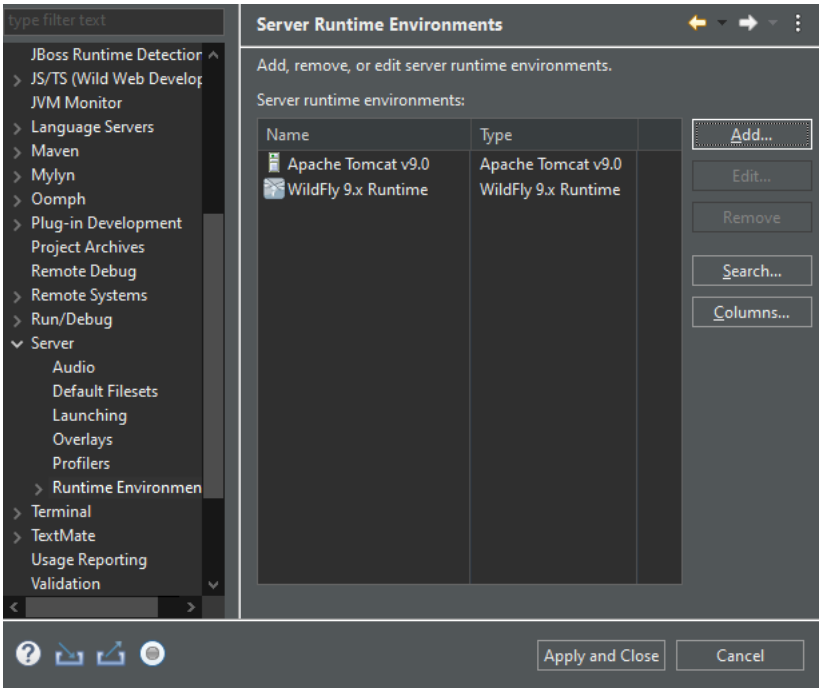
Tags	
status	sent

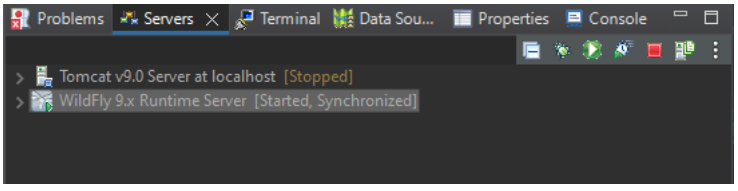
Par : IBTISSAM MASSA - GINFO2

EJB (Entreprise Java Bean)

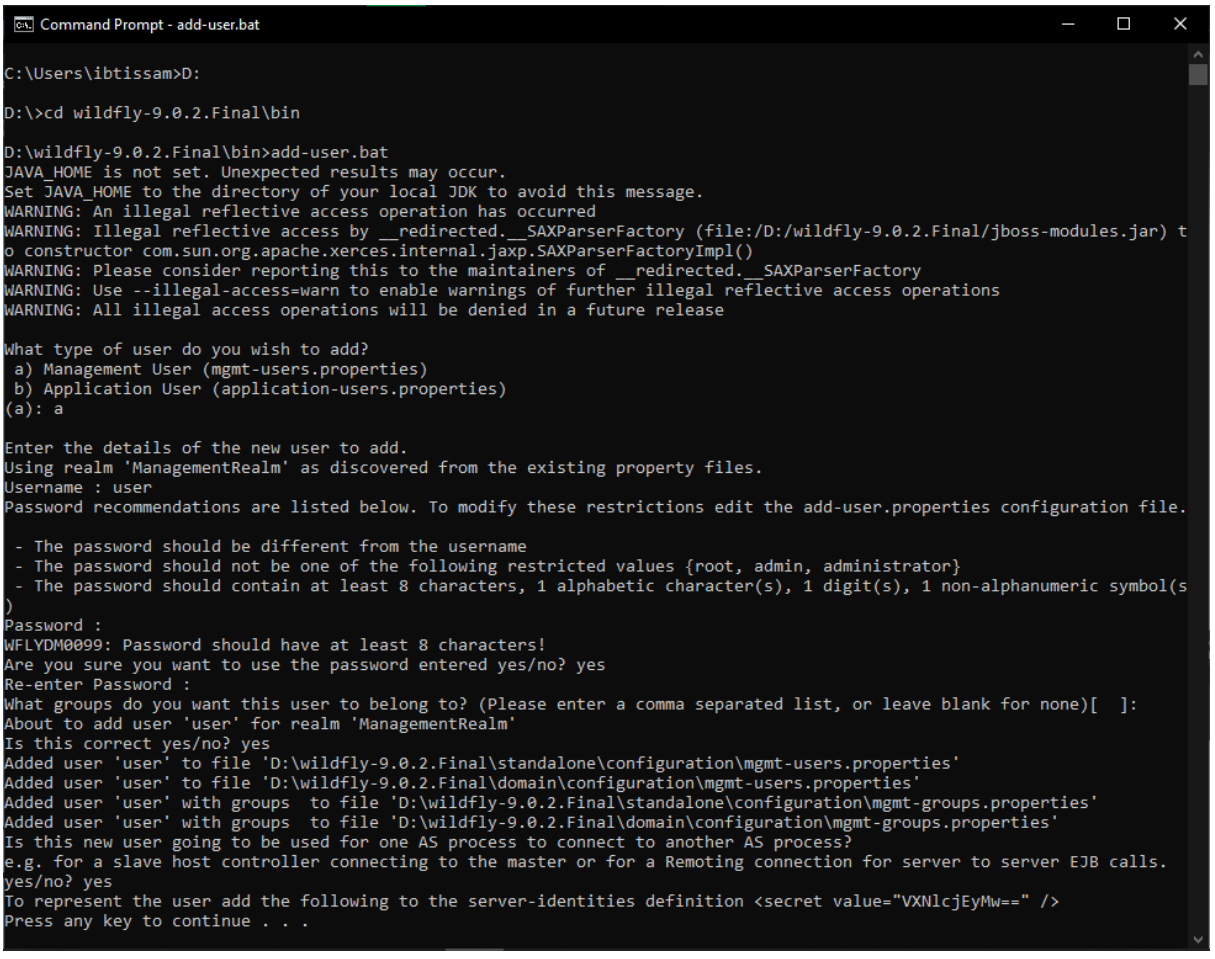
> L'objectif de ce TP est de :
> - Se familiariser avec l'utilisation des EJB
> - Développer une application 3 tiers
> - Créer un DataSource
> - Manipuler les Entities (Entity)
> - Créer de l'EJB Session
> - Développer d'une application web (MVC : JSP et Servlet)

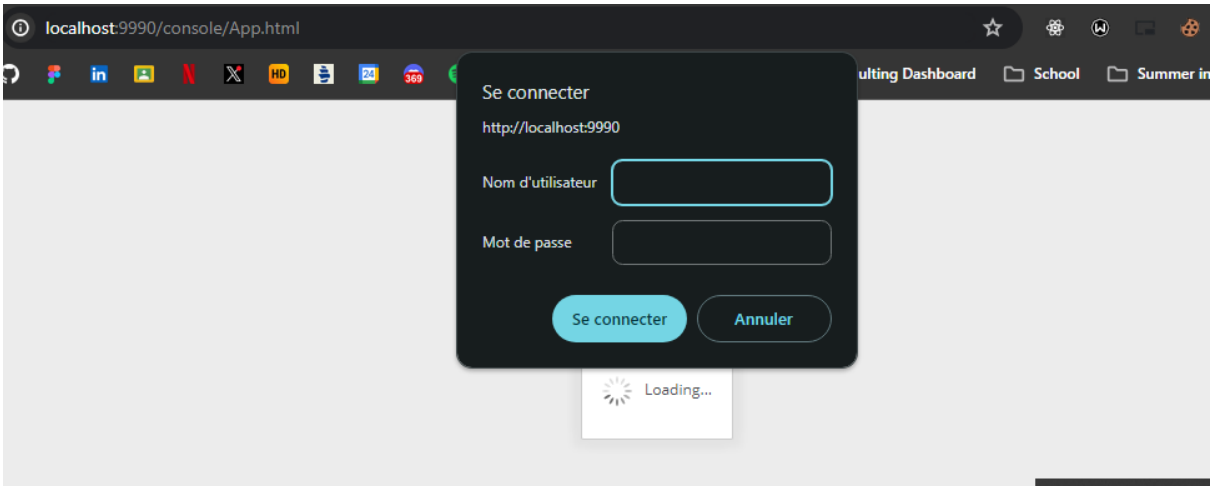


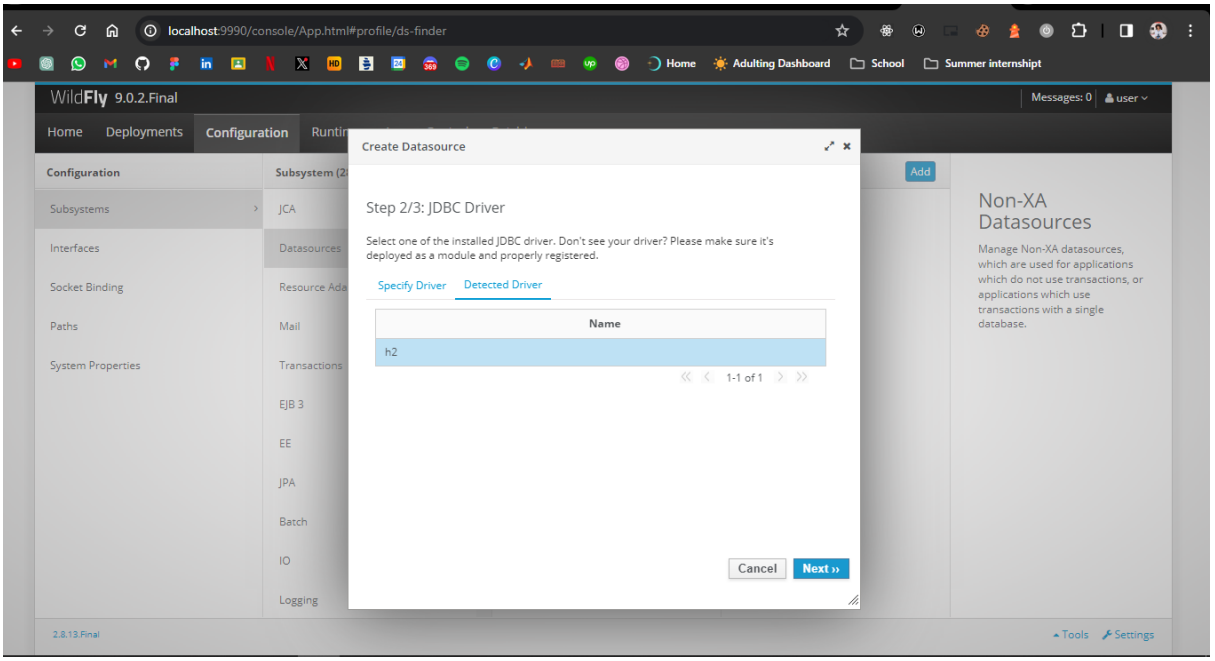


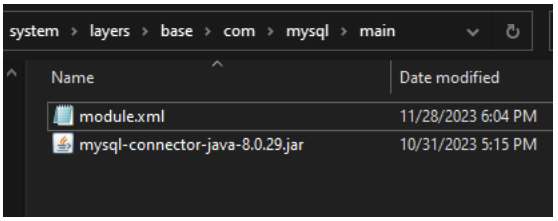



localhost:8080
Home
Adulting Dashboard
School
Summer internship
WildFly
Welcome to WildFly 9
Your WildFly 9 is running.
Documentation | Quickstarts | Administration Console
WildFly Project | User Forum | Report an issue
JBoss Community
To replace this page simply deploy your own war with / as its context path.
To disable it, remove "welcome-content" handler for location / in undertow subsystem


Command Prompt - add-user.bat
C:\Users\ibtissam>D:
D:\>cd wildfly-9.0.2.Final\bin
D:\wildfly-9.0.2.Final\bin>add-user.bat
JAVA_HOME is not set. Unexpected results may occur.
Set JAVA_HOME to the directory of your local JDK to avoid this message.
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by __redirected.__SAXParserFactory (file:/D:/wildfly-9.0.2.Final/jboss-modules.jar) t
o constructor com.sun.org.apache.xerces.internal.jaxp.SAXParserFactoryImpl()
WARNING: Please consider reporting this to the maintainers of __redirected.__SAXParserFactory
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
What type of user do you wish to add?
a) Management User (mgmt-users.properties)
b) Application User (application-users.properties)
(a): a
Enter the details of the new user to add.
Using realm 'ManagementRealm' as discovered from the existing property files.
Username : user
Password recommendations are listed below. To modify these restrictions edit the add-user.properties configuration file.
- The password should be different from the username
- The password should not be one of the following restricted values {root, admin, administrator}
- The password should contain at least 8 characters, 1 alphabetic character(s), 1 digit(s), 1 non-alphanumeric symbol(s
)
Password :
WFLYDM0099: Password should have at least 8 characters!
Are you sure you want to use the password entered yes/no? yes
Re-enter Password :
What groups do you want this user to belong to? (Please enter a comma separated list, or leave blank for none)[ ]:
About to add user 'user' for realm 'ManagementRealm'
Is this correct yes/no? yes
Added user 'user' to file 'D:\wildfly-9.0.2.Final\standalone\configuration\mgmt-users.properties'
Added user 'user' to file 'D:\wildfly-9.0.2.Final\domain\configuration\mgmt-users.properties'
Added user 'user' with groups to file 'D:\wildfly-9.0.2.Final\standalone\configuration\mgmt-groups.properties'
Added user 'user' with groups to file 'D:\wildfly-9.0.2.Final\domain\configuration\mgmt-groups.properties'
Is this new user going to be used for one AS process to connect to another AS process?
e.g. for a slave host controller connecting to the master or for a Remoting connection for server to server EJB calls.
yes/no? yes
To represent the user add the following to the server-identities definition <secret value="VXNlcjEyMw==" />
Press any key to continue . . .



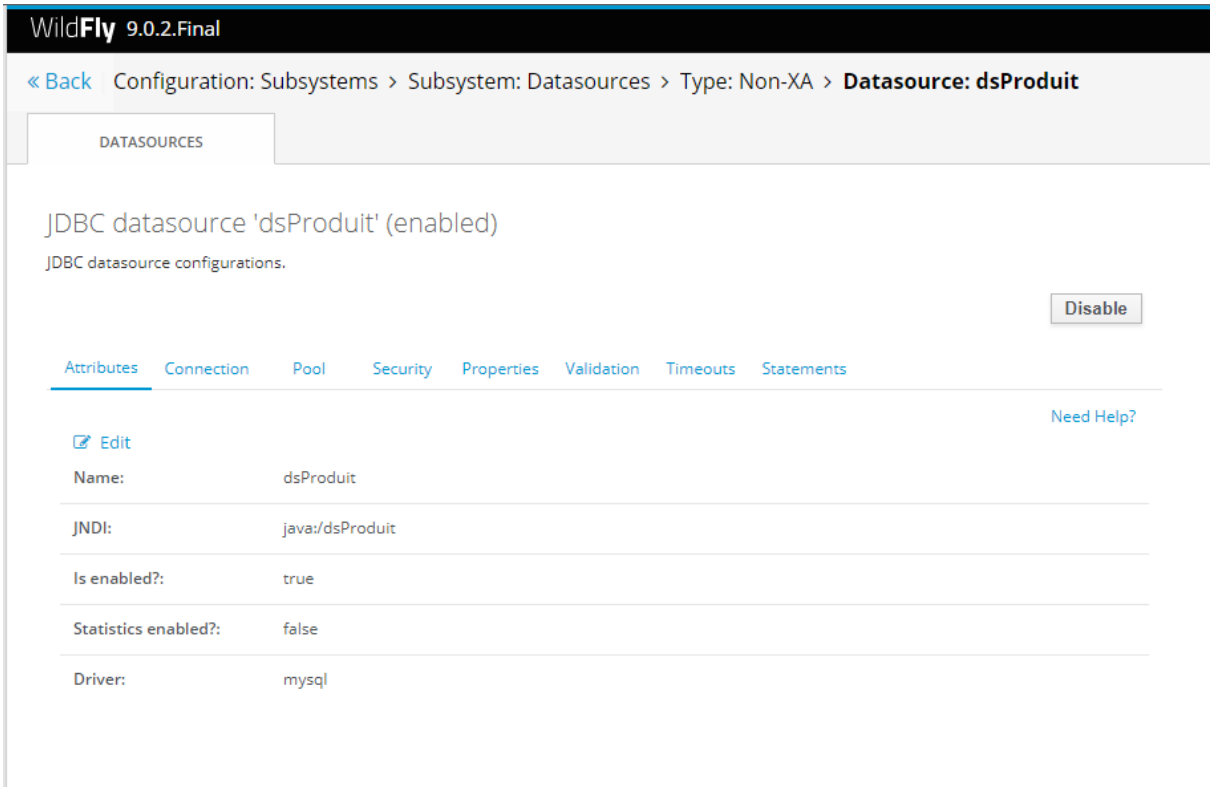


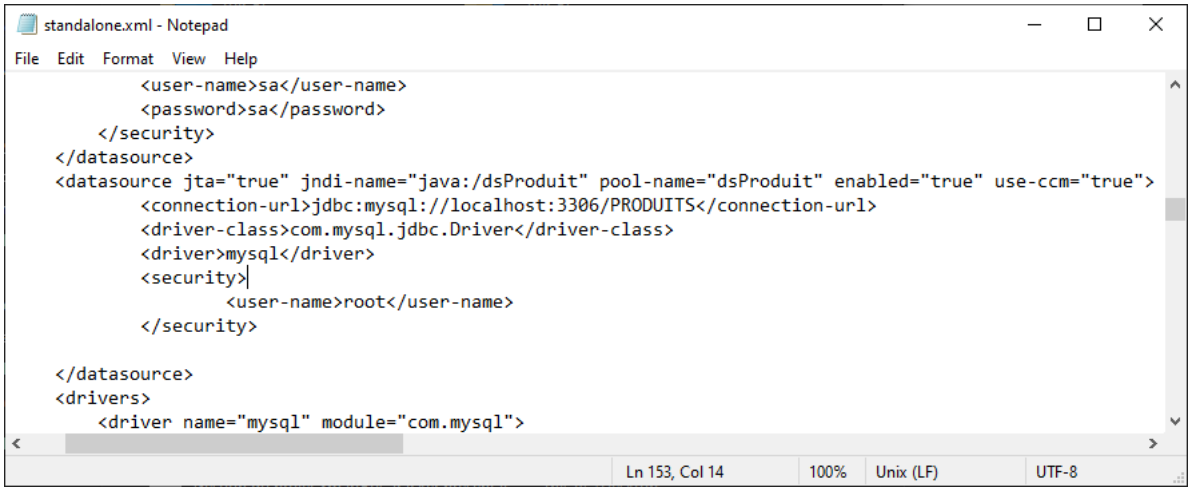


module.xml - Notepad

File Edit Format View Help

```
<module xmlns="urn:jboss:module:1.3" name="com.mysql">
        <resources>
                <resource-root path="mysql-connector-java-8.0.29.jar" />
        </resources>
        <dependencies>
                <module name="javax.api"/>
        </dependencies>
</module>
```





IproduitLocal:

```
package metier;
import java.util.List;
import javax.ejb.Local;
import metier.entities.Produit;

@Local
public interface IProduitLocal {
  public Produit addProduit(Produit produit);
  public Produit getProduit(int code);
  public void deleteProduit(int code);
  public Produit updateProduit(Produit produit);
  public List<Produit> listProduits();
  public List<Produit> listProduits(String mc);
  void ajouterQte(int code, int qt);
  void retirerQte(int code, int qt);
}
```

IProduitRemote:

```
package metier;
import java.util.List;
import javax.ejb.Remote;
import metier.entities.Produit;

@Remote
public interface IProduitRemote {
  public Produit addProduit(Produit produit);
  public Produit updateProduit(Produit produit);
  public void deleteProduit(int code);
  public Produit getProduit(int code);
  public List<Produit> listProduits();
  public List<Produit> listProduits(String mc);
  void ajouterQte(int code, int qt);
  void retirerQte(int code, int qt);
}
```

roduitEJBImpl:

```
package metier;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.TypedQuery;
import metier.entities.Produit;
@Stateless(name ="PROD")
public class ProduitEJBImpl implements IProduitLocal, IProduitRemote{
  @PersistenceContext(unitName = "GestionStockEJB")
  private EntityManager em;
  @Override
  public Produit addProduit(Produit produit) {
    em.persist(produit);
    return produit;
  }
  @Override
  public Produit getProduit(int code) {
    Produit produit = em.find(Produit.class, code);
    if(produit == null)
    throw new RuntimeException("Produit introuvable");
    return produit;
  }
  @Override
  public Produit updateProduit(Produit produit) {
    Produit p = em.find(Produit.class, produit.getId());
    if(p == null)
    throw new RuntimeException("Produit introuvable");
    p.setDesignation(produit.getDesignation());
    p.setPrix(produit.getPrix());
    p.setQuantite(produit.getQuantite());
    em.merge(p);
    return p;
  }
  @Override
  public void deleteProduit(int code) {
    Produit produit = em.find(Produit.class, code);
    if(produit == null)
    throw new RuntimeException("Produit introuvable");
    em.remove(produit);
  }
  @Override
  public List<Produit> listProduits() {
    return em.createQuery("From Produit", Produit.class).getResultList();
  }
  @Override
  public void ajouterQte(int code, int qt) {
    Produit produit = em.find(Produit.class, code);
    produit.setQuantite(produit.getQuantite() + qt);
  }
  @Override
  public void retirerQte(int code, int qt) {
    Produit produit = em.find(Produit.class, code);
    produit.setQuantite(produit.getQuantite() - qt);
  }
  @Override
  public List<Produit> listProduits(String mc) {
    TypedQuery<Produit> query = em.createQuery("select p from Produit where p.designation like :mc", Produit.class);
    query.setParameter("mc", "%" + mc + "%");
    return query.getResultList();
```

```
    }
}
```

ICategorieLocal:

```
package metier;
import java.util.List;
import javax.ejb.Local;
import metier.entities.Categorie;

@Local
public interface ICategorieLocal {
  public Categorie addCategorie(Categorie categorie);
  public Categorie getCategorie(int code);
  public void deleteCategorie(int code);
  public Categorie updateCategorie(Categorie categorie);
  public List<Categorie> listCategories();
  public List<Categorie> listCategories(String mc);
}
```

ICategorieRemote:

```
package metier;
import java.util.List;
import javax.ejb.Remote;
import metier.entities.Categorie;
@Remote
public interface ICategorieRemote {
  public Categorie addCategorie(Categorie categorie);
  public Categorie getCategorie(int code);
  public void deleteCategorie(int code);
  public Categorie updateCategorie(Categorie categorie);
  public List<Categorie> listCategories();
  public List<Categorie> listCategories(String mc);
}
```

CategorieEJBImpl:

```
package metier;
import java.util.List;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.TypedQuery;
import metier.entities.Categorie;

@Stateless(name ="CATE")
public class CategorieEJBImpl implements ICategorieLocal, ICategorieRemote{
  @PersistenceContext(unitName = "GestionStockEJB")
  private EntityManager em;
  @Override
  public Categorie addCategorie(Categorie categorie) {
  em.persist(categorie);
  return categorie;
}
@Override
public Categorie getCategorie(int code) {
Categorie categorie = em.find(Categorie.class, code);
if(categorie == null)
throw new RuntimeException("Categorie introuvable");
return categorie;
}
@Override
public Categorie updateCategorie(Categorie categorie) {
  Categorie c = em.find(Categorie.class, categorie.getId());
  if(c == null)
  throw new RuntimeException("Categorie introuvable");
  c.setLibelle(categorie.getLibelle());
  c.setDescription(categorie.getDescription());
  em.merge(c);
  return c;
}
@Override
public void deleteCategorie(int code) {
  Categorie categorie = em.find(Categorie.class, code);
```

```
  if(categorie == null)
  throw new RuntimeException("Categorie introuvable");
  em.remove(categorie);
}
@Override
public List<Categorie> listCategories() {
  return em.createQuery("From Categorie",Categorie.class).getResultList();
}
@Override
public List<Categorie> listCategories(String mc) {
  TypedQuery<Categorie> query = em.createQuery("select c from Categorie c where c.libelle like :mc", Categorie.class);query.setParamete
  return query.getResultList();
}
}
```

Categorie:

```
package metier.entities;
import java.io.Serializable;
import javax.persistence.*;

@Entity
@Table(name = "categories")
public class Categorie implements Serializable{
  private static final long serialVersionUID = 1L;
  @Id
  private int id;
  private String libelle;
  private String description;
  public Categorie() {
    super();
  }
  public Categorie(int id, String libelle, String description) {
    super();
    this.id = id;
    this.libelle = libelle;
    this.description = description;
  }
  public int getId() {
    return id;
  }
  public void setId(int id) {
    this.id = id;
  }
  public String getLibelle() {
    return libelle;
  }
  public void setLibelle(String libelle) {
    this.libelle = libelle;
  }
  public String getDescription() {
    return description;
  }
  public void setDescription(String description) {
    this.description = description;
  }
}
```

Produit:

```
package metier.entities;
import java.io.Serializable;
import javax.persistence.*;

@Entity
@Table(name = "produits")
public class Produit implements Serializable{
  private static final long serialVersionUID = 1L;
  @Id
  private int id;
  private String designation;
  private double prix;
  private int quantite;
  @ManyToOne
  @JoinColumn(name = "idCategorie")
  private Categorie categorie;
    public Produit() {
    super();
```

```
  }
  public Produit(int id, String designation, double prix, int quantite) {
    super();
    this.id = id;
    this.designation = designation;
    this.prix = prix;
    this.quantite = quantite;
  }
  public int getId() {
    return id;
  }
  public void setId(int id) {
    this.id = id;
  }
  public String getDesignation() {
    return designation;
  }
  public void setDesignation(String designation) {
    this.designation = designation;
  }
  public double getPrix() {
    return prix;
  }
  public void setPrix(double prix) {
    this.prix = prix;
  }
  public Categorie getCategorie() {
    return categorie;
  }
  public void setCategorie(Categorie categorie) {
    this.categorie = categorie;
  }
  public int getQuantite() {
    return quantite;
  }
  public void setQuantite(int quantite) {
    this.quantite = quantite;
  }
}
```

persistence :

```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1"
  xmlns="http://xmlns.jcp.org/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
  <persistence-unit name="GestionStockEJB" transaction-type="JTA">
    <jta-data-source>java:/dsGestionStock</jta-data-source>
    <class>metier.entities.Produit</class>
    <class>metier.entities.Categorie</class>
    <properties>
      <property name="javax.persistence.schemageneration.database.action" value="drop-and-create"/>
    </properties>
  </persistence-unit>
</persistence>
```

CategorieServlet :

```
package web;
import java.io.IOException;
import javax.ejb.EJB;
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 metier.ICategorieLocal;
import metier.entities.Categorie;

@WebServlet(urlPatterns = "/categories")
public class CategorieServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  @EJB
  private ICategorieLocal metier;
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponseresponse) throws ServletException, IOException {
    request.setAttribute("categories", metier.listCategories());
```

```
    request.getRequestDispatcher("Categorie.jsp").forward(request,response);
  }
  @Override
  protected void doPost(HttpServletRequest request, HttpServletResponseresponse) throws ServletException, IOException {
    String action = request.getParameter("action");
    if(action.equals("Ajouter")) {
      int code = Integer.parseInt(request.getParameter("code"));
      String libelle = request.getParameter("libelle");
      String description = request.getParameter("description");
      Categorie categorie = new Categorie(code, libelle, description);
      metier.addCategorie(categorie);
    }else if(action.equals("Modifier")) {
      int code = Integer.parseInt(request.getParameter("code"));
      String libelle = request.getParameter("libelle");
      String description = request.getParameter("description");
      Categorie categorie = new Categorie(code, libelle, description);
      metier.updateCategorie(categorie);
    }else if(action.equals("Rechercher")) {
      String libelle = request.getParameter("libelle");
      request.setAttribute("categories",
      metier.listCategories(libelle));
      request.getRequestDispatcher("Categorie.jsp").forward(request,response);
    }else if(action.contains("Supprimer")) {
      int code = Integer.parseInt(action.substring(13));
      metier.deleteCategorie(code);
    }
      request.setAttribute("categories", metier.listCategories());
      request.getRequestDispatcher("Categorie.jsp").forward(request,response);
  }
}
```

ProduitServlet:

```
package web;
import javax.ejb.EJB;
import javax.servlet.*;
import javax.servlet.http.*;
import metier.ICategorieLocal;
import metier.IProduitLocal;
import metier.entities.Categorie;
import metier.entities.Produit;
import javax.servlet.annotation.*;
import java.io.IOException;

@WebServlet(urlPatterns = "/produits")
public class ProduitServlet extends HttpServlet {
  private static final long serialVersionUID = 1L;
  @EJB
  private IProduitLocal metier;
  @EJB
  private ICategorieLocal metier1;
  @Override
  protected void doGet(HttpServletRequest request, HttpServletResponseresponse) throws ServletException, IOException {
    request.setAttribute("produits", metier.listProduits());
    request.setAttribute("categories", metier1.listCategories());
    request.getRequestDispatcher("Produit.jsp").forward(request,response);
  }
  @Override
  protected void doPost(HttpServletRequest request, HttpServletResponseresponse) throws ServletException, IOException {
    String action = request.getParameter("action");
    if(action.equals("Ajouter")) {
      int code = Integer.parseInt(request.getParameter("code"));
      String designation = request.getParameter("designation");
      double prix = Double.parseDouble(request.getParameter("prix"));
      int quantite = Integer.parseInt(request.getParameter("quantite"));
      int idCategorie =
      Integer.parseInt(request.getParameter("categorie"));
      Categorie categorie = metier1.getCategorie(idCategorie);
      Produit produit = new Produit(code, designation, prix, quantite);
      produit.setCategorie(categorie);
      metier.addProduit(produit);
    }else if(action.equals("Modifier")) {
      int code = Integer.parseInt(request.getParameter("code"));
      String designation = request.getParameter("designation");
      double prix = Double.parseDouble(request.getParameter("prix"));
      int quantite = Integer.parseInt(request.getParameter("quantite"));
      int idCategorie =
      Integer.parseInt(request.getParameter("categorie"));
      Categorie categorie = metier1.getCategorie(idCategorie);
      Produit produit = new Produit(code, designation, prix, quantite);
      produit.setCategorie(categorie);
```

```
      metier.updateProduit(produit);
    }else if(action.equals("Rechercher")) {
      String designation = request.getParameter("designation");
      request.setAttribute("produits",
      metier.listProduits(designation));
      request.getRequestDispatcher("Produit.jsp").forward(request,
      response);
    }else if(action.equals("+ Quantite")) {
      int code = Integer.parseInt(request.getParameter("code"));
      int quantite = Integer.parseInt(request.getParameter("quantite"));
      metier.ajouterQte(code, quantite);
    }else if(action.equals("- Quantite")) {
      int code = Integer.parseInt(request.getParameter("code"));
      int quantite = Integer.parseInt(request.getParameter("quantite"));
      metier.retirerQte(code, quantite);
    }else if(action.contains("Supprimer")) {
      int code = Integer.parseInt(action.substring(13));
      metier.deleteProduit(code);
    }
    request.setAttribute("produits", metier.listProduits());
    request.setAttribute("categories", metier1.listCategories());
    request.getRequestDispatcher("Produit.jsp").forward(request,response);
    }
}
```

web.xml

```
<web-app>
  <display-name>GestionStockWeb</display-name>
  <welcome-file-list>
    <welcome-file>Index.html</welcome-file>
  </welcome-file-list>
</web-app>
```